Contribution à l'analyse topologique des images : étude d'algorithmes de squelettisation pour images 2D et 3D, selon une approche topologie digitale ou topologie discrète.

Christophe Lohou

UNIVERSITÉ DE MARNE-LA-VALLÉE

THÈSE
pour obtenir le grade de
DOCTEUR DE L'UNIVERSITÉ DE MARNE-LA-VALLÉE
Discipline : Informatique Fondamentale et Applications
présentée et soutenue publiquement
par

Christophe LOHOU

le 20 décembre 2001

Titre :

Contribution à l'analyse topologique des images : étude d'algorithmes de squelettisation pour images 2D et 3D, selon une approche topologie digitale ou topologie discrète.

Directeur de thèse :
M. Gilles BERTRAND

JURY

M.	Christian	Ronse	*Président,*	*Rapporteur*
M.	Rémy	Malgouyres		*Rapporteur*
M.	Didier	Arquès		*Examinateur*
M.	Gilles	Bertrand		*Examinateur*
M.	Antoine	Manzanera		*Examinateur*

Thèse préparée au laboratoire A^2SI de l'ESIEE
(École Supérieure d'Ingénieurs en Électronique et Électrotechnique - Noisy-le-Grand)

Remerciements

Je remercie :

- M. Christian RONSE d'avoir accepté d'être rapporteur et président du jury. Sa lecture minutieuse et ses remarques ont largement contribué à améliorer mon mémoire,
- M. Rémy MALGOUYRES d'avoir accepté d'être rapporteur, ainsi que pour ses critiques constructives,
- M. Didier ARQUÈS et M. Antoine MANZANERA, qui m'ont fait l'honneur de participer au jury.

Je remercie :

- les membres du laboratoire A^2SI de l'ESIEE :
  - Gilles BERTRAND pour sa disponibilité, ses qualités pédagogiques et l'encadrement scientifique exceptionnel dont il m'a permis de bénéficier, ses précieux conseils lors de la rédaction et la correction d'articles. Je lui suis infiniment reconnaissant d'avoir partagé pistes et nombreuses idées brillantes, qui nous ont permis l'obtention de résultats significatifs dans le domaine de la squelettisation. Enfin, je le remercie de m'avoir également autorisé une grande liberté dans mes travaux et de m'avoir fait confiance. C'est un privilège d'avoir été initié à la Recherche dans de telles conditions.
  - Michel COUPRIE pour ses conseils toujours pertinents, tant scientifiques (travail sur les isométries) que rédactionnels (relecture d'articles), et ses encouragements,
  - Laurent PERROTON pour m'avoir encadré durant mon stage de DEA et m'avoir fait connaître les BDD, outils indispensables dans ma thèse, ainsi que pour son visualisateur,
  - et toutes les autres personnes du département informatique, en particulier : Dominique, Fabiana, Mohammed, Denis, Thierry, Eric et Christophe,
- mes amis stagiaires, thésards, ou docteurs :
  - Shahram, pour sa gentillesse légendaire,
  - les occupants de la "cellule" 5352 :
    - Jean-Christophe (pour ses nombreux conseils en tout genre),
    - Petr, entre autres pour la collaboration sur l'étude topologique du réseau vasculaire du foie,
    - Francisco (sa bonne humeur permanente et surtout sa disponibilité),
  - (puis par ordre d'apparition) Zouina, Hervé, Ahmed, Ailton, Junia, Khaled, Eva, Sébastien, Stéphane, Yukiko, Silvio, Xavier, Cédric, Marco, Alain ...

Je remercie mes parents Annette et Marcel, mes sœurs Nathalie et Céline (et leurs maris Christophe et Hervé) pour m'avoir encouragé durant ces années d'étude.

Je remercie infiniment ma compagne Delphine pour sa patience et son soutien, pour avoir "réglé son horloge sur la mienne".

Ce travail leur est dédié.

Merci à mes amis Carole et Hervé, Lydia et Étienne, Agnès et Fabrice, Paulo, la famille de Delphine et la mienne, Pacha ....

Table des matières

Table des figures

Liste des tableaux

Liste des algorithmes

Introduction

Avant-propos

Situation générale

Cette thèse est consacrée à l'étude d'algorithmes de squelettisation d'images, une importance plus particulière est donnée ici à la squelettisation parallèle d'images 3D binaires, même si la squelettisation pour images 2D binaires ou en niveaux de gris est étudiée.

Le but premier d'un algorithme de squelettisation est de fournir une image simplifiée, appelée *squelette*, à partir d'une image initiale en entrée ; l'algorithme opère alors par suppression de certains points de l'image initiale. Le processus de squelettisation peut être conçu de façon à contrôler les propriétés que l'on souhaite préserver de l'objet initial, et par conséquent retrouver dans le squelette. La contrainte principale concerne la préservation de la topologie : nous obtenons ainsi un squelette témoignant des propriétés topologiques de l'image initiale. Il est alors plus facile de manipuler le squelette pour en déduire des propriétés de l'image initiale. Une autre contrainte peut concerner la préservation de caractéristiques géométriques : par exemple, la préservation de courbes ou de surfaces durant le processus de squelettisation, témoignant ainsi de parties allongées dans l'image initiale. Une telle opération de suppression de points amène à la notion fondamentale de point simple : un point est dit *simple* si sa suppression d'un objet préserve la topologie de cet objet (notion séquentielle). Si l'on souhaite préserver des propriétés géométriques, un algorithme devra préserver certains points simples, appelés *points terminaux* ou *points extrémités*. Bien entendu, afin d'être efficace, un algorithme de squelettisation doit être conçu de façon à supprimer des points en parallèle (et si possible de façon symétrique, afin d'obtenir un bon centrage des squelettes). Mais la suppression parallèle de points simples peut ne pas préserver la topologie. Ce qui conduit à proposer des algorithmes utilisant des stratégies de suppression et ne supprimant qu'un sous-ensemble des points simples d'une image, à un instant donné.

Les algorithmes de squelettisation sont utilisés pour l'imagerie médicale (simplification d'angiographies [MS96], segmentation de la couche externe de la tête [BC00], du cortex cérébral [DC02], du réseau vasculaire du foie [DLPB99a] [DLPB99b] [Dok00], voir également [SNK94] [WB00]), l'extraction de caractéristiques (reconnaissance de caractères [LS95], extraction de minuties dans les empreintes digitales [CBB99] [FKVL99]), la science des matériaux (cf. la morphologie mathématique [Ser82]), la compression d'images, la polyédrisation [BM00], l'animation graphique, etc.

Dans [LS95], il est dit qu'il existe plusieurs centaines d'algorithmes de squelettisation pour images 2D. Juste avant cette thèse, il n'existait qu'une vingtaine d'algorithmes pour traiter les images 3D. Les raisons en sont que la topologie est intrinsèquement plus complexe en 3D qu'en

2D et qu'il est difficile de concevoir un algorithme de squelettisation pour images 3D préservant la topologie.

Problématique

La notion de point simple est une notion séquentielle en ce sens que la suppression itérative de points simples préserve la topologie (digitale). La suppression parallèle de points simples peut ne pas préserver la topologie. C'est le cas par exemple pour un ruban d'épaisseur 2 pour lequel tous les points sont simples ; la suppression en parallèle des points simples élimine cet objet et la topologie n'est pas préservée. En 2D, une stratégie directionnelle a été proposée par A. Rosenfeld [Ros75], [RK82] : elle consiste à éliminer des points simples ayant leur voisin selon une direction donnée dans le complémentaire de l'objet, puis à considérer d'autres directions. Mais cette stratégie ne fonctionne pas si elle est étendue directement au cas 3D.

D'autres stratégies ont été proposées : la stratégie utilisant des sous-mailles (subdivision de l'image), ou la stratégie fortement parallèle (utilisation d'un voisinage étendu).

Dans chacun de ces cas, la proposition de tels algorithmes est délicate, leur mise en œuvre est complexe (directions, isométries, voisinages étendus). Les squelettes ne sont pas bien définis (dans le sens qu'ils dépendent de l'ordre des directions, d'activation des sous-mailles) ou ne sont pas minces. De plus, certains algorithmes (non simulables directement par ordinateur) ne préservent pas la topologie, comme nous le révélons dans ce mémoire,

En 1995, G. Bertrand a proposé une solution au problème de la squelettisation parallèle par l'introduction des points P-simples. Des algorithmes, basés sur la suppression des points P-simples, ont été proposés ; ils utilisent alors les stratégies précédentes, mais exigent un voisinage étendu ou un étiquetage préliminaire.

Notre contribution

Contribution selon l'approche topologie digitale

Afin de mieux comprendre les mécanismes intrinsèques aux algorithmes de squelettisation, une part importante de notre travail a consisté à "disséquer" des algorithmes classiques de squelettisation afin de mettre en évidence leurs avantages et leurs inconvénients.

Nous proposons une démarche de conception d'algorithmes basés sur la suppression de points P^x-simples (classe de points P-simples, définie dans ce mémoire), et dérivant d'algorithmes existants ; cela afin d'obtenir des propriétés concernant la préservation de la topologie et leur "efficacité". Cette variante des points P-simples permet de s'affranchir de l'examen d'un voisinage étendu ou d'un étiquetage préliminaire. Cela nous amène à réfléchir sur la vérification systématique de la validité (au sens de la préservation de la topologie) d'algorithmes de squelettisation. Néanmoins, nos algorithmes souffrent des mêmes limitations que les autres en ce qui concerne les squelettes obtenus, puisque nos algorithmes utilisent les mêmes schémas de squelettisation (mais pas le même type de point supprimé). Nous avons également prouvé que deux algorithmes de référence ne préservaient pas la topologie.

Nécessité d'une nouvelle approche : la topologie discrète

Afin de pallier les difficultés précédentes, nous avons le choix de remettre en cause soit le cœur des schémas de squelettisation, soit le type de point supprimé. En 1999, G. Bertrand a proposé une étude concernant l'homotopie dans les ordres [Ber99]. Cette étude a permis d'une part de proposer la notion fondamentale de point α-unipolaire, puis celle de point α_n-simple ; et d'autre part de dériver aisément de nouveaux types d'algorithmes supprimant des points α_n-simples. Toute notre étude est fondée sur la définition de point unipolaire (et sur celle de point α_n-simple) et sur la propriété remarquable de pouvoir supprimer en parallèle les points α-unipolaires tout en préservant la topologie (*i.e.* la topologie discrète de l'ordre dans lequel nous travaillons).

Contribution selon l'approche topologie discrète

Nous proposons un schéma de squelettisation qui s'adapte aux images 2D binaires aussi bien qu'aux images 3D binaires ou aux images 2D en niveaux de gris. Seul le type de point éliminé varie, mais il est défini, par exemple en 3D à partir de celui des points que nous pouvons supprimer en 2D (notion générale de point α_n-simple). Nous avons proposé une notion formelle de point terminal de surface, grâce à laquelle les squelettes obtenus présentent des propriétés géométriques satisfaisantes ; cette notion est loin de celles empiriques proposées dans le cadre de la topologie digitale. De plus, à l'aide d'une transformation multipliant la taille de l'image, nous obtenons des squelettes bien centrés, bien définis (au sens qu'ils ne laissent pas de place à l'arbitraire – alors que c'est le cas avec l'approche topologie digitale, selon laquelle les squelettes dépendent de l'ordre choisi des directions, de l'activation des sous-mailles), minces et n'utilisant pas de voisinage étendu. À travers l'étude de filtrage topologique, nous illustrons le fait que ce cadre est très prometteur en ce qui concerne des traitements parallèles sur des images.

Notre contribution : conclusion

D'une part, nous avons proposé de nouveaux algorithmes de squelettisation selon l'approche topologie digitale [LB01] [LB02b] [LB]. Ceux-là sont plus efficaces que ceux dont ils dérivent. Nous avons mis en évidence deux algorithmes ne préservant pas la topologie [LB02a].

D'autre part, nous avons proposé de nouveaux algorithmes de squelettisation selon une nouvelle approche qui semble particulièrement prometteuse en ce qui concerne les opérations parallèles sur les images [LB99] [LB00a] [LB00b].

Perspectives

Nos résultats permettent d'envisager par la suite :

– selon l'approche topologie digitale, de proposer les corrections de deux algorithmes de squelettisation, d'approfondir l'étude d'une vérification systématique par ordinateur de la préservation de la topologie de certains algorithmes, et cela avec une variante des points

P^x-simples, enfin de proposer des variantes d'un autre algorithme existant, soit supprimant plus de points, soit utilisant un voisinage moindre [LB02c].

– selon l'approche topologie discrète, d'approfondir l'étude de la préservation de la topologie dans le cas 3D et l'étude réalisée dans le cadre des images 2D en niveaux de gris, de formaliser une notion de distance dans les ordres permettant l'obtention de propriétés pour nos squelettes et par exemple, une reconstruction parallèle (*i.e.* la récupération de l'objet initial – engendrant le squelette – à partir du squelette).

Plan détaillé du manuscrit

Ce document est divisé en quatre parties :

* La partie I propose un état de l'art présentant les notions de base de topologie digitale.
  – Le chapitre 1 présente les définitions de base de la topologie digitale : voisin, adjacence, chemin, connexité, composante connexe, trou.
  – Le chapitre 2 porte sur la simplicité dans le cas 2D, traitée à travers la notion de genre puis par différentes propriétés locales (dans un voisinage réduit autour d'un point) ou globales (dans l'image). Des précisions sont apportées sur ce que nous entendons par "préservation de la topologie". Différentes caractérisations d'un point simple sont proposées (notamment, celle utilisant les nombres topologiques). Des problèmes liés à la suppression parallèle de points nous amènent alors à présenter les notions d'ensemble simple et d'ensemble minimal non simple.
  – Le chapitre 3 est l'extension du chapitre 2 au cas 3D : il traite de la simplicité en 3D. Ce chapitre rend compte de la différence de difficulté entre les notions 2D et les notions 3D, à travers les exemples de calcul de genre par exemple. En 3D, la caractérisation d'un point simple la plus facile à mettre en œuvre et la plus efficace est celle effectuée à l'aide des nombres topologiques ; notons qu'en 2D cela était moins significatif. Sont ensuite précisées les notions d'ensemble simple, d'ensemble minimal non simple, dans le cadre d'une suppression parallèle de points.
  – Le chapitre 4 est un chapitre introductif aux algorithmes de squelettisation. Il décrit brièvement différents schémas d'algorithmes. Les différentes stratégies mises en œuvre par les algorithmes opérant par suppression parallèle de points sont étudiées, à savoir la division d'une itération en plusieurs sous-itérations (directionnelles, sous-mailles) ou l'examen d'un voisinage étendu (conduisant aux algorithmes fortement parallèles). Certains algorithmes existants utilisant de telles méthodes sont analysés dans les annexes A et B.
* La partie II propose de nouveaux algorithmes basés sur la suppression de points P-simples et teste si la topologie est préservée par certains algorithmes de squelettisation (détaillés dans l'annexe B).
  – Le chapitre 5 rappelle la notion remarquable de point P-simple (définie par rapport à un ensemble P). Nous avons la propriété de pouvoir supprimer en une seule fois un ensemble de points P-simples, tout en préservant la topologie. Un algorithme opérant par suppression de points P-simples sera rappelé. Un algorithme, opérant par suppression

de points P-simples, utilise soit une phase préliminaire d'étiquetage (des points à P), soit l'examen d'un voisinage étendu, soit l'utilisation de listes de contours, afin de pouvoir caractériser un point P-simple ; ce qui rend difficile la comparaison avec d'autres algorithmes de squelettisation "classiques" n'exigeant aucune de ces stratégies. Notons qu'un processus opérant par suppression de points P-simples préserve la topologie, par la définition même d'un point P-simple, et ne nécessite alors aucune preuve supplémentaire contrairement aux algorithmes n'utilisant pas cette technique. Il s'agit en quelque sorte, d'une démarche inverse de conception d'algorithmes, par rapport à celle adoptée dans le cadre des algorithmes pour images 3D binaires, déjà proposés.

– Le chapitre 6 constitue une nouvelle contribution en ce qui concerne les algorithmes de squelettisation, à plusieurs titres :
  - Nous avons d'abord introduit un nouvel ensemble P^x défini localement pour tout point x et à partir d'un ensemble P. Cela nous permet de proposer des algorithmes de suppression parallèle de points P^x-simples, ne nécessitant maintenant ni une phase préliminaire d'étiquetage, ni l'examen d'un voisinage étendu, ni les listes de contour, afin de savoir si un point est P^x-simple, et par conséquent s'il peut être éliminé. Désormais, nous pouvons comparer de tels algorithmes avec les algorithmes "classiques" existants.
  - De façon générale, nous considérons un algorithme de référence A. Puis, nous montrons que A préserve la topologie, en prouvant que les points supprimés par A sont P-simples. Ensuite, nous nous efforçons de proposer un algorithme A' (par raffinements successifs de P à l'aide d'une simulation par ordinateur), opérant par suppression de points P^x-simples, tel que A' supprime au moins les points retirés par A. La conséquence en est double : A et A' préservent la topologie, car ils opèrent par suppression de points P^x-simples (on a ainsi une autre preuve par simulation par ordinateur de la validité de A), A' retire plus de points que A, au moins lors de la première sous-itération. En fait, nous avons réussi à proposer un tel algorithme retirant au moins les points supprimés par deux algorithmes de type 6 sous-itérations, puis nous avons appliqué de nouveau cette méthodologie à base de suppression de points P^x-simples et de raffinements successifs de l'ensemble P, à un algorithme de type 12 sous-itérations.
  - Nous montrons ensuite avec les points P-simples que deux algorithmes classiques fortement parallèles basés sur les points simples ne préservent pas la topologie, et qu'un autre (le seul restant) préserve bien la topologie. En fait, nous corrigeons actuellement les deux premiers avec leur auteur à l'aide des points P^x-simples. Une perspective intéressante consisterait à proposer une variante du troisième algorithme, en espérant supprimer au moins les points qu'il supprime, ou en utilisant un plus petit voisinage.

* La partie III présente de nouveaux algorithmes selon une approche topologie discrète, suite à l'étude de l'homotopie dans les ordres, récemment proposée par G. Bertrand.

– Le chapitre 7 introduit les notions de base pour cette approche, notamment celles d'ordre, de point α-unipolaire, de point α_n-simple, d'ensemble n-contractile. Notons qu'un processus de suppression de points est au cœur même des deux dernières notions, qui sont

elles-mêmes récursives et croisées entre elles. Ces notions seront en fait explicitées et simplifiées dans les deux chapitres 8 et 9, dédiés aux images 2D binaires, et 3D binaires.

- Le chapitre 8 présente un algorithme de squelettisation d'images 2D binaires. Il s'agit d'un schéma simple, utilisant un voisinage restreint afin de décider de la suppressibilité d'un point et produisant des squelettes minces et bien centrés. Des comparaisons avec d'autres algorithmes classiques (présentés en annexe A) y sont reportées.
- Le chapitre 9 présente un algorithme de squelettisation d'images 3D binaires. Une nouvelle approche du concept de point terminal de surface, basée sur celle d'arbre de la théorie des graphes, est adoptée.
- Le chapitre 10 étend les résultats obtenus dans le chapitre 8 aux images 2D en niveaux de gris, de la même façon que cela avait été fait par l'approche topologie des coupes dérivée de la topologie digitale. Les notions de point α-destructible et d'ordre valué y sont définies. Nous présentons un algorithme de squelettisation qui consiste en l'abaissement parallèle de points α-destrutibles. Différents problèmes, non encore résolus, sont formulés concernant la valeur d'abaissement qu'un point peut prendre afin de préserver la topologie, s'il est abaissé seul ou en parallèle avec d'autres points.
- Le chapitre 11 décrit une application concernant le filtrage parallèle topologique de squelettes curvilignes 2D ou 3D ; des résultats sont également donnés dans le cas de squelettes surfaciques 3D.

* La partie IV comprend quatre annexes :

- L'annexe A passe en revue différents algorithmes 2D. Cette étude permet de mieux comprendre le déroulement d'un algorithme de squelettisation, et de mieux percevoir les problèmes liés à la suppression en parallèle (un des algorithmes détaillés ici ne préserve pas la topologie). Elle souligne aussi les différences entre les résultats obtenus par les algorithmes de classes différentes. Les différentes étapes successives de l'amélioration d'un algorithme de base, afin d'éviter ou de solutionner différents problèmes, sont également présentées. À travers cette étude, nous verrons enfin qu'un algorithme fortement parallèle peut considérer des directions (et celles-là sont le prix à payer afin d'avoir un squelette mince, mais pas forcément bien centré ; et exiger l'examen d'un voisinage plus petit que ne l'impose un algorithme fortement parallèle – qui n'aurait pas de conditions directionnelles et qui, retirant des points de façon isotrope, produit des squelettes minces et bien centrés). Cet inconvénient s'accroît en 3D ; c'est d'ailleurs ce qui a rendu difficiles la proposition d'algorithmes fortement parallèles efficaces, et les tests de sa validité (préservation de la topologie).
- L'annexe B constitue un recueil de fiches "signalétiques" d'un grand nombre d'algorithmes classiques de squelettisation d'images 3D binaires, algorithmes qui sont "disséqués" au chapitre 6.
- L'annexe C propose une étude détaillée des graphes de décision binaire. Un tel outil algorithmique nous a permis de coder efficacement les configurations correspondant à des points devant être éliminés par un algorithme (voir l'annexe B et le chapitre 9).
- L'annexe D décrit sommairement les isométries utilisées lors de l'implémentation de nos algorithmes ou (chapitres 6 et 9) celle des algorithmes "classiques" (annexe B).

Première partie

Topologie digitale, simplicité et schémas de squelettisation

Chapitre 1

Notions de base de topologie digitale

Dans ce chapitre, nous présentons les notions de base de topologie digitale dans le cas des images binaires bidimensionnelles ou tridimensionnelles. Nous avons préféré expliciter un concept 2D, puis l'expliquer ensuite dans le cas 3D. Nous introduisons les notions d'image, de pixel, de voxel, d'adjacence, de voisin, de connexité, de composante connexe. Puis avec les notions de courbe ouverte simple et de courbe fermée simple, nous illustrons le paradoxe du théorème de Jordan, dans le cas discret. Celui-là nous montre qu'il faut utiliser deux adjacences différentes pour les deux constituants de l'image binaire (à savoir l'objet et son complémentaire). Nous pouvons alors définir plus précisément les notions d'image discrète (ou digitale) (en utilisant ces deux adjacences différentes) et formaliser les notions précédentes (composante connexe pour l'objet ou son complémentaire, chemin, etc.). Nous terminerons par la notion de trou.

1.1 Espace discret de représentation d'une image binaire

1.1.1 Pavage et maillage [CM91]

Partitionnons $\mathcal{R}^2$ en un *pavage* régulier, *i.e.* par des polygones convexes réguliers (côtés égaux et angles égaux) et identiques, de telle façon que leurs sommets ne se rencontrent qu'en d'autres sommets. De tels polygones sont appelés des *tesselles*. Un *maillage* peut être associé à ce pavage de la façon suivante. On associe le centre de gravité P à l'intérieur de chacune des tesselles T_P du pavage. Les points P sont appelés *pixels* (pour *picture elements*). Puis à tout point P est associé l'ensemble des points Q tels que T_P et T_Q ont une arête commune. L'ensemble de tous les segments (P, Q) ainsi définis forme le *maillage* (Fig. 1.1). En fait, pavage et maillage sont des représentations duales du plan.

1.1.2 Image discrète binaire et maillage

Nous travaillons sur des *images discrètes (ou digitales) binaires* (voir [CM91] pour la discrétisation d'une image réelle, voir [Mar87] pour une introduction aux images selon une approche physique). Une telle image peut être vue comme un ensemble fini de tesselles. Nous nous limi-

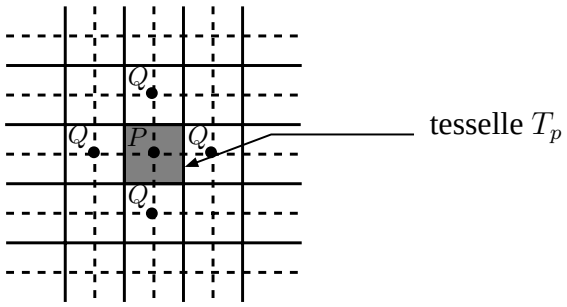


FIG. 1.1 – *Un pavage (en trait continu) et le maillage associé (en trait discontinu).*

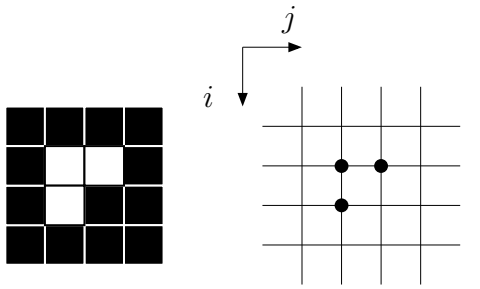
tons au cas pour lequel les tesselles sont des carrés, et utilisons alors le *maillage carré* (ou *maille carrée*). Dans un tel maillage, un pixel est repéré par ses coordonnées (i, j), i indiquant la ligne et j indiquant la colonne à l'intersection desquelles se situe le pixel. Ce maillage peut être stocké dans une structure de données de type matriciel (m_{ij}). Une image binaire peut être vue comme l'union de deux ensembles : l'ensemble constitué par les tesselles blanches (ou *tesselles-objets*) et l'ensemble constitué des tesselles noires (ou *tesselles du complémentaire*, sous-entendu de l'ensemble des tesselles-objets).

Nous pouvons alors associer à chaque tesselle T_{ij} de l'image discrète, un pixel P_{ij} de coordonnées (i, j) dans le maillage. Par convention, la présence d'un disque de couleur noire (ou *point noir*), dont le centre est de coordonnées (i, j) dans le maillage, signifiera que la tesselle T_{ij} correspondante est de couleur blanche (tesselle-objet), et dans ce cas m_{ij} vaudra 1. Par convention toujours, une tesselle T_{ij} de couleur noire (tesselle du complémentaire) sera représentée par un disque de couleur blanche (ou *point blanc*), dont le centre est de coordonnées (i, j) dans le maillage, auquel cas m_{ij} vaudra 0. Afin d'alléger la représentation, une tesselle du fond ne sera pas toujours représentée, de même qu'un point du maillage associé à une telle tesselle (*i.e.* lorsqu'un disque n'est pas dessiné dans le maillage, le pixel correspondant traduira la présence d'une tesselle du complémentaire) (Fig. 1.2).

1.1.3 Cas des images binaires tridimensionnelles

L'extension au cas 3D est directe. Les points P du maillage sont appelés *voxels* (pour *volume elements*). Les tesselles sont des cubes et on utilise alors le *maillage cubique* (ou *maille cubique*) afin de représenter une image binaire tridimensionnelle. Ce maillage sera stocké dans une matrice (m_{ijk}) avec i l'indice de la ligne, j l'indice de la colonne et k l'indice du plan (Fig. 1.3).

Remarque : Par la suite, nous parlerons d'un concept dans $\mathcal{Z}^2$ ou concept 2D (resp. $\mathcal{Z}^3$ ou 3D) lorsque nous nous référerons à un concept dans l'espace discret constitué des points du maillage carré (resp. cubique).

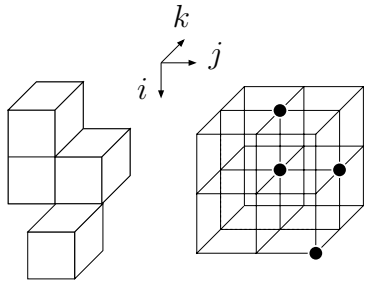


(a) (b)

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

(c)

FIG. 1.2 – (a) *Image discrète,* (b) *sa représentation dans le maillage carré,* (c) *sa matrice de stockage.*



(a) (b)

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

(c)

FIG. 1.3 – (a) *Image discrète,* (b) *sa représentation dans le maillage cubique,* (c) *sa matrice de stockage (plans successifs selon* k*).*

1.2 Distances

On définit les distances suivantes dans ces deux mailles [BM94] (on rappelle qu'une distance est définie positive, réflexive et vérifie l'inégalité triangulaire, voir également [Bor84] [Bor86]) :

– Distances 2D
Soient $x(x_1, x_2)$ et $y(y_1, y_2) \in \mathcal{Z}^2$,

$$d_4(x, y) = \sum_{i=1}^{2} |y_i - x_i| \text{ (*City Block Distance* ou *Square Distance*),}$$

$$d_8(x, y) = \underset{i=1,2}{max} |y_i - x_i| \text{ (*Chessboard Distance* ou *Diamond Distance*).}$$

– Distances 3D
Soient $x(x_1, x_2, x_3)$ et $y(y_1, y_2, y_3) \in \mathcal{Z}^3$,

$$d_6(x, y) = \sum_{i=1}^{3} |y_i - x_i|,$$

$$d_{26}(x, y) = \underset{i=1,2,3}{max} |y_i - x_i|.$$

1.3 Voisinages, voisins et points adjacents

À l'aide des distances précédentes, on définit les voisinages suivants [BM94] (voir Fig. 1.4) :

– Voisinages 2D
Soit $x \in \mathcal{Z}^2$,

$$N_4(x) = \{y \in \mathcal{Z}^2; d_4(x,y) \leq 1\},$$
$$N_8(x) = \{y \in \mathcal{Z}^2; d_8(x,y) \leq 1\}.$$

– Voisinages 3D
Soit $x \in \mathcal{Z}^3$,

$$N_6(x) = \{y \in \mathcal{Z}^3; d_6(x,y) \leq 1\},$$
$$N_{18}(x) = \{y \in \mathcal{Z}^3; d_6(x,y) \leq 2 \text{ et } d_{26}(x,y) \leq 1\},$$
$$N_{26}(x) = \{y \in \mathcal{Z}^3; d_{26}(x,y) \leq 1\}.$$

Définitions (n-voisinage, points n-adjacents) : Notons $N_n^*(x) = N_n(x) \setminus \{x\}$. Les points de $N_n^*(x)$ forment le n-*voisinage* et sont au nombre de n (Fig. 1.4 (a) à (e)). Deux points x et y de $\mathcal{Z}^2$ ou $\mathcal{Z}^3$ sont n-*adjacents* si $y \in N_n^*(x)$ (avec $n = 4$ ou 8 pour $\mathcal{Z}^2$, et $n = 6$, 18 ou 26 pour $\mathcal{Z}^3$).
On définit les types de voisins suivants :

– En 2D
Les points - de $N_4^*(x)$ sont les 4-*voisins de* x
(ou *voisins directs*, au nombre de 4) (Fig. 1.4 (f))
et correspondent à une tesselle partageant une arête avec la tesselle de x
(Fig. 1.4 (g)),
- de $N_8^*(x) \setminus N_4^*(x)$ sont les 8-*voisins de* x
(ou *voisins diagonaux ou indirects*, au nombre de 4) (Fig. 1.4 (h))
et correspondent à une tesselle partageant un sommet uniquement
avec la tesselle de x (Fig. 1.4 (i)).

– En 3D
Les points - de $N_6^*(x)$ sont les 6-*voisins de* x
(ou *voisins directs*, au nombre de 6) (Fig. 1.4 (j))
et correspondent à une tesselle partageant une face avec la tesselle de x
(Fig. 1.4 (k)),
- de $N_{18}^*(x) \setminus N_6^*(x)$ sont les 18-*voisins de* x
(ou *voisins diagonaux*, au nombre de 12) (Fig. 1.4 (l))
et correspondent à une tesselle partageant une arête et sans face commune
avec la tesselle de x (Fig. 1.4 (m)),
- de $N_{26}^*(x) \setminus N_{18}^*(x)$ sont les 26-*voisins de* x
(ou *voisins diamétralement opposés*, au nombre de 8) (Fig. 1.4 (n))
et correspondent à une tesselle partageant un sommet uniquement
avec la tesselle de x (Fig. 1.4 (o)).

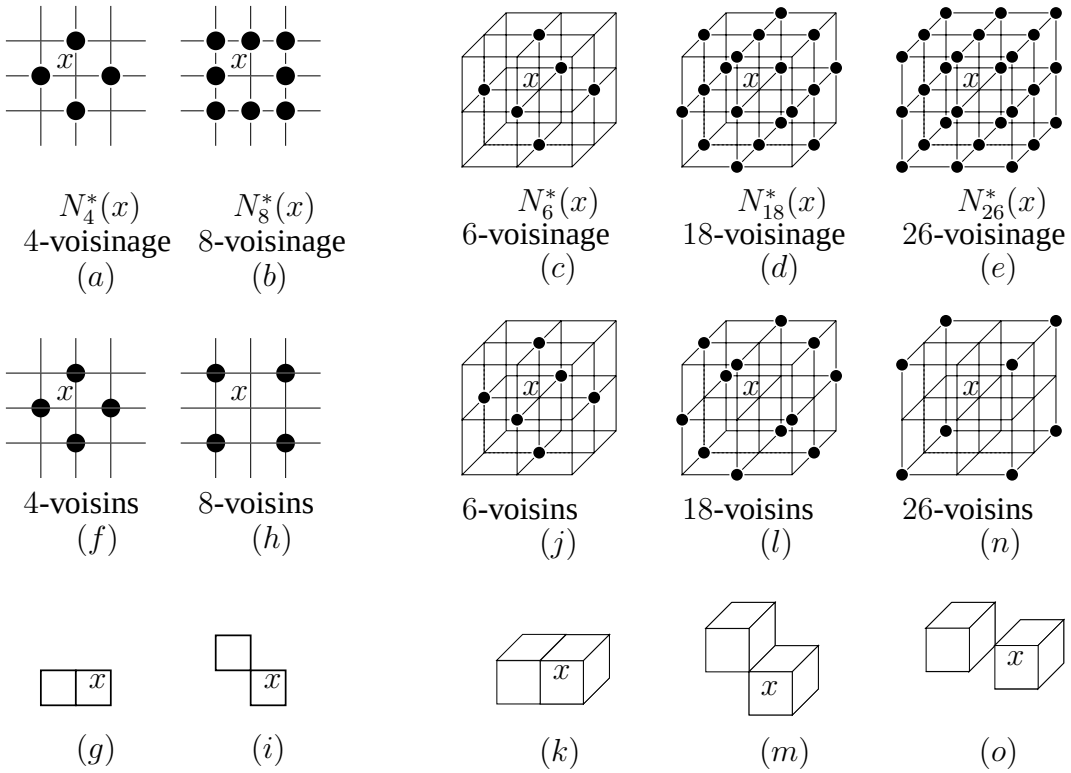


FIG. 1.4 – *Voisinages* 2*D et* 3*D.*

1.4 Théorème de Jordan dans $\mathcal{R}^2$

Dans cette section, nous illustrons le fait que si nous voulons vérifier le théorème de Jordan dans le cas 2D ou 3D (l'analogue discret du théorème de Jordan), alors il faut utiliser deux adjacences différentes pour l'objet et son complémentaire (problème connu sous le nom de *paradoxe du théorème de Jordan dans le cas discret*). Les notions utilisées de courbe et de composante connexe seront formalisées à la section suivante, elles sont utilisées ici de façon intuitive.

Théorème de Jordan dans $\mathcal{R}^2$: [CM91]
Toute courbe fermée simple (*c.-à-d.* ne se recoupant pas elle-même) sépare le plan en deux domaines (ensembles connexes) qui sont le domaine intérieur et le domaine extérieur de la courbe (voir Fig. 1.5).

1.4.1 Illustration du paradoxe de l'analogue discret du théorème de Jordan

Notons (m, n) le couple pour lequel la m-adjacence est utilisée pour l'objet et la n-adjacence pour son complémentaire (*i.e.* deux points noirs sont adjacents s'ils sont m-adjacents et deux points blancs sont adjacents s'ils sont n-adjacents). Intuitivement, une composante noire (resp.

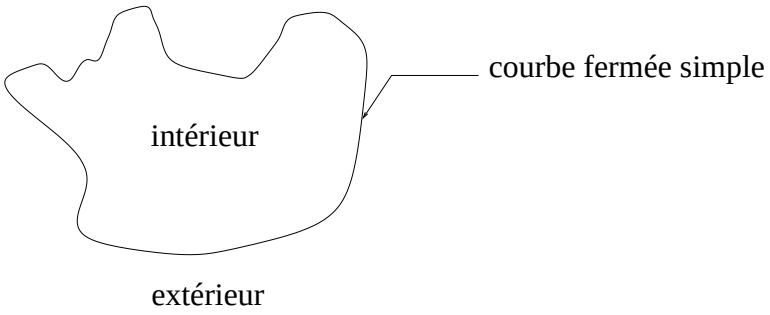


FIG. 1.5 – *Illustration du théorème de Jordan dans* $\mathcal{R}^2$.

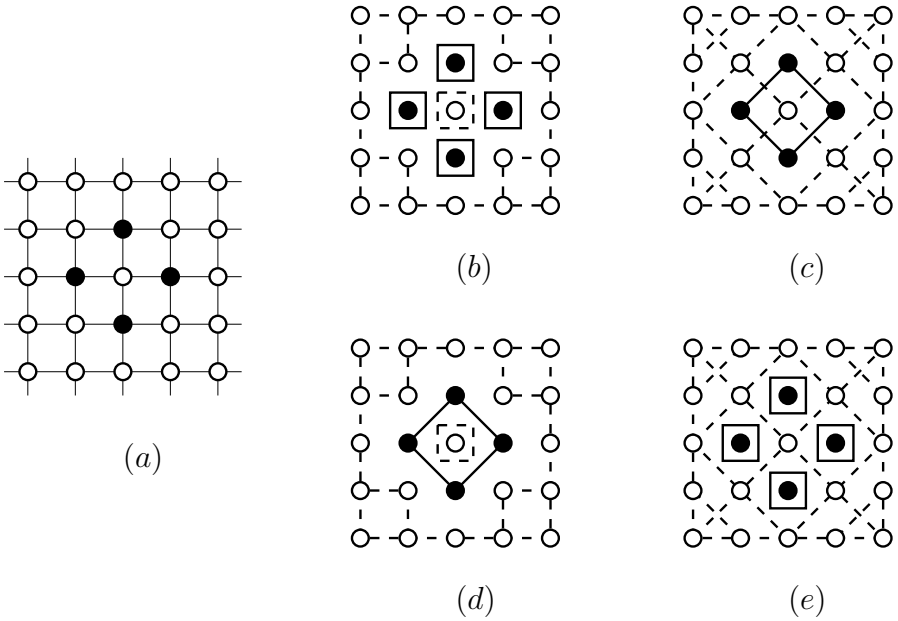


FIG. 1.6 – *Illustration du paradoxe du théorème de Jordan dans* $\mathcal{Z}^2$. (a) *Image,* (b) $(m,n) = (4,4)$, (c) $(m,n) = (8,8)$, (d) $(m,n) = (8,4)$, (e) $(m,n) = (4,8)$. *En trait continu (resp. pointillé, sont représentées les adjacences entre les points noirs (resp. points blancs) ou la délimitation des composantes noires (resp. blanches) formées d'un seul point.*

blanche) sera un ensemble maximal de points noirs m-adjacents (resp. points blancs n-adjacents).

Considérons la figure Fig. 1.6 (a). Si la 4-adjacence est utilisée pour les couples de points de même couleur, alors les points noirs ne sont pas connectés mais ils "séparent" l'ensemble des points blancs en deux composantes (Fig. 1.6 (b)) (le théorème n'est pas vérifié). Si la 8-adjacence est utilisée pour les couples de points de même couleur, alors les points noirs forment l'analogue discret d'une courbe, mais ils ne séparent pas les points blancs (Fig. 1.6 (c)) (le théorème n'est pas vérifié). La difficulté est levée si nous utilisons la 8-adjacence pour les points noirs et la 4-adjacence pour les points blancs (Fig. 1.6 (d)) (dans ce cas, une composante noire sépare deux

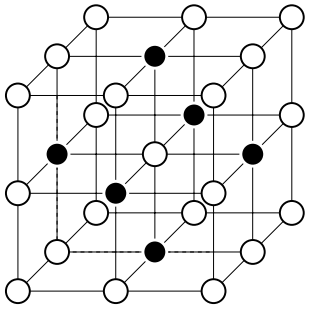


FIG. 1.7 – *Illustration du paradoxe du théorème de Jordan dans* $\mathcal{Z}^3$.

composantes blanches), ou vice versa (Fig. 1.6 (e)) (dans ce cas, nous avons une composante du fond et quatre composantes connexes de l'objet).

Dans $\mathcal{Z}^3$, apparaissent les mêmes problèmes qu'auparavant, à savoir l'existence de deux ensembles de points blancs (le fond et la composante formée du point central) non séparés par un ensemble de points noirs (cas $(6,6)$, voir la figure 1.7), ou l'existence d'une composante noire – intuitivement une surface discrète – ne séparant pas deux composantes blanches (cas $(26,26)$, $(18,18)$, $(18,26)$, $(26,18)$). Pour vérifier l'analogue discret du théorème de Jordan dans $\mathcal{Z}^3$, nous utilisons la 6-adjacence pour les points blancs et la 18- ou la 26-adjacence pour les points noirs, ou vice versa.

Le fait d'utiliser différentes sortes d'adjacence pour les points noirs et les points blancs nous permet maintenant de définir plus précisément les notions d'image discrète, d'adjacence, de chemin, de composante connexe, de courbe.

1.5 Définitions [KR89]

Nous rappelons ici les définitions proposées par Kong et Rosenfeld dans l'article fondamental concernant la topologie digitale [KR89].

1.5.1 Image et points

Une *image discrète (ou digitale)* est un quadruple (V,m,n,B) avec $V=\mathcal{Z}^2$ ou $V=\mathcal{Z}^3$, $B\subseteq V$ et $(m,n)=(4,8)$ ou $(8,4)$ si $V=\mathcal{Z}^2$ et $(m,n)=(6,26)$, $(26,6)$, $(6,18)$ ou $(18,6)$ si $V=\mathcal{Z}^3$. Nous disons que (m,n) est la *connexité choisie pour l'image*. Dans la suite de ce mémoire, pour le cas 3D, nous ne tiendrons compte que des connexités $(26,6)$ et $(6,26)$. L'image discrète $\mathcal{I}=(V,m,n,B)$ est dite *bidimensionnelle* (resp. *tridimensionnelle*) si $V=\mathcal{Z}^2$ (resp. si $V=\mathcal{Z}^3$) ; lorsqu'il n'y a pas ambiguïté, nous écrirons une (m,n)-image. Les éléments de V sont appelés les *points* de l'image discrète. Les points de B sont appelés les *points noirs* de l'image. Les points de $V\setminus B$ sont appelés les *points blancs* de l'image.

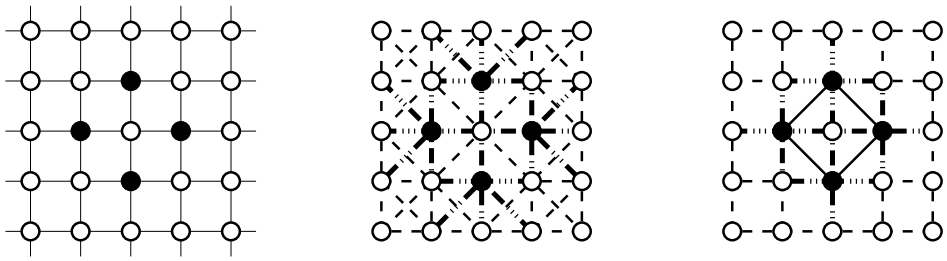


FIG. 1.8 – *Différentes relations d'adjacence.* (a) *Image,* (b) $(m,n)=(4,8)$, (c) $(m,n)=(8,4)$. *Une adjacence entre deux points noirs (resp. blancs, entre un point blanc et un point noir) est représentée en trait continu (resp. pointillé, mixte).*

1.5.2 Adjacence

Deux points noirs sont *adjacents* s'ils sont m-adjacents (nous dirons que nous utilisons la m*-connexité* pour l'objet), deux points blancs ou un point blanc et un point noir sont *adjacents* s'ils sont n-adjacents (nous dirons que nous utilisons la n*-connexité* pour le complémentaire ; ou entre l'objet et le complémentaire) (voir Fig. 1.8). Le fait d'utiliser la n-adjacence entre un point blanc et un point noir est exploité afin de caractériser les points de bord (voir section 1.5.5). Un point p est *adjacent à un ensemble* de points S si p est adjacent à au moins un point de S. Un ensemble de points S est *adjacent à un ensemble* de points T si au moins un point de S est adjacent à un point de T.

1.5.3 Connexité

Un ensemble S de points noirs et/ou de points blancs est *connexe* si S ne peut être partitionné en deux sous-ensembles non adjacents entre eux. Une *composante connexe* (abréviée parfois en CC) d'un ensemble S de points noirs et/ou de points blancs est un sous-ensemble connexe non vide de S qui n'est pas adjacent à un autre point de S. Une composante de l'ensemble des points noirs de l'image est appelée *composante noire* et une composante de l'ensemble des points blancs est appelée *composante blanche*. Dans une image discrète finie (*c.-à-d.* lorsque B est un ensemble fini), il existe une unique composante blanche infinie (délimitée par le cadre de l'image), elle est appellée *fond* (ou *background*). Les composantes blanches finies sont appelées les *cavités*.

1.5.4 Chemin

Pour tout ensemble S de points, un *chemin dans* S est une suite $< p_i; 0 \leq i \leq n >$ de points de S telle que p_i est adjacent à p_{i+1} pour tout $0 \leq i < n$. Un chemin $< p_i; 0 \leq i \leq n >$ est dit être un *chemin de* p_0 *à* p_n, et il est dit *fermé* si $p_n = p_0$. Le chemin dégénéré $< p_0 >$ constitué d'un seul point est un cas particulier de chemin fermé.

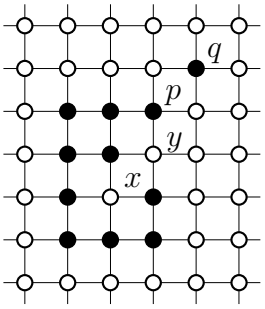


FIG. 1.9 – *Calculs de nombres de composantes connexes. Si* $(m, n) = (4, 8)$*, alors* B *est formé de* 2 *CC et* $V \setminus B$ *de* 1 *CC. Si* $(m, n) = (8, 4)$*, alors* B *est formé de* 1 *CC et* $V \setminus B$ *de* 2 *CC.*

Une autre façon de définir les composantes connexes est d'utiliser la notion de chemin. Nous disons que p et q sont *connectés dans* S si et seulement si il existe un chemin inclus dans S et reliant p à q. La relation "être connecté dans S" est une relation d'équivalence (*i.e.* réflexive, symétrique et transitive). Les composantes connexes de l'image sont les classes d'équivalence de cette relation.

Considérons la figure 1.9. Si $(m, n) = (4, 8)$, B est composé de deux 4-composantes : une 4-composante formée du point q, l'autre 4-composante formée de tous les autres points noirs (p et q ne sont pas 4-adjacents) ; et $V \setminus B$ est composé d'une 8-composante (le fond) (x et y sont 8-adjacents). Si $(m, n) = (8, 4)$, B est composé d'une 8-composante (p et q sont 8-adjacents) et $V \setminus B$ est composé de deux 4-composantes : une 4-composante formée du point x, l'autre 4-composante formée de tous les autres points blancs (x et y ne sont pas 4-adjacents).

1.5.5 Points particuliers et courbes

Un point noir est dit *isolé* s'il n'est adjacent à aucun autre point noir (au sens de m-adjacent). Un point noir est *un point de bord* s'il est adjacent à un ou plusieurs points blancs (au sens de n-adjacent), dans le cas contraire, il est appelé *point intérieur*.

Une *courbe noire fermée simple* est un ensemble connexe de points noirs tel que chacun de ses points est adjacent à exactement deux autres points de cet ensemble. Un *arc noir simple* est un ensemble connexe de points noirs tel que chacun de ses points est adjacent à seulement deux autres points de cet ensemble à l'exception de deux points – *les points extrémités d'arc* – qui sont adjacents à seulement un autre point de l'ensemble. Le terme de *courbe ouverte simple* sera employé à la place de celui d'arc, les points extrémités d'arc seront appelés *points extrémités de courbe*.

Considérons la figure 1.10 avec $(m, n) = (8, 4)$. La figure 1.10 (a) représente une courbe ouverte non simple (p_2 et p_4 ont 3 points qui leur sont 8-adjacents). La figure (b) (obtenue à partir de la figure 1.10 (a), en retirant le point p_3) représente une courbe ouverte simple (p_2 et p_4 ont maintenant 2 points qui leur sont 8-adjacents).

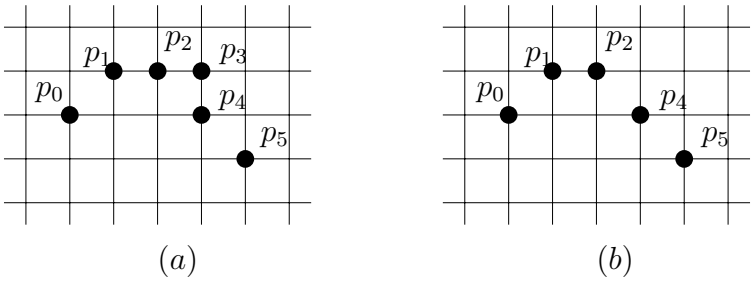


FIG. 1.10 – *Courbe ouverte* (a) *non simple* (b) *simple.*

1.5.6 Théorème de Jordan dans $\mathcal{Z}^2$

Nous formulons maintenant le théorème de Jordan dans $\mathcal{Z}^2$ de la manière suivante.

Théorème de Jordan dans $\mathcal{Z}^2$: [CM91]
Le complémentaire de toute courbe fermée simple 4-connexe (resp. 8-connexe) est formé de deux composantes 8-connexes (resp. 4-connexes).

Remarque : Se reporter à [KMW91] [Mal97] [ML00] pour une étude du théorème de Jordan dans $\mathcal{Z}^3$ (également à [KR85] d'après [KMW91]).

1.6 Notion de trou

La notion de trou n'est pas simple à définir. Par exemple, un tore plein a un trou, une balle creuse a une cavité mais pas de trou, un tore creux comporte une cavité et deux trous.

Considérons une image binaire discrète (V, m, n, B). Un trou dans B est détecté lorsqu'il existe un chemin fermé dans B qui ne peut être déformé dans B en un seul point [BM94] [Ber94]. Dans $\mathcal{Z}^3$, une déformation est une suite de déformations élémentaires telle qu'un chemin fermé Γ' est une déformation élémentaire d'un chemin fermé Γ si Γ et Γ' sont les mêmes, excepté dans un cube unité (voir [KR89] [Ber92]). Si la 6-connexité est utilisée, le cube unité doit être tel qu'il n'existe pas deux points diamétralement opposés et appartenant à $V \setminus B$ (voir Fig. 1.11, si cette restriction n'était pas utilisée, le chemin $abcPa$ pourrait être déformé en $adefgbcPa$ et traverserait ainsi le 26-chemin fermé $hQih$, il n'y aurait alors plus d'équivalence entre les trous de B et ceux de $V \setminus B$; pour plus de détails, voir [Ber92], ainsi que la discussion sur le nombre de trous de l'objet et de son complémentaire, au chapitre 3). Si la 26-connexité est utilisée, il n'y a pas de restriction sur le cube unité.

Définition (objet simplement connexe) :
Un objet connexe et sans trou est dit *simplement connexe*.

Remarque : Dans le cas 2D, une composante finie du complémentaire peut être "entourée" par un chemin fermé non déformable en un seul point, un trou est alors détecté pour chaque

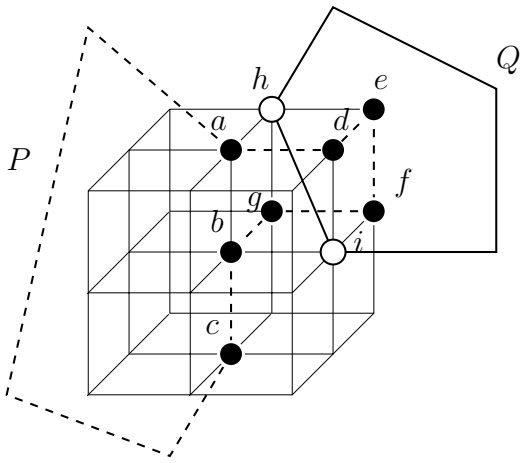


FIG. 1.11 – *Restriction sur le cube unité lors de l'utilisation de la* 6*-connexité pour la détection d'un trou.*

composante finie du complémentaire. Nous pouvons alors parler de trous pour les composantes finies du complémentaire en 2D (ces deux entités sont en même nombre), alors qu'elles sont appelées cavités en 3D (en 3D, le nombre de trous n'est pas forcément égal au nombre de cavités). Parfois, en 3D, le terme de *tunnel* est employé à la place de celui de trou [KR89] [Ma94].

Chapitre 2

Genre 2D et simplicité

Dans le rapport technique [Mor80], nous trouvons différentes formules pour calculer le genre 2D ou 3D d'un ensemble ou de son complémentaire, selon la connexité de l'image. Dans un premier temps, nous détaillons l'obtention de ces formules dans le cas 2D. Nous insistons sur le fait que le calcul du genre d'une image (le genre étant une propriété globale) peut s'effectuer par sommations d'entités calculées localement sur l'image. L'extension 3D peut être obtenue de la même façon [Mor80] [PR71], seules les formules et propositions principales seront données au chapitre 3.

Ensuite, nous introduisons la notion de point simple. Un point est dit *simple* si sa suppression préserve la topologie de l'objet, nous préciserons ultérieurement ce que cela signifie. Retenons dès à présent que la suppression de point simple est le processus mis en œuvre dans un algorithme de squelettisation (voir le chapitre 4 et les annexes A et B). Nous donnerons quelques propriétés des points simples ainsi que quelques caractérisations, notamment celle utilisant les nombres topologiques. Au chapitre suivant, seront données quelques propriétés et caractérisations de point simple en 3D utilisant également le genre ou les nombres topologiques.

Cette section se termine par la présentation des notions d'ensemble simple, d'ensemble minimal non simple (ou MNS), mises en œuvre généralement pour montrer qu'un algorithme opérant par suppression parallèle de points, préserve la topologie.

2.1 Définition du genre [Mor80]

Nous détaillons ici l'étude complète de l'obtention de la formule permettant de calculer le genre d'une image 2D (propriété globale de l'image), par calculs locaux, sommés sur toute l'image.

L'*équation d'Euler-Schaefli* :

$$\sum_{i=0}^{j}(-1)^i N_i = 1 + (-1)^j \tag{2.1}$$

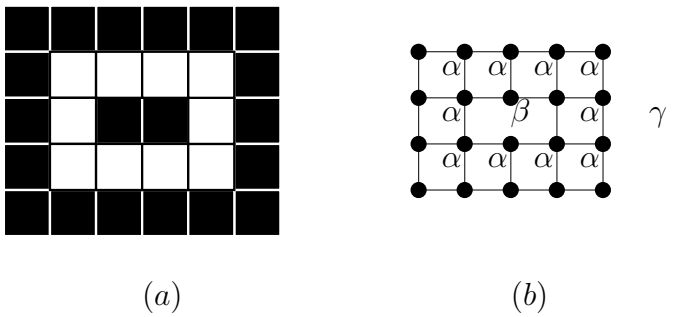


FIG. 2.1 – *Calculs du genre.* (a) *Une seule composante* S *de* B, (b) *graphe planaire associé (*α *: faces de* S, β *: faces des composantes de* $V \setminus S$ *entourées par* S, *et* γ *: (l'unique) face du background).*

décrit la relation entre les nombres d'éléments de dimension différente dans une structure multi-dimensionnelle simplement connexe, confinée à un espace euclidien. Ici, N_i est le nombre d'éléments de dimension i, et j est la dimension de la structure. En 2D ($j = 2$), cette équation (2.1) permet de retrouver l'*invariant d'Euler* pour un graphe planaire :

$$V - E + F = 2 \tag{2.2}$$

avec $V = N_0$ le nombre de sommets, $E = N_1$ le nombre d'arêtes, et $F = N_2$ le nombre de faces.

Soit une image discrète binaire bidimensionnelle $(\mathcal{Z}^2, m, n, B)$. Considérons d'abord le genre d'une seule composante 8-connexe S, avec $S \subseteq B$. Nous lui associons un graphe de la façon suivante : les sommets sont les coins des tesselles de S et les arêtes sont celles de ces tesselles. On obtient ainsi un graphe planaire et simplement connexe (hypothèse utilisée pour l'obtention de l'équation (2.1)). Sont distingués les trois types de faces suivants : les faces provenant des tesselles de S (notées α), les faces provenant des composantes de $\mathcal{Z}^2 \setminus S$ entourées par S (notées β - faces qui sont des polygones à n côtés, avec $n \geq 4$), la face du background (composante infinie de $\mathcal{Z}^2 \setminus B$) (notée γ) (voir l'exemple de la figure 2.1).

Avec l'équation (2.2), nous obtenons (en notant $\#x$, le nombre de faces de type x) :

$$V - E + F = V - E + (\#\alpha + \#\beta + \#\gamma) = 2 \tag{2.3}$$

Pour une composante connexe S, on a ($\#\gamma = 1$) :

$$1 - \#\beta = V - E + \#\alpha \tag{2.4}$$

Le nombre $\#\beta$ est le nombre de faces résultant des composantes de $\mathcal{Z}^2 \setminus S$ entourées par S, *i.e.* les cavités de S (de façon équivalente les trous de S, en 2D). On peut voir que le nombre de trous dans B est égal à la somme des nombres de trous dans chaque composante S de B. En sommant l'équation (2.4) pour toutes les composantes de B (avec $O(B)$ le nombre d'objets de B et $H(B)$ le nombre de trous de B), on obtient :

$$O(B) - H(B) = V - E + \#\alpha \tag{2.5}$$

Définition (genre 2D) : Le *genre* $G_m(B)$ d'un ensemble B est le nombre d'objets (composantes m-connexes de B) ($O(B)$), moins le nombre de trous de B ($H(B)$), (m, n) étant la connexité de l'image.

$$G_m(B) = O(B) - H(B) \overset{\text{(équ. (2.5))}}{=} V - E + \#\alpha \tag{2.6}$$

En calculant la quantité $V - E + \#\alpha$ sur l'image (somme de calculs locaux), il est alors possible de calculer le genre de B : $G_m(B) = O(B) - H(B)$. Puis, avec un algorithme de parcours de composante connexe [Sed91], on peut calculer $O(B)$; on peut alors déterminer le nombre de trous de B : $H(B)$ ($= G_m(B) - O(B)$). Avec l'équation (2.6), il est alors possible de caractériser un point tel que les nombres $O(B)$ et $H(B)$ ne changent pas avant et après la suppression d'un tel point (voir les propositions 4 et 5, page 28).

2.1.1 Genre du complémentaire

Rappelons que le genre $G(B)$ d'un ensemble B est le nombre d'objets (composantes de B) (noté $O(B)$), moins le nombre de trous de B (noté $H(B)$) (équation (2.6)). En remarquant que tout objet de B est un trou de $\mathcal{Z}^2 \setminus B$ (en 2D) (*i.e.* $O(B) = H(\mathcal{Z}^2 \setminus B)$) et que tout objet de $\mathcal{Z}^2 \setminus B$ excepté le fond est un trou de B (*i.e.* $O(\mathcal{Z}^2 \setminus B) - 1 = H(B)$), on obtient alors :

$$G(B) + G(\mathcal{Z}^2 \setminus B) = 1 \tag{2.7}$$

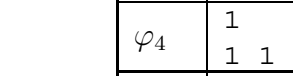
L'équation (2.7) nous permet de déterminer le genre du complément d'un objet, à partir du genre de l'objet.

2.2 Calcul du genre 2D de façon locale

À présent, nous voulons calculer ces entités directement sur l'image, sans passer par la représentation dans un graphe planaire. Il nous faut alors décomposer les objets de l'image en motifs de base et déterminer la participation de chaque motif aux calculs de V, E, et $\#\alpha$ de l'équation (2.6).

Le tableau 2.1 (resp.2.2) détaille la participation de chaque motif de base pouvant apparaître dans un objet 8-connexe (resp. 4-connexe) (un 1 symbolise la présence d'une tesselle), lors du calcul de V, E et $\#\alpha$ (*i.e.* on regarde la participation de chaque sommet, arête, face, à ces quantités, lorsqu'elles sont sommées sur l'image).

Détails concernant le tableau 2.1 : La contribution d'un motif φ_1 est de quatre sommets, quatre arêtes et d'une face, soit $4V + 4E + 1\#\alpha$. La contribution d'un motif φ_2 est de $6V + 7E + 2\#\alpha$, or les points de ces motifs ont été considérés lors du calcul de la contribution de φ_1 (comptée deux fois – car deux points). La nouvelle contribution (par rapport aux calculs sur φ_1) est alors de $6V + 7E + 2\#\alpha - 2 \times (4V + 4E + 1\#\alpha) = -2V - E$. La contribution des autres

nom	motif possible	contribution du motif	nouvelle contribution
φ_1		$4V + 4E + 1\#\alpha$	$4V + 4E + 1\#\alpha$
φ_2		$6V + 7E + 2\#\alpha$	$-2V - E$
φ_3		$7V + 8E + 2\#\alpha$	$-V$
φ_4		$8V + 10E + 3\#\alpha$	V
φ_5		$9V + 12E + 4\#\alpha$	$-V$

TAB. 2.1 – *Contribution des motifs de base,* $(m, n) = (8, 4)$.

motifs s'obtient de la même façon, sachant que pour φ_4 on a déjà calculé 3 fois φ_1, 2 fois φ_2 et 1 fois φ_3 ; et que pour φ_5 on a déjà calculé 4 fois φ_1, 4 fois φ_2, 2 fois φ_3 et 4 fois φ_4.

En effectuant le bilan de la participation de chaque motif par rapport aux nombres de sommets, d'arêtes et de faces, on obtient (avec $\#x$, le nombre de fois qu'apparaît le motif x) :

$$V = 4\#\varphi_1 - 2\#\varphi_2 - \#\varphi_3 + \#\varphi_4 - \#\varphi_5 \tag{2.8}$$

$$E = 4\#\varphi_1 - \#\varphi_2 \tag{2.9}$$

$$\#\alpha = \#\varphi_1 \tag{2.10}$$

Nous avons alors : $V - E + \#\alpha = \#\varphi_1 - \#(\varphi_2 + \varphi_3) + \#\varphi_4 - \#\varphi_5$

Finalement, nous obtenons :

$$G_8(B) \overset{\text{(équ. (2.6))}}{=} \#\varphi_1 - \#(\varphi_2 + \varphi_3) + \#\varphi_4 - \#\varphi_5 \tag{2.11}$$

Détails concernant le tableau 2.2 : Dans le cas de la connexité $(m, n) = (4, 8)$, il ne faut pas tenir compte des éléments : ou en tant que tels, car les deux points d'un de ces motifs ne sont pas 4-adjacents, un tel motif est considéré comme l'union disjointe de deux motifs ψ_1. La contribution d'un motif ψ_1 est de quatre sommets, quatre arêtes et d'une face, soit $4V+4E+1\#\alpha$. La contribution d'un motif ψ_2 est de $6V+7E+2\#\alpha$, or les points de ces motifs ont été considérés lors du calcul de la contribution de ψ_1 (comptée deux fois – car deux points). La nouvelle contribution est alors de $6V + 7E + 2\#\alpha - 2 \times (4V + 4E + 1\#\alpha) = -2V - E$. La contribution des autres motifs s'obtient de la même façon sachant que pour ψ_3 on a déjà calculé 3 fois ψ_1 et 2 fois ψ_2 ; et que pour ψ_4 on a déjà calculé 4 fois ψ_1, 4 fois ψ_2 et 4 fois $\psi 3$.

En effectuant le bilan de la participation de chaque motif par rapport aux nombres de sommets, d'arêtes et de faces, on obtient :

$$V = 4\#\psi_1 - 2\#\psi_2 + \#\psi_4 \tag{2.12}$$

nom	motif possible	contribution du motif	nouvelle contribution
ψ_1	1	$4V + 4E + 1\#\alpha$	$4V + 4E + 1\#\alpha$
ψ_2	1 1 ou 1 / 1	$6V + 7E + 2\#\alpha$	$-2V - E$
ψ_3	1 / 1 1 ou 1 1 / 1 1 ou 1 1 / 1 ou 1 1 / 1	$8V + 10E + 3\#\alpha$	0
ψ_4	1 1 / 1 1	$9V + 12E + 4\#\alpha$	V

TAB. 2.2 – *Contribution des motifs de base,* $(m, n) = (4, 8)$.

$$E = 4\#\psi_1 - \#\psi_2 \tag{2.13}$$

$$\#\alpha = \#\psi_1 \tag{2.14}$$

Nous avons alors : $V - E + \#\alpha = \#\psi_1 - \#\psi_2 + \#\psi_4$
Finalement, nous obtenons :

$$G_4(B) \overset{\text{(équ. (2.6))}}{=} \#\psi_1 - \#\psi_2 + \#\psi_4 \tag{2.15}$$

2.3 Exemples de calcul de genre 2D

2.3.1 Objet simplement connexe

(a) (b)

FIG. 2.2 – *Calcul du genre.* (a) *Objet sans trou,* (b) *graphe planaire associé.*

Considérons la figure 2.2. L'objet B (Fig. 2.2 (a)) est composé d'une seule composante connexe ($O(B) = 1$). Cet objet ne comporte pas de trou ($H(B) = 0$). On a alors $G_4(B) = G_8(B) = 1$ (équ. 2.6). Le graphe planaire associé (Fig. 2.2 (b)) comporte 8 sommets, 10 arêtes, et 4 faces (3 faces correspondant à l'objet, une pour le fond) ($V = 8, E = 10, F = 4$). Pour le cas $(m, n) = (8, 4)$, on a $\#\varphi_1 = 3, \#\varphi_2 = 2, \#\varphi_3 = \#\varphi_4 = \#\varphi_5 = 0$. Pour le cas $(m, n) = (4, 8)$, on a $\#\psi_1 = 3, \#\psi_2 = 2, \#\psi_4 = 0$. En appliquant les équations (2.8), (2.12), (2.9), (2.13), (2.11), (2.15), on retrouve $V = 8,\ E = 10,\ G_4(B) = G_8(B) = 1$.

2.3.2 Objet avec trou

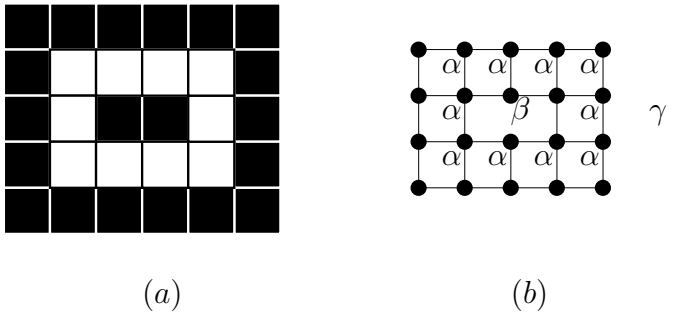


FIG. 2.3 – *Calcul du genre.* (a) *Objet avec trou,* (b) *graphe planaire associé.*

Considérons la figure 2.3. L'objet B (Fig. 2.3 (a)) est composé d'une seule composante connexe ($O(B) = 1$). Cet objet comporte un trou ($H(B) = 1$). On a alors $G_4(B) = G_8(B) = 0$ (équ. 2.6). Le graphe planaire associé (Fig. 2.3 (b)) comporte 20 sommets, 30 arêtes et 12 faces (10 faces correspondant à l'objet, 2 faces pour le complémentaire) ($V = 20, E = 30, F = 12$). Pour le cas $(m, n) = (8, 4)$, on a $\#\varphi_1 = \#\varphi_2 = 10, \#\varphi_3 = \#\varphi_4 = 4, \#\varphi_5 = 0$. Pour le cas $(m, n) = (4, 8)$, on a $\#\psi_1 = \#\psi_2 = 10, \#\psi_4 = 0$. En appliquant les équations (2.8), (2.12), (2.9), (2.13), (2.11), (2.15), on retrouve $V = 20,\ E = 30,\ G_4(B) = G_8(B) = 0$.

2.3.3 Autre exemple

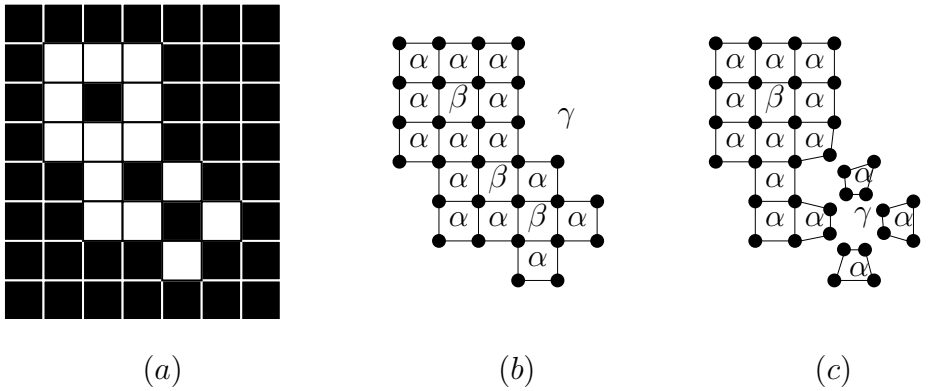


FIG. 2.4 – *Calcul du genre.* (a) *Autre objet avec trou(s). Graphe planaire associé* (b) $(m, n) = (8, 4)$, (c) $(m, n) = (4, 8)$ *(non simplement connexe).*

Considérons la figure 2.4. L'objet B (Fig. 2.4 (a)) est composé d'une seule composante connexe et de trois trous pour $(m, n) = (8, 4)$, ou de quatre composantes connexes et d'un trou pour $(m, n) = (4, 8)$. On a donc $G_8(B) = -2$ et $G_4(B) = 3$ (équ. 2.6). Le graphe planaire

associé pour $(m,n) = (8,4)$ (Fig. 2.4 (b)) comporte 29 sommets, 45 arêtes et 18 faces (14 faces correspondant à l'objet, 4 faces pour le complémentaire) ($V = 29, E = 45, F = 18$). Le graphe planaire associé pour $(m,n) = (4,8)$ (Fig. 2.4 (c)) comporte 34 sommets, 45 arêtes et 16 faces (14 faces correspondant à l'objet, 2 faces pour le complémentaire) ($V = 34, E = 45, F = 16$, notons que l'équation (2.2) n'a pas à être vérifiée car le graphe ici n'est pas connexe). De plus, le graphe est construit de façon à éviter les structures composées uniquement de 2 points en diagonale (cas de la 4-connexité) (voir la partie **Détails concernant le tableau 2.2**, page 24). Pour le cas $(m,n) = (8,4)$, on a $\#\varphi_1 = 14, \#\varphi_2 = 11, \#\varphi_3 = 12, \#\varphi_4 = 7, \#\varphi_5 = 0$. Pour le cas $(m,n) = (4,8)$, on a $\#\psi_1 = 14, \#\psi_2 = 11, \#\psi_4 = 0$. En appliquant les équations (2.8), (2.12), (2.9), (2.13), (2.11), (2.15), on retrouve pour le cas $(8,4)$: $V = 29$, $E = 45$ et $G_8(B) = -2$; et pour le cas $(4,8)$: $V = 34$, $E = 45$ et $G_4(B) = 3$.

2.4 Simplicité d'un point

Dans la suite, nous considérons un sous-ensemble S de $\mathcal{Z}^2$. Nous définissons d'abord ce qu'est un point simple, puis donnons quelques propriétés concernant sa caractérisation.

Définition (point simple en 2D) [Mor81] **:**
En 2D, le point $p \in S$ est *simple* si :

$$O([S \cap N_8^*(p)] \cup \{p\}) = O(S \cap N_8^*(p)), \tag{2.16}$$

$$O([\overline{S} \cap N_8^*(p)] \cup \{p\}) = O(\overline{S} \cap N_8^*(p)), \tag{2.17}$$

avec $O(V(p))$ étant le nombre de composantes connexes de l'ensemble $V(p)$.

En d'autres termes, p est simple si les nombres de composantes connexes de l'objet et de son complémentaire, dans $N_8^*(p)$, restent inchangés avant et après suppression de p. C'est dans ce sens que l'on dira d'un point qu'il est simple lorsque sa suppression d'un objet *préserve la topologie*.

Les relations entre les nombres de composantes connexes des objets et de leurs complémentaires dans la définition de la simplicité établissent un lien entre la simplicité et le genre ; nous mettrons en évidence ce lien dans le cas 3D, au chapitre suivant.

Proposition 1 **:** Pour $p \in S$: $O([S \cap N_8^*(p)] \cup \{p\}) = O(S \cap N_8^*(p))$ ssi $S \cap N_8^*(p)$ a exactement une composante connexe adjacente à p (au sens de S).

Proposition 2 **:** Pour $p \in S$: $O([\overline{S} \cap N_8^*(p)] \cup \{p\}) = O(\overline{S} \cap N_8^*(p))$ ssi $\overline{S} \cap N_8^*(p)$ a exactement une composante connexe adjacente à p (au sens de $\overline{S}$).

Proposition 3 **:** Pour $p \in S$: si $O([S \cap N_8^*(p)] \cup \{p\}) = O(S \cap N_8^*(p))$ et $O([\overline{S} \cap N_8^*(p)] \cup \{p\}) = O(\overline{S} \cap N_8^*(p))$ alors $O(S) = O(S \setminus \{p\})$ et $O(\overline{S}) = O(\overline{S} \cup \{p\})$.

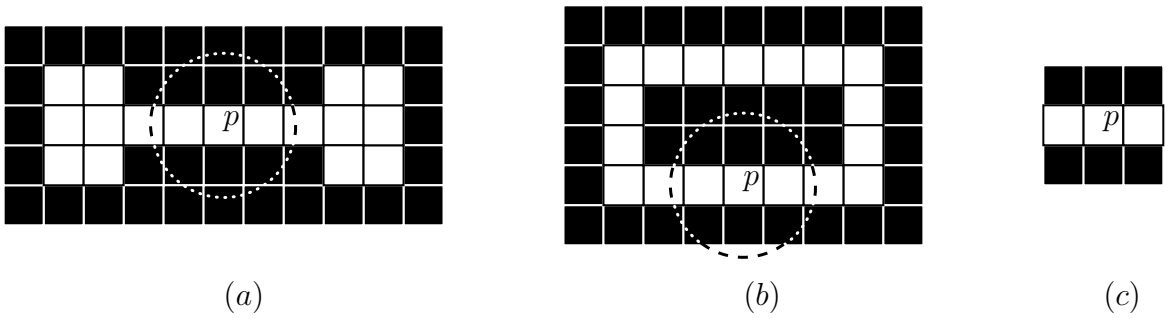


FIG. 2.5 – *Configuration de point non simple pour laquelle la raison de sa non-simplicité ne peut être décidée localement. Si l'on supprime le point p, (a) l'objet S est déconnecté en deux, (b) un trou de l'objet S est supprimé et deux composantes connexes de $V \setminus S$ sont fusionnées, (c) configuration commune de non-simplicité du point.*

La réciproque est vraie, voir la proposition 5. Nous verrons que la proposition 3 est vraie également lorsqu'elle est transposée au cas 3D, tandis que la réciproque est fausse (voir la section 3.5).

Nous avons alors les propositions suivantes :

Proposition 4 **:** Si p est simple, alors $O(S) = O(S \setminus \{p\})$, $O(\overline{S}) = O(\overline{S} \cup \{p\})$ (et $H(S) = H(S \setminus \{p\})$, $H(\overline{S}) = H(\overline{S} \cup \{p\})$, car $O(S) = H(\overline{S})$ et $O(\overline{S}) - 1 = H(S)$, voir section 2.1.1).

Proposition 5 **:** Pour un sous-ensemble S d'une image binaire 2D et un point p de S, si $O(S) = O(S \setminus \{p\})$ et $O(\overline{S}) = O(\overline{S} \cup \{p\})$, alors p est simple.

Comme nous l'avons indiqué auparavant, la proposition 5 du cas 2D est fausse si elle est transposée au cas 3D (voir la section 3.5).

Note : Soulignons l'équivalence entre la simplicité d'un point p appartenant à S et l'égalité des nombres de composantes de l'objet S et du complémentaire $\overline{S}$ avant et après suppression du point, soit dans $N_8^*(p)$ (définition) soit dans $\mathcal{Z}^2$ (propositions 4 et 5). Des exemples de calculs de ces quantités seront donnés ultérieurement (cf. Fig. 2.8, tableau 2.3, dans la section 2.5.3).

Remarque : Insistons sur le fait de pouvoir vérifier si un point p est simple ou non, et cela de façon locale (*i.e.* dans $N_8(p)$). Dans le cas où il n'est pas simple, la raison de sa non-simplicité ne peut pas toujours être décidée localement (ce qui n'est pas le cas si le point est un point intérieur ou isolé). Par exemple, à la figure 2.5 (a), la suppression du point p déconnecte l'objet initial en deux composantes ; à la figure 2.5 (b), la suppression du point p détruit un trou de l'objet initial (ou fusionne deux composantes du complémentaire, en 2D). De façon locale, nous avons la même configuration (Fig. 2.5 (c), voir également la figure 2.8 (c) et le tableau 2.3) ; en d'autres

termes, nous pouvons vérifier que le point n'est pas simple en examinant la configuration de la figure 2.5 (c) ($O([S \cap N_8^*(p)] \cup \{p\}) = 1$ et $O(S \cap N_8^*(p)) = 2$ – l'équation (2.16) n'est pas vérifiée – ou $O([\overline{S} \cap N_8^*(p)] \cup \{p\}) = 1$ et $O(\overline{S} \cap N_8^*(p)) = 2$ – l'équation (2.17) n'est pas vérifiée), sans savoir (par un examen local) si l'objet est déconnecté ou si un trou est détruit après la suppression du point p.

2.5 Caractérisations en pratique d'un point simple

Dans cette section, sont proposées différentes caractérisations de points simples en 2D (la deuxième sera d'ailleurs utilisée pour caractériser un point α-simple en 2D, au chapitre 8). Nous proposons notamment la caractérisation d'un point simple par l'utilisation de nombres topologiques. En fait, c'est par une adaptation de ces nombres que l'on pourra caractériser les points simples dans le cas 3D (une caractérisation des points simples en 3D semble impossible par une adaptation des deux premières caractérisations). Cette section se termine par le rappel de deux méthodes de stockage de configurations : la table de consultation et le diagramme de décision binaire.

2.5.1 Nombre de Hilditch

Le *nombre de Hilditch* $H(p)$ ([Hil69], d'après [KR89]), est défini par le nombre de passages d'un point blanc à un point noir lorsque les 8-voisins de p sont visités par ordre cyclique, en commençant par un 4-voisin de p, en y retournant et en sautant un coin entre deux 4-voisins noirs de p, 8-adjacents. Dans une $(8,4)$-image, $H(p) = 1 \Leftrightarrow p$ est 8-simple (Fig. 2.6 (e), (f) et (h)).

2.5.2 Nombre de Rutovitz

Le *nombre de Rutovitz* $R(p)$ ([Rut66], d'après [KR89]) est défini de la même façon que $H(p)$ sauf que le passage d'un point noir à un point blanc est également comptabilisé et que les coins ne sont pas sautés. Le nombre de Rutovitz dans une $(4,8)$-image est égale à deux fois le nombre de composantes 4-connexes noires dans $N_8^*(p)$, sauf si tous les points de $N_8^*(p)$ sont des points noirs. Dans ce cas, $R(p)$ vaut zéro (Fig. 2.6 (b)). Dans une $(4,8)$-image, si le nombre de Rutovitz d'un point p 4-adjacent à X vaut 2, alors p est 4-simple (Fig. 2.6 (e)) (p doit être 4-adjacent à X, voir Fig. 2.6 (h)) ; la réciproque est fausse (Fig. 2.6 (d)). Dans une $(8,4)$-image, si le nombre de Rutovitz d'un point de bord p vaut 2, alors p est 8-simple (Fig. 2.6 (e)) (p doit être un point de bord, ici 4-adjacent à $\overline{X}$, voir Fig. 2.6 (i)) ; la réciproque est fausse. En effet, considérons la figure 2.7 (ruban incliné d'épaisseur 2) : les points a, b, c et d sont 8-simples et $R(a) = 2$, $R(b) = R(c) = R(d) = 4$ (voir Fig. 2.6 (e) pour $R(a)$).

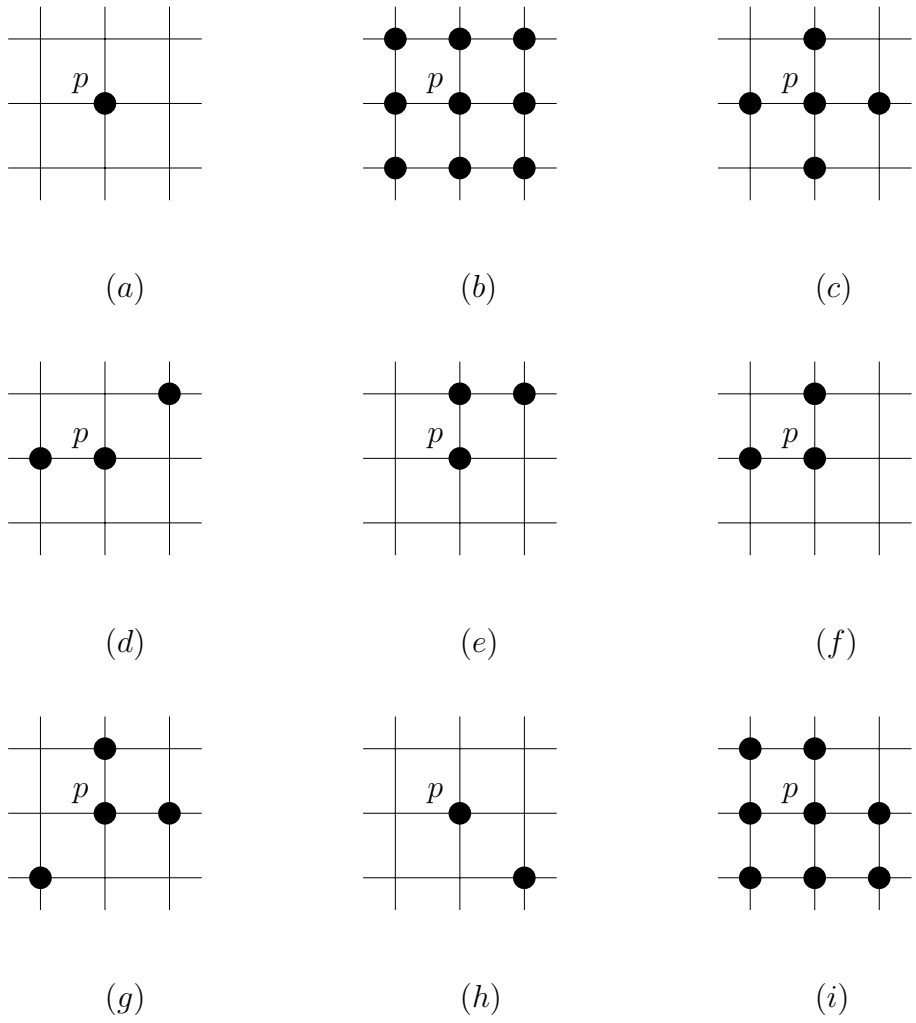


FIG. 2.6 – *Calculs des nombres de Hilditch et de Rutovitz pour différentes configurations de points :* (a) *point isolé (*$H(p) = R(p) = 0$*),* (b) *point* 4*- et* 8*-intérieur (*$H(p) = R(p) = 0$*),* (c) *point* 8*-intérieur non* 4*-intérieur (*$H(p) = 0$*,* $R(p) = 8$*),* (d) *point* 4*-simple non* 8*-simple (*$H(p) = 2$*,* $R(p) = 4$*),* (e) *point* 4*- et* 8*-simple (*$H(p) = 1$*,* $R(p) = 2$*),* (f) *point* 8*-simple non* 4*-simple (*$H(p) = 1$*,* $R(p) = 4$*),* (g) *point ni* 4*- ni* 8*-simple (*$H(p) = 2$*,* $R(p) = 6$*),* (h) *point* 8*-simple non* 4*-simple (*$H(p) = 1$*,* $R(p) = 2$*),* (i) *point* 4*-simple non* 8*-simple (*$H(p) = 0$*,* $R(p) = 2$*).*

2.5.3 Nombres topologiques

G. Bertrand et G. Malandain [BM94] ont introduit le concept de nombres topologiques dans le cas 3D. Deux nombres topologiques permettent de caractériser un point simple. La définition s'applique également dans le cas 2D. Nous retiendrons que leur utilisation présente un plus fort impact dans le cas 3D (voir chapitre 3, la caractérisation d'un point simple par les nombres topologiques et la classification topologique des points selon les nombres topologiques) et dans le

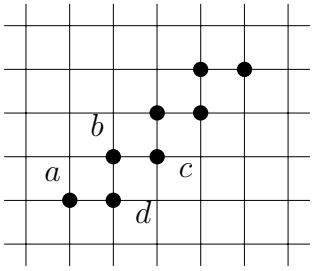
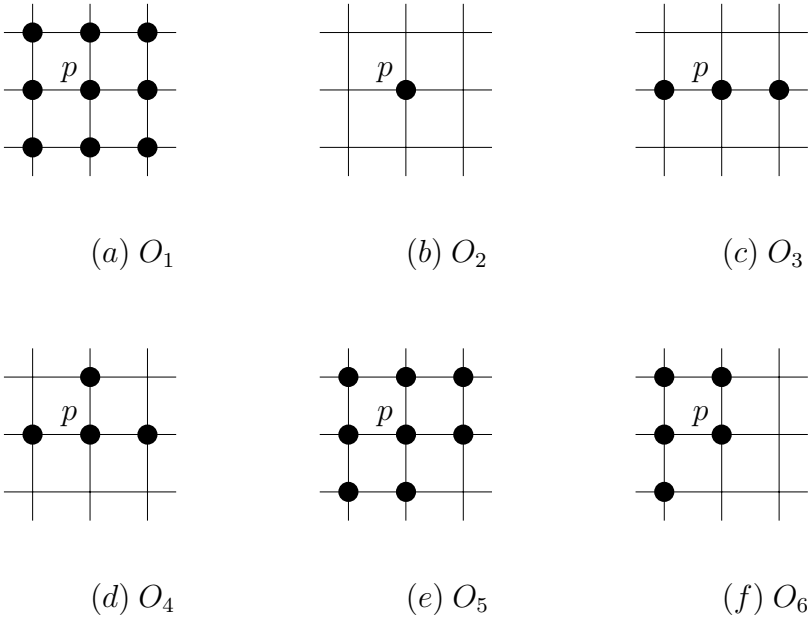


FIG. 2.7 – *Calculs des nombres de Rutovitz pour quelques points d'un ruban incliné d'épaisseur* 2 *(*$R(a) = 2$*,* $R(b) = R(c) = R(d) = 4$*).*



$(a)\ O_1$ $(b)\ O_2$ $(c)\ O_3$

$(d)\ O_4$ $(e)\ O_5$ $(f)\ O_6$

FIG. 2.8 – *Différents objets.*

cas des images 2D en niveaux de gris.

Définition (nombres topologiques en 2D) :
Soit $S \subset \mathcal{Z}^2$ et $p \in S$. On note $NCC(T)$ le nombre de composantes connexes de T, et $NCC^p(T)$ le nombre de composantes connexes de T et adjacentes au point p.

Le *nombre topologique utilisant la* 8*-connexité* est le nombre $T_8(p, S) = NCC^p[S \cap N_8^*(p)]$ $= NCC[S \cap N_8^*(p)]$. Le *nombre topologique utilisant la* 4*-connexité* est le nombre $T_4(p, S) = NCC^p[S \cap N_8^*(p)]$.

Avec l'introduction des nombres topologiques, les deux premières propositions de la section 2.4 peuvent être réécrites ((m, n) étant la connexité de l'image) :

Quantités	O_1	O_2	O_3	O_4		O_5		O_6
				$(8,4)$	$(4,8)$	$(8,4)$	$(4,8)$	
$O([S \cap N_8^*(p)] \cup \{p\})$	1	1	1	1	1	1	1	1
$O(S \cap N_8^*(p))$	1	0	2	1	3	1	1	1
$O([\overline{S} \cap N_8^*(p)] \cup \{p\})$	1	1	1	3	1	2	1	1
$O(\overline{S} \cap N_8^*(p))$	0	1	2	3	3	1	1	1
$O(S)$	1	1	1	1	1	1	1	1
$O(S \setminus \{p\})$	1	0	2	1	3	1	1	1
$O(\overline{S})$	1	1	1	1	1	1	1	1
$O(\overline{S} \cup \{p\})$	2	1	1	1	1	2	1	1
$H(S)$	0	0	0	0	0	0	0	0
$H(S \setminus \{p\})$	1	0	0	0	0	1	0	0
$H(\overline{S})$	1	1	1	1	1	1	1	1
$H(\overline{S} \cup \{p\})$	1	0	2	1	3	1	1	1
$T_m(p, S)$	1	0	2	1	3	1	1	1
$T_n(p, \overline{S})$	0	1	2	1	3	0	1	1
Point simple ?	non	non	non	oui	non	non	oui	oui

TAB. 2.3 – *Différentes quantités pour les objets S de la figure 2.8 (hypothèse : $[\mathcal{Z}^2 \setminus N_8(p)] \cap S = \emptyset$). Les quantités sont données pour les connexités $(m, n) = (8, 4)$ ou $(4, 8)$ s'il n'y a pas de précision supplémentaire.*

Proposition $1'$ **:** Pour $p \in S : O([S \cap N_8^*(p)] \cup \{p\}) = O(S \cap N_8^*(p))$ ssi $T_m(p, S) = 1$.

Proposition $2'$ **:** Pour $p \in S : O([\overline{S} \cap N_8^*(p)] \cup \{p\}) = O(\overline{S} \cap N_8^*(p))$ ssi $T_n(p, \overline{S}) = 1$.

Un point simple peut alors être caractérisé par les nombres topologiques :

Proposition 6 **(caractérisation d'un point simple par les nombres topologiques) :** Soit $S \subset \mathcal{Z}^2$ et $p \in S$. Le point p est un point 8-simple $\Leftrightarrow T_8(p, S) = 1$ et $T_4(p, \overline{S}) = 1$. Le point p est un point 4-simple $\Leftrightarrow T_4(p, S) = 1$ et $T_8(p, \overline{S}) = 1$.

Quelques exemples de calculs de nombres topologiques et autres sur les différents objets représentés à la figure 2.8 sont donnés dans le tableau 2.3.

Remarque : Considérons de nouveau les objets S des figures 2.5 (a) et (b). Ces objets présentent une même configuration locale au point p (Fig. 2.5 (c)), on a $T_8(p, S) = 2$ et $T_4(p, \overline{S}) = 2$,

$(m,n)=(4,8)$	$\overline{T}=0$	$\overline{T}=1$	$\overline{T}=2$	$\overline{T}=3$	$\overline{T}=4$
$T=0$	0	16	0	0	0
$T=1$	1	116	0	0	0
$T=2$	0	0	102	0	0
$T=3$	0	0	0	20	0
$T=4$	0	0	0	0	1

TAB. 2.4 – *Nombres de configurations ayant pour nombres topologiques le couple* $(T,\overline{T})$ *selon la connexité* $(m,n)=(4,8)$.

$(m,n)=(8,4)$	$\overline{T}=0$	$\overline{T}=1$	$\overline{T}=2$	$\overline{T}=3$	$\overline{T}=4$
$T=0$	0	1	0	0	0
$T=1$	16	116	0	0	0
$T=2$	0	0	102	0	0
$T=3$	0	0	0	20	0
$T=4$	0	0	0	0	1

TAB. 2.5 – *Nombres de configurations ayant pour nombres topologiques le couple* $(T,\overline{T})$ *selon la connexité* $(m,n)=(8,4)$.

ou $T_4(p,S)=2$ et $T_8(p,\overline{S})=2$, le point n'est alors ni 4-simple, ni 8-simple ; et l'on ne peut connaître localement la raison de la non-simplicité (déconnexion de deux composantes de S ou perte d'un trou de S). En d'autres mots, le couple de nombres topologiques peut nous indiquer si un point est simple ou non, mais dans le cas où le point n'est pas simple, on ne peut connaître la raison de sa non-simplicité, ce qui n'est évidemment pas le cas si on utilise les propositions 4 et 5 de la section 2.4.

Nous donnons les nombres de configurations ayant un couple donné de nombres topologiques selon les connexités $(m,n)=(4,8)$ ou $(8,4)$, respectivement dans les tableaux 2.4 et 2.5, configurations parmi les $256(=2^8)$ possibles dans $N_8^*(p)$ d'un point p. Dans ces tableaux, nous notons $T=T_m(p,S)$ et $\overline{T}=T_n(p,\overline{S})$. Nous vérifions que le nombre de couples tels que $T_m(p,S)=k_1$ et $T_n(p,\overline{S})=k_2$ est égal au nombre de couples tels que $T_n(p,S)=k_2$ et $T_m(p,\overline{S})=k_1$, en raison des propriétés obtenues par le passage par le complémentaire. Dans le cas 2D, si $T_m(p,S)=k_1$, $T_n(p,\overline{S})=k_2$, et k_1 ou k_2 supérieur ou égal à 2, alors k_1 égale k_2. Dans ces tableaux, nous voyons que le nombre de points 4-simples (égal au nombre de points 8-simples, car caractérisés par des nombres topologiques de même valeur) est de 116 parmi les 256 configurations possibles.

2.5.4 Table de consultation

On peut également générer tous les voisinages possibles d'un point, ne retenir que ceux qui font du point central un point simple (grâce à l'une des caractérisations précédentes) et stocker ces configurations dans une table de consultation. Cela permet dans un second temps (c.-à-d. après la création de la table) de déterminer si un point est simple ou non, uniquement par la donnée à la table de consultation, du voisinage du point.

2.5.5 Graphes de décision binaire

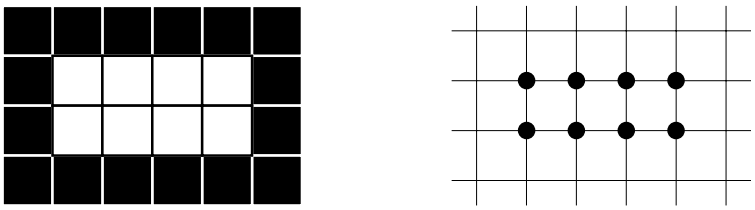
Il existe une autre méthode de stockage des configurations vérifiant une propriété donnée (en l'occurrence ici, celle donnant un point simple). Un graphe de décision binaire (*Binary Decision Diagram* ou *BDD*) est un graphe dont les sommets sont les points décrivant toute configuration. Deux arcs sortent de chaque sommet, et le choix de l'arc à suivre lors d'un parcours (*i.e.* d'une configuration à examiner) dépend de la présence ou non du point associé au nœud dans la configuration examinée. La valeur des nœuds terminaux du graphe indique si la configuration qui y amène, vérifie la propriété sur laquelle a été créé le graphe. Il existe des techniques de compression (par recherche de sous-graphes isomorphes . . .) qui rendent cet outil très performant. D'un point de vue pratique, le graphe de décision binaire peut être transformé en un fichier informatique dont la taille de stockage est généralement inférieure à celle nécessaire pour une table de consultation créée pour la même propriété (voir l'annexe C).

2.6 Suppression en parallèle de points

Nous étudions à présent la suppression en parallèle de plusieurs points.

Le processus consistant à supprimer en parallèle tous les points simples d'un objet et à reitérer cela jusqu'à stabilité peut ne pas préserver la topologie : c'est le cas du ruban d'épaisseur 2 (Fig. 2.9), pour lequel tous les points sont simples ; la suppression en parallèle de tous les points simples détruit l'objet, la topologie n'est alors pas préservée. C'est la raison pour laquelle plusieurs stratégies ont été proposées. Elles seront vues au chapitre 4 et détaillées aux annexes A et B. Néanmoins, nous décrivons succinctement maintenant les trois stratégies utilisées :

- soit chaque point de l'image est traité en parallèle (traitement *fortement parallèle*) et dans ce cas, un plus grand voisinage doit être examiné afin de savoir si un point peut être retiré ;
- soit seul un sous-ensemble de points de l'image n'est considéré de telle façon (dans le "meilleur des cas") qu'il n'y ait pas d'"interaction" entre les points considérés. C'est la méthode dite *des sous-mailles* : tous les points simples d'une même sous-maille sont retirés en parallèle (d'autres conditions peuvent être imposées selon le choix des sous-mailles) ;
- soit une itération de suppression est divisée en plusieurs sous-itérations directionnelles. Pour une sous-itération donnée, on retire les points simples vérifiant une condition supplémentaire régie par la sous-itération : en pratique, le voisin d'un tel point selon la direction associée à la sous-itération, doit appartenir au complémentaire (d'autres conditions peuvent être imposées selon le nombre de directions).



(a) (b)

FIG. 2.9 – *Exemple de non-préservation de la topologie par suppression parallèle de tous les points simples d'un objet (tous les points de ce ruban d'épaisseur* 2 *sont* 4*- ou* 8*-simples).*

Nous introduisons d'abord les concepts de suppressibilité d'un ensemble et de suppressibilité forte. Dans le cas particulier d'un ensemble constitué d'un seul point, nous verrons que la suppressibilité forte équivaut à la supressibilité et à la simplicité.

Nous proposerons enfin le concept d'ensemble minimal non simple : un tel ensemble ne doit pas être supprimé (en totalité) par un algorithme opérant par suppression parallèle de points, pour que ce dernier préserve la topologie de l'image. Finalement, nous proposons les configurations correspondant à de tels ensembles dans le cas d'une $(8,4)$-image. Cette notion sera également étudiée dans le cas 3D (voir chapitre 3).

2.6.1 Suppressibilité et suppressibilité forte [Ron86][Ron88]

C. Ronse a introduit le concept de m-suppressibilité et de m-suppressibilité forte. Cette dernière est définie par le respect de quatre conditions pour la suppression d'un ensemble D constitué d'un ou de plusieurs points.

Soit une image binaire discrète bidimensionnelle $(\mathcal{Z}^2, m, n, B)$. On note par $|A|$, la taille d'un ensemble A. Soient deux sous-ensembles S et T. On note par $C(m,T)$, l'ensemble des composantes m-connexes de T ; et $C(m,T,S)$, l'ensemble des composantes de T, m-connexes et m-adjacentes à S. On note $C_m(T) = |C(m,T)|$, $C_m(T,S) = |C(m,T,S)|$ et $W = \mathcal{Z}^2 \setminus B$.

Soit un sous-ensemble D de B (D est le sous-ensemble de points que nous souhaitons enlever de B). Nous voulons examiner le comportement de $B \setminus D$.

L'invariance du nombre de composantes connexes de l'objet et du fond avant et après suppression de D s'exprime par les deux égalités suivantes :

$$C_m(B) = C_m(B \setminus D) \tag{2.18}$$

$$C_n(W) = C_n(W \cup D) \tag{2.19}$$

Ces deux équations définissent la m*-suppressibilité.*

Définition (suppressibilité) : Soit $D \subseteq B$. Le sous-ensemble D est m*-supprimable* de B si

– B a le même nombre de composantes m-connexes que $B \setminus D$ (équ. (2.18)),
– W a le même nombre de composantes n-connexes que $W \cup D$ (équ. (2.19)).

Remarque : Nous pouvons faire le lien avec ce que nous avons vu précédemment dans le cas où D est constitué d'un seul point p ($D = \{p\}$) : la m-suppressibilité de D équivaut à la simplicité de p (propositions 4 et 5).

Considérons la figure 2.10. Sur la figure 2.10 (a), l'image est constituée de deux objets de B et d'un objet de W. En supprimant l'ensemble D (voir Fig. 2.10 (b)), ces nombres ne changent pas (les deux équations (2.18) et (2.19) sont vérifiées, D est m-supprimable), mais on a divisé une composante de B et on en a perdu une autre. À la figure 2.10 (c), l'image est constituée d'une composante de l'objet B et de deux composantes de W. En supprimant l'ensemble D, on crée une nouvelle composante de W, et on en fusionne deux autres (les deux équations (2.18) et (2.19) sont vérifiées, D est m-supprimable).

Nous nous attendons à ce que le processus de suppression maintienne une correspondance naturelle entre les composantes connexes de B et celles de W d'une part et entre celles de $B \setminus D$ et celles de $W \cup D$ d'autre part. C'est pourquoi la suppressibilité forte a été introduite.

La *m-suppressibilité forte* est définie de la façon suivante :
– une composante connexe d'un objet ne peut pas être supprimée,
– une composante connexe d'un objet ne peut pas être divisée,
– une composante connexe du fond ne peut pas être créée,
– deux (ou plusieurs) composantes connexes du fond ne peuvent pas être fusionnées en une seule.

Aux figures 2.10 (a) et (c), l'ensemble D n'est alors pas fortement m-supprimable.
Plus formellement :

Définition (suppressibilité forte) : Soit $D \subseteq B$. Le sous-ensemble D est *fortement m-supprimable* de B si
– la relation $\supseteq$ "contient" induit une bijection de l'ensemble des composantes m-connexes de B dans l'ensemble des composantes m-connexes de $B \setminus D$; *i.e.*
  – toute composante m-connexe de B contient une unique composante m-connexe de $B \setminus D$ (ce qui évite la perte d'une composante de B ou la déconnexion d'une composante de B, comme cela était le cas, Fig. 2.10 (a)) ;
  – et toute composante m-connexe de $B \setminus D$ est contenue dans une unique composante m-connexe de B (ce qui s'explique par la suppression d'un ensemble D de B),
– la relation $\subseteq$ "est inclus dans" induit une bijection de l'ensemble des composantes n-connexes de W dans l'ensemble des composantes n-connexes de $W \cup D$; *i.e.*
  – toute composante n-connexe de W est contenue dans une unique composante n-connexe de $W \cup D$ (ce qui s'explique par la suppression d'un ensemble D de B) ;
  – et toute composante n-connexe de $W \cup D$ contient une unique composante n-connexe de W (ce qui évite la création d'une composante de W ou la fusion de deux composantes de W, comme cela était le cas, Fig. 2.10 (c)).

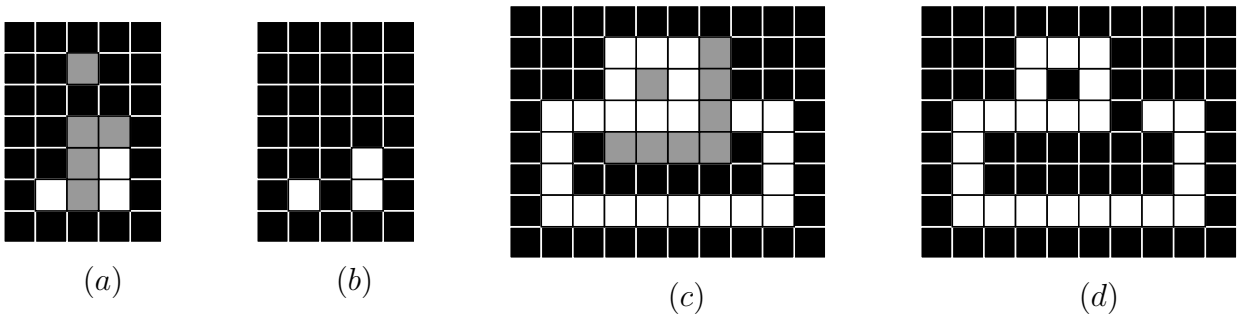


FIG. 2.10 – *Illustration de la m-suppressibilité forte (D en gris, B en blanc et gris, W en noir).*

Remarque : Notons qu'en 2D, l'équivalence entre la fusion de deux composantes de W et la destruction d'un trou de B (Fig. 2.10 (c) et (d)), et l'équivalence entre la création d'une composante de W et la création d'un trou de B (Fig. 2.10 (c) et (d)) (cf. l'équation $O(W)-1 = H(B)$, section 2.1.1) résultent de la suppression d'un ensemble de points de B. Le passage par le complémentaire exprime l'équivalence entre la perte d'une composante de B et la perte d'un trou de W (Fig. 2.10 (a) et (b)), et l'équivalence entre la déconnexion d'une composante de B et la création d'un trou dans W (Fig. 2.10 (a) et (b)) (cf. l'équation $O(B) = H(W)$, section 2.1.1).

Rappelons les propriétés suivantes données par C. Ronse :

Lemme : Soit $D \subseteq B$. Si D est 4-connexe (et en particulier si D contient un seul point), alors D est m-supprimable de B si et seulement si D est fortement m-supprimable de B.

Proposition : Soit $D \subseteq B$, $t = |D|$. On suppose $t > 1$. Les deux clauses suivantes sont équivalentes :
- D est fortement m-supprimable de B,
- les éléments de D peuvent être étiquetés $p_1, \ldots, p_t$ tels que pour tout $i = 1, \ldots, t$, le point p_i est m-supprimable de $B \setminus \{p_j, j < i\}$.

En pratique, pour montrer que D est fortement m-supprimable, il faut et il suffit de trouver une séquence de points p_i de D ($i = 1, \ldots, t$) telle que p_1 est m-supprimable (*i.e.* m-simple) pour B et p_i ($i > 1$) est m-supprimable (*i.e.* m-simple) pour $B \setminus \{p_j, j < i\}$. Un ensemble m-supprimable est également appelé *ensemble m-simple*, voir chapitre suivant.

Note : Considérons l'algorithme consistant à supprimer à partir de courbes ouvertes simples, horizontales ou verticales et de longueur strictement supérieure à 5 points, des ensembles D de deux points 4-adjacents entre eux, tels que l'un des deux soit point extrémité (l'autre point étant forcément son unique voisin dans la courbe). Cet algorithme préserve la topologie. Dans un tel sous-ensemble D, il existe une seule séquence qui indique que D est fortement m-supprimable (p_1 est le point extremité et p_2 est son voisin, car p_2 n'est pas initialement m-simple, mais le

devient après suppression du point extrémité p_1). Cela nous fait comprendre qu'un ensemble de points peut être enlevé (suppression parallèle de points), tout en préservant la topologie, si cette suppression peut être simulée par un processus séquentiel de suppression de points simples amenant au même résultat (la topologie est alors préservée, par la définition même d'un point simple).

2.6.2 Ensembles minimaux non supprimables

Nous rappelons la définition puis la caractérisation d'un ensemble minimal non supprimable, proposées par C. Ronse. Dans un second temps, nous donnerons la caractérisation équivalente, donnée par R.W. Hall, ainsi qu'une propriété liant ce concept d'ensemble minimal non supprimable à celui de la préservation de la topologie lors de la suppression en parallèle de points.

2.6.2.1 Définition et caractérisation par Ronse

Définition (ensemble minimal non supprimable) : Soit $D \subseteq B$. Si D n'est pas (fortement) m-supprimable de B, alors il contient un plus petit sous-ensemble U non vide qui n'est pas (fortement) m-supprimable. Ainsi U n'est pas (fortement) m-supprimable de B, mais tout sous-ensemble propre de U l'est. Alors U est appelé *sous-ensemble minimal non m-supprimable* de B. Par la suite, un tel ensemble est appelé *ensemble minimal non simple* (ou *MNS*, pour *Minimal Non simple Set*). En fait, cette définition ne dépend pas du fait de considérer la m-suppressibilité ou la m-suppressibilité forte (voir [Ron88]).

Théorème : Soit $U \subseteq B$, avec $U \neq \emptyset$. Alors U est un MNS de B si et seulement si l'une des conditions suivantes est vérifiée :
- U consiste en un point isolé,
- U est une paire de points 8-adjacents qui sont m-supprimables de F, mais U n'est pas supprimable de B,
- dans le cas d'une $(8,4)$-image, U est un triplet ou un quadruplet de points mutuellement 8-adjacents, et U est une composante 8-connexe de B (en d'autres mots, il n'y a pas de point de $B \setminus U$ 8-adjacent à U).

Proposition : Soit $U = \{p, q\}$, une paire de points diagonalement adjacents de B. Pour une $(8,4)$-image, si U est un MNS de B, alors U est une composante 8-connexe de B (en d'autres mots, il n'y a pas de point de $B \setminus U$ 8-adjacent à U).

Remarque : En pratique, pour vérifier qu'une paire $\{p, q\}$ de points 8-adjacents est un MNS de B, nous nous assurons que p et q sont à la fois supprimables de B, avec p non supprimable de $B \setminus \{q\}$ (ou de façon équivalente, q non supprimable de $B \setminus \{p\}$).

En fait, les motifs "interdits" (ou MNS) que D ne doit pas contenir ont été donnés explicitement par R.W. Hall, suite à la caractérisation proposée par C. Ronse. Nous les donnons ci-après, uniquement dans le cas d'une $(8,4)$-image.

a	0	1	x
b	p	q	b
x	1	0	a ’

M_1

a	0	1	x
b	p	q	0
x	1	1	x ’

M_2

0	0	1	x
0	p	q	0
0	0	1	x ’

M_3

x	1	1	x
0	p	q	0
x	1	1	x

M_4

TAB. 2.6 – *Ensembles MNS (pour les composantes, voir Tab. 2.7) pour un algorithme parallèle de squelettisation pour une $(8,4)$-image. Un ensemble $\{p,q\}$ vérifiant l’un de ces motifs ne doit pas être entièrement supprimé par un algorithme parallèle de squelettisation (afin de préserver la topologie).*

						1				1		1				1				1		1	1		1	1		1	1
1		,	1	1	,	1		,	1		,		1	,	1	1	,	1	1	,	1		,		1	,	1	1

TAB. 2.7 – *Composantes à ne pas effacer (entièrement) lors d’un algorithme parallèle de squelettisation pour une $(8,4)$-image (afin de préserver la topologie).*

2.6.2.2 Caractérisation par Hall

R.W. Hall [Hal92] a montré qu’un algorithme de supression parallèle préserve la topologie si :

- toute composante de points mutuellement adjacents entre eux n’est pas complètement supprimée,
- tout point supprimé est m-supprimable,
- si deux points p et q m-adjacents sont supprimés, avec $[[N_8(p) \cup N_8(q)] \setminus \{p,q\}] \cap S \neq \emptyset$ (sinon $\{p,q\}$ serait une composante, et ce serait le premier cas), alors $\{p,q\}$ doit être m-supprimable.

En fait, dans le cas d’une $(8,4)$-image, cela correspond à ne supprimer que des points 8-simples, à ne pas supprimer un motif (à rotation de $90°$ ou $180°$ près, à symétrie horizontale ou verticale près) parmi ceux représentés dans le tableau 2.6 (le cas $a = 1$ et $b = 0$ n’est pas permis car il faut que les points p et q soient 8-simples ; quelle que soit l’appartenance de x), et à préserver au moins un 1 dans les composantes représentées dans le tableau 2.7.

2.6.2.3 Exemple

À la figure 2.11 sont représentés des objets pour lesquels sont encerclés les points simples non extrémités, et sont encadrés en pointillé soit les MNS correspondant aux motifs M_1 à M_4 aux figures 2.11 (a) et (c), soit le MNS correspondant à la composante connexe composée de trois points simples (encerclés) dans la figure 2.11 (b).

Si l’on considère un algorithme ne supprimant aucun point des MNS, le résultat peut comporter des points simples ; d’où l’idée de supprimer certains points des MNS. Ce qui explique des conditions directionnelles à l’intérieur même de processus de suppression parallèle (voir l’annexe

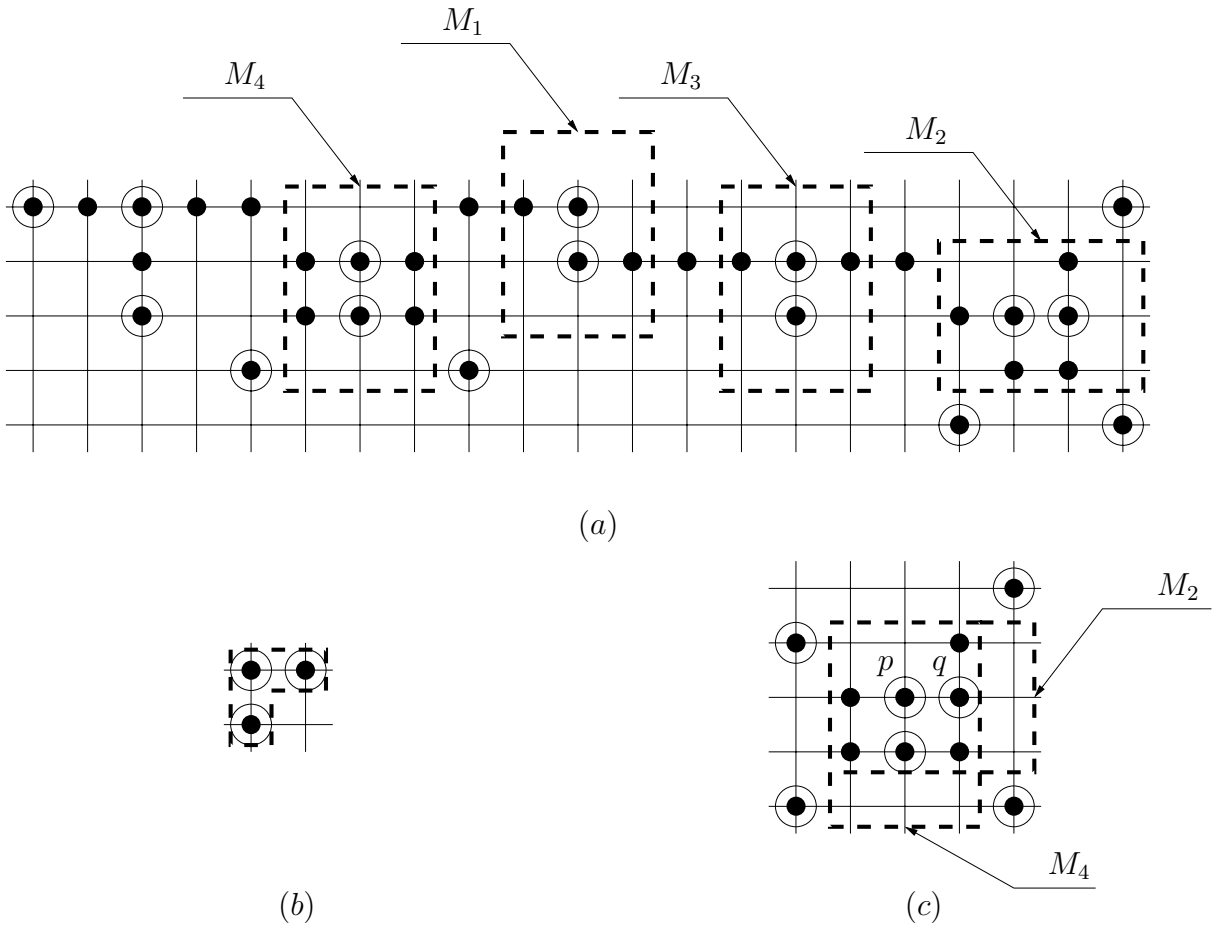


FIG. 2.11 – *Exemples de MNS* 2*D selon la connexité* $(8, 4)$.

A).

Considérons un algorithme supprimant en parallèle les points simples sauf certains d'un MNS, en adoptant le choix suivant : supprimer le point q du motif M_1, M_2 ou M_3 et le point le plus au $Nord$ (resp. à l'$Ouest$) parmi p et q du motif M_4 si ce dernier est considéré verticalement (resp. horizontalement). Considérons l'objet représenté à la figure 2.11 (c). Cet objet est tel que le point p appartient à deux MNS "imbriqués" M_2 et M_4. Le choix précédemment retenu est tel que l'ensemble $\{p, q\}$ va être éliminé de cet objet, l'objet initial va être déconnecté ; la topologie n'est alors pas préservée. Cet exemple montre qu'il faut être prudent lorsque des conditions directionnelles sont mises en jeu afin d'éliminer certains points des MNS.

2.7 Conclusion

Après avoir explicité les notions de genre 2D et de point simple, nous avons rappelé quelques relations liant les points simples aux calculs de composantes connexes, ou implicitement au genre. Puis nous avons caractérisé un point simple à l'aide des nombres topologiques. Nous verrons au chapitre suivant l'importance d'utiliser les nombres topologiques afin de caractériser efficacement les points simples ; cette caractérisation sera beaucoup plus simple à utiliser que celle utilisant le genre 3D ou que celle comptant les nombres de composantes ou de trous (voir chapitre suivant). Les problèmes de la suppression de points en parallèle ont également été abordés dans ce chapitre ; ils seront également développés dans le cas 3D, dans le chapitre suivant.

Chapitre 3

Genre 3D et simplicité

3.1 Introduction

Dans ce chapitre, après une nouvelle approche du concept de trou (cf. section 1.6), nous rappelons les formules permettant le calcul du genre 3D. Cette première partie provient du rapport technique [Mor80] (et de quelques précisions et corrections dans [NA85]). Nous ne détaillons pas l'obtention des formules, comme nous l'avions fait dans le cas 2D, au chapitre 2. La différence de difficulté peut être observée par rapport au cas 2D en ce qui concerne les formules, les motifs employés, les exemples. Nous donnons quelques exemples de calculs de genre sur de petits objets synthétiques qui, bien qu'étant de taille modeste, nécessitent un nombre important de calculs "délicats".

Nous présentons ensuite le concept de point simple en 3D, utilisant la notion de genre. Des exemples de points simples ou non simples, utilisant la caractérisation avec le genre, sont donnés. Puis nous introduisons les nombres topologiques, définis par G. Bertrand et G. Malandain [BM94]. Leur grande facilité d'utilisation pour déterminer la simplicité d'un point sera observée par rapport à la caractérisation utilisant le genre ; les nombres topologiques n'exigeant que le calcul de nombres de composantes connexes dans certains voisinages sans requérir ni ceux des cavités ni ceux des trous.

Nous rappellerons une classification de points à l'aide des nombres topologiques. Par la suite, la possibilité de caractériser de façon locale un point simple par le calcul de deux nombres topologiques, et celle de caractériser des points terminaux de surface (notion utilisée dans les algorithmes de squelettisation surfaciques – voir l'annexe B) par un certain couple de valeurs des nombres topologiques seront retenues.

Ce chapitre se terminera de façon similaire au cas 2D (chapitre 2) par la présentation du concept d'ensemble minimal non simple (ou MNS).

3.2 Définition du genre

3.2.1 Nouvelle approche du concept de trou [Mor80]

Considérons une image 2D $\mathcal{I}$ comportant O objets et H trous. Plongeons maintenant l'image $\mathcal{I}$ dans $\mathcal{Z}^3$, nous obtenons ainsi l'image 3D $\mathcal{J}$ (seuls les points appartenant à l'objet sont ceux d'un plan de $\mathcal{J}$, ce plan "correspondant" à $\mathcal{I}$). En utilisant G_4 ou G_8 sur $\mathcal{I}$ et G_6 ou G_{26} sur $\mathcal{J}$, on a : $G_4 = G_6$ et $G_8 = G_{26}$. On rappelle qu'en 2D, le genre est égal au nombre d'objets moins le nombre de trous (équation (2.6), page 23). Or l'image J n'a pas de cavités. Puisque les cavités sont les analogues 3D des trous 2D, les fonctions G_6 et G_{26} ne donnent pas le nombre d'objets 3D moins le nombre de cavités 3D ; G_6 et G_{26} sont également affectés par ce que l'on appelle les *trous* $3D$.

3.2.2 Définition du genre 3D

Soit une image discrète binaire tridimensionnelle $(\mathcal{Z}^3, m, n, B)$. Nous considérons d'abord le calcul du genre d'une seule composante S, $S \subseteq B$.

L'équation d'Euler-Schaefli (équation (2.1), page 21), pour une structure 3D simplement connexe, donne :

$$V - E + F - Q = 0 \tag{3.1}$$

avec V le nombre de sommets, E le nombre d'arêtes, F le nombre de faces et Q le nombre de cellules (en incluant le fond).

Nous distinguons les trois types de cellules suivants :
- cellule α : cellules de S (tesselles cubiques),
- cellule β : cellules de $\mathcal{Z}^3 \setminus S$ entourées par S : polyèdres de 8 sommets ou plus,
- cellule du fond γ : composante de $\mathcal{Z}^3 \setminus S$ non entourée par S.

La structure (construite de la même façon qu'en 2D) n'est pas forcément simplement connexe, nous avons alors $V-E+F-(\#\alpha+\#\beta+\#\gamma) \neq 0$, en notant $\#x$ le nombre de cellules de type x. Le *nombre de trous d'un objet* S (noté $H(S)$) est défini comme étant le montant par lequel cette somme diffère de zéro. Nous avons : $H(S) = -V + E - F + (\#\alpha + \#\beta + \#\gamma)$, avec $\beta = C(S)$ étant le nombre de cavités de l'objet S, et $\#\gamma = 1$ (correspondant au fond). Nous obtenons alors : $1 + C(S) - H(S) = V - E + F - \#\alpha$. Notons $O(B)$ (resp. $C(B)$) le nombre de composantes connexes (resp. de cavités) de B ; nous avons $O(B) = \sum_{S \subseteq B} O(S)$, et $C(B) = \sum_{S \subseteq B} C(S)$. En sommant sur toutes les composantes $S \subseteq B$, on a : $O(B) - H(B) + C(B) = V - E + F - \#\alpha$ (en supposant que $H(B) = \sum_{S \subseteq B} H(S)$, voir la justification dans [Mor80]).

Définition (genre 3D) : Le *genre* $G_m(B)$ (également appelé *caractéristique d'Euler*) d'un ensemble B est le nombre d'objets (composantes m-connexes de B) (noté $O(B)$), moins le nombre de trous de B (noté $H(B)$), plus le nombre de cavités de B (noté $C(B)$) ; (m, n) étant la connexité de l'image :

$$G_m(B) = O(B) - H(B) + C(B) \tag{3.2}$$

3.2.3 Genre du complémentaire

Le nombre de cavités de $\mathcal{Z}^3 \setminus B$ est le nombre d'objets de $B \Rightarrow C(\mathcal{Z}^3 \setminus B) = O(B)$. Le nombre d'objets de $\mathcal{Z}^3 \setminus B$ est le nombre de cavités de B plus 1 (correspondant au background) $\Rightarrow O(\mathcal{Z}^3 \setminus B) = C(B)+1$. Le nombre de trous de B est celui de $\mathcal{Z}^3 \setminus B$: $H_6(B) = H_{26}(\mathcal{Z}^3 \setminus B)$ et $H_{26}(B) = H_6(\mathcal{Z}^3 \setminus B)$ (cf. la justification dans [Mor80]) ; cette correspondance entre le nombre de trous de l'objet et de son complémentaire explique les restrictions lors de la détection d'un trou pour une $(6, 26)$-image, que nous avons précédemment rencontrées à la section 1.6.

Nous obtenons alors :

$$G_{26}(\mathcal{Z}^3 \setminus B) - G_6(B) = 1,\ G_6(\mathcal{Z}^3 \setminus B) - G_{26}(B) = 1. \tag{3.3}$$

3.3 Calcul du genre 3D de façon locale

De la même manière qu'en 2D, V, E, F et $\#\alpha$ peuvent être exprimées en fonction de motifs élémentaires pouvant composer un octant. Sur 256 ($= 2^8$) octants possibles, il y en a 22 élémentaires ; les autres sont obtenus à isométries près. Ces motifs sont donnés à la figure 3.1. Nous ne donnerons que les formules de G_6 et de G_{26} obtenues à partir de ces motifs (cf. ci-après). Avec cette méthode, il est beaucoup plus rapide de calculer G_6. Il est alors possible de calculer $G_{26}(B)$ par rapport à $G_6(\mathcal{Z}^3 \setminus B)$ et les relations par le complémentaire. Ces formules ont été données dans [Mor80] et rectifiées dans [NA85].

En raison de la propriété additive de la caractéristique d'Euler, afin de calculer la caractéristique d'un objet, nous pouvons sommer celles calculées sur chaque octant de l'image (cf. les exemples des sections 3.4.5 et 3.4.6). Nous pouvons alors créer et consulter une table stockant les caractéristiques d'Euler pour un octant parmi les 256 possibles (cf. [Ma94] et [LKC94]). Une caractéristique sur un octant peut être calculée avec les motifs de la figure 3.1, ou par passage par l'analogue continu [Ma94], comme cela l'avait déjà été fait dans [KR89] dans le cas 2D (voir également le lien avec la topologie algébrique et avec la caractéristique d'Euler des surfaces dans [LKC94]).

D'après l'étude réalisée dans [Mor80] (en fait, les formules proviennent de [NA85] - il y a des erreurs dans [Mor80]), nous avons :

$$G_6(B) = \psi_1 - \psi_2 + \psi_3 - \psi_4, \tag{3.4}$$

avec

$$\begin{aligned} \psi_1 &= \#[2] \\ \psi_2 &= \#[3] \\ \psi_3 &= \#[9] \\ \psi_4 &= \#[22], \end{aligned}$$

où le symbole $\#[x]$ désigne le nombre de fois qu'apparaît le motif x dans l'objet B, pour toute orientation ; les motifs x étant représentés à la figure 3.1.

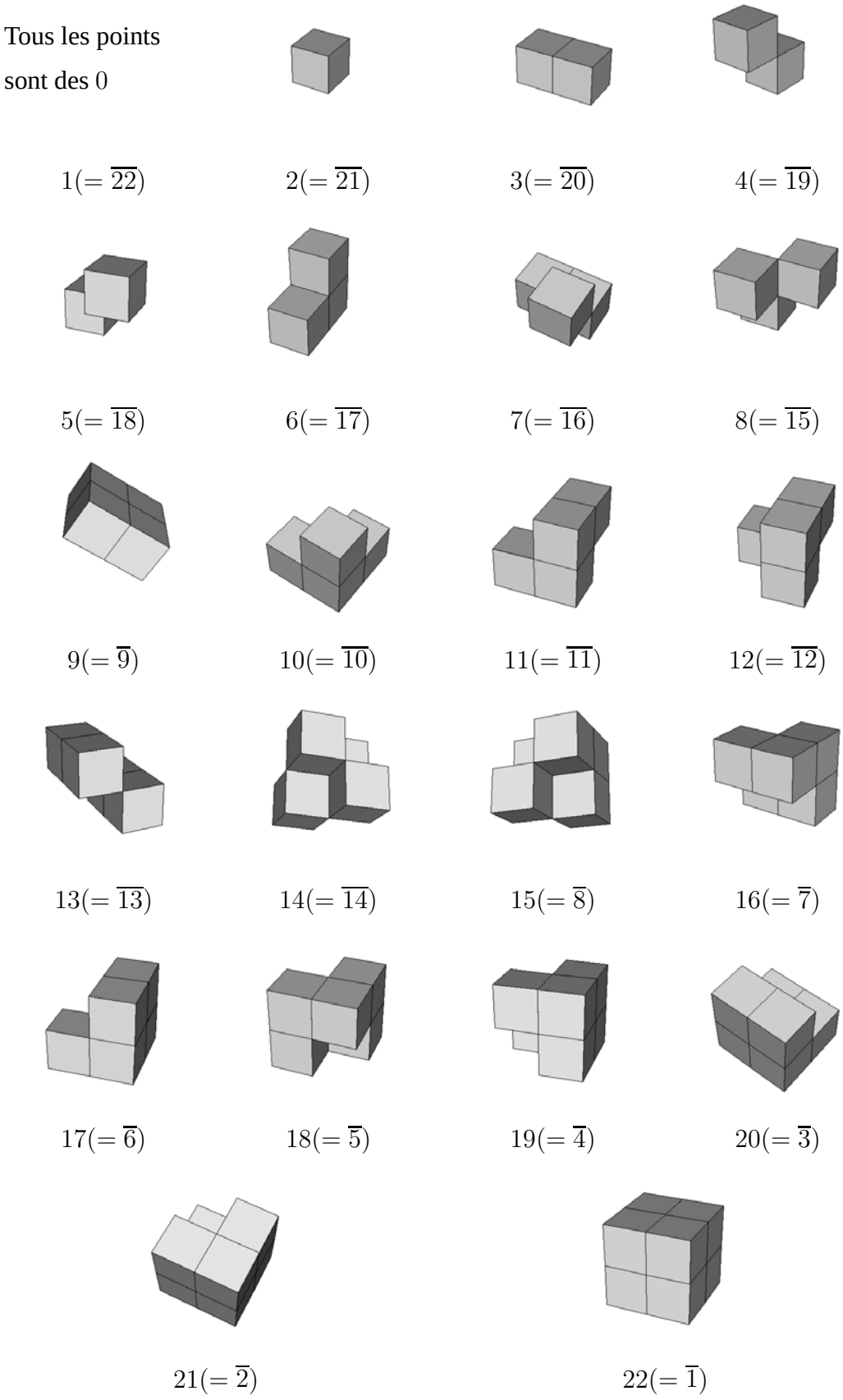


FIG. 3.1 – *Motifs utilisés pour le calcul du genre 3D (le voxel caché dans le motif 22 est à 1).*

Pour le cas $(26, 6)$, nous avons [Mor80] [NA85] :

$$G_{26}(B) = \varphi_1 - \varphi_2 + \varphi_3 - \varphi_4 + \varphi_5 - \varphi_6 + \varphi_7 - \varphi_8 \tag{3.5}$$

avec :

$$\begin{aligned}
\varphi_1 &= \#[2] \\
\varphi_2 &= \#[3] + \#[4] + \#[5] \\
\varphi_3 &= \#[6] + \#[7] + \#[8] \\
\varphi_4 &= \#[9] + \#[10] + \#[11] + \#[12] + \#[13] + \#[14] \\
\varphi_5 &= \#[15] + \#[16] + \#[17] \\
\varphi_6 &= \#[18] + \#[19] + \#[20] \\
\varphi_7 &= \#[21] \\
\varphi_8 &= \#[22]
\end{aligned}$$

3.4 Exemples de calcul de genre 3D

3.4.1 Objet avec une composante connexe

L'objet B représenté à la figure 3.2 est composé d'une composante connexe et ne comporte ni cavité ni trou : $O(B) = 1$, $H(B) = 0$, $C(B) = 0$, alors $G(B) = 1$ (équ. (3.2)). On a $\#[2] = 12$, $\#[3] = 17$, $\#[4] = 12$, $\#[6] = 24$, $\#[9] = 6$. On a $\psi_1 = 12$, $\psi_2 = 17$, $\psi_3 = 6$, alors $G_6(B) = 1$ (équ. (3.4)). On a $\varphi_1 = 12$, $\varphi_2 = 29$, $\varphi_3 = 24$, $\varphi_4 = 6$, alors $G_{26}(B) = 1$ (équ. (3.5)).

3.4.2 Objet avec une composante connexe et un trou

L'objet B représenté à la figure 3.3 est composé d'une composante connexe, présente un trou et ne comporte pas de cavité. On a $O(B) = 1$, $H(B) = 1$, $C(B) = 0$, alors $G(B) = 0$ (équ. (3.2)). On a $\#[2] = 8$, $\#[3] = 8$, $\#[4] = 4$, $\#[6] = 4$. On a $\psi_1 = 8$, $\psi_2 = 8$ alors $G_6(B) = 0$ (équ. (3.4)). On a $\varphi_1 = 8$, $\varphi_2 = 12$, $\varphi_3 = 4$ alors $G_{26}(B) = 0$ (équ. (3.5)).

3.4.3 Objet avec une composante connexe et deux trous

L'objet B représenté à la figure 3.7 est composé d'une composante connexe, présente deux trous et ne comporte pas de cavité. On a $O(B) = 1$, $H(B) = 2$, $C(B) = 0$, alors $G(B) = -1$ (équ. (3.2)). On a $\#[2] = 13$, $\#[3] = 14$, $\#[4] = 8$, $\#[6] = 8$. On a $\psi_1 = 13$, $\psi_2 = 14$ alors $G_6(B) = -1$ (équ. (3.4)). On a $\varphi_1 = 13$, $\varphi_2 = 22$, $\varphi_3 = 8$, alors $G_{26}(B) = -1$ (équ. (3.5)).

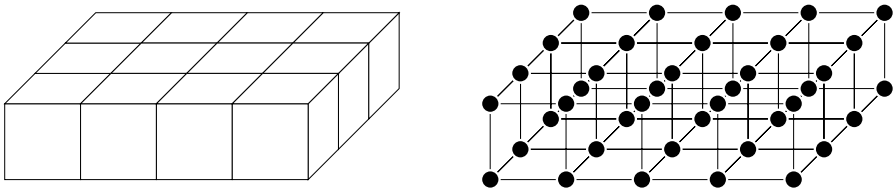


FIG. 3.2 – *Objet sans trou et le graphe associé.*

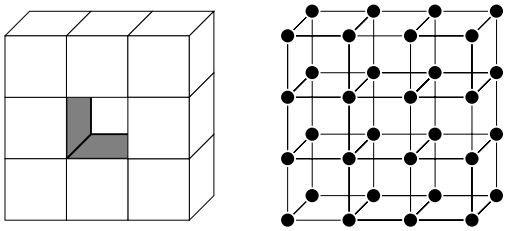


FIG. 3.3 – *Objet avec un trou et le graphe associé.*

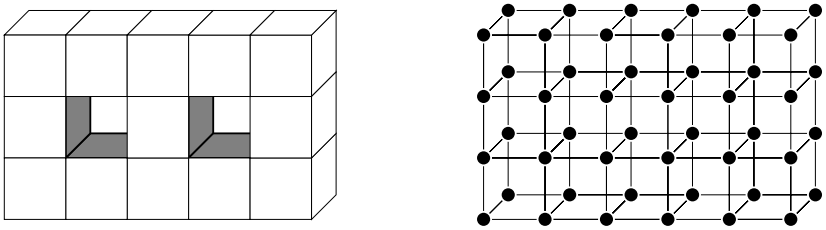


FIG. 3.4 – *Objet avec deux trous et le graphe associé.*

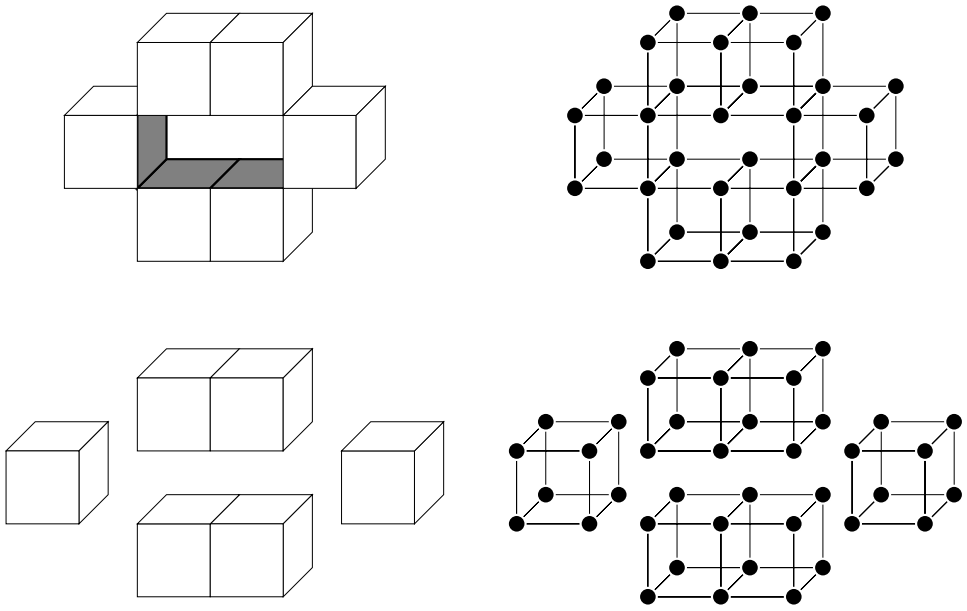


FIG. 3.5 – *Objet constitué d'une composante connexe et d'un trou (cas $(26,6)$) ou constitué de quatre composantes connexes (cas $(6,26)$).*

3.4.4 Objet avec une composante connexe et un trou (cas $(26,6)$) ou quatre composantes connexes (cas $(6,26)$)

Considérons l'objet B de la figure 3.5 (cas $(26,6)$ en haut, et cas $(6,26)$ en bas).

Pour le cas $(26,6)$, l'objet comporte une composante connexe et un trou et ne présente pas de cavité. On a $O(B)=1$, $H(B)=1$, $C(B)=0$, alors $G(B)=0$ (équ. (3.2)). On a $\#[2]=6$, $\#[3]=2$, $\#[4]=4$. On a $\varphi_1=6$, $\varphi_2=6$, alors $G_{26}(B)=0$ (équ. (3.5)).

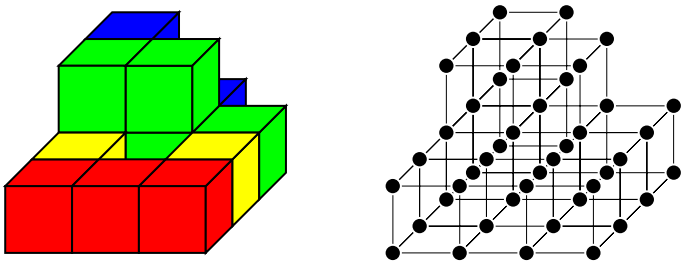
Pour le cas $(6,26)$, l'objet B comporte quatre composantes connexes et ne présente ni trou ni cavité. On a $O(B)=4$, $H(B)=0$, $C(B)=0$, alors $G(B)=4$ (équ. (3.2)). On a $\#[2]=6$, $\#[3]=2$. On a $\psi_1=6$, $\psi_2=2$ alors $G_6(B)=4$ (équ. (3.4)).

3.4.5 Objet avec une composante connexe et un trou

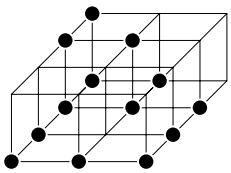
Considérons l'objet B de la figure 3.6. Cet objet comporte une composante connexe et un trou et ne présente pas de cavité. On a $O(B)=1$, $H(B)=1$, $C(B)=0$, alors $G(B)=0$ (équ. (3.2)).

Pour le cas $(6,26)$, on a $\#[2]=13$, $\#[3]=16$, $\#[9]=3$, $\#[22]=0$. On a $\psi_1=13$, $\psi_2=16$, $\psi_3=3$ alors $G_6(B)=0$ (équ. (3.4)).

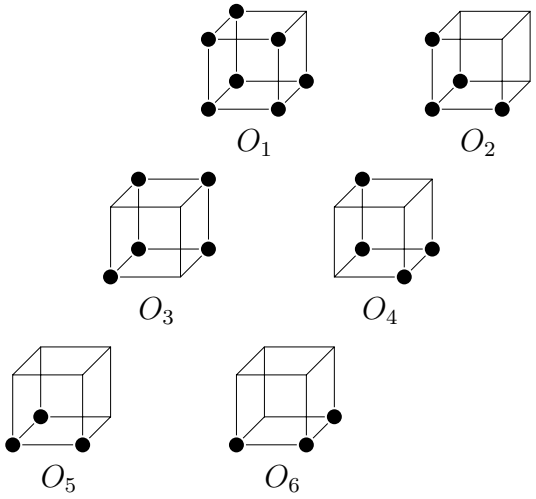
Pour le cas $(26,6)$, nous effectuons une décomposition en octants (Fig. 3.6(c)) ; le nombre de motifs est donné au tableau 3.1. Il suffit ensuite de sommer le nombre de motifs dans chaque octant, en retirant ceux qui sont redondants, *i.e.* ceux qui sont comptés plusieurs fois dans différents octants ; ou plus simplement de recompter de tels motifs – motifs $2,3,4,6$ et 9 – dans



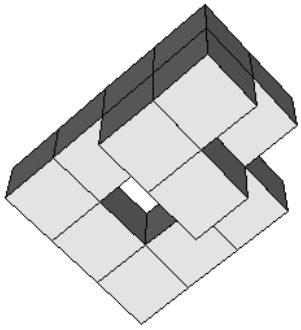
(a) Objet initial et le graphe associé



(b) Vision en octants



(c) Décomposition en octants



(d) Objet en perspective

FIG. 3.6 – *Décomposition en octants d'un objet constitué d'une composante connexe et d'un trou (pour le calcul du genre).*

$Nombre\,de\,motifs$	$Octants$						$Objet\,global$
	O_1	O_2	O_3	O_4	O_5	O_6	
#[2]	7	4	5	4	3	3	13
#[3]	9	3	5	3	2	2	16
#[4]	9	3	4	2	1	1	16
#[5]	3	0	1	1	0	0	5
#[6]	15	3	6	2	1	1	22
#[7]	15	0	3	2	0	0	20
#[8]	5	1	1	0	0	0	7
#[9]	3	0	1	0	0	0	3
#[10]	4	1	1	0	0	0	6
#[11]	12	0	2	1	0	0	15
#[12]	12	0	1	0	0	0	13
#[13]	3	0	0	0	0	0	3
#[14]	1	0	0	0	0	0	1
#[15]	3	0	0	0	0	0	3
#[16]	9	0	0	0	0	0	9
#[17]	9	0	1	0	0	0	10
#[18]	1	0	0	0	0	0	1
#[19]	3	0	0	0	0	0	3
#[20]	3	0	0	0	0	0	3
#[21]	1	0	0	0	0	0	1
#[22]	0	0	0	0	0	0	0

TAB. 3.1 – *Nombre de motifs dans les octants de la figure 3.6(c) et dans l'objet de la figure 3.6(b).*

l'objet global (Fig. 3.6(b)).

On obtient $\varphi_1 = 13$, $\varphi_2 = 37$, $\varphi_3 = 49$, $\varphi_4 = 41$, $\varphi_5 = 22$, $\varphi_6 = 7$, $\varphi_7 = 1$, $\varphi_8 = 0$, alors $G_{26}(B) = 0$ (équ. (3.5)).

3.4.6 Objet avec une composante connexe et une cavité

Considérons l'objet B de la figure 3.7. Cet objet est constitué d'une composante connexe et d'une cavité, et ne comporte pas de trou. On a $O(B) = 1$, $H(B) = 0$, $C(B) = 1$, alors $G(B) = 2$ (équ. (3.2)).

Pour le cas $(6, 26)$, on a $\#[2] = 26$, $\#[3] = 48$, $\#[9] = 24$. On a $\psi_1 = 26$, $\psi_2 = 48$, $\psi_3 = 24$ alors $G_6(B) = 2$ (équ. (3.4)).

En fait, cet objet est constitué de huit octants identiques (à isométries près) ; un tel octant a déjà été examiné dans la section 3.4.5 (octant O_1). De la même façon que pour l'objet global de la section 3.4.5, nous recomptons les motifs $2, 3, 4, 6$ et 9 ; les autres sont alors comptabilisés huit fois par rapport à leur apparition dans l'octant O_1 du tableau 3.1. Les nombres de motifs sont donnés au tableau 3.2.

On obtient $\varphi_1 = 26$, $\varphi_2 = 132$, $\varphi_3 = 268$, $\varphi_4 = 280$, $\varphi_5 = 168$, $\varphi_6 = 56$, $\varphi_7 = 8$, $\varphi_8 = 0$, alors $G_{26}(B) = 2$ (équ. (3.5)).

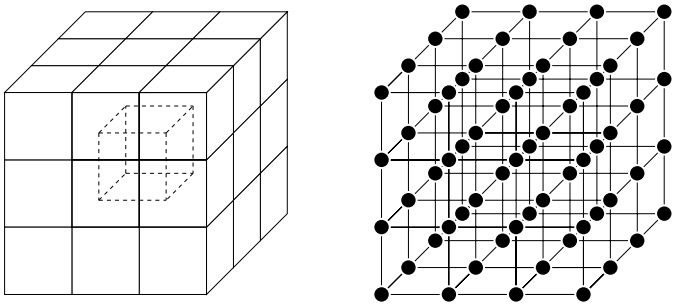


FIG. 3.7 – *Objet avec une cavité et le graphe associé.*

$\#[2]$	$= 26$	$\#[3]$	$= 48$	$\#[4]$	$= 60$	$\#[5]$	$= 24$
$\#[6]$	$= 108$	$\#[7]$	$= 120$	$\#[8]$	$= 40$	$\#[9]$	$= 24$
$\#[10]$	$= 32$	$\#[11]$	$= 96$	$\#[12]$	$= 96$	$\#[13]$	$= 24$
$\#[14]$	$= 8$	$\#[15]$	$= 24$	$\#[16]$	$= 72$	$\#[17]$	$= 72$
$\#[18]$	$= 8$	$\#[19]$	$= 24$	$\#[20]$	$= 24$	$\#[21]$	$= 8$
$\#[22]$	$= 0$						

TAB. 3.2 – *Nombre de motifs dans l'objet de la figure 3.7.*

3.5 Simplicité d'un point

Dans la suite, nous considérons un sous-ensemble S de$\mathcal{Z}^3$.

Définition (point simple en 3D) [Mor81] **:**
En 3D, le point $p \in S$ est *simple* si :

$$O([S \cap N_{26}^*(p)] \cup \{p\}) = O(S \cap N_{26}^*(p)), \tag{3.6}$$

$$O([\overline{S} \cap N_{26}^*(p)] \cup \{p\}) = O(\overline{S} \cap N_{26}^*(p)), \tag{3.7}$$

$$H([S \cap N_{26}^*(p)] \cup \{p\}) = H(S \cap N_{26}^*(p)), \tag{3.8}$$

$$H([\overline{S} \cap N_{26}^*(p)] \cup \{p\}) = H(\overline{S} \cap N_{26}^*(p)), \tag{3.9}$$

avec $O(V(p))$ (resp. $H(V(p))$) étant le nombre de composantes connexes (resp. de trous) de l'ensemble $V(p)$.

En d'autres termes, p est simple si les nombres de composantes connexes et les nombres de trous de l'objet et de son complémentaire, dans $N_{26}^*(p)$, restent inchangés avant et après suppression de p. C'est dans ce sens que l'on dira d'un point qu'il est simple lorsque sa suppression d'un objet *préserve la topologie* de l'objet.

Proposition 1 **:** Pour $p \in S$: $O([S \cap N_{26}^*(p)] \cup \{p\}) = O(S \cap N_{26}^*(p))$ ssi $S \cap N_{26}^*(p)$ a exactement une composante connexe adjacente à p (au sens de S).

Proposition 2 **:** Pour $p \in S$: $O([\overline{S} \cap N_{26}^*(p)] \cup \{p\}) = O(\overline{S} \cap N_{26}^*(p))$ ssi $\overline{S} \cap N_{26}^*(p)$ a exactement une composante connexe adjacente à p (au sens de $\overline{S}$).

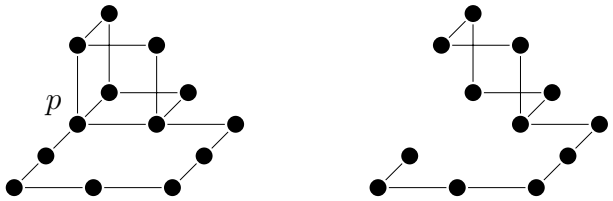
Proposition 3 **:** Pour $p \in S$: si $O([S \cap N_{26}^*(p)] \cup \{p\}) = O(S \cap N_{26}^*(p))$ et $O([\overline{S} \cap N_{26}^*(p)] \cup \{p\}) = O(\overline{S} \cap N_{26}^*(p))$ alors $O(S) = O(S \setminus \{p\})$ et $O(\overline{S}) = O(\overline{S} \cup \{p\})$.

La réciproque est fausse, voir les objets O_4 et O_{10} (cas $(26, 6)$), représentés aux figures 3.10 (d) et 3.10 (j), et le tableau 3.3. Nous avons vu que la réciproque dans le cas 2D est vraie, voir les propositions 3 (page 27) et 5 (page 28), au chapitre 2.

De plus, nous avons la proposition suivante :
Proposition 4 **:** Si p est simple, alors $H(S) = H(S \setminus \{p\})$, $H(\overline{S}) = H(\overline{S} \cup \{p\})$, $C(S) = C(S \setminus \{p\})$ et $C(\overline{S}) = C(\overline{S} \cup \{p\})$. La réciproque est fausse, comme nous le verrons après la propriété suivante.

Avec la définition d'un point simple et les propositions 3 et 4, on a :
Propriété : Si p est simple, alors $O(S) = O(S \setminus \{p\})$, $H(S) = H(S \setminus \{p\})$, $C(S) = C(S \setminus \{p\})$, $O(\overline{S}) = O(\overline{S} \cup \{p\})$, $H(\overline{S}) = H(\overline{S} \cup \{p\})$ et $C(\overline{S}) = C(\overline{S} \cup \{p\})$.

La réciproque est fausse. En effet, considérons l'objet S de la figure 3.8 (qui correspond à l'objet de la figure 3.6, précédemment étudié à la section 3.4.5). Avec la connexité $(6, 26)$,



(a) (b)

FIG. 3.8 – (a) S, (b) $S \setminus \{p\}$.

nous avons $O(S) = O(S \setminus \{p\}) = 1$, $O(\overline{S}) = O(\overline{S} \cup \{p\}) = 1$, $H(S) = H(S \setminus \{p\}) = 1$, $H(\overline{S}) = H(\overline{S} \cup \{p\}) = 1$, $C(S) = C(S \setminus \{p\}) = 0$ et $C(\overline{S}) = C(\overline{S} \cup \{p\}) = 1$.

Nous avons : $O([S \cap N^*_{26}(p)] \cup \{p\}) = 1$, $O(S \cap N^*_{26}(p)) = 2$, $O([\overline{S} \cap N^*_{26}(p)] \cup \{p\}) = 1$, $O(\overline{S} \cap N^*_{26}(p)) = 2$, $H([S \cap N^*_{26}(p)] \cup \{p\}) = 0$, $H(S \cap N^*_{26}(p)) = 1$, $H([\overline{S} \cap N^*_{26}(p)] \cup \{p\}) = 0$, $H(\overline{S} \cap N^*_{26}(p)) = 1$; la définition d'un point simple n'est alors pas vérifiée. Ces valeurs proviennent de l'objet O_9 de la figure 3.10 (i) et du tableau 3.3, O_9 correspondant au voisinage $N^*_{26}(p) \cup \{p\}$ de l'objet de la figure 3.8. En fait, nous pouvons remarquer dans cet exemple que le trou de $S \setminus \{p\}$ n'est pas le même que celui de S.

Note : Nous soulignons alors le fait que, dans le cas 3D, la préservation dans $\mathcal{Z}^3$ des nombres de composantes connexes et de trous de l'objet et de son complémentaire, avant et après suppression du point p, est insuffisante à caractériser un point simple. La définition d'un point simple p exige la préservation des nombres de composantes connexes et de trous, de l'objet et de son complémentaire, avant et après suppression du point, dans le voisinage $N^*_{26}(p)$. À titre de comparaison, nous rappelons l'équivalence entre la simplicité d'un point en 2D et la préservation des nombres de composantes connexes de l'objet et de son complémentaire, avant et après suppression du point, cela soit dans $N^*_8(p)$ soit dans $\mathcal{Z}^2$.

T.-C. Lee, R.L. Kashyap et C.-N. Chu [LKC94] ont proposé une caractérisation de point simple à l'aide du genre, suite à une mauvaise caractérisation d'un point simple, proposée par S. Lobregt, W. Verbeek et F.C.A. Groen [LVG80]. Ces derniers disent qu'un point p est un point simple si c'est un point de bord et que sa suppression préserve la caractéristique d'Euler, ce qui implique que $\delta G(S \cap N^*_{26}(p)) = 0$ et $\delta C(S \cap N^*_{26}(p)) = 0$ (comme v est un point de bord, il ne peut y avoir de création ou de suppression de cavité), avec $\delta Q(V(p)) = Q(V(p)) - Q(V(p) \cup \{p\})$. Elle implique alors $\delta O(S \cap N^*_{26}(p)) = \delta H(S \cap N^*_{26}(p))$. Néanmoins, il est possible que des points vérifient les égalités précédentes, et soient tels que les nombres d'objets et de trous ne soient pas les mêmes, et par conséquent, de tels points ne sont pas simples selon la définition de Morgenthaler. C'est le cas pour l'objet de la figure 3.9 ; le point p est un point de bord ($\delta C(S \cap N^*_{26}(p)) = 0$) et la suppression de p crée un trou dans $S \cap N^*_{26}(p)$ (cas $(26, 6)$) ($H([S \cap N^*_{26}(p)] \cup \{p\}) = 0$, $H(S \cap N^*_{26}(p)) = 1$ et $\delta H(S \cap N^*_{26}(p)) = 1$) et également un objet supplémentaire dans $S \cap N^*_{26}(p)$ ($O([S \cap N^*_{26}(p)] \cup \{p\}) = 1$, $O(S \cap N^*_{26}(p)) = 2$, et $\delta O(S \cap N^*_{26}(p)) = 1$) ; cela entraine que la caractéristique d'Euler ne change pas dans $S \cap N^*_{26}(p)$

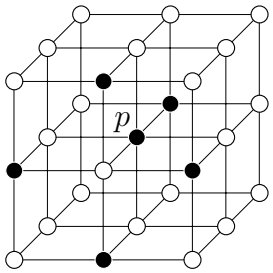


FIG. 3.9 – *Point non* 26*-simple, mais caractérisé comme tel par Lobregt, Verbeek et Groen.*

(*i.e.* $\delta G(S \cap N_{26}^*(p)) = 0$) ; mais le point p n'est pas simple selon la définition de Morgenthaler.

Nous donnons maintenant la proposition liant un point simple au genre et utilisant les remarques précédentes.

Proposition [LKC94] : Un point de bord p est un point simple si et seulement si $\delta G(S \cap N_{26}^*(p)) = 0$ et $[\delta H(S \cap N_{26}^*(p)) = 0$ ou $\delta O(S \cap N_{26}^*(p)) = 0]$, avec $\delta Q(V(p)) = Q(V(p)) - Q(V(p) \cup \{p\})$ pour un voisinage $V(p)$ du point p.

Nous avons le théorème suivant (voir [Ma94] et [KR89]) :

Théorème : Le point p est un point simple si et seulement si toutes les conditions suivantes sont vérifiées :

- (C_1) $S \cap N_{26}^*(p)$ a exactement une composante connexe adjacente à p (au sens de S) (cf. la définition et la proposition 1),
- (C_2) $\overline{S} \cap N_{26}^*(p)$ a exactement une composante connexe adjacente à p (au sens de $\overline{S}$) (cf. la définition et la proposition 2),
- (C_3) $\delta G(S \cap N_{26}^*(p)) = 0$.

On note que (C_2) implique que le point p est un point de bord.

Selon [KR89], pour la connexité $(26, 6)$, les conditions (C_1) et (C_3) impliquent ensemble la condition (C_2) ; pour la connexité $(6, 26)$, les conditions (C_2) et (C_3) impliquent ensemble la condition (C_1).

Rappelons que les deux premières conditions sont nécessaires et suffisantes dans le cas 2D, en remplaçant $N_{26}^*(p)$ par $N_8^*(p)$ (voir section 2.4) ; tandis que les deux premières conditions sont insuffisantes dans le cas 3D, voir l'objet O_1 à la figure 3.11 et le tableau 3.4, page 59.

Nous donnons quelques calculs sur quelques objets des différentes quantités passées en revue dans cette section, à la figure 3.10 et au tableau 3.3.

À la section suivante, nous introduisons les nombres topologiques 3D. Nous verrons que le test de la simplicité d'un point s'effectue beaucoup plus facilement avec ces nombres, qu'avec l'utilisation de la définition ou des propriétés données dans cette section ; néanmoins celles-ci permettent d'illustrer la difficulté intrinsèque de la topologie en 3D à travers sa préservation.

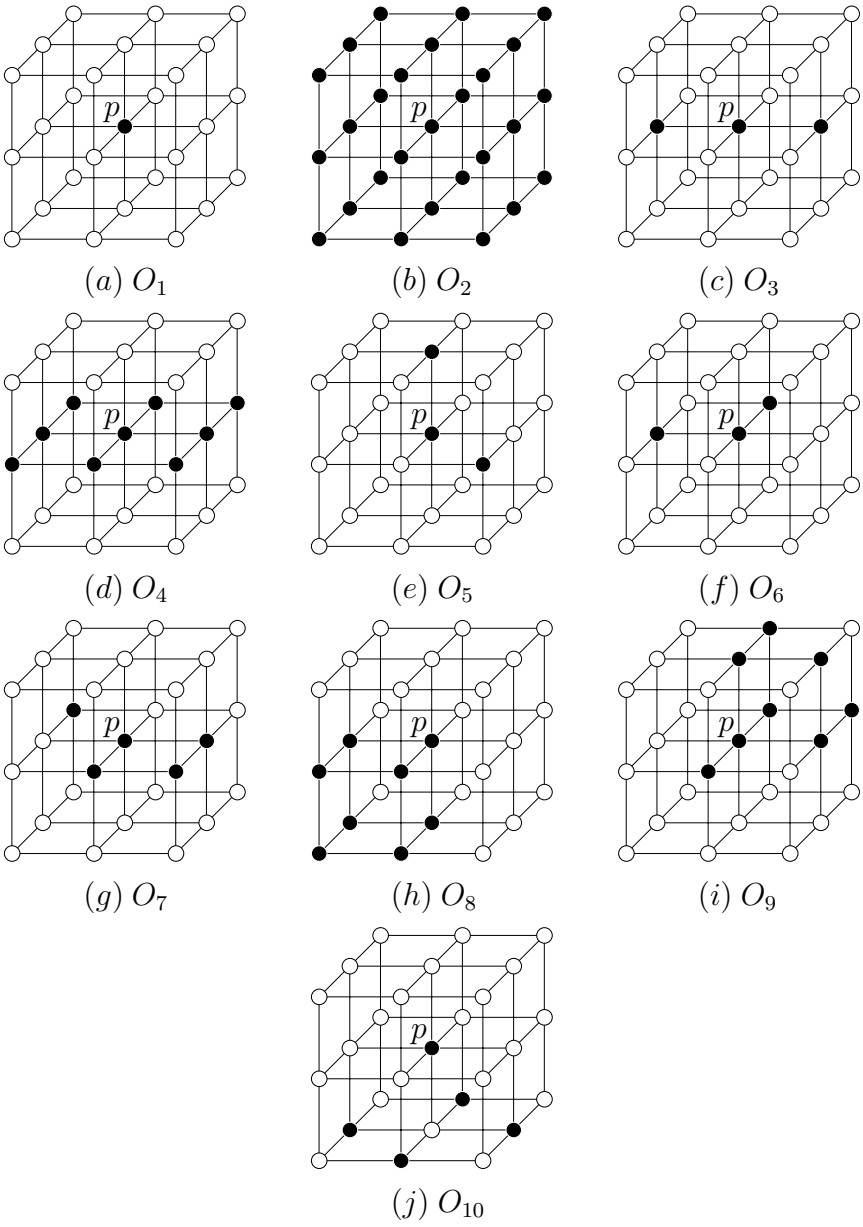


FIG. 3.10 – *Différents objets - calcul du genre.*

Quantités	O_1	O_2	O_3	O_4	O_5		O_6		O_7		O_8	O_9		O_{10}	
					$(26,6)$	$(6,26)$	$(26,6)$	$(6,26)$	$(26,6)$	$(6,26)$		$(26,6)$	$(6,26)$	$(26,6)$	$(6,26)$
$O([S \cap N_{26}^*(p)] \cup \{p\})$	1	1	1	1	1	2	1	1	1	2	1	1	1	1	5
$O(S \cap N_{26}^*(p))$	0	1	2	1	1	2	1	2	2	2	1	1	2	1	4
$O([\overline{S} \cap N_{26}^*(p)] \cup \{p\})$	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1
$O(\overline{S} \cap N_{26}^*(p))$	1	0	1	2	1	1	1	1	1	1	1	2	2	2	1
$H([S \cap N_{26}^*(p)] \cup \{p\})$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$H(S \cap N_{26}^*(p))$	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0
$H([\overline{S} \cap N_{26}^*(p)] \cup \{p\})$	0	0	0	0	1	0	1	0	0	0	0	1	0	0	0
$H(\overline{S} \cap N_{26}^*(p))$	0	0	1	0	1	0	1	1	1	0	0	1	1	0	0
$O(S)$	1	1	1	1	1	2	1	1	1	2	1	1	1	1	5
$O(S \setminus \{p\})$	0	1	2	1	1	2	1	2	2	2	1	1	2	1	4
$O(\overline{S})$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
$O(\overline{S} \cup \{p\})$	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1
$H(S)$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$H(S \setminus \{p\})$	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0
$H(\overline{S})$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$H(\overline{S} \cup \{p\})$	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0
$C(S)$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$C(S \setminus \{p\})$	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
$C(\overline{S})$	1	1	1	1	1	2	1	1	1	2	1	1	1	1	5
$C(\overline{S} \cup \{p\})$	0	1	2	1	1	2	1	2	2	2	1	1	2	1	4
$G(S)$	1	1	1	1	1	2	1	1	1	2	1	1	1	1	5
$\delta H(S \cap N_{26}^*(p))$	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0
$\delta O(S \cap N_{26}^*(p))$	-1	0	1	0	0	0	0	1	1	0	0	0	1	0	-1
$\delta C(S \cap N_{26}^*(p))$	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
$\delta G(S \cap N_{26}^*(p))$	-1	1	1	-1	0	0	0	1	1	0	0	0	0	-1	-1
$T_m(p, S)$	0	1	2	1	1	1	1	2	2	1	1	1	2	1	0
$T_n(p, \overline{S})$	1	0	1	2	1	1	1	1	1	1	1	1	2	2	1
Point simple ?	non	non	non	non	oui	oui	oui	non	non	oui	oui	oui	non	non	non

TAB. 3.3 – *Différentes quantités pour le calcul du genre des exemples de la figure 3.10 (hypothèse :* $[\mathcal{Z}^3 \setminus N_{26}(p)] \cap S = \emptyset$*. Notation :* $\delta Q(V(p)) = Q(V(p)) - Q(V(p) \cup \{p\})$*.*

3.6 Caractérisations en pratique d'un point simple

Dans cette section, nous rappelons la définition des nombres topologiques en 3D et donnons la caractérisation d'un point simple par ces nombres. La caractérisation à l'aide des BDD n'est pas explicitée ; c'est le même principe qu'en 2D (voir le chapitre 2 ainsi que l'annexe C).

3.6.1 Nombres topologiques

G. Bertrand et G. Malandain [BM94] ont introduit les nombres topologiques. Ces derniers permettent de caractériser la simplicité ou non d'un point p dans $N^*_{26}(p)$, sans calculer explicitement ni le nombre de trous (selon la définition de Morgenthaler, page 53) ni le genre (voir la condition (C_3), page 55). Le calcul des nombres topologiques n'exige que le calcul des nombres de composantes connexes de l'objet et de son complémentaire, adjacentes au point considéré et dans certains voisinages.

Définition (nombres topologiques en 3D) :
Soient $S \subset \mathcal{Z}^3$ et $p \in S$. Notons $NCC(T)$ le nombre de composantes connexes de T, et $NCC^p(T)$ le nombre de composantes connexes de T et adjacentes au point p. Le *nombre topologique utilisant la* 26*-connexité* est le nombre $T_{26}(p,S) = NCC^p[S \cap N^*_{26}(p)] = NCC[S \cap N^*_{26}(p)]$. Le *nombre topologique utilisant la* 6*-connexité* est le nombre $T_6(p,S) = NCC^p[S \cap N^*_{18}(p)]$.

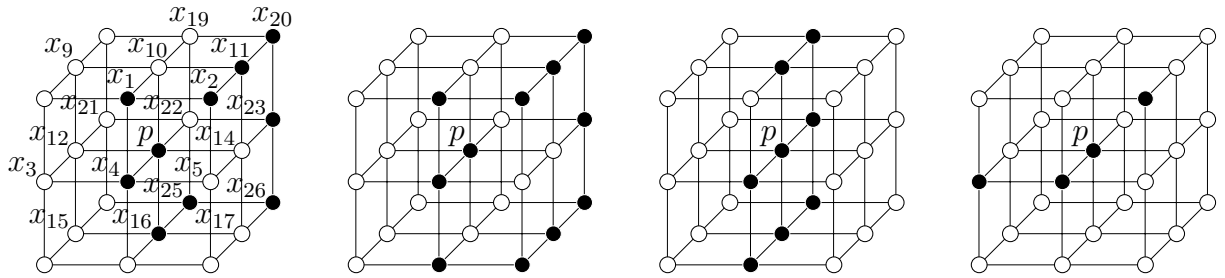
La complexité des calculs des nombres topologiques est abordée à la section 3.6.1.1.

Proposition (caractérisation d'un point simple par les nombres topologiques) :
Soient $S \subset \mathcal{Z}^3$ et $p \in S$. Le point p est un point 26-simple $\Leftrightarrow T_{26}(p,S) = 1$ et $T_6(p,\overline{S}) = 1$. Le point p est un point 6-simple $\Leftrightarrow T_6(p,S) = 1$ et $T_{26}(p,\overline{S}) = 1$.

Il est important de noter qu'il faut utiliser $N^*_{18}(p)$ dans le cas de T_6 et non $N^*_{26}(p)$, sinon des erreurs peuvent se produire (voir [LKC94] et [BM95]).

Nous illustrons maintenant la différence entre les voisinages utilisés et les trois conditions C_i du théorème de la page 55 (voir la figure 3.11 et le tableau 3.4). Par exemple, la suppression du point p crée un trou pour l'objet O_1 (Fig.3.11 (a)) avec la connexité $(26,6)$, en détruit un pour le même objet avec la connexité $(6,26)$, ou en crée un pour l'objet O_3 (Fig.3.11 (c)) avec la connexité $(26,6)$ (dans cet objet, on note que p est adjacent à 2 composantes connexes de $\overline{S}$ dans $N^*_{18}(p)$, d'où $T_6(p,\overline{S}) = 2$).

Commentaires à propos de la figure 3.11 :
Nous donnons maintenant quelques explications concernant la détection du trou de l'objet O_1, à la figure 3.11 (a). Selon la connexité $(26,6)$, un trou dans $S \cap N^*_{26}(p)$ est détecté par le 26-chemin fermé $x_1, x_2, x_{11}, x_{20}, x_{23}, x_{26}, x_{25}, x_{16}, x_4, x_1$ que l'on peut déformer en le 26-chemin fermé (non contractable en un point) : $x_4, x_{11}, x_{23}, x_{16}, x_4$. Selon la connexité $(6,26)$, un trou



(a) O_1 (b) O_2 (c) O_3 (d) O_4

FIG. 3.11 – *Exemples d'objets, caractérisation de la simplicité d'un point par les nombres topologiques.*

	O_1		O_2		O_3		O_4	
(m,n)	$(26,6)$	$(6,26)$	$(26,6)$	$(6,26)$	$(26,6)$	$(6,26)$	$(26,6)$	$(6,26)$
$T_m(p,S)$	1	2	1	1	1	1	1	1
$T_n(p,\overline{S})$	2	1	2	1	2	1	1	1
$O([S\cap N^*_{26}(p)]\cup\{p\})$	1	1	1	1	1	1	1	2
$O(S\cap N^*_{26}(p))$	1	1	1	1	1	1	1	2
$C([S\cap N^*_{26}(p)]\cup\{p\})$	0	0	0	0	0	0	0	0
$C(S\cap N^*_{26}(p))$	0	0	0	0	0	0	0	0
$H([S\cap N^*_{26}(p)]\cup\{p\})$	0	1	0	1	0	0	0	0
$H(S\cap N^*_{26}(p))$	1	0	1	1	1	0	0	0
C_1 vérifiée ?	oui	oui	oui	oui	oui	oui	oui	oui
C_2 vérifiée ?	oui	oui	non	oui	oui	oui	oui	oui
$\delta O(S\cap N^*_{26}(p))$	0	0	0	0	0	0	0	0
$\delta H(S\cap N^*_{26}(p))$	1	-1	1	0	1	0	0	0
$\delta C(S\cap N^*_{26}(p))$	0	0	0	0	0	0	0	0
$\delta G(S\cap N^*_{26}(p))$	-1	1	-1	0	-1	0	0	0
C_3 vérifiée ?	non	non	non	oui	non	oui	oui	oui
Point simple ?	non	non	non	oui	non	oui	oui	oui

TAB. 3.4 – *Calculs des nombres topologiques sur les exemples de la figure 3.11.*

dans $S\cap N^*_{26}(p)\cup\{p\}$ est détecté par le 6-chemin fermé $x_1, x_4, p, x_{16}, x_{25}, x_{26}, x_{23}, x_{20}, x_{11}, x_2, x_1$ (non réductible) ; il disparaît lorsque p est supprimé.

Détaillons quelque peu le calcul des nombres topologiques sur cette même figure. Selon la connexité $(6,26)$, $S = \{x_1, x_2, x_4, x_{11}, p, x_{16}, x_{20}, x_{23}, x_{25}, x_{26}\}$ et $\overline{S}\cap N^*_{18}(p)$ est constitué de deux composantes 6-connexes et 6-adjacentes à p : $\{x_3, x_9, x_{10}, x_{12}, x_{15}, x_{19}, x_{21}, x_{22}\}$ et

$\{x_5, x_{14}, x_{17}\}$. Ainsi $T_{26}(p, \overline{S}) = 2$; tandis qu'il n'y a qu'une composante 6-connexe et 6-adjacente à p dans $\overline{S} \cap N^*_{26}(p)$ (la condition (C_2) du théorème de la page 55 est vérifiée). Selon la connexité $(6, 26)$, $S \cap N^*_{18}(p)$ est constitué de quatre composantes 6-connexes $\{x_1, x_4\}$, $\{x_{16}, x_{25}\}$, $\{x_{11}\}$, $\{x_{23}\}$; seules les deux premières sont 6-adjacentes à p. Ainsi $T_6(p, S) = 2$; tandis qu'il n'y a qu'une composante 6-connexe et 6-adjacente à p dans $S \cap N^*_{26}(p)$ (la condition (C_1) du théorème de la page 55 est vérifiée).

Remarque (configurations non simples sans raison commune de non-simplicité globale mais avec les mêmes nombres topologiques) : La non-simplicité d'un point, selon les nombres topologiques, peut ne pas nous renseigner sur la raison de non-simplicité (en global) pour l'objet (comme dans le cas 2D, voir les remarques des sections 2.4 et 2.5.3). Considérons l'objet de la figure 3.12 (a), la suppression de p déconnecte l'objet S. Considérons l'objet de la figure 3.12 (b), la suppression de p détruit un trou de S et un trou de $\overline{S}$. On a alors deux raisons globales différentes de non-simplicité du point p, tout en ayant la même configuration locale, décrite à la figure 3.12 (c) ($T_6(p, S) = 2$ et $T_{26}(p, \overline{S}) = 1$).

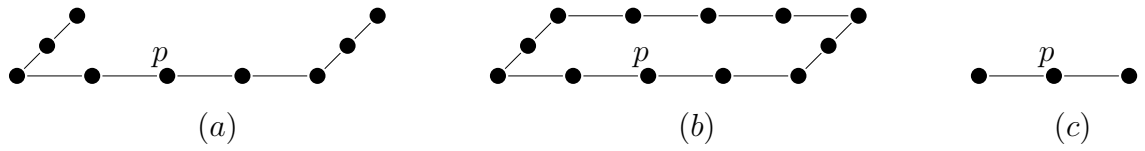


FIG. 3.12 – *Configuration de point dont la non-simplicité ne peut être décidée localement. Si l'on supprime le point p, (a) l'objet est déconnecté en deux, (b) un trou de S est supprimé , (c) configuration commune locale de non-simplicité du point.*

Remarque (configurations non simples sans raison commune de non-simplicité locale mais avec les mêmes nombres topologiques) : Considérons les figures 3.13 (a) et 3.13 (b), selon la connexité $(6, 26)$; nous avons $T_6(p, S) = 2$ et $T_{26}(p, \overline{S}) = 1$. La suppression de p dans la figure 3.13 (a) détruit un trou de S et un trou de $\overline{S}$ dans $N^*_{26}(p)$. La suppression de p dans la figure 3.13 (b) provoque la création d'une nouvelle composante de S dans $N^*_{26}(p)$ et la perte d'un trou de $\overline{S}$ dans $N^*_{26}(p)$. En conclusion, nous n'avons pas forcément la même raison de non-simplicité (locale) pour des configurations telles que les couples de nombres topologiques soient égaux.

3.6.1.1 Voisinages géodésiques

Nous introduisons maintenant le concept de voisinage géodésique. Un voisinage de ce type peut être utilisé pour exprimer de façon différente les nombres topologiques. De tels voisinages peuvent être également utilisés lors de la définition des points P-simples (voir chapitre 5).

Définitions (voisinage géodésique) [Ber94] **:**
Soient $X \subset \mathcal{Z}^3$ et $x \in \mathcal{Z}^3$. Le *n-voisinage géodésique de x à l'intérieur de X d'ordre k* ($k \geq 1$)

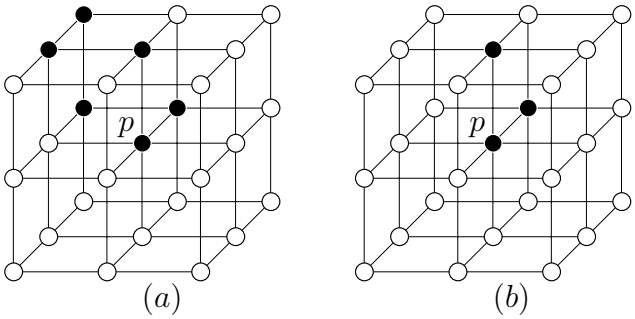


FIG. 3.13 – *Configurations de points non simples et ayant le même couple de nombres topologiques, mais pas la même raison de non-simplicité (locale).*

est l'ensemble $N_n^k(x, X)$ défini récursivement par $N_n^1(x, X) = N_n^*(x) \cap X$ pour $k = 1$ et par $N_n^k(x, X) = \cup\{N_n(y) \cap N_{26}^*(x) \cap X, y \in N_n^{k-1}(x, X)\}$ pour $k > 1$.

En d'autres termes, $N_n^k(x, X)$ est l'ensemble composé de tous les points y de $N_{26}^*(x) \cap X$ tel qu'il existe un n-chemin π de x à y de longueur inférieure ou égale à k, pour lequel tous les points de π, excepté x, appartiennent à $N_{26}^*(x) \cap X$. On définit $G_6(x, X) = N_6^2(x, X)$ et $G_{26}(x, X) = N_{26}^1(x, X)$.

Propriété :
Nous avons $T_6(x, X) = \#[G_6(x, X)]$ et $T_{26}(x, X) = \#[G_{26}(x, X)]$, où $\#S$ est le nombre d'éléments de l'ensemble S.

En utilisant des algorithmes de parcours de composantes connexes, les calculs de $T_m(x, X)$ et $T_n(x, \overline{X})$ peuvent être effectués en temps linéaire, *i.e.* respectivement en $\mathcal{O}(k)$ et $\mathcal{O}(\overline{k})$ avec $k = \#[G_m(x, X)]$ et $\overline{k} = \#[G_n(x, \overline{X})]$.

3.7 Classification topologique

Nous avons récapitulé les nombres de configurations ayant un couple donné de nombres topologiques selon les connexités $(m, n) = (6, 26)$ ou $(26, 6)$, respectivement dans les tableaux 3.5 et 3.6, configurations parmi les $67\,108\,864 (= 2^{26})$ possibles dans $N_{26}^*(p)$ d'un point p. Dans ces tableaux, on note $T = T_m(p, S)$ et $\overline{T} = T_n(p, \overline{S})$. Nous vérifions que le nombre de couples tels que $T_m(p, S) = k_1$ et $T_n(p, \overline{S}) = k_2$ est égal au nombre de couples tels que $T_n(p, S) = k_2$ et $T_m(p, \overline{S}) = k_1$, en raison des propriétés obtenues par le passage au complémentaire. Dans ces tableaux, on peut voir que le nombre de points 6-simples (égal au nombre de points 26-simples, car caractérisés par des nombres topologiques de même valeur) est de $25\,985\,144$, parmi les $67\,108\,864$ configurations possibles.

Les nombres topologiques sont utilisés pour décrire les caractéristiques topologiques locales des points dans l'espace discret (voir le tableau 3.7 et les exemples de la figure 3.14). Les nombres topologiques sont également utilisés lors du calcul de l'enveloppe topologique durant le processus de bouchage de trous (détection des points intérieurs et des isthmes 2D) [ABP96] [Akt97],

$(m,n)=(6,26)$	$\overline{T}=0$	$\overline{T}=1$	$\overline{T}=2$	$\overline{T}=3$	$\overline{T}=4$	$\overline{T}=5$	$\overline{T}=6$	$\overline{T}=7$	$\overline{T}=8$
$T=0$	0	1 048 576	0	0	0	0	0	0	0
$T=1$	1	25 985 144	3 626 396	651 784	106 054	13 512	1 180	56	1
$T=2$	0	26 073 184	904 832	49 216	2 688	96	0	0	0
$T=3$	0	7 748 352	63 744	768	0	0	0	0	0
$T=4$	0	804 352	1 024	0	0	0	0	0	0
$T=5$	0	27 648	0	0	0	0	0	0	0
$T=6$	0	256	0	0	0	0	0	0	0

TAB. 3.5 – *Nombres de configurations ayant pour nombres topologiques le couple* $(T,\overline{T})$ *selon la connexité* $(m,n)=(6,26)$.

$(m,n)=(26,6)$	$\overline{T}=0$	$\overline{T}=1$	$\overline{T}=2$	$\overline{T}=3$	$\overline{T}=4$	$\overline{T}=5$	$\overline{T}=6$
$T=0$	0	1	0	0	0	0	0
$T=1$	1 048 576	25 985 144	26 073 184	7 748 352	804 352	27 648	256
$T=2$	0	3 626 396	904 832	63 744	1 024	0	0
$T=3$	0	651 784	49 216	768	0	0	0
$T=4$	0	106 054	2 688	0	0	0	0
$T=5$	0	13 512	96	0	0	0	0
$T=6$	0	1 180	0	0	0	0	0
$T=7$	0	56	0	0	0	0	0
$T=8$	0	1	0	0	0	0	0

TAB. 3.6 – *Nombres de configurations ayant pour nombres topologiques le couple* $(T,\overline{T})$ *selon la connexité* $(m,n)=(26,6)$.

ou pour la détection de jonctions de courbes ou de surfaces, ou lors de la segmentation topologique de surfaces discrètes [MBA93]. Ils sont également utilisés dans le cadre de la topologie des coupes, sur les images 2D en niveaux de gris [BEC97] [Eve97].

3.8 Suppression en parallèle de points

Nous donnons ici les définitions d'ensemble simple, d'ensemble minimal non simple, déjà abordées dans le chapitre précédent, dans le cas 2D.

Nous rappelons le théorème proposé par C.M. Ma donnant des conditions suffisantes qu'un algorithme de suppression parallèle de points doit vérifier afin de préserver la topologie.

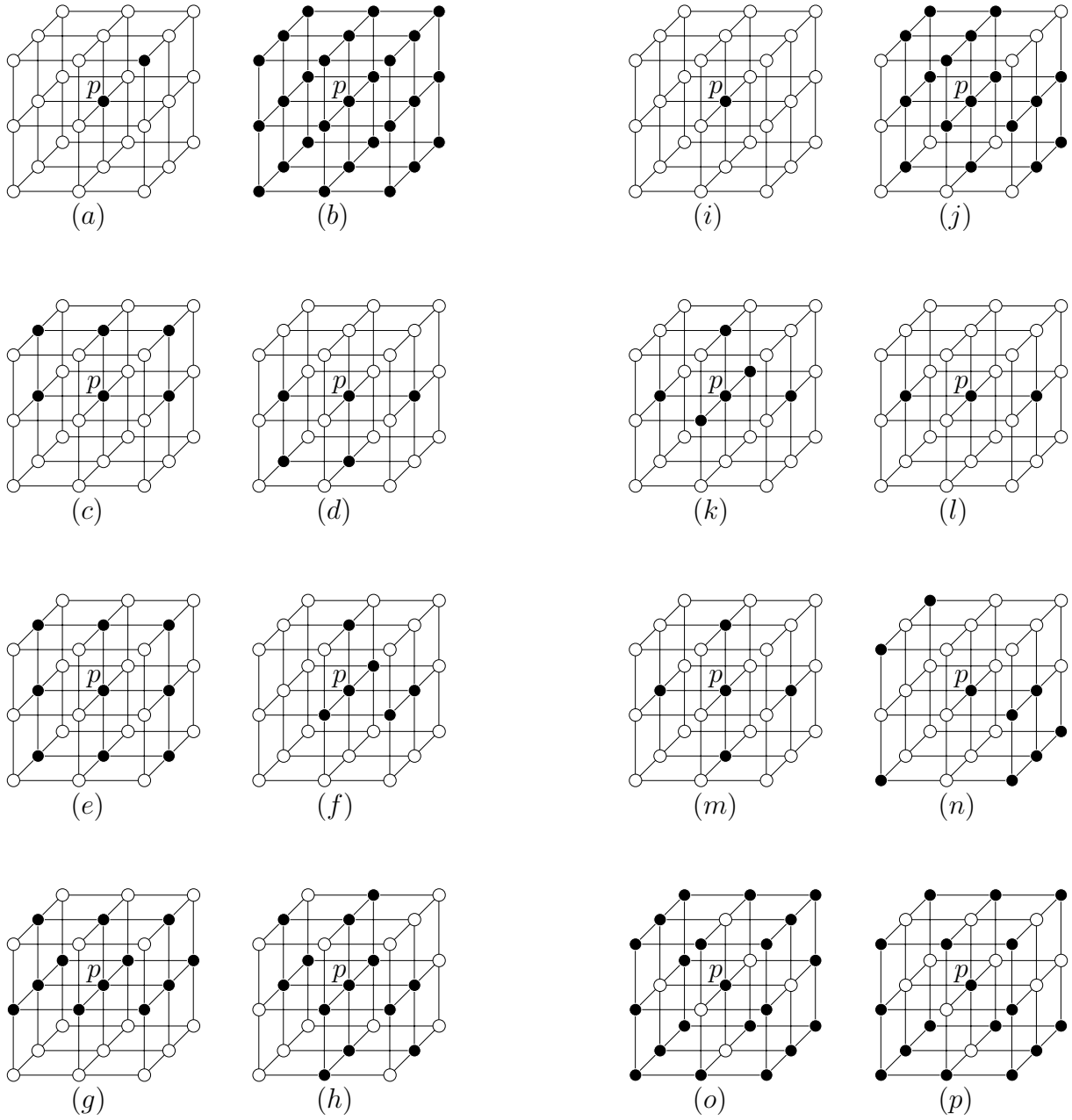


FIG. 3.14 – *Classification de points selon les nombres topologiques, voir le tableau 3.7.*

De nombreuses personnes proposant des algorithmes parallèles de squelettisation utilisent ce théorème pour montrer que l'algorithme qu'elles proposent préserve bien la topologie. Néanmoins ce théorème est difficile à utiliser car il impose de passer en revue de nombreuses configurations. Dans la seconde partie de ce manuscrit, nous proposerons des algorithmes parallèles de suppression de points P-simples (chap. 6). La définition même d'un point P-simple (chap. 5) nous assure que la topologie est bien préservée ; aucune preuve supplémentaire n'est exigée,

Valeurs des nombres topologiques	désignation	$m = 6$ (Fig. 3.14)	$m = 26$ (Fig. 3.14)
$T = 0$	*point isolé*	(a)	(i)
$\overline{T} = 0$	*point intérieur*	(b)	(j)
$\overline{T} \neq 0$	*point de bord*		
$T = 1$ et $\overline{T} = 1$	*point simple*	(c)	(k)
$T = 2$ et $\overline{T} = 1$	*isthme 1D simple* ou *point de courbe*	(d)	(l)
$T = 1$ et $\overline{T} = 2$	*isthme 2D simple* ou *point de surface*	(e)	(m)
$T \geq 3$ et $\overline{T} = 1$	*intersection 1D simple*	(f) ($T = 3$)	(n) ($T = 4$)
$T = 1$ et $\overline{T} \geq 3$	*intersection 2D simple*	(g) ($\overline{T} = 3$)	(o) ($\overline{T} = 6$)
$T \geq 2$ et $\overline{T} \geq 2$	*isthme multiple*	(h) ($T = 2$ et $\overline{T} = 3$)	(p) ($T = 2$ et $\overline{T} = 3$)

TAB. 3.7 – *Classification topologique et exemples à la figure 3.14.*

contrairement aux autres algorithmes déjà proposés, qui utilisent le théorème ci-après [PK98a] [PK99] [Ma95] [MS96] [MBPL99c].

Définition (ensemble n-simple) : Soit $X \subset \mathcal{Z}^3$. Un ensemble $S \subset X$ est un *ensemble n-simple* de X si les points de S peuvent être arrangés selon une séquence $S = \{x_1, \ldots, x_k\}$ de telle façon que x_1 soit n-simple pour X et x_i soit n-simple pour l'ensemble $X \setminus \{x_1, \ldots, x_{i-1}\}$, pour $i = 2, \ldots, k$. L'ensemble vide est un ensemble simple.

Proposons maintenant un théorème qui est une extension directe de celui donné pour le cas 2D proposé par C. Ronse ou par R.W. Hall (cf. section 2.6.2).

Théorème [Ma94][Ma93] **:**
Soit O un opérateur de réduction parallèle d'images 3D selon la connexité $(26, 6)$. Si pour toutes les $(26, 6)$-images $\mathcal{I}$, toutes les conditions suivantes sont vérifiées lorsque O est appliqué à $\mathcal{I}$, alors O préserve la topologie :

- tout ensemble de points noirs de $\mathcal{I}$ qui est contenu dans un carré unité et qui est supprimé par O, est un ensemble simple de $\mathcal{I}$;
- O ne doit pas supprimer de composante noire de $\mathcal{I}$ qui est contenue dans un cube unité.

D'autres théorèmes sont donnés pour les autres connexités $(18, 6)$, $(6, 26)$ et $(6, 18)$. Rappelons maintenant la définition d'un ensemble minimal non simple.

Définitions (ensemble minimal non n-simple) : Soit $X \subset \mathcal{Z}^3$. Un ensemble $S \subset X$ est un *ensemble minimal non n-simple* (ou *MNS* pour *Minimal Non-simple Set*) s'il n'est pas un ensemble n-simple et si chacun de ses sous-ensembles propres est n-simple (voir également la section 2.6.2).

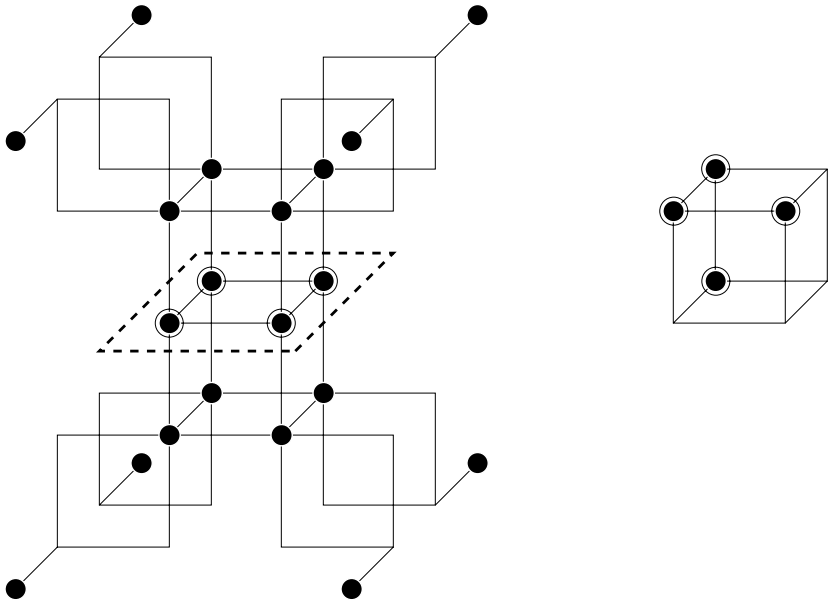


FIG. 3.15 – *Exemples de MNS* 3*D selon la connexité* $(26, 6)$.

À la figure 3.15 (a), est représenté un objet pour lequel sont encerclés les points simples non extrémités, et est encadré en pointillé un 26-MNS. L'algorithme préserve la topologie s'il ne supprime pas à la fois les quatre points simples de cet MNS. La figure 3.15 (b) représente une composante de 4 points 26-simples qui est un 26-MNS.

3.9 Conclusion

Nous avons introduit le concept de genre 3D. Puis, nous avons défini ce qu'est un point simple à travers la préservation du genre et d'autres quantités. Le test de la simplicité d'un point à travers le calcul du genre est difficile à réaliser. Différentes propriétés pour la simplicité en 3D ont été rappelées, et commentées sur des exemples.

Nous avons ensuite introduit les nombres topologiques. Ils permettent de caractériser les points simples de façon plus aisée et plus efficace. Vient ensuite une classification topologique des points de $\mathcal{Z}^3$ par l'utilisation des nombres topologiques, cette classification permet certaines applications d'analyse d'image tels le bouchage de trous et la détection de jonctions de courbes ou de surfaces. Elle est beaucoup plus utilisée actuellement, dans le cas des images 2D en niveaux de gris.

Ce chapitre se termine par les concepts de MNS et d'ensemble simple, et permet de mieux comprendre les problèmes que pose une suppression en parallèle de points.

Chapitre 4

Schémas d'algorithmes de squelettisation

4.1 Introduction

Dans les chapitres précédents, nous avons vu la notion de point simple. Un algorithme de squelettisation est un processus consistant en la suppression de points simples. Si la suppression est réalisée de façon séquentielle alors la topologie est préservée, et cela par la définition même d'un point simple. Comme nous l'avons vu précédemment (cf. sections 2.6 et 3.8), la suppression en parallèle de points simples peut ne pas préserver la topologie. C'est pourquoi différentes stratégies de suppression ont été proposées. Plutôt que de se livrer ici à une étude détaillée de quelques algorithmes de squelettisation existants, nous avons choisi de proposer un chapitre synthétique décrivant les techniques ou schémas utilisés dans de tels algorithmes. Néanmoins, cette étude détaillée a été réalisée et représente une part non négligeable et nécessaire de notre travail ; cette étude est reportée dans les annexes A et B ; cela afin d'amener plus rapidement le lecteur aux deuxième et troisième parties de ce rapport qui témoignent de notre contribution en ce qui concerne les algorithmes de squelettisation selon une approche topologie digitale et en matière d'algorithmes de squelettisation selon une approche topologie discrète.

Dans la deuxième partie de ce mémoire, deux schémas de squelettisation basés sur la suppression de points P-simples seront étudiés. L'un avait déjà été proposé par G. Bertrand (cf. section 5.3) ; le second utilise une nouvelle variante des points P-simples proposée dans ce mémoire (cf. section 6.2) ; ces schémas sont décrits dans la section 6.2.3. Dans la troisième partie, un nouveau schéma générique de squelettisation basé sur la suppression de points α_n-simples, et β_n-simples sera proposé (cf. sections 8.5, 9.5 et 10.6).

4.2 Terminologie concernant l'amincissement ou la squelettisation

4.2.1 Algorithme d'amincissement ou de squelettisation. Noyau homotopique ou squelette

Nous donnons les définitions utilisées par la suite et en reprécisons certaines, vues précédemment.

Un *algorithme d'amincissement* (ou *shrinking algorithm*) consiste en la suppression jusqu'à stabilité de points simples, le résultat obtenu s'appelle un *noyau homotopique*. Si la suppression est réalisée de façon séquentielle alors la topologie est préservée ; cela par la définition même d'un point simple.

Si le processus est modifié de façon à ce que certains points simples soient préservés durant le processus de suppression, il est alors possible de conserver des caractéristiques géométriques. Un tel processus s'appelle *algorithme de squelettisation* (ou *thinning algorithm*), et le résultat est appelé *squelette*. Les points à préserver sont appelés *points terminaux* ou *points extrémités*.

Étant donné qu'une courbe simple n'est constituée que de points non simples à l'exception de ses points extrémités, la préservation de ces derniers (*points extrémités* (ou *terminaux*) *de courbe*) durant un processus de suppression de points simples, préserve les courbes et les parties de courbes. De la même façon en 3D, on pourra préserver des (bouts de) surfaces, en conservant des *points extrémités* (ou *terminaux*) *de surface*. La définition de tels points dépend de l'auteur et du contexte (voir l'annexe B).

Un algorithme de squelettisation préservant des points terminaux de courbe (resp. de surface) est appelé *algorithme de squelettisation curviligne* (resp. *surfacique*) ; le résultat obtenu avec un tel algorithme est appelé *squelette curviligne* (resp. *squelette surfacique*). Notons qu'en 3D, certains algorithmes proposent d'abord l'obtention d'un squelette surfacique, à partir duquel peut être obtenu un squelette curviligne. D'autres proposent directement l'obtention de l'un et/ou de l'autre (voir l'annexe B).

Jusqu'à la fin de ce chapitre, nous ne parlerons que du cas de la squelettisation (et non plus de l'amincissement).

4.2.2 Différences entre l'approche itérative ou parallèle de suppression de points

4.2.2.1 Approche itérative

Un processus de squelettisation opère par suppressions de points. Il lui faut alors choisir des points à supprimer. Supposons que la suppression se fasse de manière itérative. À un instant donné nous avons un objet X ; un point x est élu, nous testons sa simplicité :

- si x est non simple, nous considérons un autre point,
- si x est simple, nous l'éliminons et obtenons ainsi un objet Y ; la simplicité du point suivant y à être examiné dépend totalement de l'objet courant et par voie de conséquence, du point

supprimé précédemment.
Cette itération est répétée jusqu'à stabilité, *i.e.* jusqu'à ce qu'il n'y ait plus de point simple. Un tel algorithme est décrit par le schéma 1.

Répéter
Pour tout point de l'image déterminé selon un balayage séquentiel **faire**
Si le point est simple **alors** il est supprimé
(**Sinon** examiner le point suivant, déterminé par le balayage)
Jusqu'à ce qu'il n'y ait plus de suppression durant un balayage complet de l'image.

Algorithme 1: *Schéma séquentiel de squelettisation.*

Retenons que le squelette obtenu par un tel processus itératif de suppression de point simple dépend du balayage de l'image : c'est lui qui élit le point candidat à la suppression. Si nous considérons un balayage vidéo, les remarques précédentes expliquent l'effet de bord qui fait "glisser" le squelette selon le sens du balayage dans l'image, d'où le mauvais centrage des squelettes. L'approche résultant d'un tel processus de suppression itérative de point simple offre comme avantages la préservation de la topologie (par la définition même d'un point simple), et la minceur des squelettes (dans le sens qu'ils ne contiennent plus de point simple).

4.2.2.2 Approche parallèle

Intuitivement, afin d'être efficace, un algorithme doit être conçu de façon à supprimer en parallèle plusieurs points. Mais la suppression en parallèle de points simples peut ne pas préserver la topologie, comme nous l'avions indiqué aux sections 2.6.2 et 3.8.

Il faut alors mettre en œuvre des stratégies permettant l'élection des points qu'un algorithme va pouvoir supprimer en parallèle ; stratégies détaillées dans la section suivante.

4.3 Stratégies mises en œuvre dans les algorithmes parallèles de squelettisation

L'un des algorithmes de squelettisation le plus connu en 2D est celui de Rosenfeld [Ros75] [RK82] (voir section A.3.4). Cet algorithme utilise la stratégie directionnelle selon la séquence $Nord, Sud, Est, Ouest$: tous les points simples ayant leur voisin $Nord$ dans le complémentaire sont supprimés en parallèle, puis tous les points simples ayant leur voisin Sud dans le complémentaire sont supprimés en parallèle, et ainsi de suite.

L'objet de la figure 4.1 illustre le fait que la stratégie directionnelle proposée par A. Rosenfeld ne fonctionne pas si elle est étendue directement au cas 3D. En effet, tous les points de cet objet sont simples, et leurs voisins selon la direction $Nord$ (voir Fig. B.1 (b)) par exemple appartiennent au complémentaire ; l'extension directe conduit alors à supprimer tous ces points (en considérant cette même direction de suppression), auquel cas l'objet disparaît : la topologie n'est ainsi pas préservée.

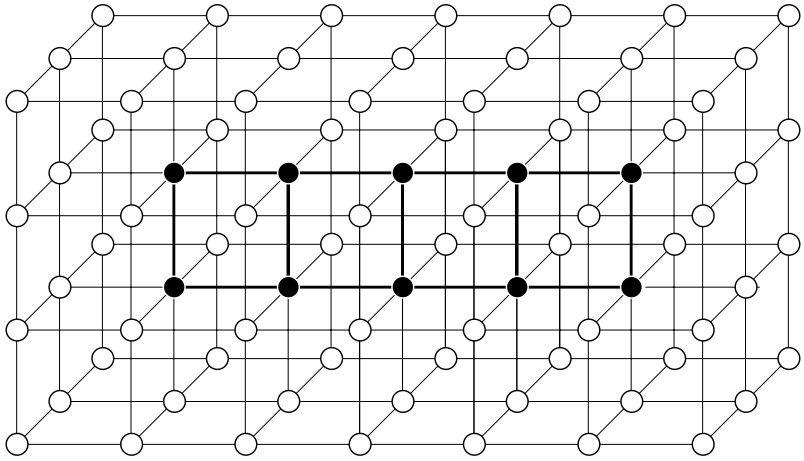


FIG. 4.1 – *Contre-exemple à l'extension directe en* 3*D de la stratégie directionnelle mise en œuvre dans l'algorithme de Rosenfeld.*

Retenons alors que la stratégie directionnelle consistant à supprimer en parallèle tous les points simples d'un objet qui ont leur voisin, déterminé selon une direction donnée, dans le complémentaire n'est uniquement valide que dans le cas 2D (au sens de préservation de la topologie).

En 3D, seul un sous-ensemble de l'ensemble des points simples peut être éliminé, à un instant donné. Il faut alors ajouter d'autres contraintes, dépendant d'une stratégie adoptée ; stratégies que nous détaillons maintenant de façon générale :

- une itération de suppression est divisée en plusieurs sous-itérations :
  - selon une *approche directionnelle* : le processus supprime en parallèle les points simples qui vérifient une condition par rapport à une direction (généralement, il s'agit du même voisinage que celui mis en jeu dans le test de la simplicité d'un point). Ce processus est répété pour chaque direction retenue, selon un ordre approprié et jusqu'à stabilité *i.e.* jusqu'à ce qu'il n'y ait plus de suppression durant une séquence "balayant" toutes les directions proposées. Le schéma 2 décrit ce processus.

$Y =$ image initiale
Répéter
    Pour chacune des directions Dir_i
        $Y = Y \setminus \{$ les points simples qui vérifient une condition selon $Dir_i \}$
Jusqu'à ce qu'il n'y ait plus de suppression durant une séquence entière de directions.

Algorithme 2: *Schéma parallèle de squelettisation, selon l'approche directionnelle.*

En pratique, un jeu de masques (ou *templates*) est fourni, décrivant les configurations qu'un point doit vérifier afin d'être supprimé, pour une direction donnée. Un point vérifiant une template est un point simple vérifiant de plus certaines conditions directionnelles. En fait, les auteurs de tels algorithmes doivent vérifier que leur algorithme préserve la topologie. C'est le cas si cet algorithme ne supprime pas de MNS. Généralement, l'obtention des templates utilisées n'est pas révélée, seule leur utilisation afin de

vérifier la préservation de la topologie par l'algorithme les utilisant est donnée. Notons également que l'utilisation des templates empêche une description "formelle" des points à éliminer, contrairement aux algorithmes que nous proposerons au chapitre 6 : un point à éliminer sera un point P-simple où P sera bien défini, par exemple, P sera l'ensemble des points dont le voisin $Nord$ appartient au complémentaire.
Certains auteurs ont proposé des algorithmes en 6 sous-itérations directionnelles [PK98a] (cf. section B.2.1) puis en 12 sous-itérations [PK97a] [PK97b] (cf. section B.2.3) [PK99] (cf. section B.2.4). Le centrage des squelettes est d'autant meilleur que le nombre de directions utilisées est élevé, mais cela accroît également la complexité d'implémentation de l'algorithme (plus de templates, d'isométries à coder). Notons enfin que les squelettes obtenus par une telle approche directionnelle dépendent de l'ordre choisi selon lequel les directions sont considérées (voir section A.3 (resp. B.2) pour des algorithmes utilisant cette approche directionnelle pour images 2D (resp. 3D)).

– selon une approche *sous-mailles* (*subfields* ou *subgrids*) : l'espace est décomposé en sous-mailles. Pour une sous-maille donnée, seuls les points y appartenant et vérifiant certaines conditions sont éliminés en parallèle, puis ce processus active une autre sous-maille et ainsi de suite, jusqu'à stabilité *i.e.* jusqu'à ce qu'il n'y ait plus de suppression durant une séquence activant toutes les sous-mailles. Un tel processus est décrit par le schéma 3.
En fait, si la subdivision est telle que deux points d'une même sous-maille ne peuvent appartenir à un MNS ; la topologie est alors préservée. C'est le cas, par exemple, pour 4 sous-mailles en 2D, ou pour 8 sous-mailles en 3D (à condition de les choisir de façon adéquate, voir par exemple la section B.3.1). Si la subdivision de l'espace est telle que deux points d'une même sous-maille peuvent être voisins, risquant ainsi d'appartenir à un MNS, il faut encore une fois ajouter d'autres conditions et ne supprimer qu'un sous-ensemble de l'ensemble des points simples d'une même sous-maille (cf. section A.4.1 (resp. B.3.1) pour un algorithme utilisant cette approche sous-mailles pour images 2D (resp. 3D)).
Le squelette obtenu dépend de la position de l'objet dans l'image, *i.e.* de la première sous-maille choisie. De plus, les squelettes présentent généralement des effets de "zig-zag".

$Y =$ image initiale
Répéter
  Pour chacune des sous-mailles S_i
    $Y = Y \setminus \{$ les points de S_i vérifiant une condition
      supplémentaire selon le choix des sous-mailles $\}$
Jusqu'à ce qu'il n'y ait plus de suppression durant une séquence
  entière d'activation de sous-mailles.

Algorithme 3: *Schéma parallèle de squelettisation, selon l'approche sous-mailles.*

- une itération de suppression utilise un voisinage étendu. C'est le cas pour :
  - les algorithmes *fortement parallèles* (ou *fully parallel*) : un processus supprime les points simples de l'objet qui vérifient une autre condition qui utilise un plus grand voisinage que celui suffisant à décider de la simplicité ou non d'un point (cf. section A.5.7) ; ce processus se termine dès qu'il n'y a plus de suppression lors d'une itération. Un tel processus est décrit par le schéma 4. En 2D, de nombreux algorithmes fortement parallèles ont été proposés. Certains ne préservaient pas la topologie (cf. sections A.5.4 et A.5.5). Il est encore plus difficile de proposer des algorithmes valides en 3D ; le voisinage utilisé étant trop étendu afin de permettre directement une simulation par ordinateur pour vérifier la validité d'un tel algorithme. Généralement, de tels algorithmes considèrent des templates, et dans celles-là, nous pouvons nous apercevoir qu'une direction "de préservation" est privilégiée : par exemple, afin de pouvoir retirer des points d'un ruban d'épaisseur 2, il faut imposer la conservation d'une bande plutôt qu'une autre (voir l'annexe A).
    À notre connaissance, trois algorithmes fortement parallèles pour images 3D (et n'utilisant pas la technique suivante des points P-simples) ont été proposés. Nous avons montré que seul l'un d'eux préservait la topologie. Il s'agit de l'algorithme de Manzanera et al. [MBPL99c] [MBPL99a] (cf. les sections B.4.3 et 6.7.4.2). La conception même de cet algorithme ne privilégie aucune direction : les squelettes obtenus sont alors bien centrés, néanmoins ils peuvent ne pas être minces ; il est à noter également que cet algorithme utilise un très grand voisinage.

$Y =$ image initiale
Répéter
    $Y = Y \setminus \{$ les points simples de Y (et vérifiant une condition supplémentaire dans un voisinage étendu) $\}$
Jusqu'à ce qu'il n'y ait plus de suppression durant une itération.

Algorithme 4: *Schéma parallèle de squelettisation, selon l'approche fortement parallèle.*

  - les algorithmes supprimant les points P-simples. Ces points doivent appartenir à un ensemble P et vérifier une condition supplémentaire par rapport à cet ensemble : ils doivent être P-simples. Deux algorithmes fortement parallèles à base de points P-simples ont déjà été proposés [MF98] [BM00]. Nous verrons que P peut dépendre également d'une direction (dans ce cas, l'algorithme est "découpé" en sous-itérations directionnelles) (voir chapitres 5 et 6). Jusqu'à présent, un algorithme supprimant des points P-simples soit utilisait une phase préliminaire d'étiquetage soit, ce qui revient au même, examinait un voisinage étendu et suffisant pour tester l'appartenance d'un point à l'ensemble P [Ber95a] (cf. section 5.3). En fait, nous avons proposé une nouvelle définition, celle de point P^x-simple, nous permettant de remédier à l'examen d'un voisinage étendu et d'éviter une phase préliminaire d'étiquetage (voir chapitre 6). Cette nouvelle approche de point P^x-simple n'a été étudiée dans ce mémoire que dans le cas des algorithmes directionnels. Des travaux sont actuellement en cours afin d'étendre ces résultats aux

cas des algorithmes fortement parallèles.

Notons également l'existence d'autres algorithmes de squelettisation. Notamment les algorithmes :

- guidés par la distance [AdB89] [Pud98]. Le choix des points candidats à la suppression est déterminé par rapport à leur distance au bord de l'objet (sont utilisées les notions d'axe médian, de lignes médianes [CM91], listes de contour [Akt97]). C'est d'ailleurs par cette notion de distance qu'a été définie la notion de squelette, à travers l'analogie avec une propagation de feux de prairies [Blu67] [Xia89],
- utilisant la morphologie mathématique (voir le squelette par zones d'influence [Ser82]).

Ces approches ne sont pas étudiées dans ce mémoire.

4.4 Complexité des algorithmes

Dans l'annexe A, nous avons donné quelques considérations concernant des comparaisons entre algorithmes de classe différente, au sujet du nombre d'itérations requis afin de squelettiser des objets synthétiques.

Par exemple, considérons une courbe ouverte horizontale de longueur n. Un parcours itératif le long de cette courbe va squelettiser cet objet en n itérations. Un processus parallèle peut nécessiter, de façon générale, environ $\frac{n}{2}$ sous-itérations de suppression (en accédant à tout point de l'image lors de chacune des sous-itérations). Un algorithme de type sous-mailles peut même exiger jusqu'à n itérations, selon la position de l'objet dans l'image (voir l'annexe A pour plus de détails).

C'est un problème difficile que de pouvoir estimer l'efficacité d'un algorithme quel que soit le type d'images à traiter. Intuitivement, pour des images réelles, un algorithme parallèle sera plus efficace qu'un algorithme itératif et produira un squelette mieux centré.

Par exemple, un algorithme A supprimant au moins les points supprimés par un algorithme B (de même classe) va généralement obtenir un squelette plus rapidement que B, mais ce n'est pas toujours le cas (cf. chapitre 6). De plus, le fait de retirer plus de points lors d'une première (sous-)itération peut éventuellement être la cause d'apparition prématurée de points extrémités : le squelette n'est alors pas forcément "meilleur" (d'un point de vue esthétique) que celui obtenu par B ; la présence de points extrémités prématurément peut même être la cause d'un manque d'efficacité de A par rapport à B.

4.5 Conclusion

Nous avons présenté les différentes classes d'algorithmes de squelettisation, opérant par suppression de points, selon l'approche topologie digitale. Retenons que les algorithmes à base de sous-itérations ne produisent pas de squelettes bien centrés, contrairement à ce que l'on devrait obtenir avec des algorithmes fortement parallèles. Dans ce but, ces derniers requièrent l'examen d'un voisinage étendu. Afin de réduire ce voisinage, il est possible d'imposer des contraintes directionnelles, absentes dans le cœur de l'algorithme, mais intrinsèquement présentes dans les

configurations correspondant aux points pouvant être supprimés (voir les annexes A et B) ; dans ce cas le centrage n'est plus aussi bon.

En conclusion, aucune classe ne prévaut ; les avantages de l'une sont les inconvénients d'une autre, au niveau soit de l'algorithme en lui-même, soit des points qu'il supprime. C'est ce qui a motivé notre recherche d'algorithmes dans un autre cadre mathématique : l'approche topologie discrète (cf. la partie III).

Deuxième partie

Points P-simples, et nouveaux algorithmes de squelettisation pour images 3D binaires, basés sur la suppression de points P-simples

Un algorithme de squelettisation opère par suppression de points simples. Si cette suppression est réalisée de façon séquentielle, la topologie est préservée (cela par la définition même d'un point simple). En revanche, la topologie peut ne pas être préservée lors d'une suppression en parallèle de points simples (cf. les sections 2.6 et 3.8, le chapitre 4 et les annexes A et B).

En 1995, G. Bertrand a proposé la notion de points P-simples [Ber95a] ; P désignant un ensemble constitué de points susceptibles d'être supprimés simultanément. Généralement, P peut être déterminé selon la nature de l'algorithme ; par exemple, dans un algorithme parallèle de type sous-itérations directionnelles, la condition pour qu'un point appartienne à P dépend d'une direction (plus précisément, P dépendra de la sous-itération courante du processus de squelettisation ; sous-itération qui est liée à une direction).

Nous avons la propriété fondamentale de pouvoir supprimer l'ensemble des points P-simples tout en préservant la topologie ; l'importance en est alors flagrante en ce qui concerne la conception d'algorithmes parallèles (et valides) de squelettisation ; encore faut-il proposer une caractérisation satisfaisante d'appartenance à l'ensemble P (voir explications à la section 5.3.2.1 et au chapitre 6). Nous dirons d'un algorithme qu'il est *P-simple*, s'il opère par suppression de points P-simples ; auquel cas, nous sommes certains qu'il préserve la topologie.

Une fois que P est connu, il est possible de tester localement (*i.e.* dans $N^*_{26}(x)$)) la P-simplicité d'un point x. En 1995, G. Bertrand a proposé un algorithme opérant par suppression de points P-simples. Celui-là consiste en la répétition jusqu'à stabilité de deux étapes : la première étiquette les points de P, la seconde détermine localement et en parallèle les points P-simples puis les élimine en parallèle. Retenons qu'un tel schéma nécessite une phase préalable d'étiquetage, ou l'utilisation de listes de contours pour lesquelles il faut gérer la création et la mise à jour. Une autre variante, détaillée dans le chapitre 6, n'a pas besoin de phase préalable d'étiquetage mais examine un plus grand voisinage.

Dans le chapitre 6, sera introduite la définition d'un ensemble P^x ; l'ensemble P^x est obtenu à partir d'un ensemble P. Cela nous a permis de proposer un schéma de squelettisation opérant par suppression de points P^x-simples et évitant les deux inconvénients précédemment cités ; à savoir une phase préliminaire d'étiquetage ou l'examen d'un voisinage étendu. Nous pourrons ainsi comparer un algorithme basé sur la suppression de points P^x-simples avec certains algorithmes (détaillés dans l'annexe B) décidant localement de la suppressibilité d'un point. En fait, nous proposerons des algorithmes supprimant des points P^x-simples, et P sera recherché de façon à ce que l'algorithme proposé retire au moins les points supprimés par un algorithme déjà existant. Cela permet d'une part, d'obtenir un algorithme supprimant généralement plus de points lors de la première sous-itération de suppression, d'autre part, de vérifier également que l'algorithme déja existant préserve bien la topologie. Remarquons que l'utilisation des points P-simples nous a permis de mettre en évidence que deux algorithmes ne préservaient pas la topologie. Actuellement, nous travaillons avec leur auteur pour les corriger.

Chapitre 5

Points P-simples

5.1 Introduction

Ce chapitre définit d'abord ce qu'est un point P-simple, puis rappelle l'algorithme opérant par suppression en parallèle de points P-simples, proposé par G. Bertrand [Ber95a] (algorithme de type sous-itérations directionnelles). Nous discuterons en fin de chapitre, de la recherche d'une démarche systématique afin de tester si un algorithme préserve la topologie.

5.2 Définitions [Ber95a]

Dans la suite, nous considérons une image binaire tridimensionnelle $(V, n, \overline{n}, B)$, avec $V = \mathcal{Z}^3$. Dans cette section, nous rappelons la définition d'un point P_n-simple (n étant la connexité utilisée pour l'objet, lorqu'il n'y a pas d'ambiguïté, nous parlerons d'un point P-simple). Nous illustrons des exemples de points P-simples ou non et donnons les caractérisations les plus récentes et la complexité pour caractériser de tels points.

Définition (point P_n-simple et ensemble P_n-simple) : Soit $X \subset \mathcal{Z}^3$. Soient $P \subset X$ et $x \in P$. Le point x est *P_n-simple* si $\forall\ S \subset P \setminus \{x\}$, x est n-simple pour $X \setminus S$. Notons $S_n(P)$ l'ensemble des points P_n-simples. Un ensemble D est *P_n-simple* (pour X) si $D \subset S_n(P)$.

Intuitivement, si P caractérise des points qui sont potentiellement candidats à la suppression, et si un point x est simple quel que soit le sous-ensemble S de P qui pourrait être supprimé de X, alors la suppression de x ne "posera pas de problème" lors de la suppression effective d'un ensemble S : la topologie est alors préservée (voir également la section 9.4).

La conséquence immédiate qui en ressort est qu'un algorithme opérant par suppression de points P-simples (*i.e.* d'ensembles P-simples) est valide (*i.e.* il préserve bien la topologie).

Nous introduisons alors une nouvelle terminologie, celle d'algorithme P-simple.

Définition (algorithme P-simple) : S'il existe une condition d'appartenance à un ensemble P, telle qu'un algorithme opère par suppression de points P-simples, alors cet algorithme est dit *algorithme P-simple*.

Proposition : Tout algorithme P-simple préserve la topologie.

A priori, le problème qui apparaît aussitôt est celui de tester la simplicité d'un point x par rapport à tous les sous-ensembles possibles S inclus dans P, afin de décider de la P-simplicité ou non du point x (processus de complexité exponentielle). En fait, G. Bertrand a proposé une caractérisation locale des points P-simples :

Proposition : Soient $P \subset X$ et $x \in P$. Notons $R = X \setminus P$.

$$x \text{ est } P_n\text{-simple pour } X \Leftrightarrow \begin{cases} (C_1) & T_n(x, R) = 1, & (5.1) \\ (C_2) & T_{\overline{n}}(x, \overline{X}) = 1, & (5.2) \\ (C_3) & \forall y \in N_n^*(x) \cap P, N_n^*(y) \cap G_n(x, R) \neq \emptyset, & (5.3) \\ (C_4) & \forall y \in N_{\overline{n}}^*(x) \cap P, N_{\overline{n}}^*(y) \cap G_{\overline{n}}(x, \overline{X}) \neq \emptyset. & (5.4) \end{cases}$$

Cette caractérisation utilise les nombres topologiques et des parcours de composantes connexes (dans des voisinages géodésiques, voir section 3.6.1.1). Notons $k = \#[G_n(x, X)]$ et $\overline{k} = \#[G_{\overline{n}}(x, \overline{X})]$, avec $\#Y$ étant le nombre d'éléments de l'ensemble Y. Les conditions (5.1) et (5.3) peuvent être implantées en $\mathcal{O}(k)$ et les conditions (5.2) et (5.4) en $\mathcal{O}(\overline{k})$: la caractérisation des points P-simples est alors réalisée en un temps linéaire (en sachant au préalable à quel ensemble R, P ou $\overline{X}$ appartient un point ; en fait, la donnée de l'image fournit X et $\overline{X}$; une fois que P est connu, nous pouvons déduire $R = X \setminus P$).

Remarque : Jusqu'à maintenant les algorithmes n'opéraient que par suppression de certains points simples ; ceux-là étant sélectionnés par exemple, par rapport à une direction de suppression, les auteurs de tels algorithmes vérifiant la validité de leurs algorithmes si ceux-là ne supprimaient pas de MNS. Ici, il s'agit de l'approche "inverse' : nous nous donnons une condition d'appartenance à un ensemble P et nous savons que l'algorithme opérant par suppression des points P-simples préserve la topologie, en raison de la définition même des points P-simples ; aucune preuve supplémentaire n'est exigée.

Le concept d'ensemble P-simple a été approché par Y.F. Tsao et K.S. Fu [TF82], (voir également le théorème 5.2, page 371, [KR89], introduction d'un ensemble R - cas 2D), mais rien n'avait été proposé plus précisément (et concrètement dans le cadre d'un algorithme de squelettisation) jusqu'aux points P-simples.

Il nous a semblé très intéressant d'approfondir l'étude déjà réalisée [Ber95a, Ber95b, Ber95c], d'essayer de proposer des algorithmes P-simples (chapitre 6) et de se positionner par rapport aux algorithmes de squelettisation existants (détaillés dans l'annexe B).

Notons encore que les points P-simples ont été également utilisés dans l'étude de surfaces

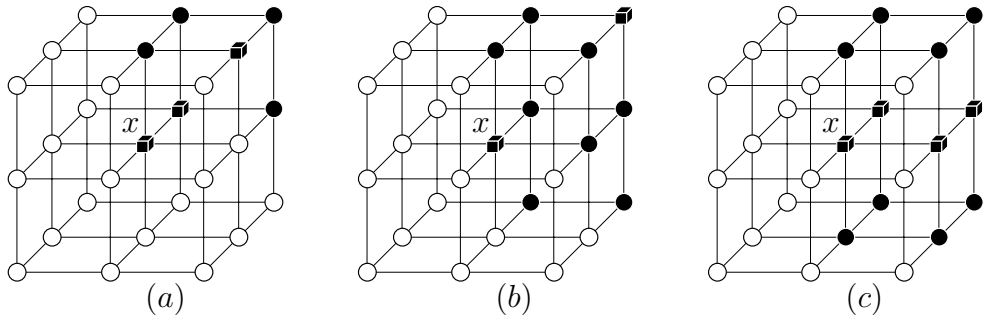


FIG. 5.1 – *Exemples de points P-simples ou non. Les points appartenant à $X \setminus P$ (= R), $\overline{X}$, P sont représentés par des boules noires, des boules blanches ou des cubes noirs, respectivement.*

discrètes [BM96] ou dans le cadre de polyédrisation [BM00].

5.2.1 Exemples

Considérons la figure 5.1. Seule la configuration (a) correspond à un point P_6-simple, et seules les configurations (a) et (b) correspondent à un point P_{26}-simple. Intuitivement, dans la configuration 5.1 (b), selon la connexité $(6, 26)$, si les points de P étaient supprimés en parallèle, un trou serait créé dans l'objet (voir Fig. 1.11 et la section 1.6) ; et c'est ici la condition (5.4) qui n'est pas vérifiée. Dans la configuration 5.1 (c), pour les 2 types de connexités, si les points de P étaient supprimés en parallèle, alors l'objet serait déconnecté ; la condition (5.1) n'est pas vérifiée pour les deux connexités et la quatrième condition n'est pas vérifiée pour la connexité $(6, 26)$.

5.2.1.1 Points non P_n-simples : mise en défaut de chacune des conditions

La figure 5.2 (mêmes notations qu'à la figure 5.1) représente différentes configurations représentant un point non P_{26}-simple, de telle façon que seule la première condition ne soit pas vérifiée (Fig. 5.2 (a)), ou la seconde uniquement (Fig. 5.2 (b)) et ainsi de suite. Les quatre configurations (Fig. 5.2 (e) à (h)) sont celles d'un point non P_6-simple selon le même procédé.

Remarque : La mise en défaut de la condition (5.2) implique que le point n'est pas n-simple. Pour la mise en défaut des autres conditions, les exemples présentés ici sont ceux de points n-simples, non P_n-simples. Le but de la figure est d'illustrer la nécessité de chacune des conditions mises en jeu dans la P-simplicité par rapport à la simplicité.

5.2.2 Autres caractérisations

Dans [Ber95c], nous avons la proposition suivante, qui est une formulation des conditions (5.3) et (5.4), plus facile à utiliser :

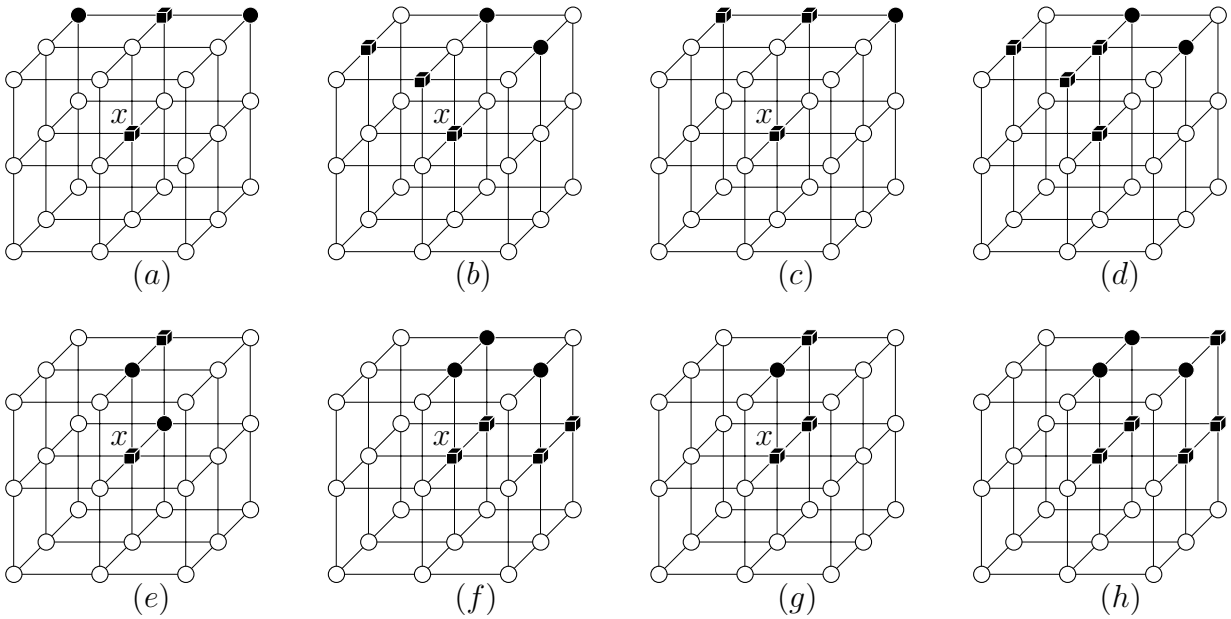


FIG. 5.2 – *Configurations de points non P-simples. Les configurations (a) à (d) (resp. (e) à (h)) correspondent à un point non P_{26}-simple (resp. non P_6-simple), en mettant en échec à chaque fois une seule des 4 conditions de P-simplicité.*

Proposition : Soient $x \in \mathcal{Z}^3$, $y \in N_n^*(x)$ et $Y \subset \mathcal{Z}^3$ tel que $y \notin Y$.

$$N_n^*(y) \cap G_n(x,Y) \neq \emptyset \Leftrightarrow \begin{cases} \text{pour } n = 26, \exists z \in Y \text{ tel que } z \text{ est } 26\text{-adjacent à } x \text{ et } y, & (5.5) \\ \text{pour } n = 6, \exists z \in Y \text{ et } \exists t \in Y \\ \qquad \text{tel que } \{x, y, z, t\} \text{ constitue un carré unité.} & (5.6) \end{cases}$$

C'est cette caractérisation qui est utilisée dans [Ber95c] afin de prouver que l'algorithme de Gong et Bertrand [GB90] préserve la topologie (démontré dans le cas $(26, 6)$). En effet, pour une direction donnée, les points candidats à la suppression appartiennent à un ensemble P_{26}-simple ; l'ensemble P étant constitué de points déterminés par la direction de suppression et vérifiant quatre conditions données dans [Ber95c] (cette démonstration est reprise à la section 6.5.1.2 ; des démonstrations de P-simplicité d'autres algorithmes sont proposées au chapitre 6).

La caractérisation des points P-simples la plus récente utilise uniquement les nombres topologiques [BM96] :

Proposition : Soient $P \subset X$ et $x \in P$ et $R = X \setminus P$.

$$x \text{ est } P_n\text{-simple pour } X \Leftrightarrow \begin{cases} T_n(x, R) = 1, \\ T_{\overline{n}}(x, \overline{X}) = 1, \\ \forall y \in N_n^*(x) \cap P, T_n(x, R \cup \{y\}) = 1, \\ \forall y \in N_{\overline{n}}^*(x) \cap P, T_{\overline{n}}(x, \overline{X} \cup \{y\}) = 1. \end{cases}$$

En pratique, cela peut nous amener à caractériser un point P-simple uniquement par l'utilisation des BDD (graphe de décision binaire, cf. l'annexe C) des nombres topologiques.

Remarque : En fait, nous pouvons nous demander s'il est possible de générer un BDD qui nous indique si un point est P-simple (de façon générale, comme cela a été fait dans le cadre des points simples et en tenant compte de toutes les configurations possibles, pour $\overline{X}$, R ou P). Dans ce cas, il faut préciser l'appartenance d'une variable (du voisinage) à $\overline{X}$, à P ou à R.

Soit nous créons, ce que nous pouvons appeler un TDD (*Ternary Decision Diagram*), par analogie avec un BDD (voir l'annexe C), consistant en un arbre pour lequel chaque nœud non terminal comporte jusqu'à trois fils (l'arbre ternaire complet comporterait $\frac{3^n-1}{2}$ nœuds non terminaux – c'est le pire des cas pour un TDD, du point de vue de la taille –, avec $n = 26$, l'arbre comporterait $1.27\ 10^{12}$ nœuds).

Soit nous créons une formule booléenne dont un terme est constitué d'une première série de 26 variables (les 26 voisins) indiquant l'appartenance ou non à $\overline{X}$, puis 26 autres variables (les mêmes voisins, mais renommés) pour l'appartenance à P ; l'arbre binaire complet comporterait $2^n - 1$ nœuds non terminaux – c'est le pire des cas pour un BDD, du point de vue de la taille –, avec $n = 52$, l'arbre comporterait $4.5\ 10^{15}$ nœuds non terminaux ; son obtention semble irréalisable.

5.3 Un schéma de squelettisation valide opérant par suppression de points P-simples [Ber95a]

Nous rappelons un schéma de squelettisation basé sur la suppression de points P-simples, proposé par G. Bertrand dans [Ber95a].

5.3.1 Description

Ce schéma utilise une stratégie directionnelle. Notons dir_i les six directions : $Haut$, Bas, Sud, $Nord$, $Ouest$, Est ($i = 0 \ldots 5$, respectivement) et $dir_i(x)$ le 6-voisin de x selon la direction dir_i. Soit $D^{dir_i}(X) = \{x \in X ; dir_i(x) \in \overline{X}\}$, l'ensemble des points x de X ayant le voisin selon la direction dir_i qui appartient à $\overline{X}$. L'ensemble des points extrémités d'un ensemble Y est noté $E(Y)$. Notons $S_n(P)$, l'ensemble des points P_n-simples. Ce schéma de squelettisation est décrit par l'algorithme 5 (avec $X^0 = X$), algorithme que nous nommons *algorithme P-simple de Bertrand* et noté BE_PS par la suite.

L'algorithme consiste alors en la répétition jusqu'à stabilité (obtenue lors d'une itération sans suppression) de sous-itérations de suppression parallèle de points P-simples – itération composée de 6 sous-itérations, correspondant chacune à 1 des 6 directions. En fait, une sous-itération est constituée de 2 phases : la première consiste à étiqueter (en parallèle) à P_i les points de l'objet courant X^i qui appartiennent à $D^{[dir_i \bmod 6]}(X^i)$, sauf les points terminaux (qui forment $E(X^i)$). La seconde consiste à déterminer (en parallèle) les points P_n^i-simples et à les supprimer.

Répéter
Faire pour chaque direction i successive :
(a) Étiqueter en parallèle les points de X^i appartenant à P^i
avec $P^i = \overline{E(X^i)} \cap D^{[dir_i \bmod 6]}(X^i)$,
(b) Supprimer en parallèle les points retenus en (a) qui sont P^i_n-simples :
nous obtenons ainsi X^{i+1} (nous avons : $X^{i+1} = X^i \setminus S_n(P^i)$)
Jusqu'à ce qu'il n'y ait plus de suppression pendant 6 sous-itérations successives.

Algorithme 5: *Algorithme P-simple de Bertrand (*BE_PS*).*

Ce processus s'arrête lorsqu'il n'y a plus de suppression lors de 6 sous-itérations (directionnelles) successives.

Ce schéma de squelettisation conduit à une large classe d'algorithmes de squelettisation car il y a plusieurs façons de définir les points extrémités. En fait, G. Bertrand a proposé un cadre plus général permettant de définir plusieurs schémas de squelettisation (voir [Ber95b]).

5.3.2 Commentaires et résultats

5.3.2.1 Remarques sur la décision d'appartenance à l'ensemble P

Lors du test de P-simplicité d'un point x, il faut déterminer lesquels de ses voisins y appartiennent à P (cf. conditions (5.3) et (5.4), page 80). Supposons que nous puissions décider localement de l'appartenance de x à P dans $N^*_{26}(x)$. Il est alors également possible de décider de l'appartenance de y à P dans $N^*_{26}(y)$ et $\bigcup_{y \in N^*_{26}(x)} N^*_{26}(y)$ est le voisinage $5 \times 5 \times 5$ centré en x.

En résumé :

- Soit nous réalisons une phase d'étiquetage préalable à P, ce qui évite de tester un voisinage $5 \times 5 \times 5$ centré en x afin de savoir lesquels des points y de $N_{26}(x)$ appartiennent à P, lors du test de la P-simplicité de x.
  R. Malgouyres et S. Fourey [MF98] ont proposé un algorithme fortement parallèle basé sur la suppression de points P-simples ; P étant défini comme étant l'ensemble des points de bord. Les résultats obtenus sont alors bien centrés (car issus d'une squelettisation homogène) mais non forcément minces, inconvénient surmonté par l'utilisation ultérieure d'un algorithme séquentiel de squelettisation. L'algorithme parallèle utilise la technique des listes de contours, l'avantage est d'éviter l'étiquetage précédemment suggéré ; répété de plus lors de chaque sous-itération de suppression, néanmoins l'utilisation des listes de contours nécessite le balayage entier de l'image (pour leur création) et leur mise à jour. En fait, cette technique peut être utilisée dans tous les algorithmes que nous proposons par la suite ou pour ceux auxquels nous les comparons ; afin d'obtenir une implémentation efficace.
- Soit nous ne réalisons pas la phase préalable d'étiquetage et dans ce cas, nous devons examiner un voisinage $5 \times 5 \times 5$ centré en x.
- Nous proposons intuitivement une autre approche qui nous a amené à proposer un ensemble P^x (voir chapitre 6) : nous décidons qu'un point y ($\in N_{26}(x)$) appartient à P^x, si

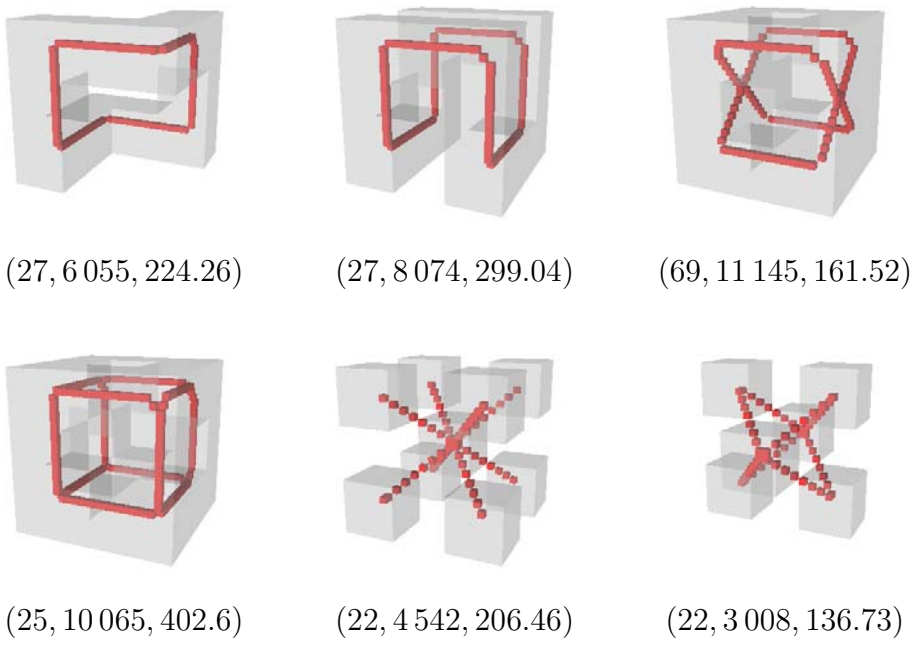


FIG. 5.3 – *Squelettisation curviligne* $(26, 6)$ *– algorithme de Bertrand (*BE_PS*) (*REPR7*).*

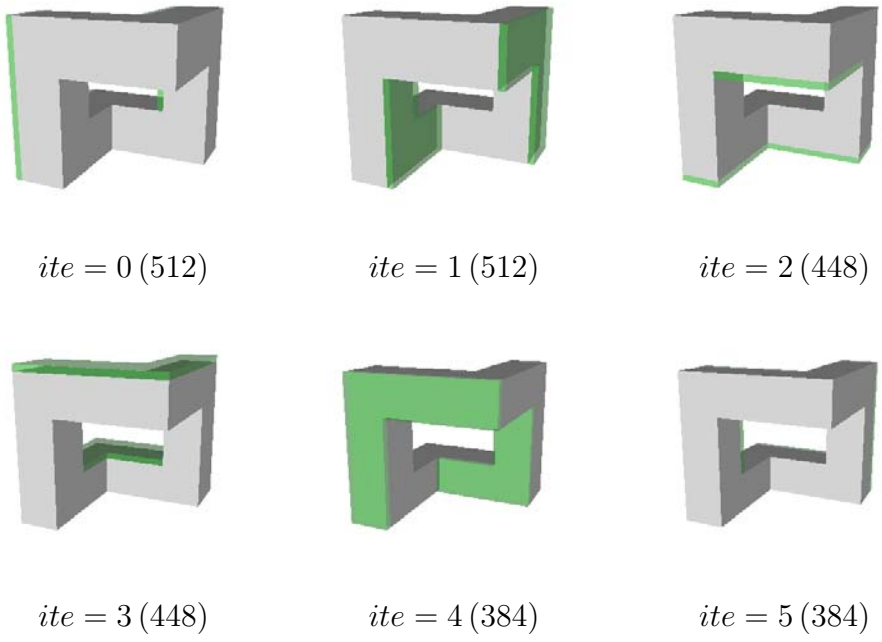
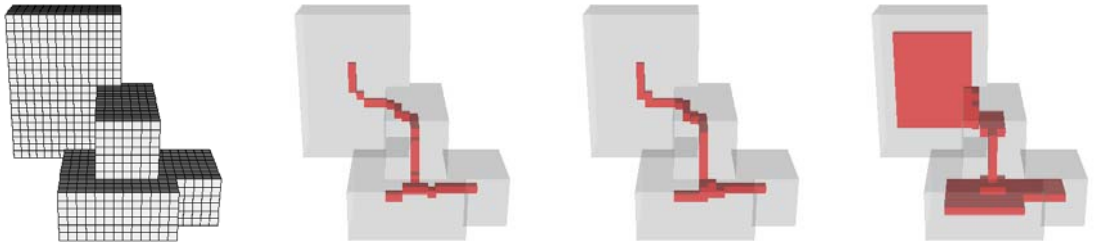
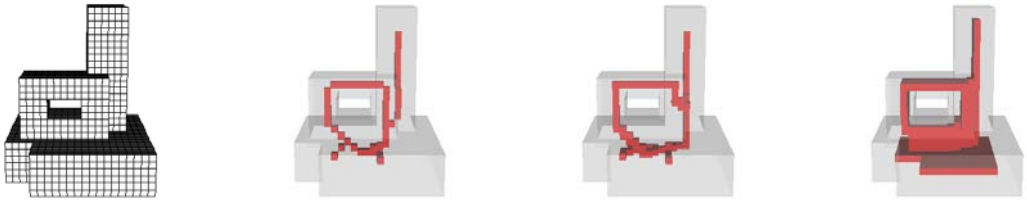


FIG. 5.4 – *Détails des* 6 *premières sous-itérations de l'algorithme de Bertrand (*BE_PS*) – squelettisation curviligne* $(26, 6)$ *sur l'objet PaKu1 (*REPR6*).*



Objet initial | Squelette curviligne $(26, 6)$ $(42, 3\,276, 78)$ | Squelette curviligne $(6, 26)$ $(41, 3\,264, 79.61)$ | Squelette surfacique $(6, 26)$ $(18, 3\,035, 168.61)$



Objet initial | Squelette curviligne $(26, 6)$ $(54, 4\,245, 78.61)$ | Squelette curviligne $(6, 26)$ $(46, 4\,218, 91.70)$ | Squelette surfacique $(6, 26)$ $(17, 3\,792, 223.06)$

FIG. 5.5 – *Autres objets, squelettisations curvilignes* $(26, 6)$ *puis* $(6, 26)$ *et surfacique* $(6, 26)$ *avec l'algorithme de Bertrand (*BE_PS*) (*REPR7*).*

les conditions d'appartenance de y à P dans $N_{26}(x)$ sont réalisées, même si elles ne le sont pas forcément toutes dans un voisinage $5 \times 5 \times 5$ centré en x. En d'autres termes, si un minimum de conditions sont vérifiées dans $N_{26}(x)$ "restrictions des conditions d'appartenance à P dans $N_{26}(x)$", $\forall y \in N_{26}(x)$, nous dirons qu'il est possible que le point testé y appartienne à P ; et nous dirons que y appartient à P^x. L'avantage est que nous accédons à moins de points (restrictions à $N_{26}(x)$), l'inconvénient intuitif est que a priori, plus de points de $N_{26}(x)$ appartiendront à P^x et x sera moins facilement P^x-simple ; en bref, un algorithme plus rapide du point de vue des accès pour déterminer les points à supprimer, mais moins rapide du point de vue des itérations (car il supprime, a priori, moins de points). Ces notions seront expliquées dans le chapitre suivant (à travers la définition d'un ensemble P^x, défini à partir d'un ensemble P). Nous proposerons ensuite de nouveaux algorithmes de squelettisation opérant par suppression de points P^x-simples.

5.3.2.2 Résultats

Considérons qu'un point extrémité pour l'obtention d'un squelette curviligne est un point avec un seul voisin. Les résultats obtenus sont illustrés aux figures 5.3 et 5.4 (la représentation utilisée est décrite à la section B.1).

Nous proposons également une caractérisation pour l'obtention de squelettes surfaciques, uniquement dans le cas de la connexité $(6, 26)$. Si nous considérons la 6-adjacence pour X, nous pouvons dire que le motif 3D le plus élémentaire est le cube unité $2 \times 2 \times 2$. Un point extrémité peut être défini comme un point qui n'appartient pas localement à une région 3D, *c.-à-d.* qui n'appartient pas à un cube unité inclus dans X [Ber95a]. Les résultats obtenus sont illustrés à la figure 5.5.

5.4 Liens entre les points P-simples, les ensembles simples et les MNS

Dans cette section, nous donnons quelques propriétés des ensembles P-simples, par rapport aux MNS (cf. chapitre 3). De manière générale, afin de tester la validité d'un algorithme, il fallait s'assurer qu'il ne retirait pas de MNS. Dans cette section, nous explicitons la démarche de la vérification de la validité d'un algorithme par le test de la P-simplicité des ensembles retirés par cet algorithme (de tels tests seront réalisés dans le chapitre suivant).

Nous expliquons brièvement pourquoi la version vérifiant si l'algorithme ne retire que des ensembles P-simples est un processus plus rapide que celui vérifiant si l'algorithme ne retire pas de MNS.

5.4.1 Propriétés des ensembles P-simples par rapport aux MNS

Rappelons les définitions d'un ensemble simple et d'un MNS (voir chapitre 3, section 3.8) :

Définitions : Soit $X \subset \mathcal{Z}^3$. Un ensemble $S \subset X$ est un *ensemble n-simple* de X si les points de S peuvent être arrangés selon une séquence $S = \{x_1, \ldots, x_k\}$ de telle façon que x_1 soit n-simple pour X et x_i soit n-simple pour l'ensemble $X \setminus \{x_1, \ldots, x_{i-1}\}$, pour $i = 2, \ldots, k$. Un ensemble $S \subset X$ est un *ensemble minimal non n-simple (n-MNS)* s'il n'est pas un ensemble simple et si chacun de ses sous-ensembles propres est n-simple.

La proposition suivante établit la relation entre les ensembles P-simples et les ensembles n-simples [Ber95c].

Proposition : Soient $X \subset \mathcal{Z}^3$ et $P \subset X$: P est P_n-simple $\Leftrightarrow$ tous les sous-ensembles de P sont n-simples.
En particulier, un point P_n-simple est n-simple.

Remarque : Soit $x \in X$. Si nous imposons à l'ensemble P d'être uniquement composé de l'élément x, alors $N_n^*(x) \cap P = \emptyset$ et $N_{\overline{n}}^*(x) \cap P = \emptyset$, les conditions (5.3) et (5.4) de P-simplicité sont donc vérifiées. Nous avons $R = X \setminus P = X \setminus \{x\}$, et la définition d'un nombre topologique implique $T_n(x, X \setminus \{x\}) = T_n(x, X)$; la condition (5.1) équivaut à $T_n(x, X) = 1$, et avec la condition (5.2), cela équivaut à ce que le point doit être simple. En résumé, lorsque P est constitué d'un seul point x, le point x est P-simple si et seulement si x est simple pour X. D'ailleurs, nous pouvons obtenir ce résultat directement par la définition de P-simplicité d'un point : avec $P = \{x\}$ et $\forall S \subset P \setminus \{x\}$ ($S = \emptyset$) et il faut x n-simple pour $X \setminus S$ ($= X$).

Intuitivement, un algorithme qui n'étiquette à P qu'un seul point (à une itération donnée), et qui le supprime (en parallèle) s'il est P-simple (sachant qu'aucun autre point de l'image ne peut être étiqueté à P), revient à un algorithme de suppression séquentiel de points simples (le choix des points candidats à la suppression par cet algorithme séquentiel équivaut au choix des ensembles successifs P).

Nous avons la proposition suivante (qui découle de la précédente) qui établit un lien entre les points P-simples et les MNS.
Proposition (lien P-simple et MNS) : Soient $X \subset \mathcal{Z}^3$ et $P \subset X$: P est P_n-simple $\Leftrightarrow$ P ne contient pas de n-MNS.

La propriété suivante de MNS est proposée par Kong ([Kon93], d'après [Ber95b]).

Propriété : Un ensemble non vide $M \subset X$ est un MNS pour X

$$\Leftrightarrow \begin{cases} \forall x \in M, x \text{ n'est pas simple pour } [X \setminus M] \cup \{x\}, & (5.7) \\ \forall x \in M, \forall N \subset M \text{ tel que } x \in N, x \text{ est simple pour } [X \setminus M] \cup N. & (5.8) \end{cases}$$

Les points P-simples permettent d'avoir une autre caractérisation des MNS :

Proposition : Un ensemble non vide $M \subset X$ est un MNS pour X

$$\Leftrightarrow \begin{cases} \forall x \in M, x \text{ n'est pas } M\text{-simple}, & (5.9) \\ \forall x \in M, \forall y \in M \text{ et } x \neq y, x \text{ est } [M \setminus \{y\}]\text{-simple}. & (5.10) \end{cases}$$

Dans la section 5.5, quelques unes de ces propriétés seront retrouvées, après l'introduction des notions d'homotopie et d'homotopie forte.

5.5 Démarche systématique de la vérification de la validité d'un algorithme

Nous illustrons une condition suffisante mais non nécessaire à la préservation de la topologie à travers l'exemple d'un algorithme supprimant ou non des MNS, et préservant la topologie. Puis, sont définies les notions d'homotopie et d'homotopie forte à l'aide desquelles nous retrouvons quelques propriétés de la section précédente. Puis, nous discutons à propos d'une démarche systématique de la vérification de la validité (au sens de préservation de la topologie) d'un algorithme.

5.5.1 Condition suffisante de la validité d'un algorithme

Dans [Ma94], il est montré que les n-MNS sont de "petits" ensembles : tout n-MNS est inclus dans un cube unité. Il a été prouvé dans le cas $(26, 6)$ qu'un opérateur parallèle de réduction O préservait la topologie s'il ne supprimait pas de MNS, *i.e.* (voir section 3.8) :

- tout ensemble contenu dans un carré unité et supprimé par O est un ensemble simple,
- O ne doit supprimer aucune composante connexe de S, contenue dans un cube unité.

Pour vérifier la validité d'un algorithme avec cette propriété, nous devons au minimum tester la simplicité d'un ensemble T inclus dans un carré S, et par conséquent accéder aux voisins des points de T afin de tester la simplicité des points de T, cela pour tout sous-ensemble de S (il faut trouver une séquence parmi toutes celles possibles), et cela pour tout sous-ensemble contenu dans un carré et supprimé par O. D'après les propositions précédentes, il suffit de regarder si l'ensemble des points x supprimés par l'algorithme est P-simple, cela se fait localement dans $N_{26}^*(x)$, une fois que P est connu.

5.5.2 Condition suffisante mais non nécessaire

Considérons un algorithme qui ne retire que des ensembles de deux points ($P = \{p_1, p_2\}$) de courbes simples ouvertes et horizontales X, de longueur supérieure à 5 ; p_1 étant un point extrémité d'une telle courbe X et p_2 étant son unique voisin. Le point p_1 n'est pas simple pour $X \setminus S$ avec $S = \{p_2\}$ ($\subset P = \{p_1, p_2\}$) ; donc p_1 n'est pas P-simple et P n'est pas P-simple (par l'utilisation de la définition, ou directement avec la proposition de la page 90 : le sous-ensemble $S = \{p_2\}$ de P n'est pas simple pour X ainsi P n'est pas P-simple). En revanche, il existe une

séquence (on retire p_1 puis p_2) caractérisant l'ensemble P comme ensemble simple (p_1 simple pour X et p_2-simple pour $X \setminus \{p_1\}$). Cela indique que si un algorithme n'est pas P-simple, il peut néanmoins préserver la topologie. Notons que dans cet exemple, l'ensemble supprimé contenait un MNS ; en effet, le point p_2 n'est pas simple pour X.

Cela montre qu'une détection systématique de la validité d'un algorithme par le test de sa P-simplicité est impossible. En revanche, si l'algorithme a été conçu de façon à ne pas supprimer de MNS (c'est la grande majorité des cas), alors la démarche systématique peut être utilisée pour vérifier sa validité avec les points P-simples (cf. la proposition (lien P-simple et MNS) à la page 88). C'est ce que nous détaillons à la section suivante.

5.5.3 Homotopie et homotopie forte

Nous introduisons des définitions provenant d'une étude sur l'homotopie et l'homotopie forte dans $\mathcal{Z}^3$, décrite dans [BM96].

Définition (ensemble (sous-) n-homotope) : Soient $X \subseteq \mathcal{Z}^3$ et $Y \subseteq X$. L'ensemble Y est *(sous) n-homotope* à l'ensemble X (si $Y = X$ ou) si Y peut être obtenu à partir de X par la suppression de points simples. Si Y est (sous) n-homotope à X, l'ensemble $X \setminus Y$ est appelé un *ensemble (sous) n-simple*.

Nous retrouvons le processus itératif de suppression de points simples, mis en œuvre dans la définition d'ensemble simple, déjà proposée page 88.

Rappelons maintenant la notion d'ensemble fortement homotope :

Définition (ensemble fortement (sous-) n-homotope) : Soient $X \subseteq \mathcal{Z}^3$ et $Y \subseteq X$. L'ensemble Y est *fortement (sous) n-homotope* à l'ensemble X (si $Y = X$ ou) si pour tout ensemble Z tel que $Y \subseteq Z \subseteq X$, Z est (sous) n-homotope à X, l'ensemble $X \setminus Y$ est appelé un *ensemble fortement (sous) n-simple*.

Nous avons alors la propriété suivante :
Propriété 1 **:** Soient $X \subseteq \mathcal{Z}^3$ et $P \subseteq X$. Pour tout ensemble D de $S_n(P)$, $X \setminus D$ est n-homotope à X. En d'autres termes, l'ensemble $X \setminus S_n(P)$ est fortement n-homotope à X.

Nous retrouvons alors les propriétés données dans la section 5.4 :

Proposition 2 **:** Soient $X \subseteq \mathcal{Z}^3$ et $P \subseteq X$: P est P_n-simple $\Leftrightarrow$ tous les sous-ensembles de P sont n-simples.

Proposition 3 **:** Soient $X \subseteq \mathcal{Z}^3$ et $P \subseteq X$: P est P_n-simple $\Leftrightarrow$ P ne contient pas de n-MNS.

Donnons également la proposition suivante [BM96] :

Proposition 4 **:** Soit $X \subseteq \mathcal{Z}^3$. Un ensemble $Y \subseteq X$ est fortement n-homotope à X si et seulement si l'ensemble $P = X \setminus Y$ est un ensemble P_n-simple pour X.

Selon la définition d'ensemble simple, si tous les sous-ensembles d'un ensemble D sont simples, cela signifie que pour tout sous-ensemble S de D, il existe une séquence telle que S soit simple. En fait, la propriété suivante signifie que pour tout sous-ensemble S de D, alors S est simple quelle que soit la séquence le décrivant. Nous énonçons alors la proposition suivante :

Proposition 5 **:** Soient $X \subseteq \mathcal{Z}^3$ et $D \subseteq X$. Tous les sous-ensembles de D sont simples $\Leftrightarrow$ tous les sous-ensembles de D sont simples quelle que soit la séquence les décrivant (voir également la notion d'ensemble héréditairement simple [Kon93]).

Preuve :
Le sens "$\Leftarrow$" est évident.
Démontrons la proposition dans le sens "$\Rightarrow$".
Soient $D = \{x_1, \ldots, x_n\}$. Tous les sous-ensembles de D sont n-simples $\Leftrightarrow$ D ne contient pas de MNS $\Leftrightarrow$ D est D_n-simple (cf. la proposition 3, page 90) $\Leftrightarrow \forall x \in D$, x est D_n-simple (cf. la définition, page 79) $\Leftrightarrow \forall x \in D, \forall S \subseteq D \setminus \{x\} x$, x est n-simple pour $X \setminus S$ (cf. la définition, page 79).

Cela entraine :

- i) On pose $S = \emptyset$. Alors $\forall x \in D$, x est n-simple pour X et $\{x\}$ est un ensemble n-simple,
- ii) On pose $S = \{x_i\}$ Alors $\forall x_j \in D$, x_j est n-simple pour $X \setminus \{x_i\}$ avec j différent de i et $1 \leq j \leq n$; cela est vrai $\forall 1 \leq i \leq n$. Ainsi $\{x_i, x_j\}$ est un ensemble n-simple, quelle que soit la séquence le décrivant (*i.e.* nous avons x_i n-simple pour X et x_j n-simple pour $X \setminus \{x_i\}$) ; et x_j n-simple pour X et x_i n-simple pour $X \setminus \{x_j\}$),
- et ainsi de suite, par récurrence.

□

Retenons alors :

- qu'un ensemble D est dit simple pour X signifie que sa suppression de X préserve la topologie, puisque nous pouvons la simuler par une séquence de suppression de points simples (pour l'objet courant),
- qu'un ensemble D est dit fortement simple pour X signifie que sa suppression de X préserve la topologie, puisque nous pouvons la simuler par toute séquence possible (décrivant D) de suppression de points simples (pour l'objet courant).

Remarque (conséquence de la proposition 5**) :** Nous savons qu'un algorithme ne supprimant pas de MNS ne supprime que des ensembles simples. La propriété précédente souligne le fait qu'il est impossible qu'il existe une séquence telle qu'un de ces sous-ensembles ne soit pas simple par cette séquence le décrivant.

Supposons qu'un algorithme soit conçu de façon à ne supprimer que des ensembles simples et que ces ensembles soient décrits par une séquence donnée : soit cet algorithme retire des MNS et il faut effectuer des tests supplémentaires pour vérifier si la topologie est préservée (voir

l'exemple dans la section 5.5.2) ; soit cet algorithme ne retire pas de MNS et dans ce cas les ensembles qu'il retire sont simples et tous leurs sous-ensembles le sont et cela quelle que soit la séquence les décrivant (cf. proposition 5). La séquence imposée lors de la conception est inutile ; cela peut simplifier alors la preuve de sa validité, permettre la simplication de l'algorithme et a posteriori son implémentation.

5.5.4 Récapitulatif concernant une démarche systématique

Considérons un algorithme A. Soit P, l'ensemble des points retirés par A.

Si A a été conçu de façon à ne pas supprimer de MNS (voir dans la preuve de sa validité) alors l'ensemble P est P-simple (cf. la proposition 3). C'est le cas si et seulement si tous les points de P sont P-simples. Nous pouvons alors le vérifier en examinant si tout point x de tout objet X pouvant être supprimé par A est P-simple. Deux cas de figure peuvent arriver :

- soit, effectivement, tout point x de tout objet X pouvant être supprimé par A est P-simple ; dans ce cas, l'algorithme préserve effectivement la topologie (cf. les sections 6.5.1, 6.5.2, 6.6.2.1 et 6.7.4).
- soit, il existe au moins une configuration C de point central x telle que x est supprimé par A et x n'est pas P-simple : il y a alors incohérence entre la conception de l'algorithme A et la preuve de sa validité.

Concrètement, pour deux algorithmes fortement parallèles (cf. sections 6.7.2 et 6.7.3) vérifiant l'hypothèse de conception sans suppression de MNS ; nous avons trouvé une configuration C pour chacun d'entre eux, telle qu'un point x de C ne soit pas P-simple ; cette configuration C est telle qu'en fait un MNS est supprimé et que la topologie n'est pas préservée ; les preuves apportées par les concepteurs de ces algorithmes sont alors fausses. Insistons sur le fait que si l'hypothèse de conception sans suppression de MNS n'est pas vérifiée, il est possible que l'algorithme préserve la topologie (cf. l'exemple de la section 5.5.2).

Retenons finalement que dans le cas de la vérification systématique, il nous faut réaliser un examen étendu du voisinage ou un étiquetage préliminaire, ce qui n'est généralement pas possible par simulation par ordinateur. C'est ce qui nous a motivé à introduire un ensemble P^x permettant de dériver la notion de point P^x-simple au chapitre suivant.

5.6 Conclusion

Ce chapitre présente le concept fondamental de point P-simple. C'est la notion adéquate pour la suppression en parallèle de points en préservant la topologie.

Il faut retenir que de nombreux algorithmes ont été proposés ; ceux-là opèrent par suppression de points simples et utilisent une certaine stratégie. Les auteurs vérifient généralement que leurs algorithmes préservent la topologie en montrant que leurs algorithmes ne suppriment pas de MNS. Un algorithme supprimant des points P-simples préserve forcément la topologie ; aucune preuve supplémentaire n'est requise. En fait, la conception d'un algorithme supprimant des points P-simples suit une démarche inverse de celle d'un algorithme classique.

Dans le chapitre suivant, nous définissons un ensemble P^x à partir d'un ensemble P. Puis, nous proposons de nouveaux algorithmes de squelettisation basés sur la suppression de points P^x-simples. Ceux-là sont tels qu'ils n'utilisent ni une phase préalable d'étiquetage, ni l'examination d'un voisinage étendu, contrairement à l'algorithme basé sur la suppression de points P-simples, décrit à la section 5.3.

Chapitre 6

P-simplicité d'algorithmes et nouveaux algorithmes basés sur les points P-simples

6.1 Introduction

Dans le chapitre précédent, nous avons rappelé un algorithme de squelettisation basé sur la suppression de points P-simples. Celui-là consistait en la répétition jusqu'à stabilité des deux étapes suivantes : une phase préliminaire d'étiquetage des points appartenant à P, ce qui pouvait être fait dans le 26-voisinage de chaque point de l'image et une seconde phase de détection locale et parallèle des points P-simples (en utilisant la caractérisation de la page 80), suivie par la suppression en parallèle de ces points. Retenons qu'une fois que P est connu, il est possible de détecter localement les points P-simples.

Une alternative au schéma précédent consiste alors en la répétition jusqu'à stabilité de la détection en parallèle des points P-simples et de leur suppression, mais en s'autorisant à examiner un voisinage étendu afin de savoir quels points y dans le 26-voisinage de x appartiennent à P. Sous l'hypothèse de pouvoir déterminer si un point x appartient à P ou non, par l'examen d'un voisinage $3 \times 3 \times 3$ centré en x, nous comprenons que l'alternative au premier schéma nécessite l'examen d'un voisinage $5 \times 5 \times 5$ centré en x ; pour déterminer si le point x est P-simple, cela sans phase préliminaire d'étiquetage (en fait, la première hypothèse pourrait être adaptée dans le cas des algorithmes fortement parallèles, car ces derniers nécessitent un voisinage plus grand que celui de $3 \times 3 \times 3$ pour décider de la suppressibilité d'un point). Une autre alternative consiste à utiliser des listes de contours (cf. section 5.3.2.1).

Nous proposons dans ce chapitre une nouvelle approche. À partir d'un ensemble P, nous définissons localement en un point x d'un objet X, un ensemble P^x. Puis, nous proposons un schéma fondé sur la suppression de points P^x-simples. Nous montrons qu'un tel algorithme préserve la topologie. L'utilisation des points P^x-simples n'utilisant ni une phase préliminaire d'étiquetage, ni un voisinage étendu, il nous est possible de comparer un algorithme supprimant des points P^x-simples avec des algorithmes déjà existants utilisant les mêmes voisinages.

Dans notre étude, il nous a été possible de proposer un ensemble P tel que l'algorithme A supprimant des points P^x-simples retire également les points supprimés par un algorithme exis-

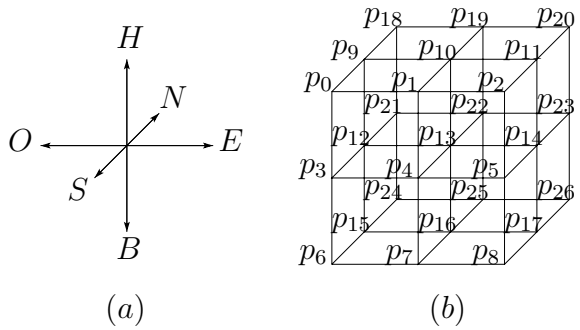


FIG. 6.1 – (a) *Les* 6 *directions principales ,* (b) *les notations utilisées.*

tant A'. Cela implique, par propriété des points P-simples, que A et A' préservent la topologie, et que A retire plus de points que A', au moins lors de la première sous-itération de suppression.

Nous commençons d'abord par définir un ensemble P^x à partir d'un ensemble P et donnons quelques propriétés. Dans la section 6.5, nous proposons un algorithme curviligne en 6 sous-itérations retirant, à la fois, au moins les points supprimés par l'algorithme de Palágyi et Kuba en 6 sous-itérations et les points supprimés par la variante curviligne de l'algorithme de Gong et Bertrand. Ensuite, dans la section 6.6, nous proposons un algorithme en 12 sous-itérations supprimant au moins les points supprimés par un algorithme de Palágyi et Kuba. Enfin nous étudions trois algorithmes fortement parallèles dans la section 6.7, celui de Ma (section 6.7.2), celui de Ma et Sonka (section 6.7.3) et celui de Manzanera, et al. (section 6.7.4). Nous montrons, par l'utilisation des points P-simples, que les deux premiers ne préservent pas la topologie. Nous montrons que l'algorithme de Manzanera, et al., quant à lui, préserve la topologie.

6.1.1 Notations utilisées

Les 6-voisins de x déterminent les six directions principales (Fig. 6.1 (a)) : $Haut$, Bas, $Nord$, Sud, $Ouest$, Est ; respectivement dénotés par H, B, N, S, O et E. Tout point de $N^*_{26}(x)$ caractérise une direction parmi les 26 qu'il est possible d'obtenir à partir des 6 principales, par exemple SO, HSO ... Notons Dir, l'une de ces 26 directions. Le point dans $N^*_{26}(x)$ selon la direction Dir est appelé le *voisin* Dir *de* x et est noté $Dir(x)$. Dans la suite, les points de $N_{26}(x)$ sont souvent notés par p_i ; $i = 0, \ldots, 26$ (Fig. 6.1 (b)) ; par exemple, p_0 est le voisin HSO de p_{13}, *i.e.* $p_0 = HSO(p_{13})$ (ou $Haut - Sud - Ouest(p_{13})$).

Considérons une direction α parmi les six directions principales (Fig. 6.1 (a)). La direction opposée à α sera notée $\overline{\alpha}$.
Nous désignons par $N^6_\alpha(x)$ (resp. $N^{18}_\alpha(x)$), les quatre 6-voisins (resp. les quatre 18-voisins) qui appartiennent à la fenêtre 3×3 perpendiculaire à la direction α et de point central x (en fait, $N^6_\alpha(x) = N^*_6(x) \backslash \{\alpha(x), \overline{\alpha}(x)\}$). Enfin, $N_\alpha(x) = N^6_\alpha(x) \cup N^{18}_\alpha(x) \cup \{x\}$ et $N^*_\alpha(x) = N_\alpha(x) \backslash \{x\}$.

Remarque : Dans la suite, les points de P seront représentés par des étoiles.

6.2 Points P^x-simples

Dans cette section, nous définissons un ensemble P^x, à partir d'un ensemble $P \subseteq X$ et pour tout point x de X, avec $X \subseteq \mathcal{Z}^3$. Nous montrons qu'un algorithme supprimant des points P^x-simples préserve la topologie. Enfin, un exemple illustre des différences entre les points P-simples et les points P^x-simples.

6.2.1 Ensemble P^x

Dans la suite, nous considérons un sous-ensemble X de $\mathcal{Z}^3$ et un sous-ensemble P de X. Pour tout x de $\mathcal{Z}^3$, nous considérons une famille finie de couples de sous-ensembles de $\mathcal{Z}^3$ $(B^k(x), W^k(x))$ avec $k \in [1, l]$, telle que $B^k(x) \cap W^k(x) = \emptyset$ et x appartient à $B^k(x)$.

Nous disons que P est "caractérisé" par une telle famille (B^k, W^k) si $P = \{x \in \mathcal{Z}^3; \exists k \in [1, l]$ tel que $B^k(x) \subseteq X$ et $W^k(x) \subseteq \overline{X}\}$. En fait, P correspond à une transformation Hit or Miss de X par (B^k, W^k) [Ser82] [Jon00].

Un algorithme de squelettisation utilisant la notion de points P-simples doit décider si un point x est P-simple ou non : dans le but de tester les quatre conditions de P-simplicité (cf. la proposition 5.2, page 80), il doit tester si le point x appartient à P, et de plus il doit tester si les points y de $N^*_{26}(x)$ appartiennent à P (voir les troisième et quatrième conditions de P-simplicité). Un tel algorithme peut se dérouler selon différentes façons pour caractériser les points appartenant à P et les points étant P-simples :

- La première stratégie consiste en la répétition de deux étapes [Ber95a]. Durant la première, les points appartenant à P sont étiquetés, par accès à $B^k(x)$, et à $W^k(x)$, pour tous les points x de $\mathcal{Z}^3$; cela étant à faire au plus pour l couples $(B^k(x), W^k(x))$. Pendant la seconde étape, les quatre conditions de P-simplicité de la proposition 5.2 (page 80) doivent être testées pour tous les points de P : le test de ces quatre conditions est rendu possible grâce à l'étiquetage préalable réalisé lors de la première étape. Le schéma correspondant à cette stratégie est décrit par l'algorithme 6 dans la section 6.2.3. Notons l'utilisation possible de listes de contours pour simuler de façon efficace cet étiquetage, pour une approche particulière de squelettisation [MF98] [BM00].
- La deuxième stratégie consiste en une unique étape de détection de points P-simples. Durant le test de P-simplicité de chaque point x de X, il est permis d'accéder à $B^k(z)$ et à $W^k(z)$ pour tout point $z \in N_{26}(x)$. Ainsi, cette stratégie exige généralement l'examen d'un plus grand voisinage que $N_{26}(x)$. Le schéma correspondant à cette stratégie est décrit par l'algorithme 7 dans la section 6.2.3.
- Nous proposons maintenant une nouvelle stratégie qui n'exige ni une phase préliminaire d'étiquetage, ni l'examen d'un voisinage étendu. Cette stratégie utilise les notions d'appartenance à un ensemble P^x que nous allons introduire maintenant. Le schéma correspondant à cette stratégie est décrit par l'algorithme 8 dans la section 6.2.3.

Pour plus de détails, se reporter à la section 6.2.3, dans laquelle sont décrites ces trois stratégies. Notons qu'elles sont données ici de façon générale, et doivent être adaptées selon la stratégie choisie, *i.e.* par exemple l'approche directionnelle (cf. section 6.2.3).

Définition (ensemble P^x) :
Pour chaque point x de X, nous définissons un nouveau sous-ensemble P^x de $\mathcal{Z}^3$, déterminé par $P^x = \{y \in N_{26}(x);\ \exists k \in [1,l]$ tel que $B^k(y) \cap N_{26}(x) \subseteq X$ et $W^k(y) \cap N_{26}(x) \subseteq \overline{X}\}$. Nous avons $P^x \supseteq P \cap N_{26}(x)$. En fait, P^x est constitué des points y de $N_{26}(x) \cap X$ qui "peuvent appartenir" à P, par la seule inspection d'appartenance à X ou à $\overline{X}$ des points appartenant à $[B^k(y) \cup W^k(y)] \cap N_{26}(x)$.

Introduction (point P^x-simple) :
Un point x de P^x est *P^x-simple* s'il est P-simple en remplaçant P par P^x dans la définition de la page 79. Il ne s'agit pas d'une nouvelle définition, mais d'une classe particulière de points P-simples.

Remarque : Pour tout y de $N_{26}(x)$ tel que $B^k(y) \cup W^k(y) \subseteq N_{26}(x)$ pour tout k dans $[1,l]$, alors $y \in P$ si et seulement si $y \in P^x$. Dans la suite, nous supposons que P est tel que $B^k(x) \cup W^k(x) \subseteq N_{26}(x)$, pour tout point x de X et pour tout k dans $[1,l]$; par conséquent $x \in P$ si et seulement si $x \in P^x$.

Avec l'hypothèse précédente, nous avons la propriété suivante :

Propriété : Tout algorithme supprimant des points x P^x-simples préserve la topologie.

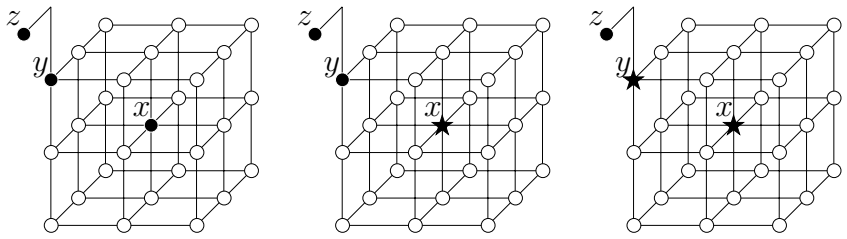
Preuve : D'après la définition d'un point P-simple, si deux sous-ensembles P_1 et P_2 de X sont tels que $P_1 \subseteq P_2$, alors si x est P_2-simple et si x appartient à P_1, alors x est P_1-simple. Nous remarquons d'une part que $P \cap N_{26}(x)$ est inclus dans P^x, et d'autre part que x appartient à P^x si et seulement si x appartient à P (cf. remarque précédente) ; par conséquent si x est P^x-simple alors x est P-simple. Nous savons que tout algorithme supprimant des points P-simples préserve la topologie, et par définition, tout algorithme supprimant des sous-ensembles de points P-simples préserve la topologie. Cela prouve que tout algorithme supprimant des points P^x-simples préserve la topologie. □

Rappelons que P^x est défini à partir de P, et que P est généralement déterminé par un ensemble de masques ou motifs (également appelés *templates*), utilisant le 26-voisinage d'un point pour savoir s'il est supprimable. Nous développons actuellement de nouveaux algorithmes fortement parallèles de squelettisation, utilisant cette notion de point P^x-simple. En fait, nous adaptons la définition de l'ensemble P^x proposée dans cette section, au voisinage mis en jeu dans les templates utilisées dans de tels algorithmes.

6.2.2 Exemple

Dans cette section, nous donnons un exemple qui illustre l'existence de points x qui sont P-simples mais non P^x-simples, pour le même sous-ensemble P, ce qui explique alors qu'un algorithme supprimant des points P^x-simples supprime moins de points qu'un algorithme supprimant des points P-simples ; mais ce dernier utiliserait alors soit une phase préliminaire d'étiquetage soit l'examen d'un voisinage étendu (c'est le "prix à payer" en contrepartie).

Nous proposons de considérer le sous-ensemble P tel que $P = \{x \in X$; le voisin $Haut{-}Sud$ de x appartient à $\overline{X}\}$. Pour tout x de $\mathcal{Z}^3$, nous avons $B^1(x) = \{x\}$, $W^1(x) = \{HS(x)\}$, et $l = 1$; nous écrivons $B(x) = B^1(x)$ et $W(x) = W^1(x)$.



(a) (b) (c)

FIG. 6.2 – Configuration initiale (a). Le point x est P-simple (b), n'est pas P^x-simple (c).

Considérons la figure 6.1 (b). Nous notons p_{13} par x. Soit U l'ensemble des points de $N_{26}(x) \cap X$ pour lesquels le voisin $Haut - Sud$ appartient à $N_{26}(x) \cap X$, *i.e.* $U = \{p_{12}, \ldots, p_{17}, p_{21}, \ldots, p_{26}\} \cap X$. Soit $V = [N_{26}(p_{13}) \cap X] \setminus U$, *i.e.* $V = \{p_0, \ldots, p_{11}, p_{18}, p_{19}, p_{20}\} \cap X$. Nous avons :

- Pour $y \in U$, $y \in P^x$ si et seulement si $B(y) \cap N_{26}(x)(= \{y\})$ est inclus dans X (toujours vérifié pour tout $y \in U$), et si $W(y) \cap N_{26}(x)(= \{HS(y)\})$ est inclus dans $\overline{X}$; par conséquent pour $y \in U$, nous avons $y \in P^x$ si et seulement si $HS(y)$ appartient à $\overline{X}$.
- Pour $y \in V$, $y \in P^x$ si et seulement si $B(y) \cap N_{26}(x)(= \{y\})$ est inclus dans X (toujours vérifié pour tout $y \in V$), et si $W(y) \cap N_{26}(x)(= \emptyset)$ est inclus dans $\overline{X}$ (toujours vérifié pour tout $y \in V$) ; par conséquent $y \in P^x$ pour tout $y \in V$.

En résumé, pour tout point x de X, $P^x = \{y \in U; HS(y) \in \overline{X}\} \cup V$.

Considérons la configuration décrite à la figure 6.2 (a). Les points de P (resp. P^x) sont représentés par une étoile à la figure 6.2 (b) (resp. figure 6.2 (c)). Dans la figure 6.2 (b), le point x appartient à P car x appartient à X et le voisin $Haut - Sud$ de x appartient à $\overline{X}$. Le point y appartient à $R(= X \setminus P)$ car $z(= HS(y))$ appartient à X et $W(y) \not\subseteq \overline{X}$. Dans ce cas, x est un point P-simple. Dans la figure 6.2 (c), le point x appartient à P^x car x appartient à U et le voisin $Haut - Sud$ de x appartient à $\overline{X}$. Le point y appartient à P^x car y appartient à V. Dans ce cas, x n'est pas un point P^x-simple car les première et troisième conditions de P^x-simplicité ne sont pas vérifiées : $T_{26}(x, R^x) = 0$ et il n'y a pas de point de R^x 26-adjacent à x et à y, avec $R^x = X \setminus P^x$.

6.2.3 Schémas d'algorithmes de squelettisation basés sur la suppression de points P-simples

Dans cette section, nous avons choisi de rappeler les trois schémas généraux de squelettisation mis en œuvre par des algorithmes supprimant des points P-simples (schémas décrits par les algorithmes 6, 7 et 8). En fait, ces trois schémas sont proposés d'une façon très générale et doivent être adaptés à nos besoins. Par exemple, lorsque le schéma indique "supprimer en parallèle tous les points P-simples", il faut tenir compte de l'approche retenue : s'il s'agit de l'approche 6 sous-itérations par exemple, alors nous retirons tous les points selon une des six directions considérées, puis successivement pour les autres directions, jusqu'à stabilité. De plus l'utilisation de listes de contours permet une implémentation plus efficace pour ces schémas.

Répéter

a) Étiqueter en parallèle les points x appartenant à P

b) Supprimer en parallèle les points P-simples x non extrémités
(nous devons accéder à $N_{26}(x)$ pour tout x retenu en a))

Jusqu'à stabilité

→ phase préliminaire d'étiquetage

Algorithme 6: *Premier schéma de squelettisation par suppression de points P-simples : utilisation d'une phase préliminaire d'étiquetage.*

Répéter

Supprimer en parallèle les points P-simples x non extrémités

Jusqu'à stabilité

→ un voisinage étendu doit être examiné

Algorithme 7: *Deuxième schéma de squelettisation par suppression de points P-simples : utilisation d'un voisinage étendu.*

Répéter

Supprimer en parallèle les points P^x-simples x non extrémités

Jusqu'à stabilité

→ Ce schéma n'exige :
- ni une phase préliminaire d'étiquetage
- ni l'examen d'un voisinage étendu

Algorithme 8: *Troisième schéma de squelettisation par suppression de points P^x-simples : n'utilise ni une phase préliminaire d'étiquetage, ni l'examen d'un voisinage étendu.*

6.3 Mode opératoire de tests et de propositions d'algorithmes

Le but de ce chapitre est de pouvoir proposer de nouveaux algorithmes pour images 3D binaires retirant au moins les points supprimés par des algorithmes déjà proposés.

Pour un algorithme A donné, nous vérifions d'abord qu'il préserve bien la topologie. Généralement, cela a déjà été proposé dans l'article proposant l'algorithme, par l'utilisation des MNS (preuve combinatoire, "toutes" les configurations sont testées). En utilisant les points P-simples, nous pouvons vérifier que A préserve la topologie de la façon suivante :

- Soit P l'ensemble correspondant exactement aux points supprimés par A. Nous prouvons que l'ensemble P des points supprimés par A est P-simple si les points retirés par A sont P-simples ; si c'est le cas alors l'algorithme préserve bien la topologie. Par rapport à ce qui a déjà été indiqué, il s'agit encore d'une preuve combinatoire ; nous "testons" toutes les configurations possibles, néanmoins, cela semble plus "aisé" à réaliser avec les points P-simples qu'avec les MNS (cf. section 5.5.1).
- Une seconde possibilité consiste à tester si l'algorithme A est P^x-simple ; cela pouvant être simulé par ordinateur (P étant l'ensemble des points pouvant être supprimés par A). Si c'est le cas, alors l'algorithme A préserve la topologie. Notons que les algorithmes testés dans ce chapitre s'ils préservent la topologie sont P^x-simples.

Ce chapitre rend compte des deux possibilités : preuve écrite "manuellement" par l'utilisation des points P-simples et simulation par ordinateur par l'utilisation des points P^x-simples.

Une variante de la deuxième possibilité consiste à proposer un ensemble P' tel que l'algorithme A' supprimant les points P'^x-simples (voir section 6.2) supprime au moins les points retirés par un algorithme A. De cette façon, d'une part A et A' préservent la topologie (preuve par ordinateur de la validité de A) ; et d'autre part l'algorithme A' retire plus ou le même nombre de points que l'algorithme A, au moins lors de la première itération de suppression, ce qui laisse à penser qu'un squelette peut être obtenu par A' plus rapidement qu'avec A, pour un même objet initial. En fait, il est difficile de généraliser, étant donné que le squelette n'a pas le même centrage selon les deux algorithmes. La démarche permettant de trouver P par raffinements successifs sera décrite en détails pour deux algorithmes aux sections 6.5.3 et 6.6.3.

6.4 Algorithmes à base de sous-mailles

Considérons un algorithme de squelettisation utilisant 8 sous-mailles. Les 8 sous-mailles sont choisies de façon à ce que tout couple de deux points différents x et y d'une même sous-maille soit tel que $y \notin N^*_{26}(x)$. Supposons que cet algorithme retire en parallèle les points simples d'une même sous-maille. Nous étiquetons à P tous les points de la sous-maille examinée. Il n'existe pas d'autres points étiquetés à P, dans le voisinage $N^*_{26}(x)$ du point x testé de la sous-maille. Dans ce cas, nous avons l'équivalence entre la P-simplicité et la simplicité de x ; *i.e.* un point simple retiré par cet algorithme est en fait un point P-simple, et vice versa. Cela se rapproche de la remarque pour laquelle P est réduit en un point (remarque de la page 88) ; ici c'est P restreint dans $N_{26}(x)$ qui est réduit en un point et nous utilisons le fait que la caractérisation d'un point P-simple x est locale - *i.e.* s'effectue dans $N^*_{26}(x)$ (voir la proposition page 80).

En conséquence, nous ne pouvons pas proposer d'ensemble P' de telle façon à ce qu'un algorithme supprimant des points P-simples retire plus de points qu'avec l'algorithme proposé.

Retenons qu'il est plus facile de démontrer qu'un algorithme utilisant 8 sous-mailles, telles que deux points x et y d'une même sous-maille sont tels que $y \notin N^*_{26}(x)$, préserve la topologie, qu'avec les autres types d'algorithmes. À noter le renouveau actuel d'intérêt pour de tels algorithmes ; la recherche se portant alors sur de nouvelles définitions de points extrémités afin de pallier l'aspect zigzag, montré en 2D dans l'annexe A, et également sur un nombre différent de sous-mailles. Voir également l'annexe B, pour des algorithmes à base de sous-mailles pour images 3D.

Nous ne reviendrons pas sur l'approche sous-mailles par la suite.

6.5 Algorithmes en 6 sous-itérations directionnelles

Nous montrons que l'algorithme de Gong et Bertrand et celui de Palágyi et Kuba sont P-simples, puis nous proposons un algorithme qui permet de supprimer au moins les points supprimés par les deux précédemment cités.

6.5.1 Étude de la P-simplicité de l'algorithme de Gong et Bertrand

6.5.1.1 Rappels de l'algorithme de Gong et Bertrand (GOBE) [GB90]

Cet algorithme consiste en la répétition jusqu'à stabilité de 6 sous-itérations directionnelles $Haut$, $Nord$, Est, Bas, Sud, $Ouest$ de suppression parallèle de points. Pour une direction α (parmi les 6 principales précédemment citées), l'algorithme élimine en parallèle les points qui vérifient les conditions suivantes et qui ne sont pas extrémités (voir également la figure 6.3 ; des notations ont été introduites à la section 6.1.1) :

$$\begin{cases} (G_1) : \alpha(x) \in \overline{X}, \\ (G_2) : \overline{\alpha}(x) \in X, \\ (G_3) : \forall y \in N_\alpha^6(x), \text{ si } y \in \overline{X} \text{ alors } \alpha(y) \in \overline{X}, \\ (G_4) : \forall y \in N_\alpha^6(x), \forall z \in N_\alpha^{18}(y) \cap N_\alpha^6(x), \text{ soit } t \text{ le seul élément de } N_\alpha^6(z) \cap N_\alpha^6(y) \cap N_\alpha^{18}(x), \\ \qquad \text{si } y \in \overline{X} \text{ et } z \in \overline{X} \text{ et } t \in \overline{X} \text{ alors } \alpha(t) \in \overline{X}. \end{cases}$$

Notons que dans la condition (G_4), $\alpha(y) \in \overline{X}$ et $\alpha(z) \in \overline{X}$ (d'après (G_3)).

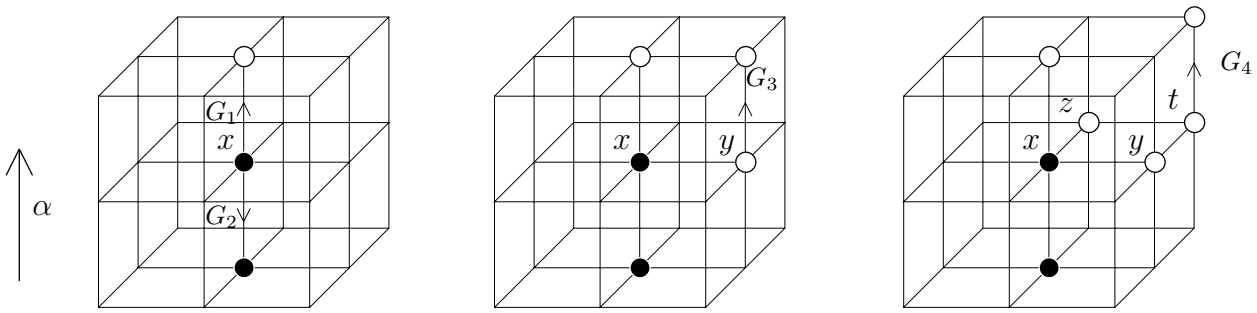


FIG. 6.3 – *Conditions imposées dans l'algorithme de Gong et Bertrand (*GOBE*) (*$\alpha = Haut$*).*

Pour plus d'informations, se reporter à la description de cet algorithme dans la section B.2.2, page 293.

6.5.1.2 P_{26}-simplicité de GOBE

Propriété : L'algorithme de Gong et Bertrand (GOBE) est P_{26}-simple.

Nous démontrons maintenant cette propriété ; la démonstration a déjà été proposée par G. Bertrand dans [Ber95c].

L'intérêt de cette démonstration est de comprendre comment utiliser les quatre conditions de P-simplicité d'un point et de pouvoir les appliquer pour un ensemble de points susceptibles d'être supprimés par un algorithme A, et ensuite de montrer que cet ensemble est P-simple, ce qui entraine que l'algorithme A l'est également. En fait, cette preuve est assez compacte, comparée à celles qui viendront dans la suite du chapitre ; d'où l'intérêt d'en comprendre le principe sur cet algorithme.

Preuve :

Soit $G^\alpha(X)$, l'ensemble des points $x \in X$ vérifiant les quatre conditions précédentes (pour la direction α). Pour prouver que l'algorithme GOBE préserve la topologie, il est suffisant de montrer que :

$$\forall X \subset \mathcal{Z}^3,\ P = G^\alpha(X) \text{ est } P_{26}\text{-simple.}$$

Soient $P = G^\alpha(X),\ R = X \setminus P$ et $x \in P$.

Remarque 0 (rmq_0) : Si x appartient à P, alors $\overline{\alpha}(x) \in R$ (car $\overline{\alpha}(x) \in X\ (G_2)$, et comme $\alpha(\overline{\alpha}(x)) = x \in X$, alors $\overline{\alpha}(x)$ ne vérifie pas (G_1)).

Remarque 1 (rmq_1) : Si x appartient à X, et si $\overline{\alpha}(x) \in X$, alors $\overline{\alpha}(x) \in R$ (comme $\alpha(\overline{\alpha}(x)) = x \in X$, alors $\overline{\alpha}(x)$ ne vérifie pas (G_1)).

Montrons qu'un point x appartenant à $G^\alpha(X)$ est P_{26}-simple (*i.e.* qu'il vérifie les quatre conditions de P-simplicité, cf. les propositions aux pages 80 et 82).

Condition (C_1) :

1) $\forall y \in N^*_\alpha(x) \cup N^*_\alpha(\overline{\alpha}(x)), y$ est 26-adjacent à $\overline{\alpha}(x)$.

2) • Soit $y \in N^6_\alpha(\alpha(x)) \cap X$. Alors $z = \overline{\alpha}(y)$ appartient à X (sinon $x \notin P$ d'après (G_3), car $y \in X$). De plus $z \in R\,(rmq_1)$.

• Soient $y \in N^{18}_\alpha(\alpha(x)) \cap X$ et $z = \overline{\alpha}(y)$.

Si $z \in X$ alors $z \in R\,(rmq_1)$.

Si $z \in \overline{X}$ alors il existe $u \in N^6_\alpha(x) \cap N^6_\alpha(z)$ tel que $u \in X$ (sinon $x \notin P$ d'après (G_4), car $y \in X$). Comme $z \in N^6_\alpha(u) \cap \overline{X}$ et $y = \alpha(z) \in X$, alors u ne vérifie pas (G_3) et $u \in R$.

Dans tous ces cas (1) et 2)), les points de $N^*_{26}(x) \cap R$ sont 26-adjacents à $\overline{\alpha}(x)$ qui appartient à $R\ (rmq_0)$, ou à un point z ou u appartenant à R et 26-adjacent à $\overline{\alpha}(x)$.

Nous avons alors $T_{26}(x, R) = 1$.

Condition (C_2) :

$\forall y \in N^6_\alpha(x) \cap \overline{X}, \alpha(y) \in \overline{X}\ (G_3)$, or $\alpha(y)$ est 6-adjacent à $\alpha(x)$ qui appartient à $\overline{X}\ (G_1)$ (et qui est 6-adjacent à x). De plus $\overline{\alpha}(x) \in X\ (G_2)$. Nous avons alors $T_6(x, \overline{X}) = 1$.

Condition (C_3) :

$\forall y \in N^*_\alpha(x) \cup N^*_\alpha(\overline{\alpha}(x)), \exists z = \overline{\alpha}(x) \in R\,(rmq_0)$ tel que z est 26-adjacent à x et à y. $\forall y \in$

$N_\alpha(\alpha(x))$, si $y \in P$ alors $y \neq \alpha(x)$ car $\alpha(x) \in \overline{X}(G_1)$ et $\exists z = \overline{\alpha}(y) \in R\,(rmq_0)$, 26-adjacent à x ($z \in N^*_\alpha(x)$) et à y.

Condition (C_4) :
$\forall y \in N^*_6(x) \cap P, y \in N^6_\alpha(x)$ car $\alpha(x) \in \overline{X}\,(G_1)$ et $\overline{\alpha}(x) \in R\,(rmq_0)$. $\forall y \in N^6_\alpha(x) \cap P, \alpha(y) \in \overline{X}\,(G_1)$ et $\exists z = \alpha(y) \in \overline{X}$ et $\exists t = \alpha(x) \in \overline{X}(G_1)$ tel que $\{x, y, z, t\}$ constitue un carré unité.

(N.B. : lire "$y \in P$" à la place de "$y \in \overline{X}$" dans le test de la condition 4 du paragraphe 6 dans [Ber95c])
□

6.5.2 Étude de la P-simplicité de l'algorithme de Palágyi et Kuba en 6 sous-itérations

6.5.2.1 Rappels de l'algorithme de l'algorithme de Palágyi et Kuba en 6 sous-itérations (PAKU6) [PK98a]

Cet algorithme consiste en la répétition jusqu'à stabilité de 6 sous-itérations directionnelles $Haut, Bas, Nord, Sud, Est, Ouest$ de suppression parallèle de points. Pour une direction donnée, l'algorithme élimine en parallèle les points dont le voisinage vérifie au moins l'une des templates proposées dans le masque (lié à la direction donnée).

À la figure 6.4, est représentée une partie des templates du masque pour la direction $Haut$; il faut également prendre en compte les templates obtenues par les rotations selon l'axe $HautBas$ (90, 180 et 270 degrés). L'ensemble des masques pour la direction α est noté $\mathcal{T}_\alpha$.

Pour plus d'informations, se reporter à la description de cet algorithme dans la section B.2.1, page 288.

6.5.2.2 P_{26}-simplicité de PAKU6

Propriété : L'algorithme de Palágyi et Kuba en 6 sous-itérations est P_{26}-simple.

Preuve :
Nous considérons $\alpha = Haut$, lorsque nous nous reportons aux templates (Fig. 6.4).

Nous pouvons classer les templates en deux catégories : les templates pour lesquelles $\overline{\alpha}(x) \in X$ (templates M_1 à M_4) et les templates pour lesquelles $\overline{\alpha}(x) \in \overline{X}$ (templates M_5 et M_6). Nous nous apercevons que certains points apparaissent dans chaque template d'une même catégorie, définissant ainsi les *motifs récurrents* MR_1 et MR_2 pour les deux catégories successives (voir Fig. 6.5).

Soit $PK^\alpha_6(X)$, l'ensemble des points x de X pouvant vérifier au moins l'une des templates (pour la direction α). Pour prouver que l'algorithme PAKU6 préserve la topologie, il est suffisant de montrer que :

$$\forall X \subset \mathcal{Z}^3,\ P = PK^\alpha_6(X) \text{ est } P_{26}\text{-simple}.$$

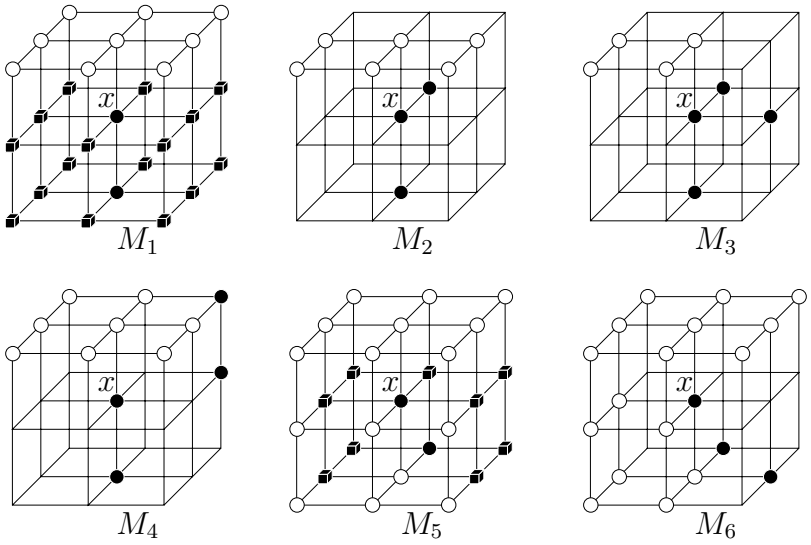


FIG. 6.4 – *Masque Haut : templates utilisées lors de la sous-itération de suppression selon la direction $Haut$, de l'algorithme* PAKU6 *(à rotations près selon l'axe $HautBas$). Au minimum un point représenté par un cube doit appartenir à l'objet dans les masques M_1 et M_5.*

Soient $P = PK_6^{\alpha}(X)$, $R = X \setminus P$ et $x \in P$.

Remarque 1 (rmq_1) **:** Si $x \in P$, alors $\alpha(x) \in \overline{X}$ (les points appartenant à P appartiennent au sous-ensemble des points de bord selon la direction α).

Remarque 2 (rmq_2) **:** Si $x \in X$, et si $\overline{\alpha}(x) \in X$ alors $\overline{\alpha}(x) \in R$ (car $\alpha(\overline{\alpha}(x)) = x \notin \overline{X}$, $\overline{\alpha}(x)$ ne vérifie pas rmq_1).

Remarque 3 (rmq_3) **:** Si x vérifie l'une des templates, alors $\forall y \in N_{\alpha}^{6}(x)$, si $y \in \overline{X}$ alors $\alpha(y) \in \overline{X}$.

Remarque 4 (rmq_4) **:** Si x vérifie l'une des templates M_2 ou M_3, $\forall y \in N_{\alpha}^{6}(\alpha(x))$, si $y \in X$ alors $\overline{\alpha}(y) \in X$. Dans ce cas, $\overline{\alpha}(y) \in R$ (rmq_2).

Remarque 5 (rmq_5) **:** Dans les motifs M_5 ou M_6 (voir MR_2), le point $u_1 \in R$ (il ne peut ni vérifier M_1, M_4, M_5 ou M_6 car $x \in X$, ni vérifier M_2, M_3 car $u(= \overline{\alpha}(x)) \in \overline{X}$).

Remarque 6 (rmq_6) **:** Dans le motif M_6, le point $u_2 \in R$ (voir explications rmq_5).

Condition (C_1) **et** (C_3) **:**
Pour M_1 : $\forall y \in N_{26}^{*}(x) \cap X$, y est 26-adjacent à $\overline{\alpha}(x) \in R$ (rmq_2).
Pour M_4 : $\forall y \in N_{26}^{*}(x) \cap X$, y est 26-adjacent à $\overline{\alpha}(x) \in R$ (rmq_2) ou à $z_4 \in R$ (car $y_3 \in X$ et (rmq_2)) (voir MR_1), et z_4 est 26-adjacent à $\overline{\alpha}(x)$.

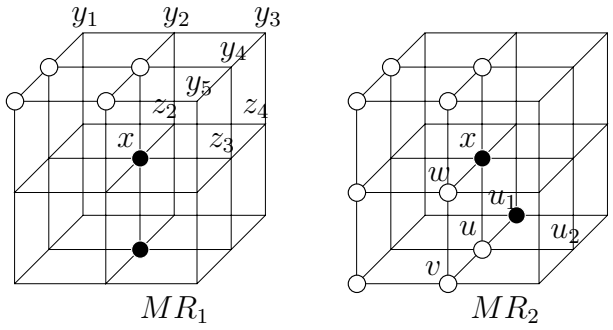


FIG. 6.5 – *Motifs récurrents (algorithme* PAKU6*) dans les masques M_1 à M_4 : MR_1 ; et dans les masques M_5 à M_6 : MR_2. Sont également indiquées les notations utilisées dans la démonstration.*

Pour M_5 : $\forall y \in N_{26}^*(x) \cap X$, y est 26-adjacent à $u_1 \in R$ (rmq_5) (voir MR_2).
Pour M_6 : $\forall y \in N_{26}^*(x) \cap X$, y est 26-adjacent à $u_1 \in R$ (rmq_5) ou à $u_2 \in R$ (rmq_6) et u_1 est 26-adjacent à u_2.
Pour M_2,

$\diamond$ si $z_2 \in P$ (voir MR_1) $\overset{rmq_1}{\Rightarrow}$ $y_2 \in \overline{X}$. Soit $A = \{N_\alpha^6(\alpha(z_2)) \cap N_{26}^*(x) \cap X\}$,

• si $A \neq \emptyset$ alors z_2 ne peut vérifier que M_2 ou M_3 et $\forall u \in A$, on a $\overline{\alpha}(u) \in R$ (rmq_4) et $\forall y \in N_{26}^*(x) \cap X$ alors y est 26-adjacent à $\overline{\alpha}(x) \in R$ (rmq_2) ou à $\overline{\alpha}(u) \in R$ (et $\forall u \in A$, $\overline{\alpha}(u)$ est 26-adjacent à $\overline{\alpha}(x)$),

• sinon, $\forall y \in N_{26}^*(x) \cap X$, on a $y \in N_\alpha^*(x) \cup N_\alpha(\overline{\alpha}(x))$ et y est 26-adjacent à $\overline{\alpha}(x) \in R$ (rmq_2).

$\diamond$ si $z_2 \in R$ alors $\forall y \in N_{26}^*(x) \cap X$, y est 26-adjacent à $\overline{\alpha}(x) \in R$ (rmq_2) ou à $z_2 \in R$ (et z_2 est 26-adjacent à $\overline{\alpha}(x)$).

Pour M_3. Soient $A = \{N_\alpha^6(\alpha(z_2)) \cap N_{26}^*(x) \cap X\}$ et $B = \{N_\alpha^6(\alpha(z_3)) \cap N_{26}^*(x) \cap X\}$,

$\diamond$ si $z_3 \in P$ (voir MR_1) $\overset{rmq_1}{\Rightarrow}$ $y_4 \in \overline{X}$.

• si $z_2 \in P$, $\forall y \in N_{26}^*(x) \cap X$, y est 26-adjacent à $\overline{\alpha}(x) \in R$ (rmq_2) ou à $\overline{\alpha}(u) \in R$ (rmq_4) avec $u \in A$ si $A \neq \emptyset$ (z_2 ne peut alors vérifier que M_2 ou M_3), et dans ce cas, $\overline{\alpha}(u)$ est 26-adjacent à $\overline{\alpha}(x)$, $\forall u \in A$; ou à $\overline{\alpha}(v) \in R$ (rmq_4) avec $v \in B$ si $B \neq \emptyset$ (z_3 ne peut alors vérifier que M_2 ou M_3), et dans ce cas, $\overline{\alpha}(v)$ est 26-adjacent à $\overline{\alpha}(x)$, $\forall v \in B$,

• si $z_2 \in R$, $\forall y \in N_{26}^*(x) \cap X$, y est 26-adjacent à $\overline{\alpha}(x) \in R$ (rmq_2) ou à $z_2 \in R$ (et z_2 est 26-adjacent à $\overline{\alpha}(x)$) ou à $\overline{\alpha}(v) \in R$ (rmq_4) avec $v \in B$ si $B \neq \emptyset$ (z_3 ne peut alors vérfier que M_2 ou M_3), et dans ce cas, $\overline{\alpha}(v)$ est 26-adjacent à $\overline{\alpha}(x)$, $\forall v \in B$,

$\diamond$ si $z_3 \in R$

• si $z_2 \in P$, $\forall y \in N_{26}^*(x) \cap X$, y est 26-adjacent à $\overline{\alpha}(x) \in R$ (rmq_2) ou à $z_3 \in R$ (et z_3 est 26-adjacent à $\overline{\alpha}(x)$) ou à $\overline{\alpha}(u) \in R$ (rmq_4) avec $u \in A$ si $A \neq \emptyset$ (z_2 ne peut alors vérifier que M_2 ou M_3), et dans ce cas, $\overline{\alpha}(u)$ est 26-adjacent à $\overline{\alpha}(x)$, $\forall u \in A$,

• si $z_2 \in R$, $\forall y \in N_{26}^*(x) \cap X$, y est 26-adjacent à $\overline{\alpha}(x) \in R$ (rmq_2) ou à $z_3 \in R$ ou à $z_2 \in R$ (et ces trois points sont mutuellement 26-adjacents).

Dans tous ces cas de figure $T_{26}(x, R) = 1$ et $\forall y \in N_{26}^*(x) \cap P$, $\exists z \in R$ tel que z est 26-adjacent à x et y.

Condition (C_2) **:**
$x \in P \overset{rmq_1}{\Rightarrow} \alpha(x) = z \in \overline{X}$. Dans M_1 à M_6, $\forall y \in N_\alpha^6(x) \cap \overline{X}$, alors $\exists t\,(= \alpha(y)) \in N_\alpha^6(\alpha(x)) \cap \overline{X}$ (rmq_3), t est 6-adjacent à y et à z $(*_1)$.
Considérons $u = \overline{\alpha}(x)$. Dans M_1 à M_4, $u \in X$ et $(*_1)$ implique $T_6(x, \overline{X}) = 1$. Dans M_5 à M_6, $u \in \overline{X}$, et $\exists v \in N_\alpha^6(u) \cap \overline{X}$ et $\exists w \in N_\alpha^6(x) \cap \overline{X}$ tel que $w = \alpha(v)$ (voir MR_2) et d'après $(*_1)$ (on pose $y = w$), on a $T_6(x, \overline{X}) = 1$.

Condition (C_4) **:**
$\forall y \in N_6^*(x) \cap P$ alors $y \in N_\alpha^6(x)$, car $\alpha(x) \in \overline{X}$ (rmq_1) et soit $\overline{\alpha}(x) \in \overline{X}$ (templates M_5 et M_6 – ou MR_2), soit $\overline{\alpha}(x) \in R$ (templates M_1 à M_4 – ou MR_1 – et rmq_2).
$\forall y \in N_\alpha^6(x) \cap P$ alors $\exists z = \alpha(y) \in \overline{X}$ (rmq_1) et $\exists t = \alpha(x) \in \overline{X}$ (rmq_1), et nous avons le carré unité $\{x, y, z, t\}$.

□

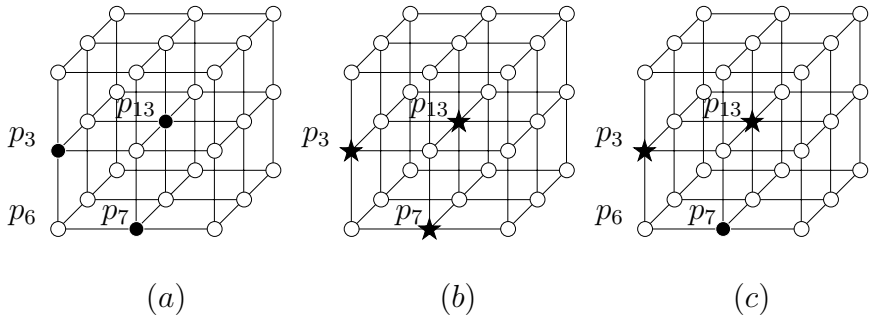


FIG. 6.6 – *Cette configuration* (a) *est non* P_1^x*-simple* (b)*, et est* P_2^x*-simple* (c).

6.5.3 Nouvel algorithme proposé [LB02b]

Dans cette section est relaté le raisonnement qui nous a permis de proposer deux conditions successives d'appartenance à un ensemble P de façon à ce que notre algorithme final (noté à partir de maintenant LOBE) supprimant des points P^x-simples, supprime au moins les points retirés par l'algorithme de Palágyi et Kuba en 6 sous-itérations (PAKU6) (algorithme déjà vu dans la section 6.5.2.1, voir détails supplémentaires dans la section B.2.1). En fait, notre algorithme final supprime également les points retirés par l'algorithme de Gong et Bertrand (GOBE) [GB90] proposé dans sa version curviligne par F. Rolland, et al. [RCM92] (algorithme déjà vu dans la section 6.5.1.1, voir détails supplémentaires dans la section B.2.2).

Dans un premier temps, la recherche s'effectue pour la sous-itération selon la direction $Haut$ et elle est guidée uniquement par rapport à l'algorithme PAKU6.

6.5.3.1 Premier essai (recherche de P_1)

Nous observons que $\mathcal{T}_H$ supprime certains points de X dont le voisin $Haut$ appartient à $\overline{X}$ (voir les templates à la figure 6.4). C'est le cas également pour l'algorithme GOBE dans la direction $\alpha = Haut$.

Nous proposons alors de considérer $P_1 = \{x \in X;$ le voisin $Haut$ de x appartient à $\overline{X}\}$ ou plus généralement $P_1 = \{x \in X; \alpha(x) \in \overline{X}\}$; pour la discussion qui suit, nous ne considérons que le cas $\alpha = Haut$. Parmi les 2^{26} configurations possibles, nous obtenons $4\,423\,259$ configurations correspondant à des points P_1^x-simples et non extrémités selon la direction $Haut$. Rappelons que nous adaptons la définition suivante ici (utilisée par K. Palágyi et A. Kuba, et par F. Rolland, et al.) : un point x est *extrémité (de courbe)* s'il n'existe qu'un seul autre point de X dans $N_{26}^*(x)$.

Considérons la configuration de la figure 6.6 (a). Les trois points p_3, p_7 et p_{13} appartiennent à P_1^x (voir Fig. 6.6 (b)) car ils appartiennent à X, et le voisin $Haut$ de chacun de ces points appartient à $\overline{X}$. Les première et troisième conditions de P_1^x-simplicité ne sont pas vérifiées par le point central p_{13}. Ainsi, le point p_{13} n'est pas P_1^x-simple. Néanmoins, ce point peut être éliminé par une rotation selon l'axe vertical de la template M_5 de $\mathcal{T}_H$. Par conséquent, cette configuration devrait être éliminée par l'algorithme que nous cherchons.

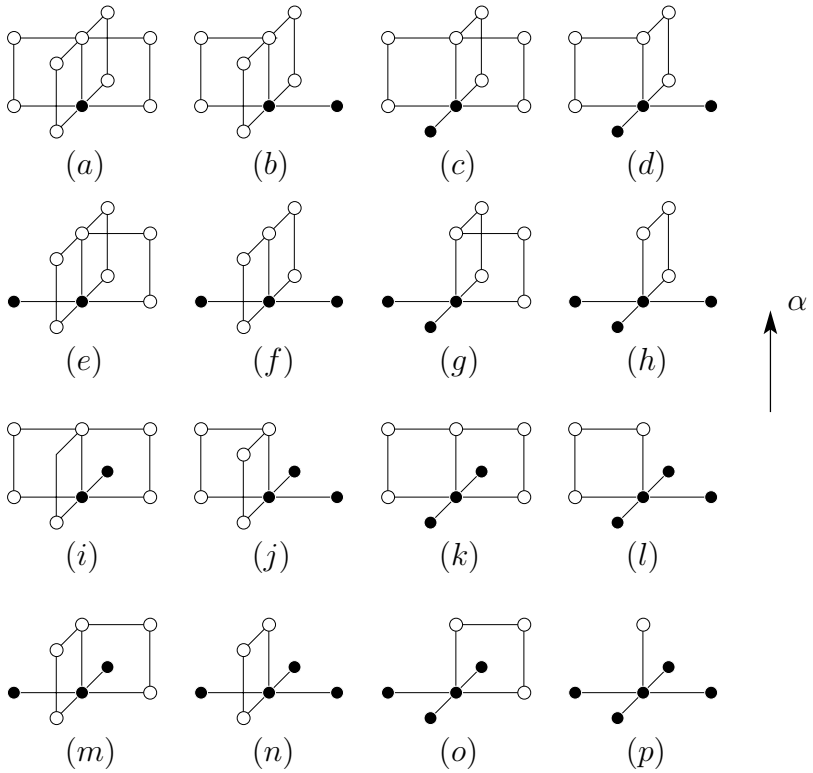


FIG. 6.7 – *Un point appartient à P_2 ssi il vérifie l'une de ces templates.*

Étudions le comportement des autres points de cette configuration avec les templates de $\mathcal{T}_H$ (voir Fig. 6.6 (a)). Le point p_3 peut vérifier une rotation selon l'axe vertical de la template M_5 ou de la template M_6. Le point p_7 ne peut pas être supprimé, car $p_6(= O(p_7))$ appartient à $\overline{X}$ et $p_3(= H(p_6))$ appartient à X, et les templates sont telles que si x est supprimé alors pour tout y appartenant à $N_6^*(x) \cap \overline{X}$, y n'étant ni $H(x)$ ni $B(x)$ (*c.-à-d.* pour tout $y \in N_\alpha^6(x)$ avec $\alpha = Haut$), alors le point $H(y)$ doit appartenir à $\overline{X}$.

Avec ces remarques, nous proposons un nouvel ensemble P_2.

6.5.3.2 Deuxième essai (recherche de P_2)

Nous proposons de considérer $P_2 = \{x \in X; \alpha(x) \in \overline{X}$ et pour tout point y appartenant à $N_\alpha^6(x)$, si y appartient à $\overline{X}$ alors $\alpha(y)$ doit appartenir à $\overline{X}\}$; la discussion qui suit se déroule pour le cas $\alpha = Haut$.

Avec les notations introduites dans la section 6.2, l'ensemble P_2 peut être décrit par la famille composée de 16 couples de sous-ensembles de $\mathcal{Z}^3$ $(B^k(x), W^k(x))$ décrites à la figure 6.7, pour $k = 1, \ldots, 16$, pour la direction $\alpha = Haut$; en fait, il y a 6 templates principales, les autres sont obtenues à rotations de 90, 180 ou 270 degrés selon l'axe $(\alpha(x), \overline{\alpha}(x))$.

Considérons la configuration non P_1^x-simple de la figure 6.6 (b) (voir notations, Fig. 6.6 (c)). Le point p_{13} appartient à P_2^x, car il vérifie la template Fig. 6.7 (a). Le point p_3 appartient à P_2^x,

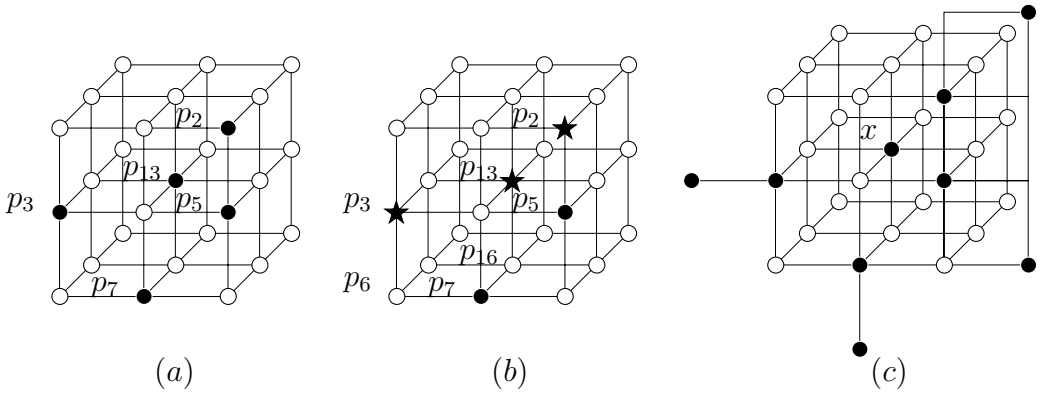


FIG. 6.8 – (a) *Cette configuration ne peut pas être supprimée par* PAKU6 *quelle que soit la direction de suppression et est P_2^x-simple* (b)*, dans* (c) *(obtenue à partir de* (a)*), aucun point n'est supprimé par* PAKU6*, néanmoins x est supprimé par* LOBE.

car p_3 peut vérifier les templates des figures 6.7 (a), (c), (e), ou (g). Le point p_7 n'appartient pas à P_2^x car il existe un point y dans $N_H^6(p_7) \cap N_{26}(x)$ qui appartient à $\overline{X}$, par exemple $p_6(= O(p_7))$, et tel que $p_3(= H(y))$ appartient à X ; ou plus directement car p_7 ne vérifie aucune des templates de la figure 6.7. Ainsi, cette configuration non P_1^x-simple (Fig. 6.6 (b)) est maintenant P_2^x-simple (Fig. 6.6 (c)).

Nous obtenons $6\,129\,527$ configurations correspondant à des points P_2^x-simples et non extrémités, pour la direction $Haut$. Les $2\,124\,283$ configurations supprimées par $\mathcal{T}_H$ sont également P_2^x-simples. Le fait que les configurations supprimées par PAKU6 soient P_2^x-simples (pour chaque direction et par conséquent pour l'algorithme en entier) garantit que la topologie est préservée par PAKU6 (car PAKU6 supprime des sous-ensembles de points P_2^x-simples, voir chapitre 5).

Pour une meilleure comparaison entre PAKU6 et LOBE, nous générons les configurations supprimées par ces algorithmes pour chaque direction : PAKU6 supprime $9\,916\,926$ configurations, *i.e.* il existe au moins une direction telle qu'une configuration parmi celles-là est supprimée par PAKU6 selon cette direction ; LOBE supprime $23\,721\,982$ configurations (139.2% de plus). Nous rappelons qu'il y a $25\,985\,118$ points simples et non extrémités parmi les $67\,108\,864 (= 2^{26})$ configurations possibles dans un voisinage $3 \times 3 \times 3$.

La configuration décrite à la figure 6.8 (a) ne peut pas être supprimée par PAKU6, quelle que soit la direction de suppression. Cette configuration est P_2^x-simple (Fig. 6.8 (b)), avec $\alpha = Haut$. En effet, le point p_2 appartient à P_2^x car p_2 peut vérifier les templates des figures 6.7 (a), (b), (c) ou (d) ; p_3 appartient à P_2^x car p_3 peut vérifier les templates des figures 6.7 (a), (c), (e) ou (g) ; p_{13} appartient à P_2^x car il vérifie la template de la figure 6.7 (a) ; p_5 n'appartient pas à P_2^x car $p_2(= H(p_5))$ appartient à X (ou plus directement, car p_5 ne vérifie aucune template de la figure 6.7) ; et p_7 n'appartient pas à P_2^x car il existe un point y dans $N_H^6(p_7) \cap N_{26}(x)$ ($y = p_6(= O(p_7))$) qui appartient à $\overline{X}$ et tel que $H(y)$ ($= p_3$) appartient à X (ou plus directement, car p_7 ne vérifie aucune template de la figure 6.7).

La figure 6.8 (c) montre une image construite à partir de la configuration de la figure 6.8 (a)

telle que tout point est soit non simple (excepté x) soit extrémité, et aucun point n'est supprimé par PAKU6, néanmoins le point x est supprimé par LOBE.

6.5.3.3 Propriétés de LOBE par rapport à PAKU6

Résumons cette étude par les deux propriétés suivantes :

Propriété : Les points supprimés par PAKU6 sont des points P_2^x-simples ; par conséquent PAKU6 préserve la topologie.

Propriété : L'algorithme LOBE opérant par suppression de points P_2^x-simples non terminaux de courbe supprime au moins les points retirés par PAKU6, tout en préservant les mêmes points terminaux.

6.5.3.4 Par rapport à l'algorithme GOBE

Les $4\,772\,095$ configurations supprimées par GOBE, selon la direction $Haut$, sont P_2^x-simples. Cela implique que GOBE préserve la topologie (car cet algorithme supprime des sous-ensembles de points P_2^x-simples, voir chapitre 5).

De même, pour une meilleure comparaison entre GOBE et LOBE, nous générons les configurations supprimées pour chaque direction. L'algorithme GOBE supprime $21\,194\,234$ configurations, notre algorithme en supprime $23\,721\,982$ (11.9% de plus).

En fait, la configuration de la figure 6.8 (a) ne peut être supprimée par GOBE, quelle que soit la direction de suppression considérée, car p_{13} est tel que $N_6^*(p_{13}) \cap X = \emptyset$ et la condition (G_2) ne peut être vérifiée.

6.5.3.5 Propriétés de LOBE par rapport à GOBE

Résumons cette étude par les deux propriétés suivantes :

Propriété : Les points supprimés par GOBE sont des points P_2^x-simples ; par conséquent GOBE préserve la topologie.

Propriété : L'algorithme LOBE opérant par suppression de points P_2^x-simples non terminaux de courbe supprime au moins les points retirés par GOBE, tout en préservant les mêmes points terminaux.

6.5.4 Récapitulatif pour l'algorithme en 6 sous-itérations

Les figures 6.9 et 6.10 proposent des configurations P_1^x, P_2^x-simples ou non, éliminées par GOBE, par PAKU6 ou non, pour une sous-itération (Fig. 6.9) ou pour une itération (Fig. 6.10).

Sur ces figures, si une configuration est reliée à un “nœud”-algorithme par un trait, alors elle est éliminée par cet algorithme, sauf dans le cas où le trait est barré. À proximité de chaque “nœud”-algorithme, est indiqué le nombre de configurations que l’algorithme supprime, pour une sous-itération (Fig. 6.9) ou pour une itération (Fig. 6.10).

Nous avons choisi de ne pas proposer une version surfacique ; cette possibilité sera en revanche étudiée dans la section 6.6, pour un algorithme en 12 sous-itérations.

6.5.4.1 Résultats observés

Les squelettes de quelques images, obtenues respectivement par PAKU6 et LOBE, sont montrés à la figure 6.11 (la représentation est décrite dans la section B.1). Nous observons que le nombre de sous-itérations de suppression exigé par LOBE est inférieur ou égal à ceux requis par PAKU6. Le nombre de points supprimés par LOBE est inférieur ou égal à celui des points retirés par PAKU6. Rappelons qu’il est possible que LOBE exige plus d’itérations pour obtenir un squelette que ne l’exige PAKU6 ou GOBE (cf. Fig. 6.8 (c)).

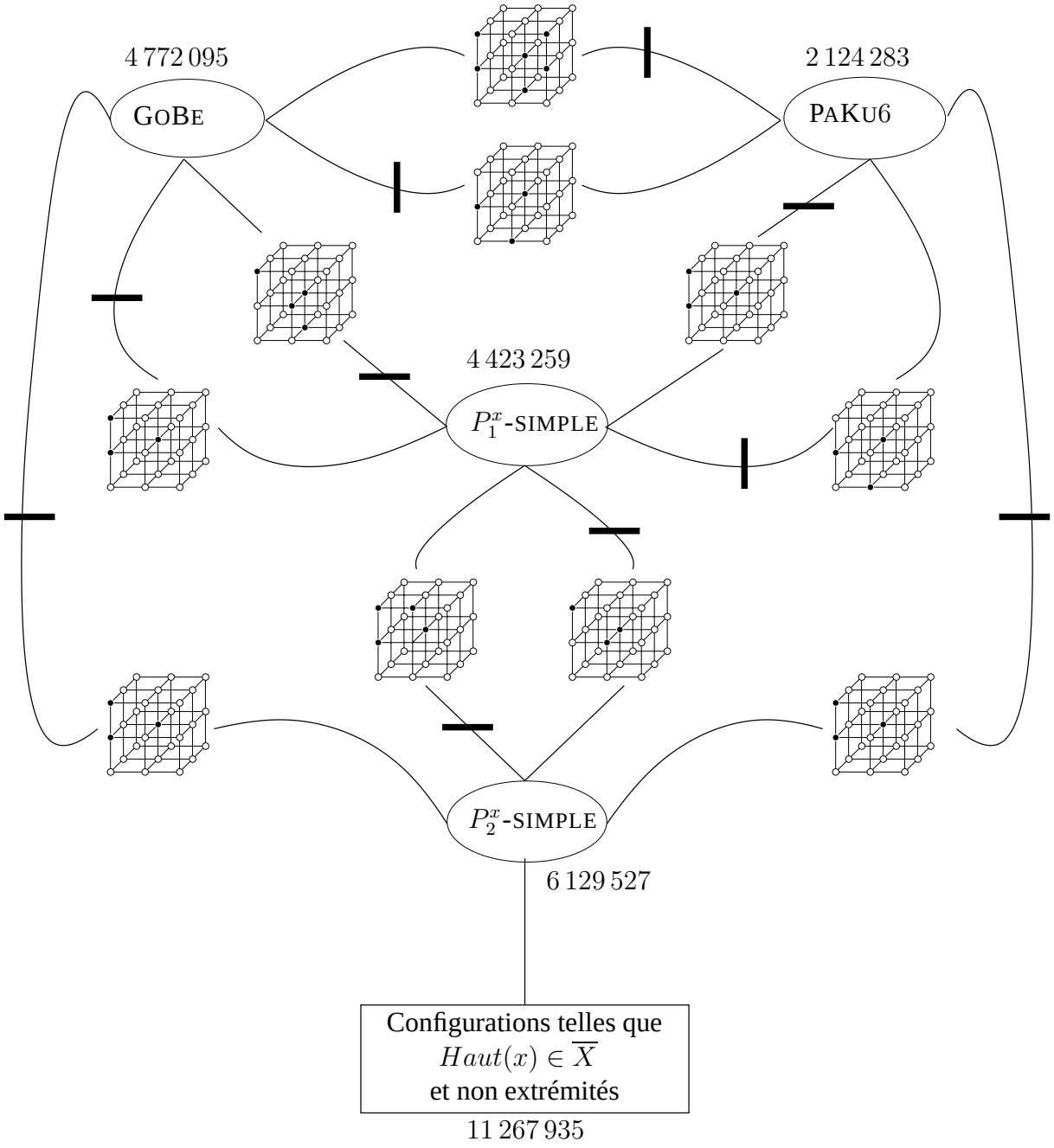


FIG. 6.9 – *Récapitulatif : sous-itération (comparaisons algorithmes* GOBE*,* PAKU6 *et* LOBE*).*

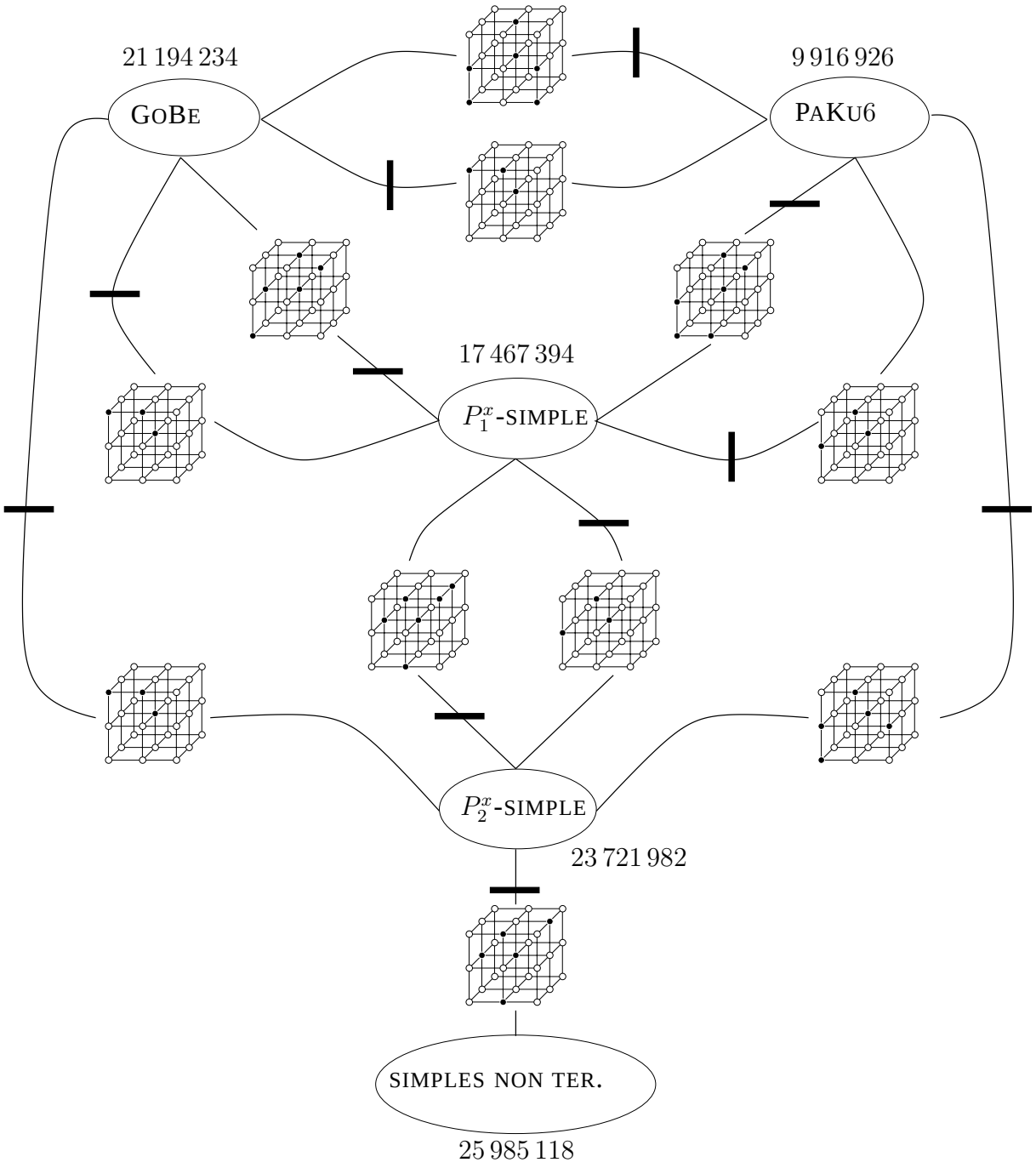
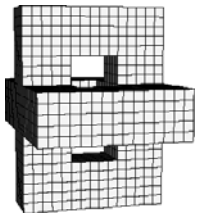


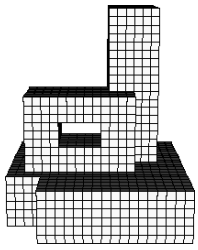
FIG. 6.10 – *Récapitulatif : itération (comparaisons algorithmes* GOBE*,* PAKU6 *et* LOBE*).*



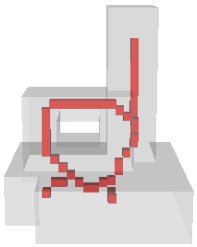
Objet initial

$16 - 2\,256$

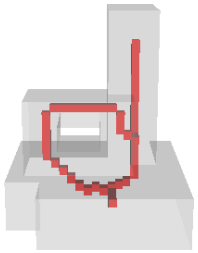
$16 - 2\,256$



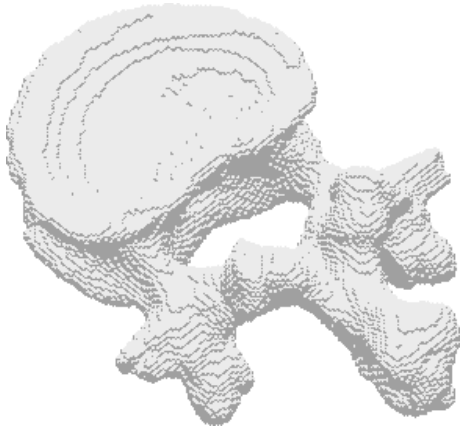
Objet initial



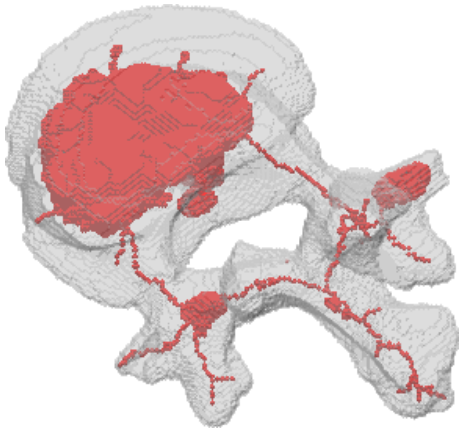
$54 - 4\,251$



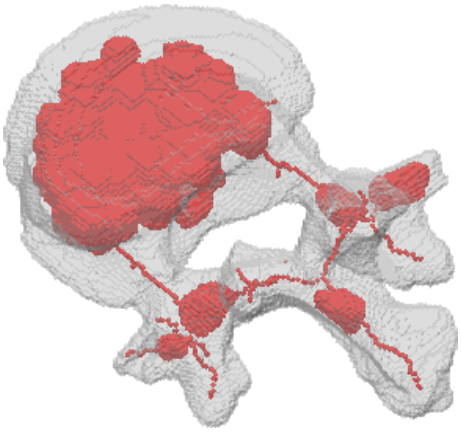
$45 - 4\,248$



Objet initial



$142 - 170\,001$



$73 - 163\,874$

FIG. 6.11 – *Respectivement : un objet initial, les squelettes curvilignes obtenus avec* PAKU6 *et* LOBE. *Sous chaque squelette, sont donnés le numéro de la dernière sous-itération effective de suppression et le nombre de points supprimés (*REPR7*).*

6.6 Algorithmes en 12 sous-itérations directionnelles

K. Palágyi et A. Kuba ont proposé un algorithme en 12 sous-itérations, à base de templates, permettant l'obtention de squelettes curvilignes ou surfaciques. Nous montrons d'abord que cet algorithme est P_{26}-simple. Nous proposons ensuite un algorithme basé sur la suppression de points P^x-simples, et supprimant au moins les points supprimés par l'algorithme de Palágyi et Kuba. Nous le déclinons en deux versions : curviligne ou surfacique. Par ailleurs, nous avons obtenu un ensemble restreint de templates qu'un point doit vérifier pour être supprimé par notre algorithme surfacique, ces templates ont été obtenues par l'utilisation des BDD. Voir également [LB01].

6.6.1 Rappels de l'algorithme de l'algorithme de Palágyi et Kuba en 12 sous-itérations (PAKU12) [PK99]

K. Palágyi et A. Kuba ont proposé un algorithme de squelettisation (que nous nommerons PAKU12, à partir de maintenant) consistant en la répétition jusqu'à stabilité de 12 sous-itérations directionnelles $Haut-Sud$, $Nord-Est$, $Bas-Ouest$, $Sud-Est$, $Haut-Ouest$, $Bas-Nord$, $Sud-Ouest$, $Haut-Nord$, $Bas-Est$, $Nord-Ouest$, $Haut-Est$, $Bas-Sud$ de suppression parallèle de points ; ces 12 sous-itérations correspondent à toutes les paires de directions principales non opposées. Pour une direction donnée, l'algorithme élimine en parallèle les points dont le voisinage vérifie au moins l'une des templates proposées, pour cette direction.

Ces templates ont été obtenues de façon à supprimer un sous-ensemble des points simples ayant un voisin dans le complémentaire, voisin dans une direction donnée, tout en préservant la topologie, mais également afin de ne pas supprimer les points extrémités, et de façon à respecter certaines conditions géométriques (les templates comportent au moins deux plans de symétrie, ce qui permet alors un meilleur centrage). Deux ensembles de templates ont été proposés afin d'obtenir soit un squelette curviligne, soit un squelette surfacique.

Pour plus d'informations, se reporter à la description de cet algorithme dans la section B.2.4, page 301.

6.6.2 Algorithme de Palágyi - version squelette curviligne

À la figure 6.12, est représenté l'ensemble $\mathcal{T}_{HS}$ des templates pour la direction $Haut-Sud$, utilisé pour l'obtention du squelette curviligne.

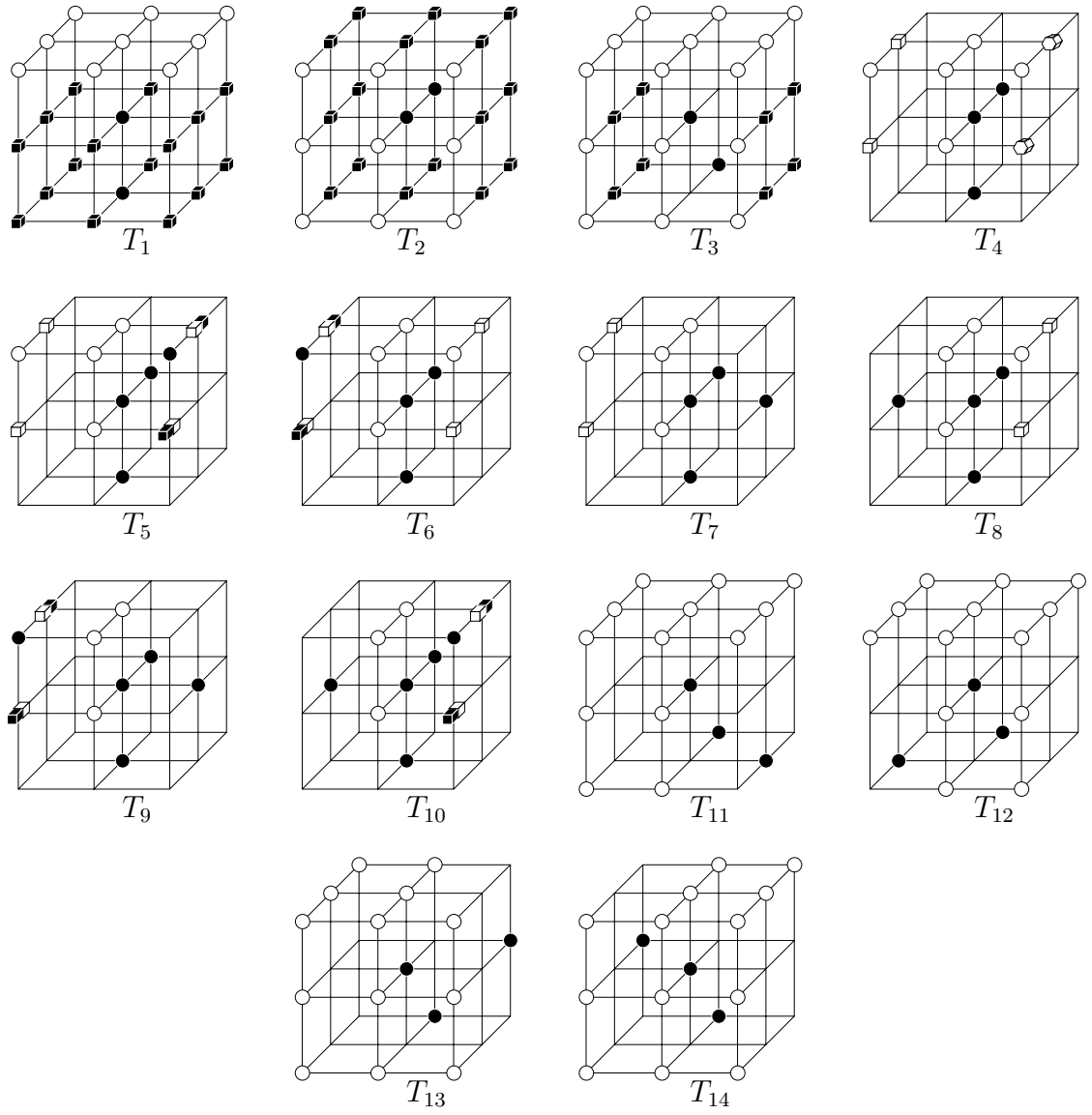


FIG. 6.12 – *Templates utilisées lors de la sous-itération de suppression selon la direction $Haut-Sud$, pour l'obtention du squelette curviligne. Au minimum un point représenté par un cube noir doit appartenir à l'objet dans les templates T_1 à T_3. Au minimum un point représenté par un cube blanc ou un hexagone en perspective doit appartenir au background dans les templates T_4 à T_8. Deux points marqués par deux parallélépipèdes bicolores doivent être différents l'un de l'autre dans les templates T_5, T_6, T_9 et T_{10}. Les points non représentés appartiennent soit à l'objet, soit au background.*

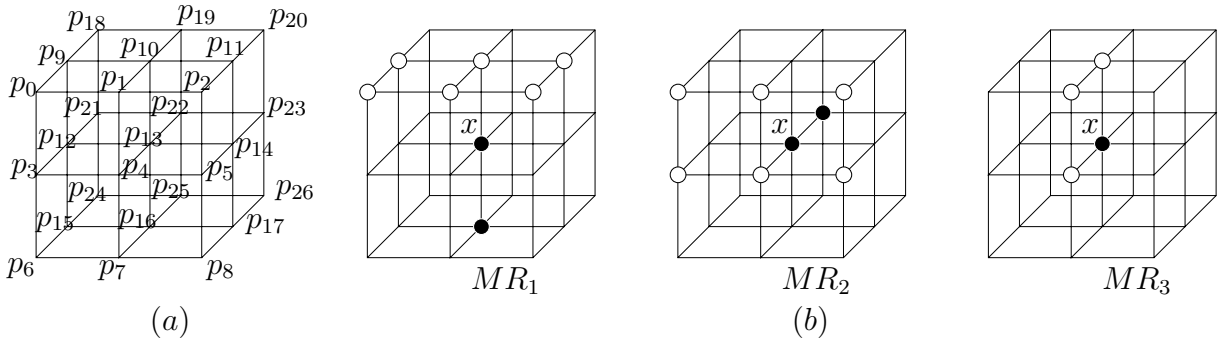


FIG. 6.13 – (a) *Notations utilisées et* (b) *motifs récurrents.*

6.6.2.1 P_{26}-simplicité de PAKU12

Propriété : L'algorithme de Palágyi et Kuba en 12 sous-itérations (PAKU12) est P_{26}-simple.

Preuve :

Les templates peuvent être classées en trois catégories : la template T_1, la template T_2 et les templates T_3 à T_{14}, pour lesquelles apparaissent respectivement les motifs récurrents illustrés à la figure 6.13 (b) (*i.e.* apparaissant dans chaque template de chacune de ces classes). Les notations utilisées pour démontrer que cet algorithme est P_{26}-simple sont reportées à la figure 6.13 (a).

Soit $PK_{12}^{\alpha}(X)$, l'ensemble des points x de X pouvant vérifier au moins l'une des templates (pour la direction α). Pour prouver que l'algorithme PAKU12 préserve la topologie, il est suffisant de montrer que :

$$\forall X \subset \mathcal{Z}^3,\ P = PK_{12}^{\alpha}(X) \text{ est } P_{26}\text{-simple.}$$

Soient $P = PK_{12}^{\alpha}(X),\ R = X \setminus P$ et $x \in P$.

Nous avons choisi de montrer qu'un point vérifiant une template est P-simple (*i.e.* qu'il vérifie les 4 conditions de P-simplicité) et cela pour chacune des templates.

Pour des raisons de symétrie (imposées lors de la conception des templates, symétries selon la direction $HautSud$), il est suffisant d'examiner les templates T_1, T_2, T_3, T_4, T_5, T_7, T_{10}, T_{11} et T_{13}.

Il est aisé de constater que (C_2) est vérifiée par ces templates. La preuve utilise les remarques suivantes (voir notations 6.13 (a)).

Remarque 1 (rmq_1) **:**
Si $x \in P \Rightarrow HautSud(x) \in \overline{X}$ et $(Haut(x) \in \overline{X}$ ou $Sud(x) \in \overline{X})$ (voir les templates ou les motifs récurrents).
Conséquence : si $x \in X$ et $HautSud(x) \in X \Rightarrow x \in R$.

Remarque 2 (rmq_2) **:**
Si $x \in P$ et $Sud(x) \in X$ alors x ne peut vérifier que T_1 (motif récurrent MR_1).

Remarque 3 (rmq_3) **:**
Si $x \in P$ et $Haut(x) \in X$ alors x ne peut vérifier que T_2 (motif récurrent MR_2).

Remarque 4 (rmq_4, S) **:**
Supposons qu'il existe un ensemble 26-connexe $S \subseteq N_{26}^*(x) \cap R$.
Si $\forall y \in N_{26}^*(x) \cap X$, y est 26-adjacent à au moins un point de S, alors
$\begin{cases} T_{26}(x, R) = 1, \text{ c'est la condition } (C_1) \\ \forall y \in N_{26}^*(x) \cap P, \exists z \in R \text{ tel que } z \text{ est 26-adjacent à } x \text{ et à } y \text{ (car } S \subseteq N_{26}^*(x) \cap R) \\ \quad \text{c'est la condition } (C_3). \end{cases}$
Dans une telle situation, nous indiquerons alors (rmq_4, S) ; l'ensemble S étant décrit par l'ensemble de points $\{x_0, \ldots, x_i, \ldots, x_n\}$ avec $n \geq 0$ et tel que l'ensemble S est 26-connexe et est inclus dans $N_{26}^*(x) \cap R$ (l'ensemble S est un paramètre de cette remarque).

Considérons maintenant un motif particulier (octant supérieur avant droit, faisant intervenir deux points opposés), motif apparaissant dans les templates T_5 et T_{10} parmi celles à examiner.
Si $p_{11} \in \overline{X} \Rightarrow p_5 \in X$. Si $p_5 \in P$, alors il ne peut vérifier que T_2 (rmq_3) $\overset{MR_2}{\Rightarrow}$ $p_{14}(= S(p_5)) \in X$, or $p_2(= HS(p_{14})) \in X$ $\overset{rmq_1}{\Rightarrow}$ $p_{14} \in R$.
De même (par symétrie), si $p_{11} \in X \Rightarrow p_5 \in \overline{X}$. Si $p_{11} \in P$, alors il ne peut vérifier que T_1 (rmq_2) $\overset{MR_1}{\Rightarrow}$ $p_{14}(= B(p_{11})) \in X$, or $p_2(= HS(p_{14})) \in X$ $\overset{rmq_1}{\Rightarrow}$ $p_{14} \in R$.

Sous ces conditions (octant supérieur avant droit des templates T_5 et T_{10}), nous avons les trois remarques suivantes :

Remarque 5 (rmq_5) **:**
Si $p_5 \in X$ alors $p_5 \in R$ ou $p_{14} \in R$. Notons que l'un ou l'autre est 26-adjacent à p_{16}.

Remarque 6 (rmq_6) **:**
Si $p_{11} \in X$ alors $p_{11} \in R$ ou $p_{14} \in R$. Notons que l'un ou l'autre est 26-adjacent à p_{22}.

Remarque 7 (rmq_7) **:** (déduite des 2 précédentes)
Il existe un point $z \in \{p_5, p_{14}, p_{11}\}$ tel que z appartient à R et z est 26-adjacent à l'ensemble 26-connexe $\{p_{16}, p_{22}\}$.

Template T_1 :

• Si $p_{16} \in P$ alors il ne peut vérifier que T_2 (rmq_3)

$\overset{MR_2}{\Rightarrow}$ $\begin{cases} \{p_3, p_4, p_5, p_6, p_7, p_8\} \subset \overline{X}. \ (C_4) \text{ est vérifiée,} \\ p_{25} \in X, \text{ or } p_{13} \in X \overset{rmq_1}{\Rightarrow} p_{25} \in R. \ (C_1) \text{ et } (C_3) \text{ sont vérifiées } (rmq_4, \{p_{25}\}). \end{cases}$

• Si $p_{16} \in R$ alors (C_1), (C_3) $(rmq_4, \{p_{16}\})$ et (C_4) sont vérifiées.

Template T_2 :

• Si $p_{22} \in P$ alors il ne peut vérifier que T_1 (rmq_2)

$\overset{MR_1}{\Rightarrow}$ $\begin{cases} \{p_9, p_{10}, p_{11}, p_{18}, p_{19}, p_{20}\} \subset \overline{X}. \ (C_4) \text{ est vérifiée,} \\ p_{25} \in X, \text{ or } p_{13} \in X \overset{rmq_1}{\Rightarrow} p_{25} \in R. \ (C_1) \text{ et } (C_3) \text{ sont vérifiées } (rmq_4, \{p_{25}\}). \end{cases}$

• Si $p_{22} \in R$ alors (C_1), (C_3) $(rmq_4, \{p_{22}\})$ et (C_4) sont vérifiées.

Template T_3 :

(C_4) est vérifiée. $p_{25} \in R$, car $p_{13} \in X$ (rmq_1). (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{25}\})$.

Template T_4 :

• Si $p_{16} \in P$ alors il ne peut vérifier que T_2 (rmq_3)

$\overset{MR_2}{\Rightarrow}$ $\begin{cases} \{p_3, p_5, p_6, p_7, p_8\} \subset \overline{X} \\ p_{25} \in X \text{ or } p_{13} \in X \overset{rmq_1}{\Rightarrow} p_{25} \in R \end{cases}$

* Si $p_{22} \in P$ alors il ne peut vérifier que T_1 (rmq_2)

$\overset{MR_1}{\Rightarrow}$ $\{p_9, p_{11}, p_{18}, p_{19}, p_{20}\} \subset \overline{X}$. (C_1), (C_3) $(rmq_4, \{p_{25}\})$ et (C_4) sont vérifiées.

* Si $p_{22} \in R$. (C_1), (C_3) $(rmq_4, \{p_{22}\})$ et (C_4) sont vérifiées.

• Si $p_{16} \in R$,

* Si $p_{22} \in P$ alors il ne peut vérifier que T_1 (rmq_2)

$\overset{MR_1}{\Rightarrow}$ $\{p_9, p_{11}, p_{18}, p_{19}, p_{20}\} \subset \overline{X}$. (C_1), (C_3) $(rmq_4, \{p_{16}\})$ et (C_4) sont vérifiées.

* Si $p_{22} \in R$. (C_1), (C_3) $(rmq_4, \{p_{16}, p_{22}\})$ et (C_4) sont vérifiées.

Template T_5 :

• Si $p_{22} \in P$ alors il ne peut vérifier que T_1 (rmq_2)

$\overset{MR_1}{\Rightarrow}$ $\{p_9, p_{11}, p_{18}, p_{19}, p_{20}\} \subset \overline{X}$ et $p_{11} \in \overline{X} \Rightarrow p_5 \in X$

Et $p_{16} \in R$ ($\notin P$ à cause de p_{13} et de p_5). (C_4) est vérifiée.

$p_5 \in X \overset{rmq_5}{\Rightarrow} p_5 \in R$ ou $p_{14} \in R$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_5$ ou $p_{14}, p_{16}\})$.

• Si $p_{22} \in R$

* Si $p_{16} \in P$ alors il ne peut vérifier que T_2 (rmq_3)

$\overset{MR_2}{\Rightarrow}$ $\{p_3, p_5, p_6, p_7, p_8\} \subset \overline{X}$ et $p_5 \in \overline{X} \Rightarrow p_{11} \in X$. (C_4) est vérifiée.

$p_{11} \in X \overset{rmq_6}{\Rightarrow} p_{11} \in R$ ou $p_{14} \in R$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{11}$ ou $p_{14}, p_{22}\})$.

* Si $p_{16} \in R$. (C_4) est vérifiée.

Il existe un point $z \in R$, avec $z \in \{p_5, p_{14}, p_{11}\}$ 26-adjacent à $\{p_{16}, p_{22}\}$ (rmq_7).

(C_1) et (C_3) sont vérifiées $(rmq_4, \{z, p_{16}, p_{22}\})$.

Template T_7 :

• Si $p_{22} \in P$ alors il ne peut vérifier que T_1 (rmq_2)

$\overset{MR_1}{\Rightarrow}$ $\begin{cases} \{p_9, p_{11}, p_{18}, p_{19}, p_{20}\} \subset \overline{X} \\ p_{25} \in X \text{ or } p_{13} \in X \overset{rmq_1}{\Rightarrow} p_{25} \in R \end{cases}$

* Si $p_{16} \in P$ alors il ne peut vérifier que T_2 (rmq_3)

$\overset{MR_2}{\Rightarrow} \{p_3, p_5, p_6, p_7, p_8\} \subset \overline{X}$. (C_4) est vérifiée.

♢ Si $p_{14} \in P \overset{rmq_1}{\Rightarrow} p_2 \in \overline{X}$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{25}\})$.

♢ Si $p_{14} \in R$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{25}, p_{14}\})$.

* Si $p_{16} \in R$,

(C_4) est vérifiée.

♢ Si $p_{14} \in P \overset{rmq_1}{\Rightarrow} p_2 \in \overline{X}$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{16}\})$.

♢ Si $p_{14} \in R$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{16}, p_{14}\})$.

• Si $p_{22} \in R$,

* Si $p_{16} \in P$ alors il ne peut vérifier que T_2 (rmq_3)

$\overset{MR_2}{\Rightarrow} \{p_3, p_5, p_6, p_7, p_8\} \subset \overline{X}$. (C_4) est vérifiée.

♢ Si $p_{14} \in P \overset{rmq_1}{\Rightarrow} p_2 \in \overline{X}$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{22}\})$.

♢ Si $p_{14} \in R$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{22}, p_{14}\})$.

* Si $p_{16} \in R$,

♢ Si $p_{14} \in P$

$\overset{rmq_1}{\Rightarrow}$ $\begin{cases} p_2 \in \overline{X}. (C_1) \text{ et } (C_3) \text{ sont vérifiées.} (rmq_4, \{p_{16}, p_{22}\}) \\ p_5 \in \overline{X} \text{ ou } p_{11} \in \overline{X} \text{ (d'après les templates).} (C_4) \text{ est vérifiée.} \end{cases}$

♢ Si $p_{14} \in R$. (C_1), (C_3) $(rmq_4, \{p_{14}, p_{16}, p_{22}\})$ et (C_4) sont vérifiées.

Template T_{10} :

• Si $p_{16} \in P$ alors il ne peut vérifier que T_2 (rmq_3)

$\overset{MR_2}{\Rightarrow} \begin{cases} \{p_3, p_5, p_6, p_7, p_8\} \subset \overline{X} \text{ et } p_5 \in \overline{X} \Rightarrow p_{11} \in X \\ p_{25} \in X \text{ or } p_{13} \in X \overset{rmq_1}{\Rightarrow} p_{25} \in R \end{cases}$

Et $p_{22} \in R$ ($\notin P$ à cause de p_{13}, et de p_{11}). (C_4) est vérifiée.

$p_{11} \in X \overset{rmq_6}{\Rightarrow} p_{11} \in R$ ou $p_{14} \in R$ (26-adjacent à p_{22}).

* Si $p_{12} \in P \overset{rmq_1}{\Rightarrow} p_0 \in \overline{X}$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{11}$ ou $p_{14}, p_{22}, \})$.

* Si $p_{12} \in R$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{11}$ ou $p_{14}, p_{22}, p_{12}\})$.

• Si $p_{16} \in R$

* Si $p_{22} \in P$ alors il ne peut vérifier que T_1 (rmq_2)

$\overset{MR_1}{\Rightarrow} \begin{cases} \{p_9, p_{11}, p_{18}, p_{19}, p_{20}\} \subset \overline{X} \text{ et } p_{11} \in \overline{X} \Rightarrow p_5 \in X. \ (C_4) \text{ est vérifiée.} \\ p_{25} \in X \text{ or } p_{13} \in X \overset{rmq_1}{\Rightarrow} p_{25} \in R \end{cases}$

$p_5 \in X \overset{rmq_5}{\Rightarrow} p_5 \in R$ ou $p_{14} \in R$ (26-adjacent à p_{16}).

$\diamondsuit$ Si $p_{12} \in P \overset{rmq_1}{\Rightarrow} p_0 \in \overline{X}$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_5$ ou $p_{14}, p_{16}\})$.

$\diamondsuit$ Si $p_{12} \in R$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_5$ ou $p_{14}, p_{16}, p_{12}\})$.

* Si $p_{22} \in R$,

$\exists z \in R\,(z \in \{p_{11}, p_5, p_{14}\})$ 26-adjacent à $\{p_{16}, p_{22}\}$ (rmq_7).

$\diamondsuit$ Si $p_{12} \in P \overset{rmq_1}{\Rightarrow} p_0 \in \overline{X}$ et ($p_3 \in \overline{X}$ ou $p_9 \in \overline{X}$) (d'après les templates).

(C_1), (C_3) $(rmq_4, \{z, p_{16}, p_{22}\})$ et (C_4) sont vérifiées.

$\diamondsuit$ Si $p_{12} \in R$. (C_1), (C_3) $(rmq_4, \{z, p_{16}, p_{22}, p_{12}\})$ et (C_4) sont vérifiées.

Template T_{11} :

(C_4) est vérifiée.

$p_{25} \in R$, car $p_{13} \in X$ (rmq_1).

• Si $p_{17} \in P$ alors il ne peut vérifier que $T_2, T_4, T_5, T_7, T_8, T_{10}$ ou T_{14} car $p_{13} \in X$ et $p_4 \in \overline{X}$

$\overset{MR_2 \cap MR_3}{\Rightarrow} \{p_5, p_8\} \subset \overline{X}$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{25}\})$.

• Si $p_{17} \in R$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{25}, p_{17}\})$.

Template T_{13} :

(C_4) est vérifiée.

$p_{25} \in R$, car $p_{13} \in X$ (rmq_1).

• Si $p_{23} \in P$ alors il ne peut vérifier que $T_1, T_4, T_5, T_7, T_8, T_{10}$ ou T_{12} car $p_{13} \in X$ et $p_{10} \in \overline{X}$

$\overset{MR_1 \cap MR_3}{\Rightarrow} \{p_{11}, p_{20}\} \subset \overline{X}$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{25}\})$.

• Si $p_{23} \in R$. (C_1) et (C_3) sont vérifiées $(rmq_4, \{p_{25}, p_{23}\})$.

□

6.6.3 Nouvel algorithme proposé - version squelette curviligne

Nous proposons ici une recherche par raffinements successifs de l'ensemble P nous permettant d'obtenir un ensemble de configurations P-simples, incluant celui des configurations pouvant être éliminées par l'algorithme de Palágyi et Kuba, dans sa version curviligne (dénoté à partir de maintenant par PK_C).
Dans un premier temps, la recherche s'effectue pour la sous-itération selon la direction $Haut - Sud$. Nous noterons LB_C, l'algorithme final, que nous obtenons ici à la suite de trois tentatives.

6.6.3.1 Premier essai (recherche de P_1)

La première sous-itération de PAKU12 élimine en parallèle les points dont le voisinage vérifie au moins l'une des 14 templates dans la direction $Haut - Sud$. Les points éliminés forment un sous-ensemble des points simples dont le voisin selon la direction $Haut - Sud$ appartient à $\overline{X}$. Nous proposons alors de considérer $P_1 = \{x \in X$ et $Haut - Sud(x) \in \overline{X}\}$.

Parmi les 2^{26} configurations possibles, nous obtenons $923\,551$ configurations correspondant à des points P_1^x-simples et non extrémités (un point x étant *extrémité (de courbe)* s'il n'y a qu'un seul autre point de X dans $N_{26}^*(x)$).

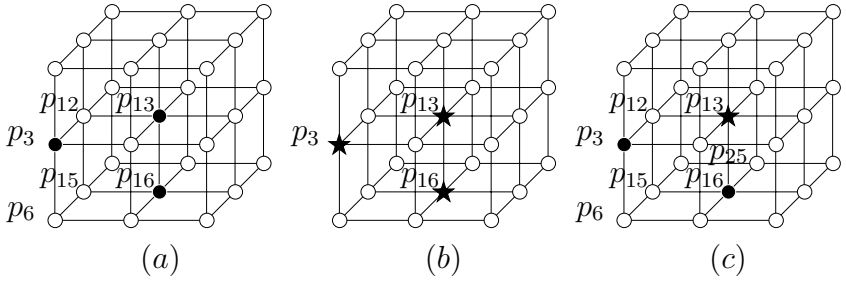


FIG. 6.14 – *La configuration (a) n'est pas P_1^x-simple (b), et est P_2^x-simple (c).*

Considérons la configuration de la figure 6.14 (a). Les trois points p_3, p_{13} et p_{16} appartiennent à P_1^x (Fig. 6.14 (b)) car ils appartiennent à X, le voisin $Haut - Sud$ de p_{13} et celui de p_{16} appartiennent à $\overline{X}$, et le voisin $Haut - Sud$ de p_3 pourrait appartenir à $\overline{X}$. Les première et troisième conditions de P_1^x-simplicité ne sont pas vérifiées par le point central p_{13} : avec $R_1^x = X \setminus P_1^x$, $T_{26}(p_{13}, R_1^x) = 0$, et par exemple, pour p_{16} de $N_{26}^*(p_{13}) \cap P_1^x$ il n'y a pas de point de R_1^x 26-adjacent à p_{16} et à p_{13}. Ainsi, le point p_{13} n'est pas P_1^x-simple. Néanmoins, il peut être éliminé par la template T_1 de $\mathcal{T}_{HS}$. Par conséquent, il devrait être éliminé par l'algorithme que nous cherchons à élaborer.

Étudions le comportement des autres points de cette configuration avec les templates de $\mathcal{T}_{HS}$ (voir Fig. 6.14 (a)). Le point p_{16} ne peut être supprimé ni par T_2, parce que p_3 $(= HSO(p_{16}))$ appartient à X, ni par les autres templates parce que p_{13} $(= H(p_{16}))$ appartient à X. Le point p_3 ne peut pas être supprimé parce que p_6, p_{15} et p_{12} appartiennent à $\overline{X}$, *i.e.* les voisins Bas, $Bas - Nord$ et $Nord$ de p_3, et toutes les templates imposent qu'au moins un de ces points

doit appartenir à X dans le but de supprimer un point central d'une configuration. Avec ces remarques, nous proposons un nouvel ensemble P_2.

6.6.3.2 Deuxième essai (recherche de P_2)

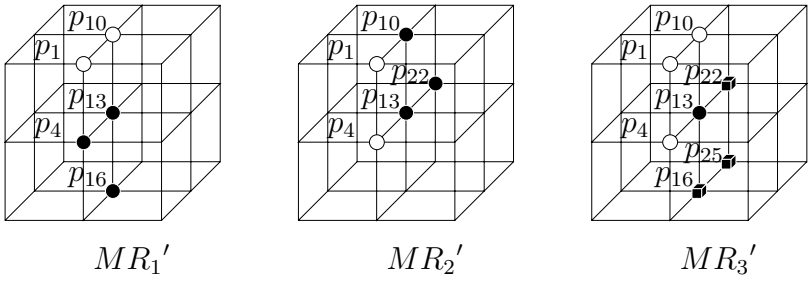


FIG. 6.15 – *Un point doit vérifier au moins l'une de ces templates pour appartenir à P_2. Au minimum un point représenté par un cube noir doit appartenir à l'objet (motif $MR_3{}'$).*

Nous imposons : $p_{13} \in X$ et $p_1(= HS(p_{13})) \in \overline{X}$ (seuls les points dont le voisin $Haut - Sud(x)$ appartient à $\overline{X}$, peuvent être supprimés). Discutons de l'appartenance des points $p_4(= S(p_{13}))$ et $p_{10}(= H(p_{13}))$ (voir également la figure 6.15).

- Si $p_4 \in X$ et $p_{10} \in \overline{X}$ alors p_{13} ne peut vérifier que T_1 et $p_{16}(= B(p_{13})) \in X$ (motif $MR_1{}'$).
- Si $p_4 \in \overline{X}$ et $p_{10} \in X$ alors p_{13} ne peut vérifier que T_2 et $p_{22}(= N(p_{13})) \in X$ (motif $MR_2{}'$).
- Si $p_4 \in \overline{X}$ et $p_{10} \in \overline{X}$ (motif $MR_3{}'$). Une condition nécessaire imposée par les templates de $\mathcal{T}_{HS}$ afin d'éliminer le point central p_{13} est qu'au moins un des points parmi les points $Bas(p_{13})$, $Nord - Bas(p_{13})$ ou $Nord(p_{13})$ (*i.e.* p_{16}, p_{25} ou p_{22}) appartienne à X (cf. discussion lors de l'analyse de la configuration de la figure 6.14 dans la section 6.6.3.1). Nous imposons alors cette condition pour le cas $p_4 \in \overline{X}$ et $p_{10} \in X$.
- Si $p_4 \in X$ et $p_{10} \in X$ alors une telle configuration n'est pas éliminée par les templates. Nous ne cherchons pas à l'éliminer non plus.

Finalement, nous proposons $P_2 = \{x \in \mathcal{Z}^3 ;\ x$ verifie au moins l'une des templates de la figure 6.15 $\}$. Notons que la configuration non P_1^x-simple de la figure 6.14 (b) est P_2^x-simple (Fig. 6.14 (c)). En effet, p_{13} appartient à P_2^x car il vérifie $MR_3{}'$; p_3 appartient à $R_2^x(= X \setminus P_2^x)$ car il ne vérifie ni $MR_1{}'$ ni $MR_2{}'$ ni $MR_3{}'$ étant donné que ses voisins Bas , $Bas - Nord$ et $Nord$ appartiennent à $\overline{X}$ (*i.e.* resp. p_6, p_{15} et p_{12}) ; p_{16} appartient à R_2^x car il ne peut vérifier ni $MR_2{}'$ car $p_{25}(= N(p_{16}))$ appartient à $\overline{X}$, ni $MR_1{}'$ ni $MR_3{}'$ car $p_{13}(= H(p_{16}))$ appartient à X. Nous obtenons $4\,672\,557$ configurations correspondant à des points P_2^x-simples et non extrémités.

Considérons la configuration de la figure 6.16 (a). Les points p_{13}, p_6 et p_{15} appartiennent à P_2^x (voir Fig. 6.16 (b)) car p_6 peut vérifier $MR_1{}'$ ou $MR_3{}'$, p_{15} peut vérifier $MR_1{}'$, p_{13} vérifie $MR_3{}'$; le point p_{25} appartient à R_2^x car $p_{13}(= HS(p_{25}))$ appartient à X. La troisième condition

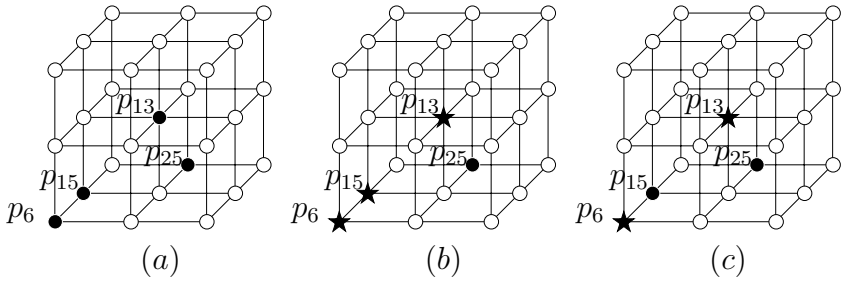


FIG. 6.16 – *La configuration* (a) *n'est pas* P_2^x*-simple* (b)*, et est* P_3^x*-simple* (c).

de P_2^x-simplicité n'est pas vérifiée : pour p_6 de $N_{26}^*(p_{13}) \cap P_2^x$, il n'y a pas de point de R_2^x 26-adjacent à p_6 et à p_{13}. Ainsi, le point p_{13} n'est pas P_2^x-simple. Néanmoins, il peut être supprimé par la template T_{12} de $\mathcal{T}_{HS}$. Une telle configuration devrait être éliminée par l'algorithme que nous recherchons.

Par l'utilisation des templates de $\mathcal{T}_{HS}$, le point p_{25} ne peut pas être supprimé car $p_{13}(= HS(p_{25}))$ appartient à X ; le point p_6 peut être supprimé au moins par la template T_2 ; mais le point p_{15} ne peut pas être supprimé ni par T_1 car p_{13} $(= HE(p_{15}))$ appartient à X, ni par les autres templates car p_6 $(= S(p_{15}))$ appartient à X. Nous proposons maintenant un ensemble P_3, de sorte que le point p_{15} de la configuration de la figure 6.16 (b) ne puisse appartenir à P_3^x.

6.6.3.3 Troisième essai (recherche de P_3)

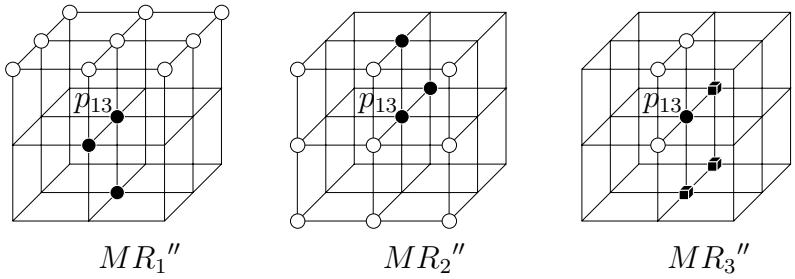


FIG. 6.17 – *Un point doit vérifier au moins l'une de ces templates pour appartenir à* P_3*. Au mininmum un point représenté par un cube noir doit appartenir à l'objet.*

En fait, dans la configuration non P_2^x-simple de la figure 6.16 (b), le point p_{15} peut vérifier $MR_1{}'$; et $MR_1{}'$ a été proposé par rapport à la template T_1. Mais p_{15} ne vérifie pas T_1, car $p_{13}(= HE(p_{15}))$ appartient à X dans la configuration mais le voisin $Haut - Est$ du point central dans T_1 n'appartient pas à X. Nous ajoutons alors à $MR_1{}'$ les autres points de la template T_1 et qui appartiennent à $\overline{X}$, obtenant ainsi le motif $MR_1{}''$. Nous faisons de même pour $MR_2{}'$ avec T_2 et nous obtenons $MR_2{}''$. Nous gardons $MR_3{}'$ et nous le renommons $MR_3{}''$.

Ainsi, nous proposons $P_3 = \{x \in \mathcal{Z}^3;\, x$ vérifie au moins l'une des templates de la figure 6.17 $\}$. Notons que la configuration non P_2^x-simple de la figure 6.16 (b) est P_3^x-simple (voir

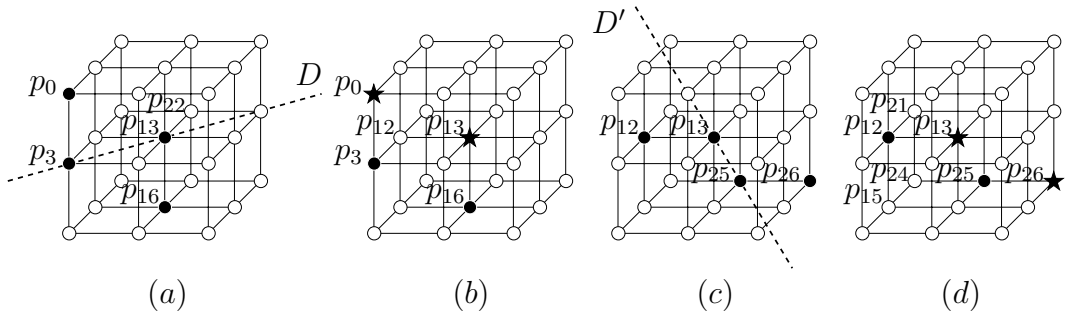


FIG. 6.18 – *Cette configuration* (a) *n'est pas supprimée par* $\mathcal{T}_{HS}$ *et* (b) *est* P_3^x*-simple,* (c) *montre une isométrie de* (a)*, elle est supprimée par* $\mathcal{T}_{HS}$ *et est* P_3^x*-simple* (d)*.*

Fig. 6.16 (c)). En effet, p_6 appartient à P_3^x car il peut vérifier $MR_3{}''$; p_{13} appartient à P_3^x car il vérifie $MR_3{}''$; le point p_{25} appartient à $R_3^x (= X \setminus P_3^x)$ car $p_{13} (= HS(p_{25}))$ appartient à X ; et le point p_{15} appartient à R_3^x car il ne peut vérifier ni $MR_2{}''$ ni $MR_3{}''$ car $p_6 (= S(p_{15}))$ appartient à X, ni $MR_1{}''$ car $p_{13} (= HE(p_{15}))$ appartient à X. Nous obtenons $2\,803\,838$ configurations correspondant à des points P_3^x-simples et non extrémités. Ces dernières incluent les $1\,379\,581$ supprimées par $\mathcal{T}_{HS}$. Le fait que les configurations supprimables par PK_C soient P_3^x-simples (pour chaque direction et par conséquent pour l'algorithme en entier) garantit que la topologie est préservée par PK_C (car PK_C supprime des sous-ensembles de points P_3^x-simples, voir chapitre 5).

Considérons la configuration de la figure 6.18 (a). Cette configuration est P_3^x-simple (voir Fig. 6.18 (b)). En effet, p_{13} appartient à P_3^x car il vérifie $MR_3{}''$; p_0 appartient à P_3^x car il peut vérifier $MR_1{}''$ ou $MR_3{}''$; p_3 appartient à R_3^x car il ne vérifie ni $MR_1{}''$ ni $MR_3{}''$ car $p_0 (= H(p_3))$ appartient à X, ni $MR_2{}''$ car $p_{12} (= N(p_3))$ appartient à $\overline{X}$; le point p_{16} appartient à R_3^x car il ne vérifie ni $MR_1{}''$, ni $MR_3{}''$ car $p_{13} (= H(p_{16}))$ appartient à X, ni $MR_2{}''$ car $p_3 (= HSO(p_{16}))$ appartient à X. Cette configuration n'est pas supprimée par $\mathcal{T}_{HS}$ (voir Fig. 6.18 (a)) : ni par les templates T_1 à T_4, ou T_{11} à T_{14} car $p_0 (= HSO(p_{13}))$ appartient à X, ni par les templates T_5 à T_{10} car $p_{22} (= N(p_{13}))$ appartient à $\overline{X}$.

La figure 6.18 (c) montre une isométrie de la configuration de la figure 6.18 (a) obtenue lorsque la droite D (passant par les points p_3 et $p_{13} (= NE(p_3))$) le long de la direction NE dans (a) est considérée selon la direction HS dans (c) ; obtenant ainsi D' (passant par les points p_{25} et $p_{13} (= HS(p_{25}))$). Cette configuration est supprimée par T_3 de $\mathcal{T}_{HS}$; ou plus directement il existe une direction de suppression Dir telle que la configuration de la figure 6.18 (a) est supprimée par T_3 de $\mathcal{T}_{Dir}$. Remarquons que cette configuration est P_3^x-simple (voir Fig. 6.18 (d)), en effet, p_{13} appartient à P_3^x car il vérifie $MR_3{}''$; p_{26} appartient à P_3^x car il peut vérifier $MR_3{}''$; p_{25} appartient à R_3^x car $p_{13} (= HS(p_{25}))$ appartient à X ; et p_{12} appartient à R_3^x car les voisins Bas, $Bas - Nord$ et $Nord$ de p_{12} (*i.e.* resp. p_{15}, p_{24} et p_{21}) appartiennent à $\overline{X}$.

Pour une meilleure comparaison entre PK_C et LB_C, nous générons les configurations supprimées par ces algorithmes pour chaque direction : PK_C supprime $11\,268\,606$ configurations, *i.e.* il existe au moins une direction de suppression telle qu'une configuration donnée parmi celles-là est supprimée selon cette direction par PK_C ; LB_C supprime $19\,327\,098$ configura-

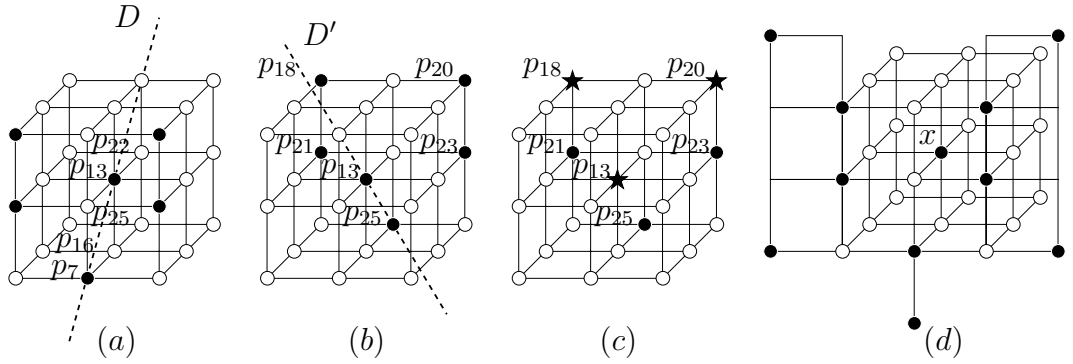


FIG. 6.19 – (a) *Cette configuration ne peut pas être supprimée par* PK_C *quelle que soit la direction de suppression, et n'est pas* P_3^x*-simple,* (b) *montre une isométrie de* (a)*, elle est* P_3^x*-simple* (c)*, dans* (d) *(obtenue à partir de* (a)*), aucun point n'est supprimée par* PK_C*, néanmoins* x *est supprimé par* LB_C*.*

tions (70.6% de plus). La configuration décrite à la figure 6.19 (a) ne peut pas être supprimée par PK_C, quelle que soit la direction de suppression considérée. Le point p_{13} appartient à R_3^x car les voisins Bas, $Bas - Nord$ et $Nord$ de p_{13} (*i.e.* resp. p_{16}, p_{25} et p_{22}) appartiennent à $\overline{X}$, ainsi p_{13} n'est pas P_3^x-simple. Cependant, lorsque la droite D dans (a) (passant par les points p_7 et $p_{13}(= HN(p_7))$) est considérée selon la direction $Haut - Sud$ dans (b), obtenant ainsi D' (passant par les points p_{25} et $p_{13}(= HS(p_{25}))$), alors la configuration obtenue est P_3^x-simple (Fig. 6.19 (c)). En effet, les points p_{18} et p_{20} appartiennent à P_3^x car ils peuvent vérifier ${MR_3}''$; p_{13} appartient à P_3^x car il vérifie ${MR_3}''$; p_{25} appartient à R_3^x car $p_{13}(= HS(p_{25}))$ appartient à X ; p_{21} appartient à R_3^x car il ne vérifie ni ${MR_1}''$, ni ${MR_3}''$ car $p_{18}(= H(p_{21}))$ appartient à X, ni ${MR_2}''$ car $p_{13}(= SE(p_{21}))$ appartient à X ; p_{23} appartient à R_3^x car il ne vérifie ni ${MR_1}''$, ni ${MR_3}''$ car $p_{20}(= H(p_{23}))$ appartient à X, ni ${MR_2}''$ car $p_{13}(= SO(p_{23}))$ appartient à X. À la figure 6.19 (d) est montrée une image construite à partir de la configuration de la figure 6.19 (a) telle que tout point est soit non simple (sauf x) soit extrémité, et qu'aucun point ne peut être supprimé par PK_C ; seul le point x peut être supprimé par LB_C, selon la même direction donnant l'isométrie dans la figure 6.19 (b).

Avec ce troisième exemple, nous allons obtenir les configurations correspondant aux points P_3^x-simples et non extrémités de surface.

6.6.3.4 Autre essai

Il est possible de proposer la condition d'appartenance suivante afin de mieux respecter des symétries : $P_4 = \{x \in \mathcal{Z}^3 \,;\, x$ vérifie au moins l'une des templates de la figure 6.20$\}$.

6.6.3.5 Propriétés de PK_C et de LB_C

Résumons cette étude par les deux propriétés suivantes :

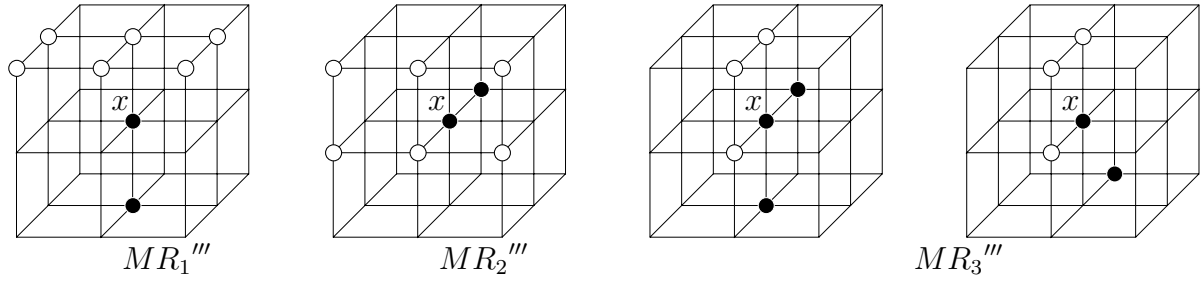


FIG. 6.20 – *Autre proposition.*

Propriété : Les points supprimés par PK_C sont des points P_3^x-simples ; par conséquent PK_C préserve la topologie.

Propriété : L'algorithme LB_C opérant par suppression de points P_3^x-simples non terminaux de courbe supprime au moins les points retirés par PK_C, tout en préservant les mêmes points terminaux.

6.6.4 Récapitulatif

Les figures 6.21 et 6.22 proposent des configurations P_1^x, P_2^x, P_3^x-simples ou non, éliminées par PK_C ou non, pour une sous-itération (Fig. 6.21) ou pour une itération (Fig. 6.22).
Sur ces figures, si une configuration est reliée à un "nœud"-algorithme par un trait, alors elle est éliminée par celui-là, sauf dans le cas où le trait est barré. À proximité de chaque "nœud"-algorithme, est indiqué le nombre de configurations que l'algorithme supprime, pour une sous-itération (Fig. 6.21) ou pour une itération (Fig. 6.22).

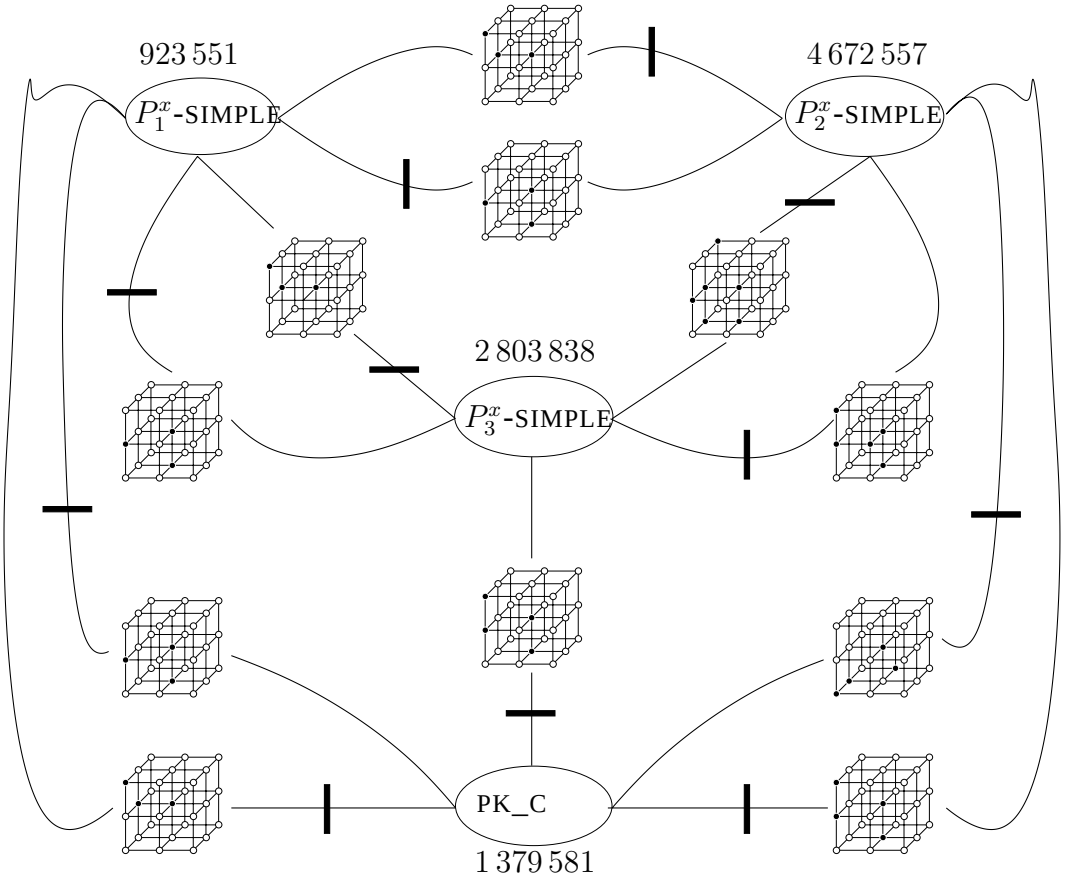


FIG. 6.21 – *Récapitulatif : sous-itération (version squelette curviligne).*

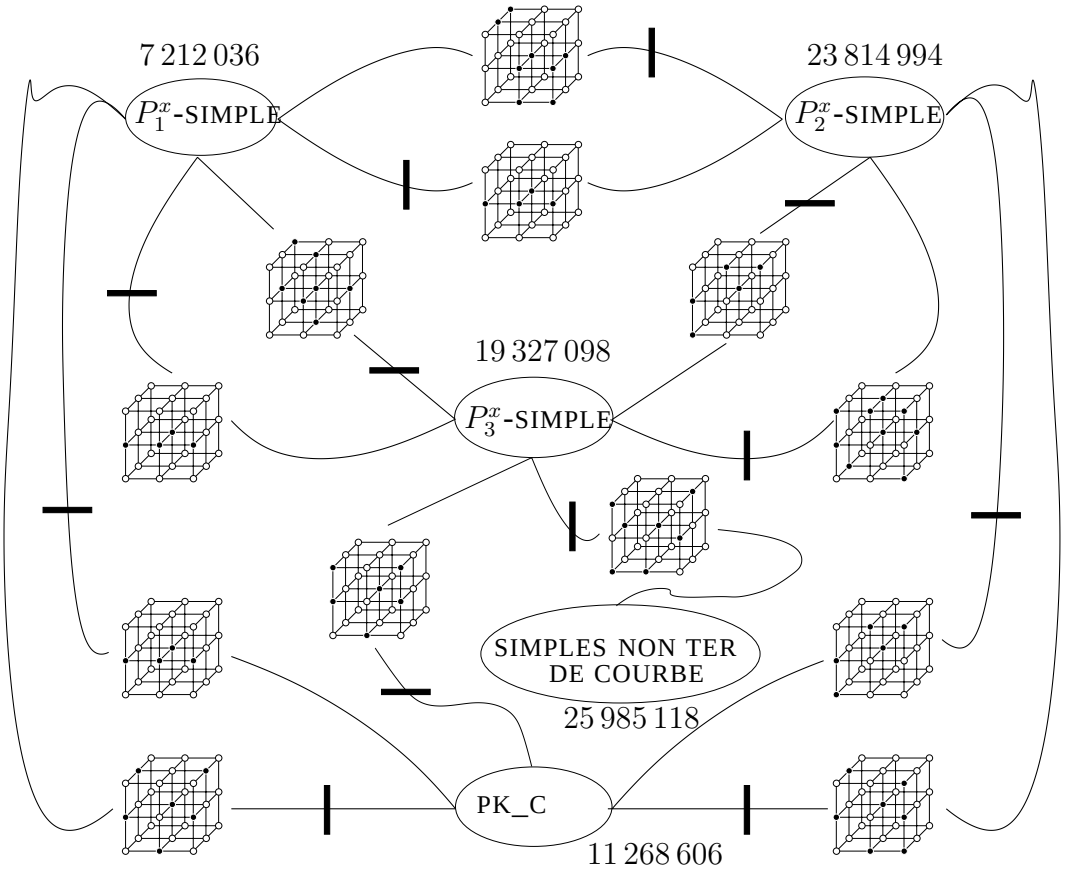


FIG. 6.22 – *Récapitulatif : itération (version squelette curviligne).*

6.6.5 Algorithme de Palágyi - version squelette surfacique

L'algorithme de Palágyi et Kuba, proposé pour l'obtention de squelettes surfaciques (noté PK_S, par la suite), a été donné sous forme de templates. Ces dernières ont été obtenues en ne sélectionnant que les configurations non terminales de surface obtenues à partir des templates permettant l'obtention de squelettes curvilignes. La caractérisation utilisée pour un point terminal de surface est la suivante : un point $p \in X$ est *terminal de surface* si $N_6(p)$ contient au moins une paire de points opposés (par leur direction) et appartenant à $\overline{X}$. À la figure 6.23, est représenté l'ensemble $\mathcal{T}'_{HS}$ des templates pour la direction $Haut - Sud$, utilisé pour l'obtention du squelette surfacique.

Dans la section 6.6.2.1, nous avons vu que l'algorithme de Palágyi et Kuba, selon la version curviligne, était P-simple. Comme la version surfacique ne supprime que certaines configurations parmi celles supprimées dans la version curviligne, alors d'après les propriétés sur les points P-simples, l'algorithme décliné dans sa version surfacique préserve également la topologie.

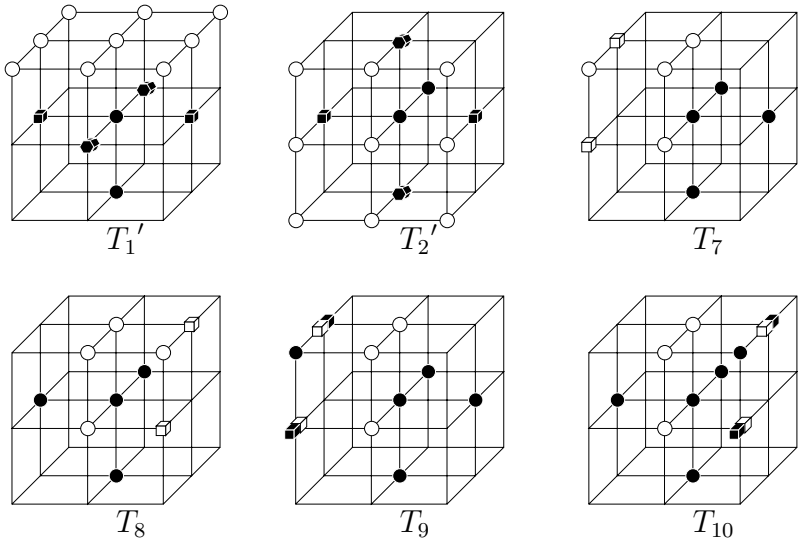


FIG. 6.23 – *Templates utilisées lors de la sous-itération de suppression selon la direction $Haut - Sud$, pour l'obtention du squelette surfacique, avec l'algorithme* PK_S. *Au minimum un point représenté par un cube noir et un hexagone noir en perspective doivent appartenir à l'objet dans les templates $T_1{}'$ à $T_2{}'$. Au minimum un point représenté par un cube blanc doit appartenir au background dans les templates T_7 et T_8. Deux points marqués par deux parallélépipèdes bicolores doivent être différents l'un de l'autre dans les templates T_9 et T_{10}. Les points non représentés appartiennent soit à l'objet soit au complémentaire.*

6.6.6 Nouvel algorithme proposé

De même que K. Palágyi et A. Kuba, nous ne retenons que les configurations qui ne sont pas terminales de surface parmi les configurations précédentes P_3^x-simples permettant l'obtention de

squelettes curvilignes. Nous notons LB_S, l'algorithme supprimant les points P_3^x-simples non terminaux de surface.

Nous avons obtenu $1\,228\,800$ configurations P_3^x-simples et non terminales de surface ; l'algorithme de Palágyi et Kuba en retient $1\,155\,072$, pour une même sous-itération.

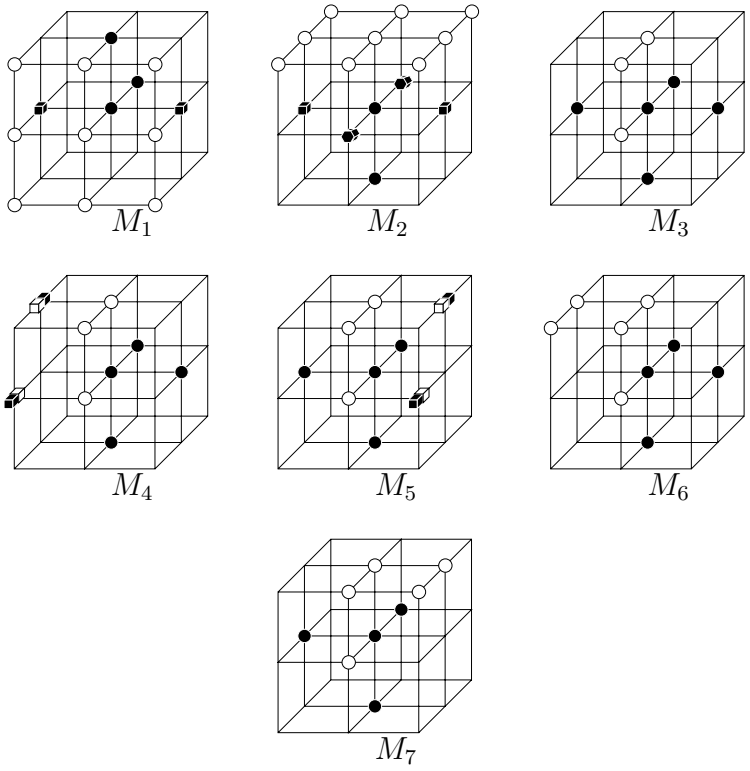


FIG. 6.24 – *Templates proposées pour l'algorithme de squelettisation surfacique en 12 sous-itérations (*LB_S*), selon la stratégie des points P-simples (ensemble P_3), et pour la direction $Haut - Sud$. Au minimum un point représenté par un cube noir doit appartenir à l'objet dans les templates M_1 et M_2. Au minimum un point représenté par un hexagone noir en perspective doit appartenir à l'objet dans la template M_2. Deux points marqués par les parallélépipèdes bicolores doivent être différents l'un de l'autre dans les templates M_4 et M_5. Les points non représentés appartiennent soit à l'objet soit au complémentaire.*

Contrairement à LB_C, nous avons réussi à obtenir un jeu restreint de templates décrivant les configurations P_3^x-simples et non terminales de surface. Cet ensemble de templates a été obtenu à l'aide des BDD (voir l'annexe C). L'ensemble de ces templates est représenté selon la direction $Haut - Sud$ à la figure 6.24. Un point qui vérifie au moins l'une d'entre elles, sera supprimé par LB_S, pour la direction HS. Ainsi, il est possible d'implémenter LB_S, de la même façon que PK_S, seulement à partir de templates ; nous évitons ainsi de tester les conditions de P-simplicité et celle d'être un point terminal de surface. Nous pouvons aussi voir que les templates de $\mathcal{T}'_{HS}$ sont strictement "incluses" dans les nôtres : par exemple, $T'_1 = M_2$, $T'_2 \subseteq [M_1 \cup M_3 \cup M_4 \cup M_5 \cup M_6 \cup M_7]$, $T_7 \subseteq [M_4 \cup M_6]$, $T_8 \subseteq [M_5 \cup M_7]$, $T_9 \subseteq M_4$, $T_{10} \subseteq M_5$;

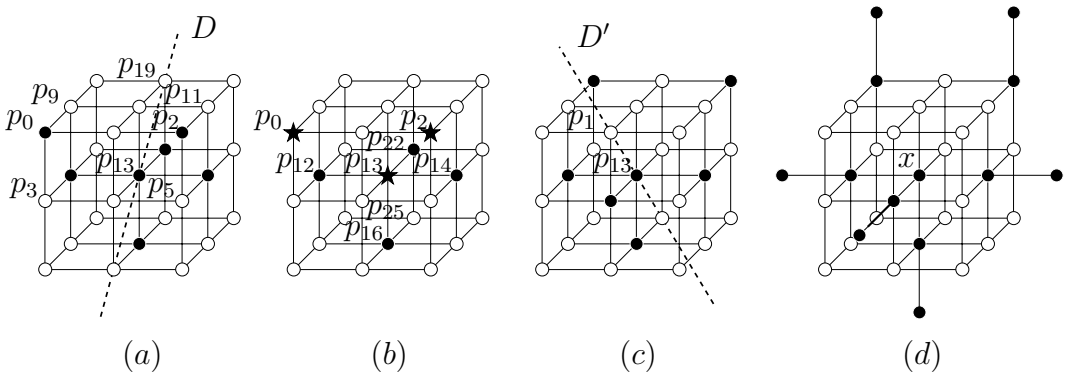


FIG. 6.25 – (a) *Cette configuration ne peut pas être supprimée par* PK_S *quelle que soit la direction de suppression, et est P_3^x-simple* (b)*,* (c) *montre une isométrie de* (a)*, elle est P_3^x-simple* (c)*, dans* (d) *(obtenue à partir de* (c)*) aucun point n'est supprimé par* PK_S*, néanmoins x est supprimé par* LB_S.

$T_i \subseteq M_j$ (resp. $T_i = M_j$) signifiant que les configurations supprimées par T_i sont parmi (resp. sont exactement) celles qui sont supprimées par M_j, ou par une union de M_j. Cela confirme que nous pouvons supprimer au moins les configurations supprimées par PK_S. Nous pouvons également vérifier que nos templates interdisent la suppression de points terminaux de surface.

Considérons la configuration de la figure 6.25 (a). Elle n'est pas supprimée par $\mathcal{T}'_{HS}$: ni par les templates T'_1, T'_2 ou T_7 car $p_0(= HSO(p_{13}))$ appartient à X ; ni par T_8 car $p_2(= HSE(p_{13}))$ appartient à X ; ni par T_9 car $p_3(= SO(p_{13}))$ et $p_9(= HO(p_{13}))$ appartiennent à $\overline{X}$; ni par T_{10} car $p_5(= SE(p_{13}))$ et $p_{11}(= HE(p_{13}))$ appartiennent à $\overline{X}$. Cependant, elle correspond à un point P_3^x-simple et non terminal de surface (Fig. 6.25 (b)). En effet, les points p_0 et p_2 appartiennent à P_3^x car ils peuvent vérifier ${MR_3}''$; le point p_{13} appartient à P_3^x car il vérifie ${MR_3}''$; p_{16} appartient à R_3^x car il ne vérifie ni ${MR_1}''$ ni ${MR_3}''$ car $p_{13}(= H(p_{16}))$ appartient à X, ni ${MR_2}''$ car $p_{25}(= N(p_{16}))$ appartient à $\overline{X}$; p_{12} appartient à R_3^x car il ne vérifie ni ${MR_1}''$ ni ${MR_2}''$ ni ${MR_3}''$, car $p_0(= HS(p_{12}))$ appartient à X ; p_{14} appartient à R_3^x car il ne vérifie ni ${MR_1}''$ ni ${MR_2}''$ ni ${MR_3}''$, car $p_2(= HS(p_{14}))$ appartient à X ; p_{22} appartient à R_3^x car il ne vérifie ni ${MR_2}''$ ni ${MR_3}''$ car $p_{13}(= S(p_{22}))$ appartient à X, ni ${MR_1}''$ car $p_{25}(= B(p_{22}))$ appartient à $\overline{X}$. En fait, cette configuration peut être supprimée par M_3, l'une de nos templates, donnée à la figure 6.24. Cependant, cette configuration n'est pas supprimée par PK_S, quelle que soit la direction de suppression.

Nous avons de même généré les configurations supprimées par LB_S, pour chaque direction. PK_S supprime $9\,101\,312$ configurations ; LB_S en supprime $9\,986\,048$ (9.7% en plus). La figure 6.25 (c) montre une isométrie de la configuration de la figure 6.25 (a), obtenue lorsque la ligne D (passant par les points p_{13} et $p_{19}(= HN(p_{13}))$) selon la direction HN dans (a) est considérée selon la direction HS dans (c), obtenant ainsi D' (passant par les points p_{13} et $p_1(= HS(p_{13}))$). Cette configuration n'est pas supprimée par PK_S, comme nous l'avons dit auparavant. La figure 6.25 (d) montre une image construite à partir de la configuration de la figure 6.25 (c) telle que tout point est soit non simple (sauf x) soit terminal de surface ; et aucun point ne peut être

supprimé par PK_S ; seul le point x peut être supprimé par LB_S. En fait, Palágyi et Kuba ont exclu la configuration de la figure 6.25 (a) (voir [PK99], p. 207, Fig. 6). Ils prétendent que si l'ensemble des templates $\mathcal{T}_{HS}$ peut la supprimer, alors de mauvais segments de courbes ou de surface peuvent être créés. Cela n'est peut-être pas le cas de notre algorithme car il supprime plus de points que l'algorithme de Palágyi et Kuba.

6.6.6.1 Propriétés de PK_S et de LB_S

Résumons cette étude par les quatre propriétés suivantes :

Propriété : Les points supprimés par PK_S sont des points P_3^x-simples ; par conséquent PK_S préserve la topologie.

Propriété : Les points retirés par PK_S sont retirés par PK_C.

Propriété : L'algorithme LB_S opérant par suppression de points P_3^x-simples non terminaux de surface supprime au moins les points retirés par PK_S, tout en préservant les mêmes points terminaux.

Propriété : Les points retirés par LB_S sont retirés par LB_C.

6.6.7 Récapitulatif

Les figures 6.26 et 6.27 proposent des configurations P_2^x, P_3^x-simples ou non, éliminées par PK_S ou non, pour une sous-itération (Fig. 6.21) ou pour une itération (Fig. 6.22). Bien qu'il ne retire pas les configurations retirées par PK_S, nous avons également testé l'algorithme supprimant les points P_2-simples, celui-là supprime le plus grand nombre de configurations 26-simples et non terminales de surface parmi les algorithmes que nous avons proposés (comme dans le cas curviligne).

6.6.8 Récapitulatif pour l'algorithme en 12 sous-itérations

Nous récapitulons les nombres de configurations P-simples (non extrémités) ou vérifiant les masques de PK pour une sous-itération ou pour une itération (c'est-à-dire pour les 12 directions).

6.6.8.1 Squelette curviligne

	PK_C	P_1^x-simples	P_2^x-simples	P_3^x-simples
sous-itération	1 379 581	923 551	4 672 557	2 803 838
itération	11 268 606	7 212 036	23 814 994	19 327 098

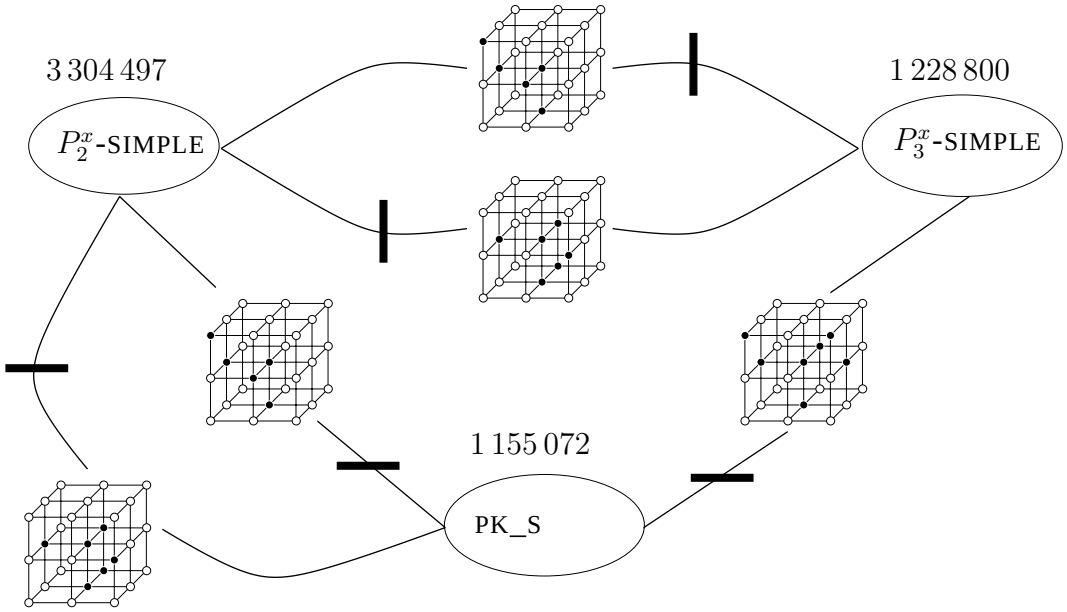


FIG. 6.26 – *Récapitulatif : sous-itération (version squelette surfacique).*

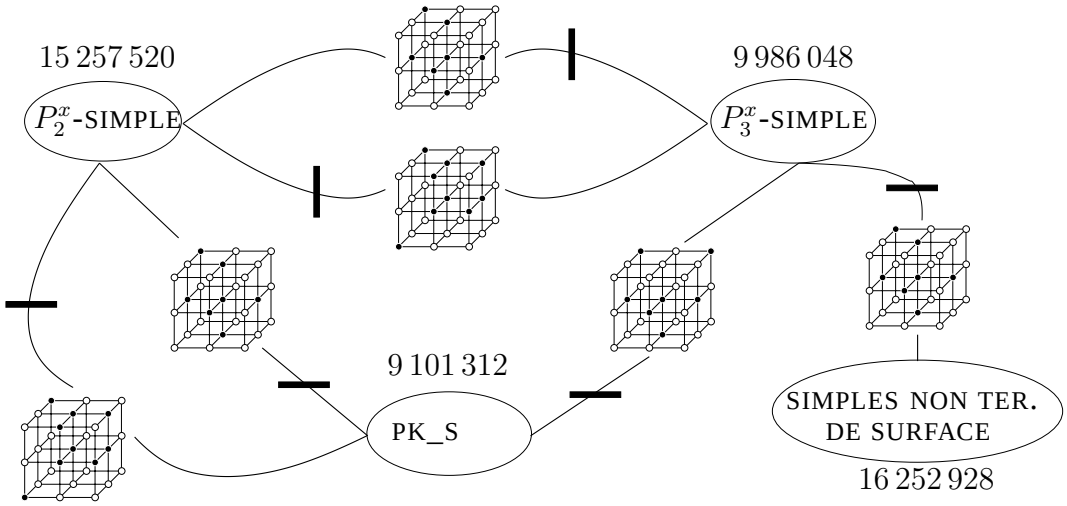


FIG. 6.27 – *Récapitulatif : itération (version squelette surfacique).*

6.6.8.2 Squelette surfacique

	PK_S	P_2^x-simples	P_3^x-simples
sous-itération	1 155 072	3 304 497	1 228 800
itération	9 101 312	15 257 520	9 986 048

6.6.8.3 Autres résultats

26-simples	:	25 985 144
et non terminales de courbe	:	25 985 118
et non terminales de surface	:	16 252 928

6.6.8.4 Résultats observés

Les squelettes de quelques images, obtenus respectivement par PK_C, LB_C, PK_S et LB_S, sont donnés à la figure 6.28 (la représentation est décrite dans la section B.1). Nous observons que :

- L'apparence géométrique est la même entre PK_C et LB_C, ou entre PK_S et LB_S.
- Le nombre de sous-itérations de suppression exigé par LB_C est inférieur ou égal à celui de PK_C. Le nombre de points supprimés par LB_C est inférieur ou égal à celui de PK_C. Le centrage résultant n'est pas le même. Nous rappelons qu'il est possible que LB exige plus de sous-itérations pour obtenir un squelette qu'avec PK (voir les figures 6.19 (d) et 6.25 (d)).
- Sur ces exemples, le nombre de sous-itérations de suppression, le nombre de points supprimés et les squelettes sont les mêmes pour PK_S et LB_S.

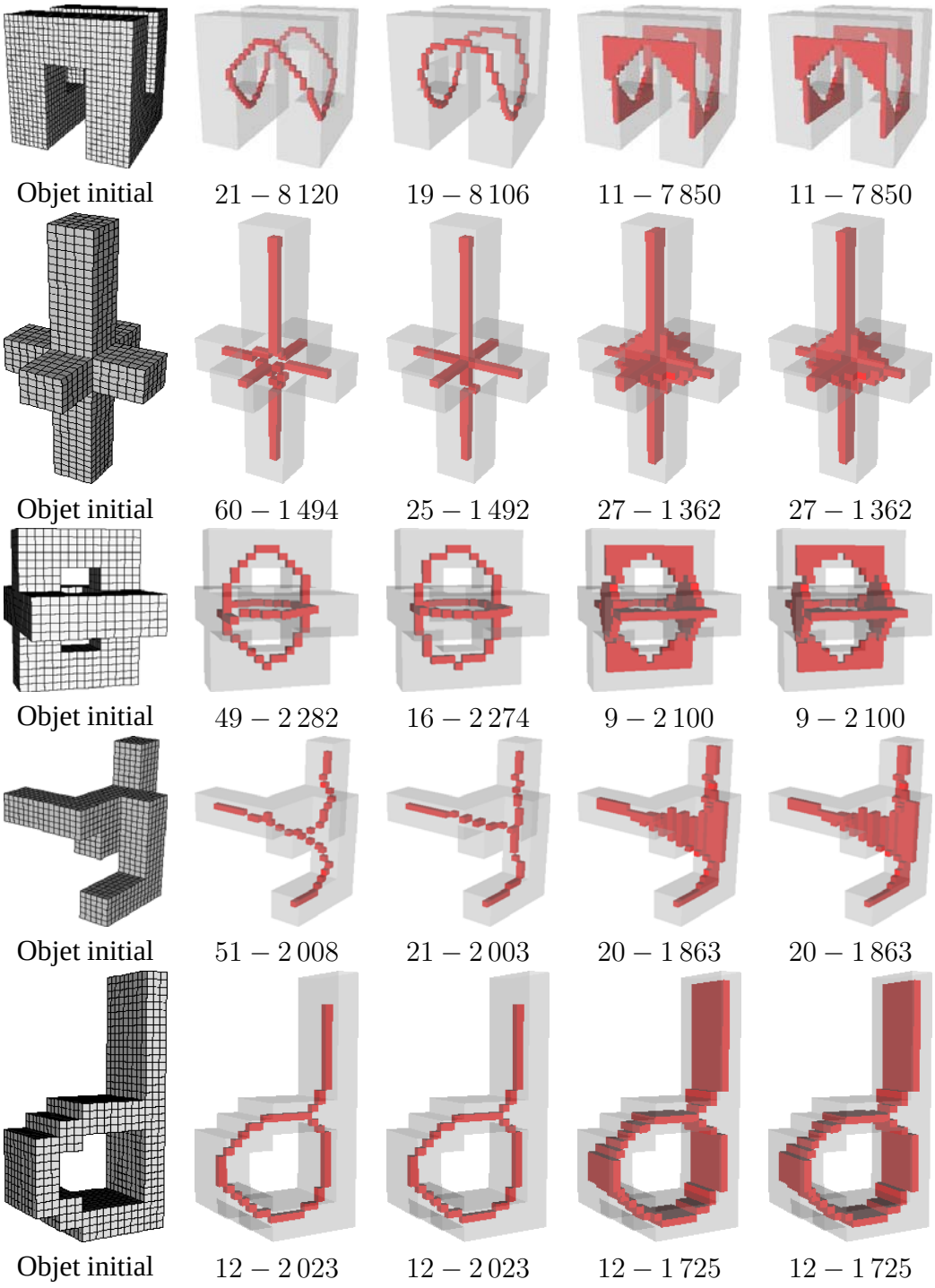


FIG. 6.28 – *Par colonne, respectivement : un objet initial, les squelettes curvilignes avec* PK_C *et* LB_C*, puis les squelettes surfaciques avec* PK_S *et* LB_S*. Sous chaque figure, sont donnés le numéro de la dernière sous-itération effective de suppression et le nombre de points supprimés (*REPR7*).*

6.7 Algorithmes fortement parallèles

6.7.1 Introduction

Nous étudions trois algorithmes de squelettisation de type fortement parallèles pour des images 3D. Par ordre chronologique, le premier a été proposé par C.M. Ma [Ma95], le second a été proposé par C.M. Ma et M. Sonka [MS96], le plus récent a été proposé par A. Manzanera, et al. [MBPL99b].

6.7.1.1 Notes à propos de la recherche sur l'algorithme de Ma, et sur celui de Ma et Sonka

En essayant de montrer la P-simplicité de chacun des deux premiers algorithmes, nous avons exhibé une configuration non P-simple pour chaque algorithme. Nous savons que la non P-simplicité d'un algorithme n'entraine pas forcément que cet algorithme ne préserve pas la topologie. Néanmoins, lorsque nous avons exhibé une configuration qui est telle que l'algorithme n'est pas P-simple, celle-là peut nous aider à examiner si l'algorithme préserve la topologie (par rapport à cette configuration).

Les deux configurations exhibées sont telles que la topologie n'est pas préservée. Nous avons choisi de retracer la démarche nous amenant à la première configuration, cela afin d'illustrer plus concrètement les tests de P-simplicité sur le premier algorithme ; nous donnerons directement (afin d'alléger la rédaction) la seconde configuration mettant en échec le second algorithme.

Nous avons eu la confirmation auprès du Professeur C.M. Ma que ces configurations montrent effectivement la non-validité de ses algorithmes. Nous travaillons actuellement avec lui, afin de les corriger, en adoptant une approche basée sur une extension des points P^x-simples dans un voisinage étendu (voir la section 6.8).

6.7.1.2 Notes à propos de la recherche sur l'algorithme de Manzanera, et al.

L'algorithme de Manzanera a été proposé dans le cas des images 2D [MBPL99b] et a été étendu aux images 3D [MBPL99a] (voir n-D [MBPL99c]). Nous allons montrer que cet algorithme est P-simple (cas 2D et 3D).

En plus d'être concis et efficace [BM99], cet algorithme utilise une approche originale en testant la présence d'un point intérieur dans le voisinage d'un point pouvant être supprimé. Cet algorithme n'utilise pas de conditions directionnelles, comme le faisaient les deux premiers algorithmes cités auparavant. Il a la propriété de retirer des points de façon isotrope, le squelette obtenu est alors bien centré ; à sa décharge, il utilise un plus grand support. Notons également qu'il n'utilise pas explicitement de condition de point terminal.

C'est à partir de l'idée de supprimer des points voisins d'un point intérieur (avec d'autres conditions), que nous avons proposé un ensemble P, pour le cas 2D, de sorte que les configurations P-simples "incluent" les configurations pouvant être supprimées par l'algorithme de Manzanera (la version 2D de cet algorithme est décrite dans la section A.5.6). Nous avons exploité la possibilité d'étendre directement nos résultats au cas des images 3D (comme l'ont fait A. Manzanera, et al.).

En proposant une extension directe de l'algorithme de Manzanera, et al. dans le cas d'une squelettisation curviligne (définition classique des points extrémités), il nous est également possible de proposer un algorithme P-simple retirant au moins les points supprimés par l'algorithme de Manzanera, et al., en 2D, et en 3D. Nous avons également proposé un algorithme P-simple dans le cadre d'une squelettisation surfacique qui est équivalent à l'algorithme de Manzanera, et al., dans le sens qu'il retire les mêmes points.

En fait, notre travail actuel consiste à finaliser ces recherches ; ces dernières ne sont pas détaillées davantage dans ce rapport. Seule la preuve de la P-simplicité de cet algorithme y est décrite.

6.7.2 Algorithme de Ma [Ma95]

6.7.2.1 Rappels sur l'algorithme

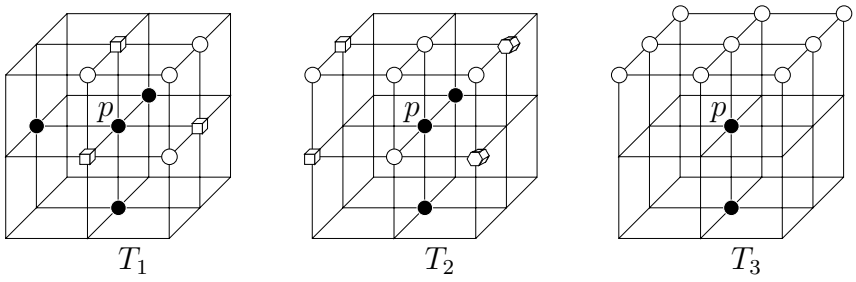


FIG. 6.29 – *Templates utilisées par l'algorithme de squelettisation de Ma. Au minimum un point représenté par un cube blanc ou un "hexagone 3D" blanc en perspective doivent appartenir au complémentaire.*

Points extrémités :
Un point x de l'objet est dit *extrémité* s'il y a exactement un seul point appartenant à l'objet, soit dans $N_{26}^{*}(x)$, soit dans n'importe lequel des trois plans orthogonaux 3×3 le contenant.

Règle 2.1 **:**
Si les quatre coins d'un carré unité doivent être supprimés, alors le coin dont la somme des coordonnées est la plus petite doit être préservé, si et seulement si, il est non simple après la suppression des trois autres.

Conditions directionnelles et de non-extrémité :

$$\begin{cases} (C\_Ma_1)\ p \text{ ne doit pas être un point extrémité,} \\ (C\_Ma_2) \text{ si } p \text{ est un point de bord } Nord, & \text{alors } Sud - Sud(p) & \text{doit appartenir à } X, \\ (C\_Ma_3) \text{ si } p \text{ est un point de bord } Est, & \text{alors } Ouest - Ouest(p) & \text{doit appartenir à } X, \\ (C\_Ma_4) \text{ si } p \text{ est un point de bord } Haut, & \text{alors } Bas - Bas(p) & \text{doit appartenir à } X. \end{cases}$$

L'algorithme consiste à supprimer en parallèle tous les points non préservés par la règle 2.1 et qui vérifient les conditions directionnelles et de non-extrémité précédentes, et au moins une template, à isométries près, parmi celles décrites à la figure 6.29.

Pour plus d'informations, se reporter à la description de cet algorithme à la section B.4.1.

6.7.2.2 Tests de la P-simplicité de l'algorithme

Lors du test de la P-simplicité de l'algorithme de Ma, nous avons exhibé une configuration qui n'était pas P-simple, plus précisément qui ne vérifiait pas la condition (C_4) (cf. la proposition, page 80). Ensuite, nous avons constaté que cette configuration montrait que l'algorithme de Ma ne préservait pas la topologie. Nous décrivons maintenant l'obtention de cette configuration.

Soit P, l'ensemble des configurations pouvant être éliminées par l'algorithme. Nous cherchons à mettre en défaut la condition (C_4) de P-simplicité sur une configuration X. Considérons un point x de X appartenant à P. Supposons que le point x vérifie la template T_3 selon la direction $Haut$ (Fig. 6.30 (a)). Soit $y = Bas(x)$, puisque x est un point de bord $Haut$, alors $z = Bas - Bas(x)$ doit appartenir à X (C_Ma_4). Supposons ensuite que y vérifie une isométrie de la template T_1 (symétrie d'axe $Nord - Sud$, isométrie acceptée d'après le texte), la condition qu'exige la template T_1 pour qu'au moins un des trois points, indiqués à la figure 6.29, appartienne au complémentaire, doit être vérifiée (c.-à-d. $z_1 \in \overline{X}$ ou $z_2 \in \overline{X}$, z a été imposé comme appartenant à X) (Fig. 6.30 (b)).

Le point y vérifie la template T_1 ; nous supposons que y vérifie les autres conditions requises de façon à ce que y appartienne à P, il faudra le vérifier par la suite. La condition (C_4) "essaye de trouver" un carré unité $\{x, y, z, t\}$ avec $z \in \overline{X}$ et $t \in \overline{X}$. Nous imposons maintenant que z_1 et z_2 appartiennent à $\overline{X}$. Une façon de ne pas vérifier (C_4) consiste à imposer que t_1 et t_2 appartiennent à X (Fig. 6.30 (c)). Nous avons alors une configuration qui ne vérifie pas la condition (C_4) de P-simplicité (x n'est pas P-simple), encore faut-il vérifier que cette configuration réponde à toutes les exigences imposées par l'algorithme de Ma.

Nous supposons maintenant que les points non représentés dans la figure 6.30 (c) appartiennent à $\overline{X}$; les conditions directionnelles de Ma, afin de supprimer x et y font que certains de ces points doivent en fait appartenir à X : x est un point de bord $Nord$ alors $(C\_Ma_2) \Rightarrow Sud - Sud(x) \in X$, x est un point de bord Est alors $(C\_Ma_3) \Rightarrow Ouest - Ouest(x) \in X$. Nous pouvons constater que les points supprimés par les templates sont des points simples ; par la suite, lorsqu'un point est détecté non simple (par les nombres topologiques, par exemple) alors ce point ne peut être supprimé par les templates et a fortiori par l'algorithme de Ma. Nous complétons avec d'autres points : $Est - Est(y)$, $Nord - Nord(y)$ et $Bas - Bas(y)$ appartiennent à X (la configuration obtenue est décrite à la figure 6.30 (d)) ; de façon à ce que seuls les points x et y soient simples et non-extrémités ; les points t_1, t_2, y_1, y_2 et z sont non simples car $\forall v \in \{t_1, t_2, y_1, y_2, z\}$, on a $T_{26}(v, X) = 2$; et les autres points sont extrémités. Il n'y a pas non plus de carré unité de points de X dans cette configuration ; la règle 2.1 ne peut donc intervenir. L'algorithme élimine alors les deux points x et y.

Nous obtenons l'objet de la figure 6.30 (e), celui-là comporte un trou (pouvant être matérialisé par le chemin fermé non déformable : t_2, t_1, y_1, y_2, t_2), la topologie n'est alors pas préservée. Nous pouvons vérifier que l'algorithme s'arrête à ce stade : en effet, les points t_1, t_2, y_1, y_2, et z

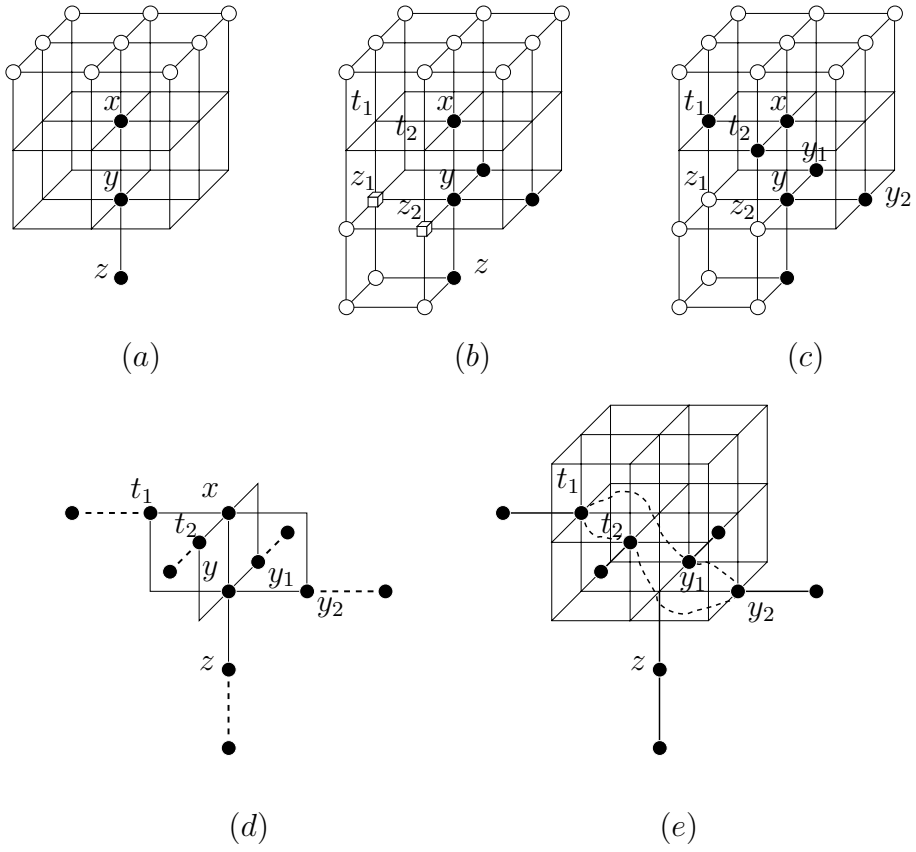


FIG. 6.30 – *Configuration montrant que l'algorithme de Ma ne préserve pas la topologie.* (a) *Le point* x *vérifie* T_3*,* (b) *le point* y *vérifie* T_1*,* (c) *ajout de certains points,* (d) *prolongement par des courbes afin de rendre tous les points, autres que* x *et* y*, extrémités ou non simples,* (e) *création d'un trou après suppression de* x *et de* y*.*

sont non simples pour $X \setminus \{x, y\}$ car $\forall v \in \{t_1, t_2, y_1, y_2\}$, on a $T_{26}(v, X \setminus \{x, y\}) = 3$, z est tel que $T_{26}(z, X \setminus \{x, y\}) = 2$, et les autres points sont extrémités.

En fait, les points x et y sont simples pour X. Mais x n'est pas simple pour $X \setminus \{y\}$ ($T_6(x, \overline{X \setminus \{y\}}) = 2$) et y n'est pas simple pour $X \setminus \{x\}$ ($T_6(y, \overline{X \setminus \{x\}}) = 2$). L'ensemble $\{x, y\}$ est alors un ensemble non simple pour lequel chaque sous-ensemble propre est simple ; c'est un MNS (cf. section 3.8).

Nous avons localisé l'erreur dans l'article [Ma95] concernant le problème apparaissant dans la configuration de la figure 6.30.
Introduisons les notations suivantes : un point p vérifie Ω s'il vérifie une template à isométrie près, les conditions directionnelles et de non-extremité. Soient p et q deux points qui vérifient Ω. La notation "$q \in \Omega(p)$" signifie que p ne satisfait plus Ω après la suppression de q, et "$q \notin \Omega(p)$" signifie que p satisfait encore Ω après la suppression de q.

Rappelons le lemme 3.2 ([Ma95], p. 86) : Soient p et q, deux points 6-adjacents tels que p et

q satisfont Ω. Alors $q \notin \Omega(p)$ ou $p \notin \Omega(q)$.

La preuve de ce lemme considère le morceau de droite $a - q - p - b$ (avec a 6-adjacent à q et b 6-adjacent à p). Puis sont supposées à la fois les conditions $p \in \Omega(q)$ et $q \in \Omega(p)$. D'après l'observation des templates : si $p \in \Omega(q)$, alors a appartient au complémentaire $(*_1)$; si $q \in \Omega(p)$ alors b appartient au complémentaire $(*_2)$. Au moins l'un des deux points p ou q est alors un point de bord et vérifie une condition directionnelle ; cela entraine que a ou b appartient à l'objet, d'où la contradiction.

La démonstration de ce lemme n'est pas valide. En effet, considérons la figure 6.30 (d), x et y vérifient tous deux Ω et sont 6-adjacents entre eux. Si nous supprimons x, alors y devient non simple, *c.-à-d.* $x \in \Omega(y)$ ou vice versa si nous supprimons d'abord y alors x devient non simple, *c.-à-d.* $y \in \Omega(x)$; et ici, il existe un 6-voisin de x ou de y (aligné sur le morceau de droite (xy)), en fait c'est le point $(z = Bas(y))$, non dans le complémentaire, ce qui contredit $(*_1)$ ou $(*_2)$. Nous avons donc une configuration telle que x et y sont deux points 6-adjacents tels que x et y satisfont Ω et $x \in \Omega(y)$ et $y \in \Omega(x)$. En conclusion, le lemme 3.2 est faux, et l'algorithme n'est pas valide car il ne préserve pas la topologie.

Comme ce lemme est également utilisé de la même façon dans l'algorithme fortement parallèle proposé par C.M. Ma et M. Sonka ; ce dernier ne préserve pas la topologie. C'est ce que nous illustrons dans la section suivante. Résumons cette section par la propriété suivante :

Propriété : L'algorithme de Ma ne préserve pas la topologie.

6.7.3 Algorithme de Ma et Sonka [MS96]

Par un raisonnement similaire, nous avons exhibé une configuration éliminée par l'algorithme de Ma et Sonka, mettant en échec une condition de P-simplicité. De plus, nous avons constaté que la suppression des points dans cette configuration, par cet algorithme, ne préservait pas la topologie.

Contrairement à la section précédente, seule la configuration est donnée ainsi que quelques explications, sans relater la démarche nécessaire à l'obtention de cette configuration (nombreux essais du fait de l'ambiguïté des définitions des points extrémités et des conditions directionnelles liées aux templates).

6.7.3.1 Rappels sur l'algorithme

En raison de la complexité de cet algorithme, nous proposons au lecteur de se reporter directement à la section B.4.2, pour son entière description.

6.7.3.2 Tests de la P-simplicité de l'algorithme

Considérons la configuration de la figure 6.31 (a). Les points p et q n'ont qu'un seul 6-voisin (p ou q), les points p_1 et q_1 les empêchent d'appartenir à la classe A ; s'ils peuvent être éliminés, ils ne peuvent alors qu'appartenir à la classe D uniquement. Dans ce cas, ils doivent respecter les conditions (a) à (g) de la définition $2.1(2)$ (cf. [MS96]) ; les points p et q ne peuvent être forcés

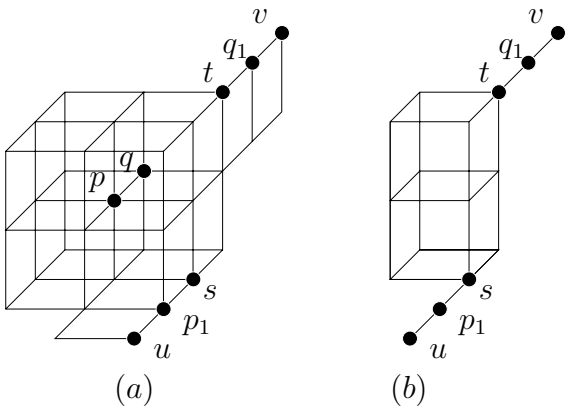


FIG. 6.31 – *Configuration montrant que l'algorithme de Ma et Sonka ne préserve pas la topologie.*

à respecter les conditions (a) à (c) de la définition $2.1(1)$, car ils n'appartiennent pas aux classes A, B ou C. Comme aucun des points $Haut - Sud(x)$, $Bas - Sud(x)$, $Sud - Est(x)$, $Sud - Ouest(x)$, $Haut - Ouest(x)$ et $Bas - Ouest(x)$ n'appartient à X ($\forall x \in \{p, q\}$), les conditions (b) à (g) sont vérifiées. Plus directement, p et q vérifient respectivement les templates D_4 et D_3 de la figure B.32, page 321. De plus, les points p et q sont 26-simples (c'est bien la connexité utilisée – cf. la définition 1.1, page 421 dans l'article) et ne sont pas extrémités. L'algorithme élimine alors ces 2 points, les autres sont préservés, puisqu'ils sont non simples ou terminaux. En effet, les points u et v sont extrémités car ils sont 26-adjacents à exactement un seul point de X, respectivement le point p_1 ou le point q_1. Les points p_1 et q_1 sont conservés ; ils ne vérifient pas les templates – par ailleurs, ils ne sont pas 26-simples ; $T_{26}(x, X) = 2$, $\forall x \in \{p_1, q_1\}$. Les points s et t sont 26-simples et non extrémités, ils ne peuvent vérifier qu'une template de la classe D, mais la condition (f) (resp. (g)) de la définition $2.1(2)$ n'est pas vérifiée pour s (resp. t) ; ou plus directement aucun de ces points ne vérifie de template. Les points s et t ne sont alors pas retirés. L'objet finalement obtenu est représenté à la figure 6.31 (b) ; l'objet initial a été déconnecté en deux ; la topologie n'est alors pas préservée. Dans cette configuration, les points u, v, s et t sont extrémités, et les points q_1 et p_1 ne peuvent être éliminés par les templates ; par ailleurs, ils sont non simples.

Rappel des notations (ce sont celles de la section 6.7.2) : un point p vérifie Ω, s'il vérifie une template à isométrie près. Soient p et q deux points tels que p satisfait Ω. La notation "$q \in \Omega(p)$" signifie que p ne satisfait plus Ω après la suppression de q, et "$q \notin \Omega(p)$" signifie que p satisfait encore Ω après la suppression de q.

En fait, le point p qui est un point simple pour X, ne l'est plus pour $X \setminus \{q\}$, on a $q \in \Omega(p)$. De même $p \in \Omega(q)$. En fait, l'ensemble $\{p, q\}$ de la configuration 6.31 (a) est un ensemble minimal non simple.

Détaillons cela : les points p et q sont simples pour X. Les points p et q satisfont la classe D ; p et q appartiennent alors à Ω. Mais le point p n'est pas simple pour $X \setminus \{q\}$ ($T_{26}(x, X \setminus \{q\}) = 2$) et le point q n'est pas simple pour $X \setminus \{p\}$ ($T_{26}(x, X \setminus \{p\}) = 2$). L'ensemble $\{p, q\}$ est un ensemble non simple, pour lequel chaque sous-ensemble propre est simple ; c'est un MNS. Comme il

n'existe pas de point 6-adjacent à p dans $[X \setminus \{q\}] \cap N^*_{26}(p)$; le point p afin d'être supprimé après la suppression de q, doit vérifier la classe D. Mais la condition (a) de la définition $2.1(2)$ n'est alors pas vérifiée (le point p n'étant pas simple pour $X \setminus \{q\}$) ; nous avons alors $p \in \Omega(q)$. De la même façon, $q \in \Omega(p)$.

Le lemme 3.5, p.430 de [MS96] n'est alors pas valide. Il est identique au lemme 3.2 [Ma95], la démonstration s'adaptant aux templates actuelles.

Examinons la démonstration de l'article. Considérons quatre points alignés sur un morceau de droite : $a-p-q-b$. Il est écrit que si $p \in \Omega(q)$ et $q \in \Omega(p)$ alors les templates impliquent que a et b appartiennent au background et que les points p et q ne peuvent vérifier qu'une template des trois premières classes. Le point q est tel que $Sud(q) \in X(= p)$ par exemple, ce qui implique que $Sud-Sud(q)(= a)$ appartienne à X, selon la définition $2.1(1(c))$ (que l'on peut utiliser car p et q sont 6-adjacents uniquement dans les trois premières classes), ce qui entraine une contradiction. L'erreur se situe ici, car nous avons proposé une configuration pour laquelle les conditions sont vérifiées (deux points 6-adjacents), mais les points p et q n'appartiennent pas aux classes A, B et C, ils appartiennent uniquement à la classe D ; ils n'ont alors pas à vérifier la définition $2.1(1)$.

Remarque : soit P l'ensemble des points pouvant être supprimés par l'algorithme de Ma et Sonka. Détaillons l'appartenance des points de $N_{26}(p)$: $p \in P$ car il vérifie D_4, $s \in R$ car il ne peut vérifier qu'une template de la classe D mais la condition (f) n'est pas satisfaite, $p_1 \in R$ car il ne peut vérifier aucune template, $t \in R$ car il ne peut vérifier qu'une template de la classe D mais la condition (g) n'est pas satisfaite, $q \in P$ car il vérifie D_3. Nous trouvons $T_{26}(p, R) = 2$; la première condition de P-simplicité n'est pas vérifiée et le point p n'est pas P-simple.

Résumons cette étude par la propriété suivante :
Propriété : L'algorithme de Ma et Sonka ne préserve pas la topologie.

6.7.4 Algorithme 3D de Manzanera, et al. (MB_3D) [MBPL99c] [MBPL99a]

6.7.4.1 Rappels sur l'algorithme

Cet algorithme consiste à supprimer jusqu'à stabilité et en parallèle tout point x s'il vérifie α_1, α_2 ou α_3 (à isométries près), sauf si son 18-voisinage contient le motif β_1, ou si son 26-voisinage contient le motif β_2 (Fig. 6.32). Notons que si une configuration vérifie β_1 alors le point x appartient au motif (c'est l'un des points représentés), en revanche si la configuration vérifie β_2, le point x n'est pas forcément l'un des deux points noirs de l'objet, représentés dans le motif β_2.

Pour plus d'informations, se reporter à la description de cet algorithme à la section B.4.3.

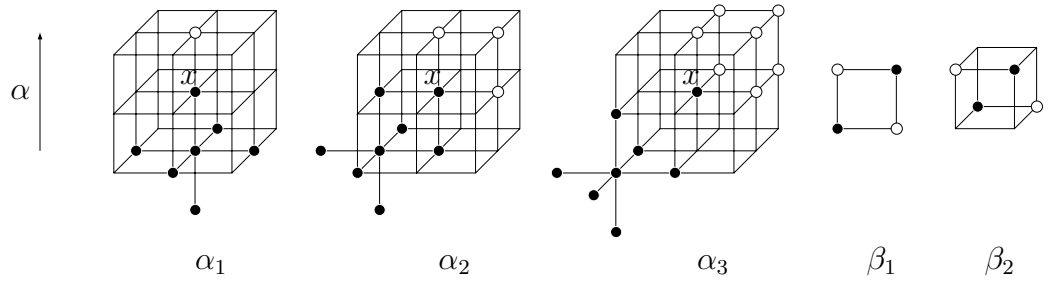


FIG. 6.32 – *Un point x est éliminé s'il vérifie α_1, α_2 ou α_3, sauf si son 18-voisinage contient le motif β_1, ou si son 26-voisinage contient le motif β_2.*

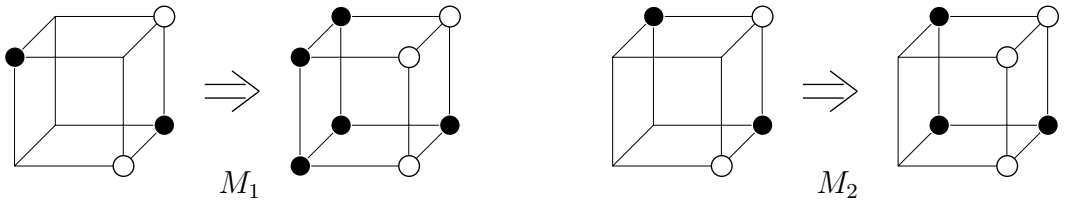


FIG. 6.33 – *Transformations utilisées pour la démonstration (pour les première et troisième conditions de P-simplicité).*

6.7.4.2 P_{26}-simplicité de MB_3D

Propriété : L'algorithme de Manzanera, et al. (MB_3D) est P_{26}-simple.

Preuve :

Soit $MB(X)$, l'ensemble des points x de X vérifiant les conditions de suppressibilité. Pour prouver que l'algorithme MB_3D préserve la topologie, il est suffisant de montrer que :

$$\forall X \subset \mathcal{Z}^3,\ P = MB(X) \text{ est } P_{26}\text{-simple.}$$

Soient $P = MB(X)$, $R = X \setminus P$ et $x \in P$.

Notation :
Remarque 1 (rmq_1) : si un point ne vérifie ni α_1, ni α_2, ni α_3, alors il appartient à R. En particulier, un point intérieur appartient à R.
Le raisonnement suivant s'effectue à isométrie près.

• Conditions (C_1) et (C_3) de P-simplicité :

Nous allons montrer que $\forall y \in N^*_{26}(x) \cap X$, $\exists z \in R$ tel que z est 26-adjacent à x et à y.
• Si x vérifie α_1 (voir Fig. 6.34 (a)).
Remarquons d'abord que $\overline{\alpha}(x) \in R$ car c'est un point intérieur.

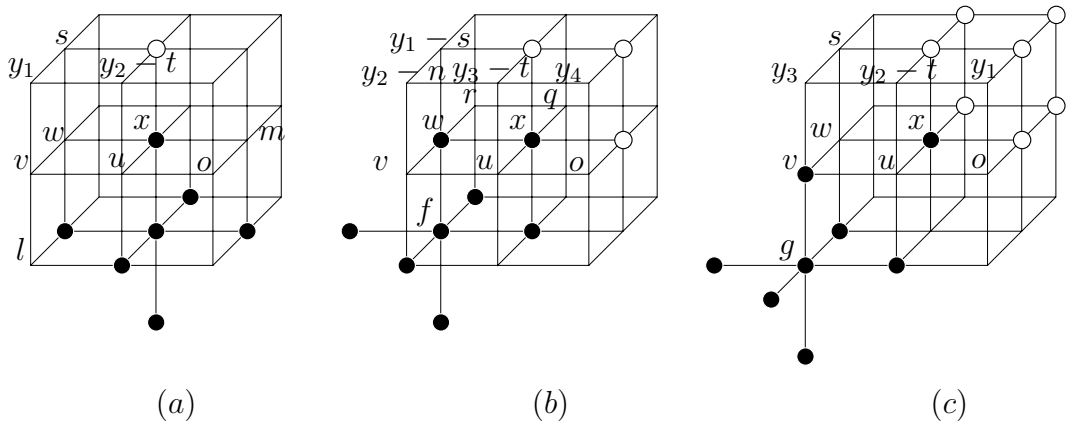


FIG. 6.34 – *Notations : $y_2 - n$, par exemple signifie que le point s'appelle y_2 ou n, double notation utilisée dans la démonstration.*

◇ Si $y_1 \in X \overset{\beta_2}{\Rightarrow} v \in X$,
- ∘ si v vérifie α_1,
  - ⊛ soit u ou w appartient à R (rmq_1),
  - ⊛ soit u ou w appartient à $\overline{X}$, mais pas les deux à la fois (β_1),

par exemple, w appartient à $\overline{X} \overset{M_1}{\Rightarrow} u \in X$, $t \in X$ et $s \in \overline{X} \overset{rmq_1}{\Rightarrow} u \in R$ (Fig. 6.35 (a)) (le choix de u ou de w importe peu, rôle symétrique de u ou de w dans la configuration, selon la direction α)

⊛ soient $l \in \overline{X}$ et $t \in X \overset{\beta_1}{\Rightarrow} u \in X$ et $w \in X$ et si u vérifie α_2 alors $w \in R$ sinon $u \in R$ (rmq_1) (Fig. 6.35 (b)).

∘ si v vérifie α_2,

alors u ou w appartiennent à $\overline{X}$, mais pas les deux à la fois (β_1),

par exemple, w appartient à $\overline{X} \overset{M_1}{\Rightarrow} u \in X$ et $t \in X \overset{rmq_1}{\Rightarrow} u \in R$ (Fig. 6.35 (a)).

(même remarque pour le choix de w)

∘ sinon v ne peut vérifier $\alpha_3 \overset{rmq_1}{\Rightarrow} v \in R$.

Dans ces cas, $\exists e \in \{u, v, w\}$ tel que y_1 est 26-adjacent à $e \in R$ et e est 26-adjacent à $\overline{\alpha}(x) \in R$.

◇ Si $y_2 \in X \overset{\beta_1}{\Rightarrow} u \in X$,
- ∘ si u vérifie α_1, alors o ou $v \in R$ (rmq_1),
- ∘ si u vérifie α_2, alors w ou $m \in R$ (rmq_1),
- ∘ sinon u ne peut vérifier $\alpha_3 \overset{rmq_1}{\Rightarrow} u \in R$.

Dans ces cas, $\exists e \in \{o, m, u, v, w\}$ tel que y_2 est 26-adjacent à $e \in R$ et e est 26-adjacent à $\overline{\alpha}(x) \in R$.

◇ Si $y \in X$ et y n'appartenant pas à une isométrie d'une configuration déjà vue, alors y est 26-adjacent à $\overline{\alpha}(x) \in R$.

• Si x vérifie α_2 (voir Fig. 6.34 (b)).

Remarquons d'abord que $f \in R$ car c'est un point intérieur.

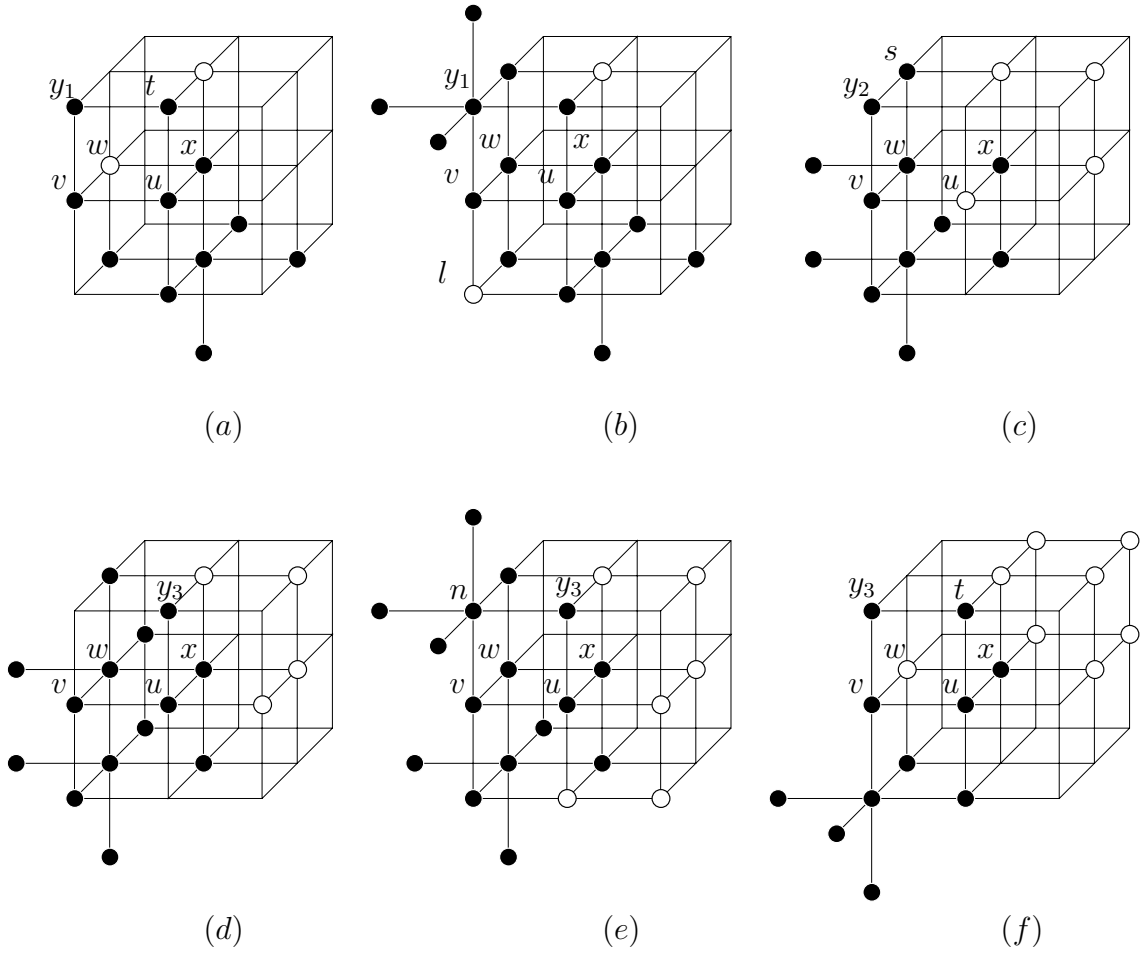


FIG. 6.35 – *Figures utilisées lors de la démonstration de la P-simplicité (conditions (C_1) et (C_3)).*

◇ Si $y_1 \in X$,
- ∘ si w vérifie α_1, alors r ou v appartiennent à R,
- ∘ si w vérifie α_2, alors q ou u appartiennent à R,
- ∘ sinon w ne peut vérifier $\alpha_3 \overset{rmq_1}{\Rightarrow} w \in R$.

Dans ces cas, $\exists e \in \{r, v, q, u\}$ tel que y_1 est 26-adjacent à $e \in R$ et e est 26-adjacent à $f \in R$.

◇ Si $y_2 \in X \overset{\beta_2}{\Rightarrow} v \in X$,
- ∘ si v vérifie α_1,
  - ⊛ soit u ou w appartiennent à R,
  - ⊛ soit $u \in \overline{X} \overset{M_1}{\Rightarrow} s \in X \overset{rmq_1}{\Rightarrow} w \in R$ (Fig. 6.35 (c)),
- ∘ si v vérifie α_2, alors $u \in \overline{X} \overset{M_1}{\Rightarrow} s \in X \overset{rmq_1}{\Rightarrow} w \in R$ (Fig. 6.35 (c)).
- ∘ sinon v ne peut vérifier $\alpha_3 \overset{rmq_1}{\Rightarrow} v \in R$.

Dans ces cas, $\exists e \in \{u, v, w\}$ tel que y_2 est 26-adjacent à $e \in R$ et e est 26-adjacent à $f \in R$.

◇ Si $y_3 \in X \overset{M_2}{\Rightarrow} u \in X \overset{\beta_2}{\Rightarrow} v \in X$.
∘ si u vérifie $\alpha_1 \Rightarrow v \in R$,
∘ si u vérifie α_2,
⊛ soit $w \in R$ (Fig. 6.35 (d)),
⊛ soit $n \in X$, (en fait, n est un point intérieur et appartient à R) (Fig. 6.35 (e)),
⊠ si v vérifie α_1, alors $w \in R$
⊠ sinon v ne peut vérifier ni α_2, ni $\alpha_3 \overset{rmq_1}{\Rightarrow} v \in R$.
∘ sinon u ne peut vérifier $\alpha_3 \overset{rmq_1}{\Rightarrow} u \in R$.
Dans ces cas, $\exists e \in \{u, v, w\}$ tel que y_3 est 26-adjacent à $e \in R$ et e est 26-adjacent à $f \in R$.

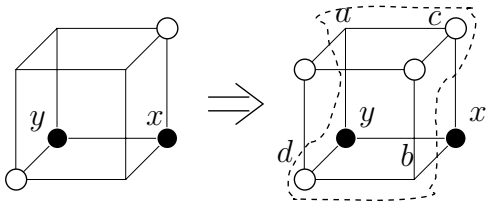
◇ Si $y_4 \in X \overset{M_1}{\Rightarrow} t$, u et o appartiennent à $X \overset{rmq_1}{\Rightarrow} u \in R$, et y_4 est 26-adjacent à $u \in R$ qui est 26-adjacent à $f \in R$.

◇ Si $y \in X$ et y n'appartenant pas à une isométrie d'une configuration déjà vue, alors y est 26-adjacent à $f \in R$.

• Si x vérifie α_3 (voir Fig. 6.34 (c)).
Remarquons d'abord que $g \in R$ car c'est un point intérieur.

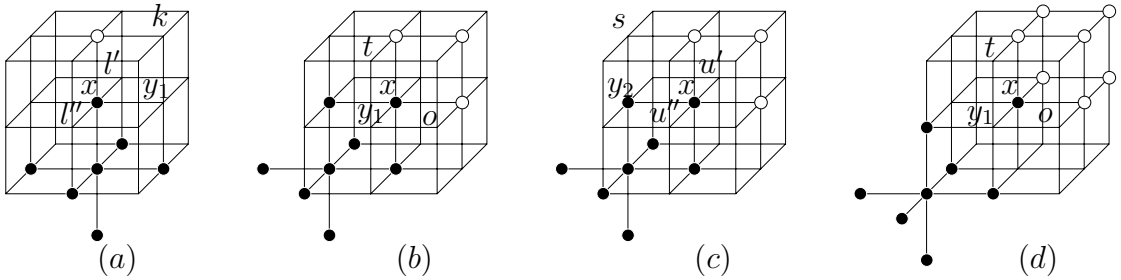
◇ Si $y_1 \in X \overset{M_1}{\Rightarrow} t \in X$, $o \in X$, $u \in X \overset{rmq_1}{\Rightarrow} u \in R$, et y_1 est 26-adjacent à $u \in R$ qui est 26-adjacent à $g \in R$,
◇ Si $y_2 \in X \overset{\beta_1}{\Rightarrow} u \in X$,
∘ si u vérifie α_1, alors $v \in R$,
∘ si u vérifie α_2, alors $w \in R$,
∘ sinon il ne peut vérifier $\alpha_3 \overset{rmq_1}{\Rightarrow} u \in R$.
Dans ces cas, $\exists e \in \{u, v, w\}$ tel que y_2 est 26-adjacent à $e \in R$ et e est 26-adjacent à $g \in R$.
◇ Si $y_3 \in X$
∘ si v vérifie α_1,
⊛ soit u ou w appartient à R,
⊛ soit u ou w appartient à $\overline{X}$, mais pas les deux à la fois (β_1),
par exemple, w appartient à $\overline{X} \overset{M_1}{\Rightarrow} u \in X$, $t \in X \overset{rmq_1}{\Rightarrow} u \in R$ (Fig. 6.35 (f))
(même remarque pour le choix de w),
∘ si v vérifie α_2, alors
u ou w appartient à $\overline{X}$, mais pas les deux à la fois (β_1),
par exemple, w appartient à $\overline{X} \overset{M_1}{\Rightarrow} u \in X$ et $t \in X \overset{rmq_1}{\Rightarrow} u \in R$ (Fig. 6.35 (f))
(même remarque pour le choix de w),
∘ sinon, v ne peut vérifier $\alpha_3 \overset{rmq_1}{\Rightarrow} v \in R$.
Dans ces cas, $\exists e \in \{u, v, w\}$ tel que y_3 est 26-adjacent à $e \in R$ et e est 26-adjacent à $g \in R$.
◇ Si $y \in X$ et y n'appartenant pas à une isométrie déjà vue, alors y est 26-adjacent à un point intérieur (ce dernier appartient à R).

Tout cela implique que les conditions (C_1) et (C_3) sont à la fois vérifiées, car nous n'avons pas imposé au point y d'appartenir explicitement à R ou à P.



M_3

FIG. 6.36 – *Figure utilisée (motif M_3) lors de la démonstration de la P-simplicité (condition (C_4)).*



(a) (b) (c) (d)

FIG. 6.37 – *Figures utilisées lors de la démonstration de la P-simplicité (condition (C_4)).*

• Condition (C_2) de P-simplicité.
Si x vérifie α_1, avec $\alpha(x) \in \overline{X}$ (voisin de x selon la direction α, imposé par le motif α_1), alors $\forall y \in N_6^*(x) \cap \overline{X}$, avec $y \neq \alpha(x)$, le point y appartient à $N_\alpha^6(x)$ (car $\overline{\alpha}(x) \in R$, car c'est un point intérieur). Or $\alpha(y) \in \overline{X}$, sinon x vérifie β_1 ; puisque $\alpha(y)$ est 6-adjacent à $\alpha(x)$, alors (C_2) est vérifiée.

Le raisonnement est le même, si x vérifie α_2 ou α_3, en considérant les points appartenant à $\overline{X}$, imposés par le motif α_2 ou α_3.

• Condition (C_4) de P-simplicité.

Considérons d'abord la figure 6.36. Le motif β_2 et les points présents dans la configuration de gauche (appelé motif M_3 dans la suite) entrainent l'ajout de nouveaux points (configuration de droite). De plus, au moins l'un des points a ou b appartient à $\overline{X}$, afin d'éviter le motif β_2 (délimité en pointillé par les points a, c, d, b). Il existe donc un carré unité $\{x, y, a, c\}$ ou $\{x, y, d, b\}$, dans un tel octant, qui vérifie (C_4).

◇ Si x vérifie α_1 (Fig. 6.37 (a)),
  ○ si $y_1 \in P$ (seule position à étudier pour $y \in N_6^*(x) \cap P$, à isométries près),
    ⊛ si y_1 vérifie α_1 ou α_2, soit $k \in \overline{X}$, soit nous avons le motif M_3, et k, l' ou l'' appartiennent à $\overline{X}$,

⊛ si y_1 vérifie $\alpha_3 \Rightarrow k \in \overline{X}$.

◇ Si x vérifie α_2 (les deux seules positions pour $y \in N_6^*(x) \cap P$ à isométries près sont y_1 (Fig. 6.37 (b)) ou y_2 (Fig. 6.37 (c))),
- ○ si $y_1 \in P$ (Fig. 6.37 (b)),
  - ⊛ si y_1 vérifie α_1 ou α_2 alors o ou t appartiennent à $\overline{X}$,
  - ⊛ si y_1 vérifie α_3 alors o et t appartiennent à $\overline{X}$,
- ○ si $y_2 \in P$ (Fig. 6.37 (c)),
  - ⊛ si y_2 vérifie α_1 ou α_2 alors soit $s \in \overline{X}$, soit nous avons le motif M_3, et s, u' ou u'' appartiennent à $\overline{X}$,
  - ⊛ si y_2 vérifie α_3 alors $s \in \overline{X}$.

◇ Si x vérifie α_3 (Fig. 6.37 (d)),
- ○ si $y_1 \in P$ (seule position à étudier pour $y \in N_6^*(x) \cap P$, à isométries près),
  - ⊛ si y_1 vérifie α_1 ou α_2 alors o ou t appartiennent à $\overline{X}$,
  - ⊛ si y_1 vérifie α_3 alors o et t appartiennent à $\overline{X}$.

Dans tous ces cas, (C_4) est vérifiée. □

6.8 À propos d'une démarche systématique avec les points P^x-simples

Parmi les six algorithmes que nous avons testés précédemment, trois utilisaient des templates ou des conditions dans $N_{26}^*(x)$; le seul examen de ce voisinage était suffisant pour décider de la suppressibilité d'un point. Nous avons prouvé "manuellement" que ces trois algorithmes étaient P-simples. Puis en proposant des versions plus puissantes, nous avons montré qu'ils étaient P^x-simples, par conséquent la preuve de la préservation de la topologie est effectuée "par" simulation par ordinateur (encore faut-il actuellement rechercher "manuellement" un ensemble P adéquat). Remarquons qu'en posant précisément P comme étant l'ensemble des points candidats à la suppression, nous pouvons tester entièrement par ordinateur si l'algorithme est P^x-simple. Si c'est le cas alors l'algorithme préserve la topologie. Sinon, nous ne pouvons conclure directement.

Il semblerait (nous essayons actuellement de le démontrer) que tout algorithme ne supprimant pas de MNS (cf. section 5.5) et décidant de la suppressibilité de tout point x par le seul examen de $N_{26}^*(x)$ (resp. d'un voisinage V) soit P^x-simple (soit $P(V,x)$-simple, $P(V,x)$ étant une généralisation de l'ensemble P^x au voisinage V). Si cela est prouvé, alors nous pourrons tester si un algorithme préserve la topologie, par simulation par ordinateur, évitant ainsi les preuves combinatoires difficiles proposées jusqu'ici (cf. les sections 6.5.1, 6.5.2, 6.6.2.1 et 6.7.4.2) ; simulation consistant à tester si tous les points pouvant être supprimés par un algorithme donné sont $P(V,x)$-simples (à condition que la taille de V rende la simulation réalisable par ordinateur).

6.9 Conclusion

Dans ce chapitre, avec la seule notion de point P-simple, nous avons montré que les algorithmes de Palágyi et Kuba en 6 et 12 sous-itérations et que celui de Manzanera, et al. préservaient la topologie, que celui de Ma et celui de Ma et Sonka ne préservaient pas la topologie. Par l'introduction d'un nouvel ensemble P^x défini à partir d'un ensemble P, nous avons proposé un algorithme en 6 sous-itérations supprimant à la fois les points retirés par l'algorithme de Gong et Bertrand et les points retirés par celui de Palágyi et Kuba, tout en préservant les mêmes points terminaux, et un algorithme en 12 sous-itérations dans une version curviligne ou surfacique et supprimant au moins les points retirés par l'algorithme de Palágyi et Kuba, tout en préservant les mêmes points terminaux. Cela implique également que ces algorithmes préservent la topologie, cela peut alors être considéré comme une preuve partiellement automatisée.

Nous collaborons actuellement avec le Professeur C.M. Ma afin de proposer de nouveaux algorithmes fortement parallèles préservant la topologie, en adoptant l'approche utilisant les points P^x-simples. Avec cette même technique, une autre étude peut être réalisée afin de proposer un algorithme supprimant plus de points que celui de Manzanera, et al., ou utilisant un support moindre et supprimant les mêmes points [LB02c].

Troisième partie

Topologie discrète, et algorithmes parallèles de squelettisation pour images 2D et 3D binaires et images 2D en niveaux de gris

Nous avons étudié des algorithmes de squelettisation dans les deux premières parties de ce rapport (voir également les annexes A et B) selon l'approche topologie digitale. Nous avons passé en revue différentes classes existantes pour de tels algorithmes : les algorithmes séquentiels (produisant des squelettes mal centrés, un tel processus étant plus long qu'un processus parallèle), les algorithmes parallèles pour lesquels un sous-ensemble de points simples est retiré lors d'une (sous-)itération donnée. Ce processus de suppression pouvant être réalisé selon les différentes façons suivantes : une approche directionnelle (le processus n'est pas homogène, le résultat dépend de l'ordre des directions et peut être mal centré), une approche sous-mailles (produisant des effets de dentelles), une approche fortement parallèle. Pour ces derniers, nous avons vu précédemment que deux d'entre eux ne préservaient pas la topologie. Nous avons montré que l'algorithme de Manzanera et Bernard préserve la topologie. Celui-là se déroule de façon homogène mais nécessite un support étendu afin de décider de la suppressibilité d'un point ; les squelettes obtenus sont bien centrés mais peuvent ne pas être minces. La version corrigée des deux autres algorithmes utilisera également des considérations directionnelles, ce qui aura pour conséquence des squelettes non fins ou mal centrés.

Récemment, G. Bertrand a introduit le concept de point α_n- ou β_n-simple [Ber99]. Cela nous a permis de concevoir aisément des schémas simples et génériques d'amincissement ou de squelettisation. En adaptant la condition pour un point d'être terminal pour un type donné de squelette, ces schémas sont valables dans le cas d'images 2D binaires ou en niveaux de gris ou 3D binaires. En plus d'avoir des schémas simples de squelettisation, nous avons la propriété remarquable de pouvoir supprimer en parallèle des points α_n-simples, puis de pouvoir supprimer en parallèle des points β_n-simples, tout en préservant la topologie. En outre, le processus basé sur la suppression de tels points est indépendant des directions, ce qui a pour conséquence des squelettes bien centrés. Nous avons également la propriété de pouvoir décider de la suppressibilité d'un point en examinant un support d'au plus 26 points. Nous avons codé les configurations des points α_3-simples (pour images 3D) dans des BDD, obtenant ainsi une caractérisation rapide de tels points, et par conséquent, un algorithme efficace. Il est à noter également un effort important afin de proposer une nouvelle approche du concept de point terminal de surface avec la notion d'arbre de la théorie des graphes, ce qui nous a permis d'obtenir des squelettes surfaciques satisfaisants. En contrepartie, toutes ses propriétés sont obtenues par le passage d'une image dans un ordre qui nécessite une multiplication de la taille de l'image ; mais, cela n'implique pas généralement un nombre d'itérations, afin d'obtenir un squelette dans un ordre, plus important que celui requis par des algorithmes classiques de la topologie digitale.

Retenons que, dans le cas de la topologie digitale, nous avons examiné de nombreux algorithmes. Il nous a semblé intéressant de proposer de nouveaux algorithmes utilisant les points P-simples car nous avions une description précise des points à éliminer, contrairement aux algorithmes utilisant des templates, dont l'obtention ne nous est généralement pas dévoilée. Mais, nos algorithmes souffraient des mêmes limitations, à savoir que le processus utilisait des directions ou un support plus grand et que les définitions des points terminaux de surface n'étaient pas satisfaisantes dans tous les cas de figure. Du fait des propriétés des ordres, il semble qu'il s'agisse d'un cadre très prometteur en ce qui concerne la squelettisation parallèle ; il semble également plus facile de paralléliser d'autres opérateurs, tels que le filtrage de squelette, la ligne de partage des eaux, etc.

Chapitre 7

Généralités

Après avoir situé la nouvelle approche utilisant les ordres par rapport aux autres existantes étudiant des propriétés topologiques des images, nous définirons quelques notions de base dans les ordres. Seront vues notamment les notions de point α-unipolaire et d'α-noyau. Cette dernière notion conduit directement à la conception d'un algorithme de suppression parallèle de points. Nous donnerons ensuite un ordre associé à $\mathcal{Z}^n$, qui lorsqu'il sera dérivé pour $\mathcal{Z}^2$ ou $\mathcal{Z}^3$ nous permettra de traiter les images 2D (chap. 8) ou 3D (chap. 9). Nous introduisons ensuite les notions de point α_n-simple et de point β_n-simple ; notions provenant de l'étude de l'homotopie et de la simplicité dans les ordres, proposées récemment dans [Ber99]. De telles définitions sont récursives croisées avec celle d'ensemble n-contractible. Elles conduisent directement à un processus parallèle de suppression de points. Ces notions seront étudiées en détail dans le cas 2D (chap. 8) ou 3D (chap. 9).

7.1 Introduction [Ber99]

Différentes approches ont été proposées pour l'étude des propriétés topologiques des images digitales binaires :

- l'approche topologie digitale introduite par Rosenfeld [KR89]. Les éléments de $\mathcal{Z}^n$ sont liés par des relations d'adjacence qui permettent de définir la connexité. Il n'y a pas de voie directe afin de construire une topologie qui corresponde à cette notion de connexité (voir [Ber99]). C'est cette approche qui a été décrite et utilisée dans les deux premières parties de ce rapport.
- l'approche espace topologique connexe ordonné – ou *connected ordered topological space* – introduite par Khalimsky [Kha60] [KKM90] [KKM91] [Kop94]. Le plus petit voisinage de chaque point de $\mathcal{Z}^n$ diffère d'un point à un autre. Cette approche permet de retrouver la structure d'une topologie.
- l'approche complexe cellulaire. Un objet est vu comme une structure constituée d'éléments de différentes dimensions, appelés cellules. Il est possible de retrouver la structure d'une topologie [Kov89].

La topologie utilisée dans les deux dernières approches est une *topologie d'Alexandroff* [Ale37], appelée également *topologie discrète*. Dans une telle topologie, toute intersection d'ouverts (*c.-à-d.* intersection finie ou non) est un ouvert. Il existe un lien entre les topologies d'Alexandroff et les ensembles partiellement ordonnés ou ordres – ou *partially ordered sets* (*posets*, *orders*). Un *ordre* est un ensemble sur lequel une relation réflexive, antisymétrique et transitive est définie. Il est possible de définir un ordre sur les points d'un espace topologique T_0-séparable et, réciproquement, un ordre détermine une topologie d'Alexandroff (voir [Ber99] pour plus de détails).

7.2 Notions de base

Nous présentons maintenant quelques notions de topologie discrète, proposées dans [Ber99].

Soit X un ensemble. $\mathcal{P}(X)$ désigne l'ensemble composé de tous les sous-ensembles de X. Si S est un sous-ensemble de X, $\overline{S}$ désigne le complément de S dans X. Si S est un sous-ensemble de T, on écrit $S \subseteq T$. La notation $S \subset T$ signifie $S \subseteq T$ et $S \neq T$. Si γ est une application de $\mathcal{P}(X)$ dans $\mathcal{P}(X)$, le *dual* de γ est l'application $*\gamma$ de $\mathcal{P}(X)$ dans $\mathcal{P}(X)$ telle que $\forall S \subseteq X$, $*\gamma(S) = \overline{\gamma(\overline{S})}$.

Soit δ une relation binaire sur X, *c.-à-d.* un sous-ensemble de $X \times X$. L'*inverse* de δ est la relation binaire $\{(x, y) \in X \times X;\ (y, x) \in \delta\}$. Nous désignons aussi par δ l'application de X dans $\mathcal{P}(X)$ telle que, pour tout $x \in X$, $\delta(x) = \{y \in X;\ (x, y) \in \delta\}$. Nous définissons la relation binaire $\delta^{\square} = \delta \setminus \{(x, x);\ x \in X\}$.

Soient x_0 et x_k, deux éléments de X. Un *δ-chemin* de x_0 à x_k est une séquence x_0, x_1, ..., x_k d'éléments de X telle que $x_i \in \delta(x_{i-1})$, avec $i \in \{1, \ldots, k\}$; le nombre k est la *longueur* du chemin et x_0 est son *origine*.

7.2.1 Ordres et topologie discrète

Un *ordre* $|X|$ est un couple (X, α) dans lequel X est un ensemble et α est une *relation d'ordre* sur X, *c.-à-d.* une relation binaire réflexive, antisymétrique et transitive sur X. Un élément de X est aussi appelé un *point*. L'ensemble $\alpha(x)$ est appelé l'*α-adhérence* de x, si $y \in \alpha(x)$; on dit que y est *α-adhérent* à x.

Nous notons aussi α l'application de $\mathcal{P}(X)$ dans $\mathcal{P}(X)$ telle que pour tout sous-ensemble S de X, $\alpha(S) = \cup\{\alpha(x);\ x \in S\}$, $\alpha(S)$ est appelé l'*α-fermeture de S*, $*\alpha(S)$ est appelé l'*α-intérieur de S*.

Un sous-ensemble S de X est *α-fermé* si $S = \alpha(S)$, S est *α-ouvert* si $S = *\alpha(S)$. Soit la relation $\beta = \{(x, y);\ (y, x) \in \alpha\}$, β est l'*inverse* de la relation α. Parfois, afin de rappeler cette notation, nous appellerons également un ordre tout triplet (X, α, β) dans lequel (X, α) est un ordre comme défini ci-avant, et β est l'inverse de α. Le *dual de l'ordre* (X, α, β) est l'ordre (X, β, α).

Notons que $*\alpha(S) = \{x; \beta(x) \subseteq S\}$.

Preuve : Remarquons d'abord que $\beta(x) = \{y \in X; (y, x) \in \alpha\} = \{y \in X; x \in \alpha(y)\}$ ($*_0$). Puis, $x \in *\alpha(S) \Leftrightarrow x \in \overline{\alpha(\overline{S})} \Leftrightarrow x \notin \alpha(\overline{S}) \Leftrightarrow x \notin \cup\{\alpha(y), y \in \overline{S}\} \Leftrightarrow \nexists y \in \overline{S}$ tel que $x \in$

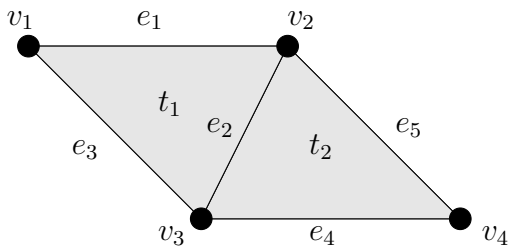


FIG. 7.1 – *Figure d'illustration de quelques notions de base dans les ordres.*

$\alpha(y) \Leftrightarrow \{y \in \overline{S}; x \in \alpha(y)\} = \emptyset \Leftrightarrow \{y \in X; x \in \alpha(y)\} \subseteq S \overset{*0}{\Leftrightarrow} \beta(x) \subseteq S$. □

Le même raisonnement tient pour $*\beta(S) = \{x; \alpha(x) \subseteq S\}$.

La famille $\mathcal{O}_\alpha$ composée de tous les sous-ensembles α-ouverts de X satisfait les conditions d'une famille de sous-ensembles ouverts d'une topologie ; cette remarque est également valable pour la famille $\mathcal{O}_\beta$ composée de tous les sous-ensembles β-ouverts de X. Notons respectivement $\mathcal{T}_\alpha = (X, \mathcal{O}_\alpha)$ et $\mathcal{T}_\beta = (X, \mathcal{O}_\beta)$ ces deux topologies. Ces topologies sont en fait des *topologies discrètes*, ou *topologies d'Alexandroff*, *c.-à-d.* des topologies telles que toute intersection d'ensembles ouverts est un ouvert [Ale37] (voir également le théorème de spécialisation d'Alexandroff [Ron98]). Les sous-ensembles fermés pour $\mathcal{T}_\alpha$ (resp. $\mathcal{T}_\beta$) sont les sous-ensembles α-fermés (resp. β-fermés) comme définis ci-avant. L'α-adhérence (resp. β-adhérence) d'un point x est le plus petit voisinage pour la topologie $\mathcal{T}_\beta$ (resp. $\mathcal{T}_\alpha$). Soient (X, α) un ordre et S un sous-ensemble de X, l'*ordre relatif à* S est l'ordre $|S| = (S, \alpha \cap (S \times S))$, nous notons $\alpha_S = \alpha \cap (S \times S)$. Dans la suite de l'article, nous notons $\theta = \alpha \cup \beta$.

Un ordre (X, α) est *dénombrable* si X est dénombrable. L'ordre est *localement fini* si $\forall x \in X$, $\theta(x)$ est un ensemble fini. Dans la suite, les ordres considérés sont dénombrables et localement finis.

Soit $|X|$ un ordre. Un *chemin* dans $|X|$ est un θ-chemin. Nous pouvons voir qu'un ordre (X, α, β) est connexe pour $\mathcal{T}_\alpha$ (ou pour $\mathcal{T}_\beta$) si et seulement si il est connexe par chemin, *i.e.* si pour tout x et y dans X, il existe un chemin de x à y [Ber99]. Un élément x de $|X|$ est un *point α-terminal* si $\alpha^{\square}(x) = \emptyset$.

7.2.2 Illustration de quelques notions générales [CBK02]

Nous illustrons quelques notions précédentes à la figure 7.1. Dans cet exemple, nous avons un ensemble X composé de deux triangles t_1 et t_2, de cinq arêtes e_1, e_2, e_3, e_4 et e_5 et de quatre points v_1, v_2, v_3 et v_4. Nous définissons la relation α par : $\alpha(t_1) = \{t_1, e_1, e_2, e_3, v_1, v_2, v_3\}$, $\alpha(t_2) = \{t_2, e_2, e_4, e_5, v_2, v_3, v_4\}$, $\alpha(e_1) = \{e_1, v_1, v_2\}$, $\alpha(e_2) = \{e_2, v_2, v_3\}$, $\alpha(e_3) = \{e_3, v_1, v_3\}$, $\alpha(e_4) = \{e_4, v_3, v_4\}$, $\alpha(e_5) = \{e_5, v_2, v_4\}$, et $\forall i \in \{1, \ldots, 4\}$, $\alpha(v_i) = \{v_i\}$. Considérons alors l'ordre (X, α).

Soit $S_1 = \{t_1, e_1, e_5, v_2\}$. Nous avons $\alpha(S_1) = \{t_1, e_1, e_2, e_3, e_5, v_1, v_2, v_3, v_4\} \neq S_1$, ainsi S_1 n'est pas un α-fermé. On a $\alpha(\overline{S_1}) = \cup\{\alpha(x), x \in \overline{S_1}\} = \{t_2, e_2, e_3, e_4, e_5, v_1, v_2, v_3, v_4\}$ et $\overline{\alpha(\overline{S_1})} = \{t_1, e_1\} \neq S_1$, ainsi S_1 n'est pas un α-ouvert.

Posons $S_2 = \alpha(t_1), S_3 = \{e_2, e_5, v_2, v_3, v_4\}$ et $S_4 = \{v_1\}$. Puisque ces sous-ensembles sont tels que $\alpha(S_i) = S_i$, pour $i \in \{2, 3, 4\}$, alors ils sont α-fermés. Posons $S_5 = \{t_1\}, S_6 = \{t_1, t_2, e_2\}$. Puisque ces sous-ensembles sont tels que $\overline{\alpha(\overline{S_i})} = S_i$, pour $i \in \{5, 6\}$, alors ils sont α-ouverts.

Avec $\beta(x) = \{y \in X; (y, x) \in \alpha\}$ et $\theta = \alpha \cup \beta$; nous avons $\beta(v_2) = \theta(v_2) = \{t_1, t_2, e_1, e_2, e_5, v_2\}$; $\beta(e_2) = \{t_1, t_2, e_2\}$ et puisque $\alpha(e_2) = \{e_2, v_2, v_3\}$ alors $\theta(e_2) = \{t_1, t_2, e_2, v_2, v_3\}$; $\beta(t_1) = \{t_1\}$ et puisque $\alpha(t_1) = \{t_1, e_1, e_2, e_3, v_1, v_2, v_3\}$ alors $\theta(t_1) = \{t_1, e_1, e_2, e_3, v_1, v_2, v_3\}$.

7.3 Définitions : point α-unipolaire, α-noyau

Nous introduisons la notion de point α-unipolaire (un point α-unipolaire peut être vu comme un “point inessentiel pour la topologie”), puis la notion d’α-noyau.

Définitions (point α-unipolaire, point α-libre, α-noyau) : Soient $|X| = (X, \alpha)$ un ordre, x et y deux points de X. Nous définissons $\alpha^\bullet$, la relation sur X telle que $y \in \alpha^\bullet(x)$ si et seulement si $y \in \alpha^\square(x)$ et $\alpha^\square(x) \cap \beta^\square(y) = \emptyset$. Un point x de X est *α-unipolaire* si $\alpha^\bullet(x)$ consiste en un seul point. Un élément x est *α-libre* si x est α-unipolaire ou s’il existe une séquence $x_0, \ldots, x_k$, avec $x_k = x$ telle que x_0 est α-unipolaire et x_i est α_i-unipolaire avec $\alpha_i = \alpha \cap (S_i \times S_i),\ S_i = X \setminus \{x_0, \ldots, x_{i-1}\}$, avec $i \in \{1, \ldots, k\}$. Un point qui n’est pas α-libre est appelé un *α-lien*. L’*α-noyau* de X est le sous-ensemble de X composé de tous les α-liens de X.

Propriétés : Soient $|X| = (X, \alpha)$ un ordre, f un point α-unipolaire et x un point de X avec $x \neq f$. Si x est α-unipolaire, alors x est α_S-unipolaire pour $S = X \setminus \{f\}$. Le point x est α-libre si et seulement si x est α_S-libre pour $S = X \setminus \{f\}$.
Preuve (de la première propriété, cf. [Ber99]) :
Soit x, un point α-unipolaire. Notons x' (resp. f') l’unique point de $\alpha^\bullet(x)$ (resp. $\alpha^\bullet(f)$).

- Supposons $f \neq x'$. Alors $x' \in \alpha_S^\bullet(x)$. Supposons qu’il existe un point y, avec $y \neq x'$ et $y \in \alpha_S^\bullet(x)$ (dans ce cas, x ne serait pas α_S-unipolaire, puisque $x' \in \alpha_S^\bullet(x)$, et nous avons x non α_S-unipolaire s’il existe un point $y \neq x'$ tel que $y \in \alpha_S^\bullet(x)$) ; et puisque $y \in \alpha_S^\bullet(x)$, nous avons $\alpha_S^\square(x) \cap \beta_S^\square(y) = \emptyset$.
  Or $y \in \alpha_S^\bullet(x) \Leftrightarrow y \in \alpha_S^\square(x)$ et $\alpha_S^\square(x) \cap \beta_S^\square(y) = \emptyset \Rightarrow y \in \alpha^\square(x)$ et si $\alpha^\square(x) \cap \beta^\square(y) = \emptyset$ alors $y \in \alpha^\bullet(x)$ (or x' est l’unique point de $\alpha^\bullet(x)$ et $x' \neq y$). Nous devons avoir $\alpha^\square(x) \cap \beta^\square(y) \neq \emptyset$, or $\alpha_S^\square(x) \cap \beta_S^\square(y) = \emptyset$, cela implique que $\alpha^\square(x) \cap \beta^\square(y) = \{f\}$; ainsi $f \in \alpha^\square(x)$ et $\alpha^\square(x) \cap \beta^\square(f) = \emptyset$: f devrait appartenir à $\alpha^\bullet(x)$, ce qui est impossible puisque $\alpha^\bullet(x) = \{x'\}$ et $f \neq x'$. Ainsi x est α_S-unipolaire,
- Supposons $f = x'$. Alors $\alpha_S^\bullet(x) = \{f'\}$ et x est α_S-unipolaire.

□

Cette première propriété peut être reformulée de la façon suivante : soit $|X| = (X, \alpha)$ un ordre. Si $f \in X$ est α-unipolaire, alors l’α_S-noyau pour $S = X \setminus \{f\}$ est précisément l’α-noyau. Cela indique une façon d’obtenir l’α-noyau : nous choisissons arbitrairement un point α-unipolaire et nous le supprimons ; nous répétons cette opération jusqu’à idempotence.

Remarque : Rappelons que dans le cadre de la topologie digitale, si deux points x et y de X sont simples pour X, avec $X \subseteq \mathcal{Z}^n$ et $n \in \{2, 3\}$, alors cela n'implique pas forcément que x soit simple pour $X \setminus \{y\}$, ou que y soit simple pour $X \setminus \{x\}$ (cf. MNS, sections 2.6.2 et 3.8).

7.4 Ordres associés à $\mathcal{Z}^n$

Nous proposons maintenant une présentation de quelques ordres qui peuvent être associés à $\mathcal{Z}^n$, ces ordres sont équivalents à ceux obtenus dans l'étude des espaces topologiques ordonnés connexes introduits par Khalimsky [Kha60] (voir également [KKM90] [KKM91] [Kop94]).

Soient E un ensemble et E^n le produit cartésien de n copies de E. Un élément a de E^n peut être vu comme une application de $\{1, \ldots, n\}$ dans E, $a(i)$ est la i-ième coordonnée de a, avec $i \in \{1, \ldots, n\}$. Si S est un sous-ensemble de E^n, la i-ième projection de S est l'ensemble $S(i) = \{a(i); a \in S\}$, avec $i \in \{1, \ldots, n\}$.

Soit $\mathcal{Z}$, l'ensemble des entiers relatifs. Nous considérons les familles d'ensembles H^1_0, H^1_1 et H^1, telles que $H^1_0 = \{\{a\};\ a \in \mathcal{Z}\}$, $H^1_1 = \{\{a, a+1\}; a \in \mathcal{Z}\}$ et $H^1 = H^1_0 \cup H^1_1$.

Un sous-ensemble S de $\mathcal{Z}^n$ qui est le produit cartésien d'exactement m éléments de H^1_1 et de $(n-m)$ éléments de H^1_0 est appelé un *m-cube* de $\mathcal{Z}^n$. Soit H^n l'ensemble composé de tous les m-cubes de $\mathcal{Z}^n$, $m \in \{0, \ldots, n\}$. Un m-cube de $\mathcal{Z}^n$ est appelé un *singleton*, un *intervalle unité*, un *carré unité* ou un *cube unité* pour $m = 0, 1, 2$ ou 3 respectivement.

Notons $*\mathcal{Z} = \{a + 1/2;\ a \in \mathcal{Z}\}$ et $\mathcal{Z}_+ = \mathcal{Z} \cup *\mathcal{Z}$. Soit l'application ψ_1 de H^1 dans $\mathcal{Z}_+$ telle que pour tout x de H^1, $\psi_1(x) = a$ si $x = \{a\}$, $a \in \mathcal{Z}$ et $\psi_1(x) = a+1/2$ si $x = \{a, a+1\}$, $a \in \mathcal{Z}$. Considérons également l'application ψ_n de H^n dans $\mathcal{Z}^n_+$ telle que pour tout x de H^n, la i-ième coordonnée de $\psi_n(x)$ est l'image de la i-ième coordonnée de x par ψ_1, *i.e.* $[\psi_n(x)](i) = \psi_1[x(i)]$, avec $i \in \{1, \ldots, n\}$. Si $S \subseteq H^n$, alors $\psi_n(S) = \{\psi_n(x), x \in S\}$ peut être vu comme une représentation de S, appelée *représentation tableau* [Ber99].

Considérons comme ordre de base associé à $\mathcal{Z}^n$, l'ordre (H^n, α) avec $\alpha = \supseteq$, ainsi $y \in \alpha(x)$ si $x \supseteq y$; rappelons que la relation $\supseteq$ est une relation d'ordre partiel.

Notons que la relation α utilisée dans la section 7.2.2 pour illustrer les concepts d'α-ouvert et d'α-fermé est la relation $\supseteq$. Les ordres de base associés à $\mathcal{Z}^2$ et à $\mathcal{Z}^3$, ainsi que différentes façons de considérer des objets 2D et 3D dans ces ordres seront détaillés respectivement aux sections 8.2 et 9.2.

7.5 Simplicité

Nous introduisons maintenant les définitions de point α_n-simple et de point β_n-simple, définitions récursives et croisées avec celle d'ordre n-contractible. Elles sont à la base des algorithmes de squelettisation que nous proposons. Nous avons illustré que la seule notion de point unipolaire ne suffisait pas à obtenir des algorithmes efficaces d'amincissement dans le cas d'images 2D (section 8.4.2) ou 3D (section 9.4). C'est la raison pour laquelle la notion de point α_n-simple

a été introduite.

Définitions (point α_n-simple, point β_n-simple, ordre n-contractile) : Soit $|X| = (X, \alpha, \beta)$ un ordre dénombrable, localement fini et non vide. L'ordre $|X|$ est 0-*contractile* s'il est composé d'un unique point. Un point x est α_n-*simple* pour $|X|$ si $|\alpha^{\square}(x)|$ est $(n-1)$-contractile, $n \geq 1$. L'ensemble composé de tous les points de X qui sont α_n-simples pour $|X|$ est noté par X_{α_n}. L'ordre $|X|$ est n-*contractile*, $n > 0$, s'il existe une séquence $X^0, \ldots, X^k$, avec $X^0 = X$, $X^i = X^{i-1} \setminus X^{i-1}_{\alpha_n}$ si i est impair, $X^i = X^{i-1} \setminus X^{i-1}_{\beta_n}$ si i est pair, pour $i \in \{1, \ldots, k\}$ et tel que $X^k = \{a\}, a \in X$.

En fait, la notion de point α_2-simple (resp. α_3-simple) est appropriée dans le cas de la squelettisation 2D (resp. 3D). Nous avons d'abord proposé une caractérisation efficace des points α_2-simples (cf. chapitre 8). Cette caractérisation a été utilisée afin de caractériser plus rapidement un point α_3-simple (selon la définition proposée ici). Puis nous avons créé un BDD stockant toutes les configurations correspondant à un point α_3-simple. Nous avons ainsi une caractérisation efficace des points α_3-simples. Celle-là n'a besoin d'examiner uniquement que la présence de points dans $\alpha^{\square}(x)$ pour déterminer si x est α_3-simple ou non (voir le chapitre 9 et l'annexe C).

7.6 Conclusion

Il est à noter que des liens ont été établis entre les points simples dans $\mathcal{Z}^3$ et les points α_3-simples pour les ordres obtenus à partir d'α-noyaux de sous-ensembles conçus à partir de transformations "Hit or Miss" (voir [BC99]).

Des notions de surface ont été définies dans le cadre des ordres et sont comparées par rapport à celles proposées dans le cadre de la topologie digitale [CBK02]. Nous avons également défini la notion d'ordre valué afin de traiter des images en niveaux de gris [LB00a], et proposer des processus d'abaissement de points en bénéficiant de l'approche parallèle (cf. chapitre 10). Selon cette approche topologie discrètre, des travaux ont également été réalisés, par exemple : l'obtention de lignes de partage des eaux [CB00], le filtrage parallèle de squelettes (cf. chapitre 11), une étude de discrétisation [CBK02], la segmentation du cortex cérébral à partir de données IRM [DC02].

Chapitre 8

Ordre pour images binaires 2D

8.1 Introduction

Nous présentons ici l'étude d'algorithmes de squelettisation dans l'ordre de base associé à $\mathcal{Z}^2$ (voir également [LB00b]).

8.2 Ordre associé à $\mathcal{Z}^2$

Dans cette section, nous définissons l'ordre $(H^2, \supseteq)$ associé à $\mathcal{Z}^2$ et nous proposons différentes façons de considérer des objets binaires 2D dans cet ordre.

8.2.1 L'ordre $|H^2|$

Soit $\mathcal{Z}$ l'ensemble des entiers relatifs. Nous considérons les familles d'ensembles H_0^2, $H_{1_h}^2$, $H_{1_v}^2$, H_2^2 et H^2, telles que $H_0^2 = \{\{(i,j)\}, i,j \in \mathcal{Z}\}$, $H_{1_h}^2 = \{\{(i,j),(i+1,j)\}, i,j \in \mathcal{Z}\}$,

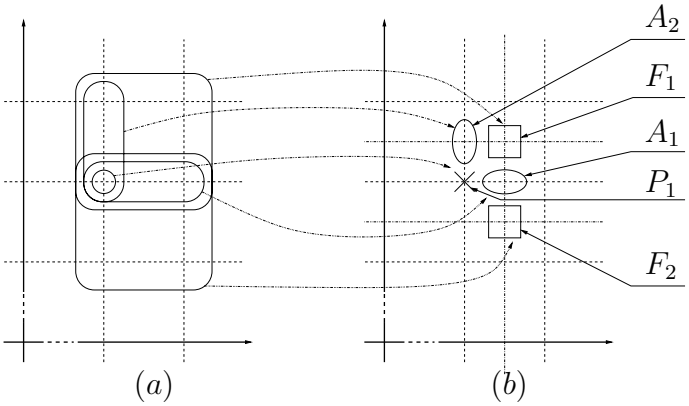
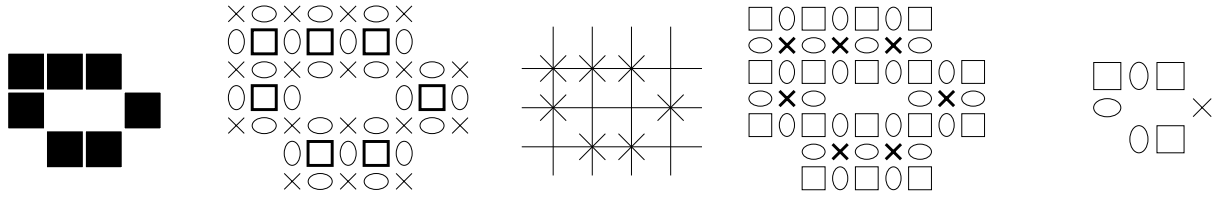


FIG. 8.1 – (a) *Un sous-ensemble* S *de* H^2 *et* (b) *sa représentation tableau.*



(a) (b) (c) (d) (e)

FIG. 8.2 – (a) *Un ensemble* S *de pixels,* (b) $\Phi_1(S)$*,* (c) *un sous-ensemble* S *de* $\mathcal{Z}^2$*,* (d) $\Phi_2(S)$*,* (e) $\Phi_3(S)$.

$H^2_{1_v} = \{\{(i,j),(i,j+1)\},\ i,j \in \mathcal{Z}\}$, $H^2_2 = \{\{(i,j),(i,j+1),(i+1,j),(i+1,j+1)\},\ i,j \in \mathcal{Z}\}$, et $H^2 = H^2_0 \cup H^2_{1_h} \cup H^2_{1_v} \cup H^2_2$.

Un sous-ensemble de H^2_0 (resp. $H^2_{1_h}$, $H^2_{1_v}$, H^2_2) est appelé un *singleton* (resp. un *intervalle horizontal unité*, un *intervalle vertical unité*, un *carré unité*). En fait, H^2 est une structure 2D qui est un complexe cellulaire [Kov89].

Notons $*\mathcal{Z} = \{a + 1/2;\ a \in \mathcal{Z}\}$ et $\mathcal{Z}_+ = \mathcal{Z} \cup *\mathcal{Z}$.

Soit l'application ψ de H^2 dans $\mathcal{Z}^2_+$ telle que $\forall x \in H^2 : \psi(x) = (i,j)$ si $x = \{(i,j)\}$, $\psi(x) = (i+1/2, j)$ si $x = \{(i,j),(i+1,j)\}$, $\psi(x) = (i, j+1/2)$ si $x = \{(i,j),(i,j+1)\}$, et $\psi(x) = (i+1/2, j+1/2)$ si $x = \{(i,j),(i,j+1),(i+1,j),(i+1,j+1)\}$, avec $i,j \in \mathcal{Z}$.

Si $S \subseteq H^2$, $\psi(S) = \{\psi(x);\ x \in S\}$ peut être vu comme une représentation de S, appelée *représentation tableau* [Ber99]. En fait, $\psi(H^2)$ est la grille digitale hétérogène qui est le produit de deux copies de la ligne de Khalimsky [KKM90] [KKM91] [Kop94]. À partir de maintenant, nous utiliserons la représentation tableau pour illustrer certaines notions. L'ordre associé à $\mathcal{Z}^2$ est l'ordre $|H^2| = (H^2, \supseteq)$, ainsi $y \in \alpha(x)$ si $x \supseteq y$.

Considérons le sous-ensemble S de H^2 de la figure 8.1 (a). Ce sous-ensemble est composé de deux carrés unités, ces deux carrés unités contiennent un intervalle horizontal unité, un des deux carrés contient en plus un intervalle vertical unité, et chacun de ces intervalles contient un singleton. Les deux carrés unités contiennent également le singleton (transitivité de la relation α). La figure 8.1 (b) montre la représentation tableau associée au sous-ensemble S. Nous avons adopté la convention suivante : un élément de H^2_0 (resp. $H^2_{1_h}$, $H^2_{1_v}$, H^2_2) sera représenté par une croix (resp. un ovale horizontal, un ovale vertical, un carré) dans la représentation tableau.

8.2.2 Objets 2D

Nous allons maintenant proposer différentes façons de considérer des objets binaires 2D dans $|H^2|$.

En analyse d'images 2D, un objet est souvent défini comme un ensemble de pixels. Rappelons [Kon97] qu'un pixel dans $\mathcal{R}^2$ est le produit cartésien de la forme $[i_1, i_1+1] \times [i_2, i_2+1]$, avec $i_1,\ i_2 \in \mathcal{Z}$. Une image binaire bidimensionnelle, ou 2-image, est définie comme un ensemble fini de pixels de $\mathcal{R}^2$. À chaque 2-image I est associé un sous-ensemble S de H^2 comme suit : à chaque pixel de I est associé le carré unité de H^2 qui est le produit cartésien de la forme $\{i_1, i_1 +$

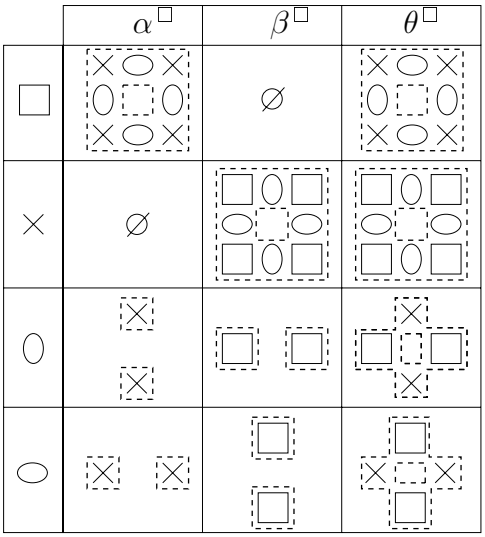


FIG. 8.3 – *Voisinages dans* $|H^2|$.

$1\} \times \{i_2, i_2 + 1\}$ ainsi que les éléments de H^2 qu'il contient. Soit Φ_1 une telle transformation (voir figures 8.2 (a) et (b)). Une autre approche est de considérer un objet 2D, comme un sous-ensemble de $\mathcal{Z}^2$. À chaque sous-ensemble I de $\mathcal{Z}^2$ est associé un sous-ensemble S de H^2 comme suit : à chaque point de I de coordonnées (i, j) dans $\mathcal{Z}^2$, est associé le singleton $\{(i, j)\}$ de H^2 ainsi que les éléments de H^2 qui contiennent ce singleton. Soit Φ_2 une telle transformation (voir figures 8.2 (c) et (d)). Remarquons que ces deux approches multiplient environ par 4 la taille de l'image. Une autre façon consiste à associer à chaque point p de coordonnées (i, j) dans $\mathcal{Z}^2$, le point p' de H^2 tel que $\psi(p') = (i/2, j/2)$. Soit Φ_3 une telle transformation. Celle-là a l'avantage d'associer un seul point de H^2 à un point de $\mathcal{Z}^2$ (voir figures 8.2 (c) et (e)) et préserve donc la taille de l'image initiale. Néanmoins, elle a l'inconvénient de ne pas être invariante par translation (cf. section 8.7.2).

8.3 Voisinages, chemin, composante connexe dans $|H^2|$

Nous présentons maintenant des notions classiques de topologie transposées aux ordres. Nous rappelons que $\theta = \alpha \cup \beta$.

Soient $S \subseteq H^2$ et x un point de S. L'ensemble $\alpha^{\square}_S(x)$ est appelé le $\alpha^{\square}$-*voisinage de* x *dans* S. Si $y \in \alpha^{\square}_S(x)$, nous disons que y est $\alpha^{\square}$-*voisin de* x *dans* S.

Le $\beta^{\square}$-voisinage et le $\theta^{\square}$-voisinage peuvent être définis de la même manière. De façon générale, nous dirons que deux points sont *voisins* si un point appartient au $\theta^{\square}$-voisinage de l'autre. La figure 8.3 représente les différents voisinages possibles (zone délimitée par les pointillés) selon le type de point de H^2 considéré (colonne de gauche) (voir aussi [Kov89]). Rappelons que la relation α correspond à la relation "contient" (le voisinage $\alpha^{\square}$ pour un singleton est vide, cet élément est toujours α-terminal) ; par dualité, la relation β correspond à la relation "est contenu"

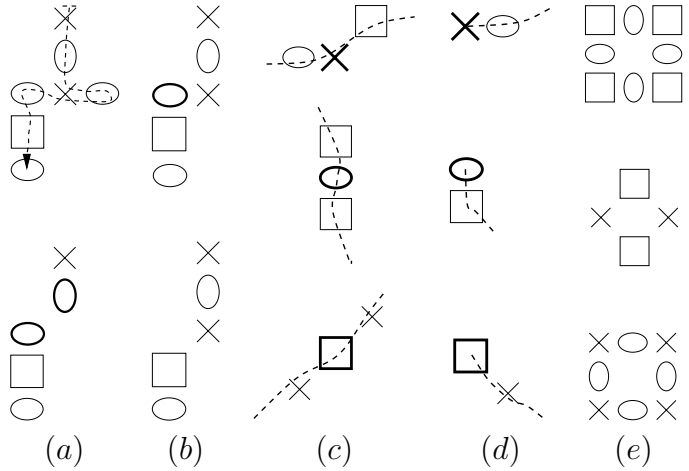


FIG. 8.4 – (a) *Chemins et composantes connexes,* (b) *courbes ouvertes,* (c) *points de courbes,* (d) *points extrémités,* (e) *courbes fermées.*

(le voisinage $\beta^{\square}$ pour un carré est vide, cet élément est toujours β-terminal), et $\theta^{\square}$ est la réunion de $\alpha^{\square}$ et $\beta^{\square}$.

Définition (chemin) : Soit $S \subseteq H^2$. Soient x_0 et x_n deux éléments de S. Un *chemin dans l'ordre* $|S|$ de x_0 à x_n est une suite $\mathcal{C} = x_0, \ldots, x_n$ de points de S, telle que $x_i \in \theta_S^{\square}(x_{i-1})$ avec $i \in \{1, \ldots, n\}$, avec $n > 0$. Le nombre n est la *longueur* du chemin et x_0 est son *origine*.

Définition (composante connexe) : Soit $S \subseteq H^2$. Le sous-ensemble S est dit *connexe* si tout couple de points de S est relié par un chemin dans $|S|$. La relation "est relié par un chemin dans $|S|$" est une relation d'équivalence dont les classes d'équivalence sont les *composantes connexes de l'ordre* $|S|$.

Nous illustrons cette notion dans la figure 8.4 (a). Dans la figure du dessus, il y a une seule composante connexe (un chemin la parcourant est dessiné en pointillé), dans la figure du dessous, deux composantes connexes sont présentes, en effet les deux éléments ovales (en gras sur la figure) ne sont pas voisins.

Définition (courbe ouverte simple) : Soit $S \subseteq H^2$. Le sous-ensemble S est une *courbe ouverte simple* si c'est un ensemble connexe tel que tout point de S a deux voisins dans S, ce sont les *points de courbe* (Fig. 8.4 (c)), à l'exception de deux points qui n'ont qu'un voisin dans S, ce sont les *extrémités de courbe* (Fig. 8.4 (d)). Nous étendons la définition au cas dégénéré suivant : si S est réduit en un point alors S est également une courbe ouverte simple.

La notion de courbe ouverte simple est illustrée sur la figure 8.4 (b), dans l'exemple du dessus, l'élément carré a plus de deux voisins ; si l'élément ovale (en gras sur la figure) est retiré alors nous obtenons une courbe ouverte simple (c'est l'exemple du dessous de la même figure).

Définition (courbe fermée simple) : Soit $S \subseteq H^2$. Le sous-ensemble S est une *courbe fermée simple* si c'est un ensemble connexe tel que tout point de S a exactement deux voisins x et y dans S, et tel que x et y ne sont pas voisins.

Trois exemples de courbes fermées simples sont données à la figure 8.4 (e). Nous proposons quelques autres définitions. Soit $S \subseteq H^2$. Un point x de S est un *point isolé* de S si $\theta^{\square}_S(x)$ est vide ; c'est un *point intérieur* de S si $\theta^{\square}_S(x) = \theta^{\square}_{H^2}(x)$. Nous vérifions facilement que x est un point intérieur de S si et seulement si $\theta^{\square}_S(x)$ est une courbe simple fermée.

8.4 Étude de la simplicité dans $|H^1|$ et dans $|H^2|$

À la section 7.5, nous avons introduit les définitions récursives croisées de point α_n-simple, de point β_n-simple et d'ordre n-contractile.

Il est intéressant de comprendre comment ont été proposées ces définitions récursives. C'est ce que nous nous proposons d'étudier dans cette section à travers les études successives de la simplicité dans $|H^1|$ puis dans $|H^2|$.

Rappelons la définition d'un point α-unipolaire et la propriété concernant leur suppression en parallèle, données dans la section 7.5 [Ber99].

Définition 1 **(point α-unipolaire) :** Soient $|X| = (X, \alpha)$ un ordre, x et y deux points de X. Soit $\alpha^\bullet$, la relation sur X telle que $y \in \alpha^\bullet(x)$ si et seulement si $y \in \alpha^{\square}(x)$ et $\alpha^{\square}(x) \cap \beta^{\square}(y) = \emptyset$. Un point x de X est α-*unipolaire* si $\#[\alpha^\bullet(x)] = 1$, avec $\#S$ étant le nombre d'éléments d'un ensemble S.

Propriété 1 **:** Soient $|X| = (X, \alpha)$ un ordre, y un point de X α-unipolaire et x un point de X avec $x \neq y$. Si x est α-unipolaire, alors x est α_S-unipolaire pour $S = X \setminus \{y\}$.

En d'autres termes, les points α-unipolaires pour $|X|$ peuvent être supprimés en parallèle (sous-entendu tout en préservant les propriétés topologiques de l'ordre $|X|$, voir [Ber99]).

8.4.1 Étude de la simplicité dans $|H^1|$

Soit $|H^1|(= (H^1, \supseteq))$, l'ordre de base associé à $\mathcal{Z}$, défini de la même façon que dans les sections 7.4 et 8.2.1. Les éléments de H^1 sont des singletons (les 0-cubes) et des intervalles unités (les 1-cubes) (cf. la section 7.4).

Propriété 2 **:** Soient $S \subseteq H^1$ et $x \in S$. Nous avons $\alpha^\bullet_S(x) = \alpha^{\square}_S(x)$ et $\beta^\bullet_S(x) = \beta^{\square}_S(x)$.

Preuve :
• Soit s un singleton de S. Nous avons $\alpha^{\square}_{H^1}(s) = \emptyset$ (en particulier $\alpha^{\square}_S(s) = \emptyset$) alors $\alpha^\bullet_S(s) = \emptyset$; par conséquent s n'est jamais α_S-unipolaire.

Notons d'abord que $y \in \beta^{\bullet}(x) \Leftrightarrow y \in \beta^{\square}(x)$ et $\beta^{\square}(x) \cap \alpha^{\square}(y)$.

Dans $|S|$, un singleton peut être contenu par aucun, 1 ou 2 intervalle(s) i_j :
$\diamond$ soit $\beta_S^{\square}(s) = \emptyset$, alors $\beta_S^{\bullet}(s) = \emptyset$,
$\diamond$ soit $\beta_S^{\square}(s) = \{i_1\}$, alors $\beta_S^{\square}(s) \cap \alpha_S^{\square}(i_1) = \emptyset$ et $\beta_S^{\bullet}(s) = \{i_1\}$,
$\diamond$ soit $\beta_S^{\square}(s) = \{i_1, i_2\}$, alors $\forall j \in \{1,2\}, \beta_S^{\square}(s) \cap \alpha_S^{\square}(i_j) = \emptyset$ et $\beta_S^{\bullet}(s) = \{i_1, i_2\}$.
• Soit i un intervalle de S.
Remarquons que $\alpha^{\square}(i)$ est de la "même forme" que $\beta^{\square}(s)$: à savoir deux composantes connexes constituées d'un seul élément, étant soit un singleton pour le cas $\alpha^{\square}(i)$, soit un intervalle pour le cas $\beta^{\square}(s)$. Nous obtenons alors les mêmes résultats que précédemment, en changeant α par β, et "le rôle joué" par un intervalle dans le cas $\beta^{\square}(s)$ par celui d'un singleton dans le cas $\alpha^{\square}(i)$. Nous détaillons néanmoins les résultats :

Dans $|H^1|$, $\beta^{\square}(i) = \emptyset$ (en particulier $\beta_S^{\square}(i) = \emptyset$) alors $\beta_S^{\bullet}(i) = \emptyset$; par conséquent i n'est jamais β_S-unipolaire. Dans $|S|$, un intervalle peut contenir aucun, 1 ou 2 singleton(s) s_j :
$\diamond$ soit $\alpha_S^{\square}(i) = \emptyset$, alors $\alpha_S^{\bullet}(i) = \emptyset$,
$\diamond$ soit $\alpha_S^{\square}(i) = \{s_1\}$, alors $\alpha_S^{\square}(i) \cap \beta_S^{\square}(s_1) = \emptyset$ et $\alpha_S^{\bullet}(i) = \{s_1\}$,
$\diamond$ soit $\alpha_S^{\square}(i) = \{s_1, s_2\}$, alors $\forall j \in \{1,2\}, \alpha_S^{\square}(i) \cap \beta_S^{\square}(i_j) = \emptyset$ et $\alpha_S^{\bullet}(i) = \{s_1, s_2\}$.
□

Propriété 3 **:** Soient $S \subseteq H^1$ et $x \in S$; x est α_S-unipolaire ssi $\#[\alpha_S^{\square}(x)] = 1$, x est β_S-unipolaire ssi $\#[\beta_S^{\square}(x)] = 1$. D'après les résultats de la proposition 2, x ne peut être α-unipolaire et β-unipolaire à la fois. Si x est un singleton, il n'est jamais α-unipolaire, si x est un intervalle, il n'est jamais β-unipolaire.

→ Amorçons la récursivité par la notion d'ordre 0-contractile.

Définition 2 **(ordre** 0**-contractile) :** Soit $|X|$ un ordre. $|X|$ est 0-*contractile* si X est composé d'un seul point (*i.e.* $\#X = 1$).

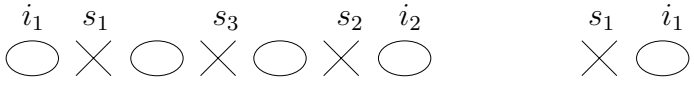
Nous donnons maintenant une propriété liant un point α-unipolaire et un ordre 0-contractile :
Propriété 4 **:** Soient $|X|$ un ordre et $x \in X$. Alors x est α-unipolaire ssi $|\alpha^{\bullet}(x)|$ est 0-contractile (cf. la définition 1).

La propriété 4 et la proposition 2 donnent :
Propriété 5 **:** Soient $S \subseteq H^1$ et $x \in S$; x est α_S-unipolaire ssi $|\alpha_S^{\square}(x)|$ est 0-contractile.

→ Continuons la recherche d'une définition récursive par la notion de point α_1-simple.

Définition 3 **(point** α_1**-simple) :** Soit $|X|$ un ordre. Un point x de X est α_1-*simple* pour $|X|$ si $|\alpha^{\square}(x)|$ est 0-contractile.

Nous en déduisons la propriété suivante indiquant l'équivalence dans $|H^1|$ entre les points α_1-simples et les points α-unipolaires.



(a) (b)

FIG. 8.5 – *Figures utilisées dans la discussion sur la simplicité dans $|H^1|$.*

Propriété 6 **:** Soient $S \subseteq H^1$ et $x \in S$; x est α_1-simple pour $|S|$ ssi x est α_S-unipolaire ssi $\#[\alpha_S^{\square}(x)] = 1$. Un singleton n'est jamais α_1-simple, un intervalle n'est jamais β_1-simple.

De la propriété 6 et de la propriété 1, nous déduisons la propriété suivante :

Propriété 7 **:** Soient $S \subseteq H^1$, x un point α_1-simple pour $|S|$ et y un point de S avec $y \neq x$. Si y est α_1-simple pour $|S|$, alors y est α_1-simple pour $|T|$, avec $T = S \setminus \{x\}$. En d'autres termes, les points α_1-simples pour $|S|$ peuvent être éliminés en parallèle.

Considérons un processus consistant en la suppression en parallèle de points α-unipolaires. Ce processus supprime les points i_1 et i_2 de l'objet de la figure 8.5(a). Nous nous attendons à amincir plus cet objet. Nous remarquons que les singletons s_1 et s_2 sont β-unipolaires (après la suppression de i_1 et i_2).

Notons qu'une suppression simultanée de points α- ou β-unipolaires ne préserve pas la topologie. En effet, considérons la figure 8.5(b) ; une telle suppression retirerait l'intervalle α-unipolaire i_1 et le singleton β-unipolaire s_1, et l'objet disparaîtrait.

Nous proposons alors un processus consistant en la répétition jusqu'à stabilité d'itérations de suppression parallèle de points α-unipolaires alternées avec des itérations de suppression parallèle de points β-unipolaires.

L'objet de la figure 8.5(a) est alors réduit en un point : s_3.

Introduisons maintenant d'autres définitions que nous utiliserons par la suite :

Définition 4 **(ordre contractile) :** Soit $|X|$ un ordre. L'ordre $|X|$ est *contractile* si X peut être réduit en un seul point par la suppression séquentielle de points α-unipolaires, ou de points β-unipolaires.

Avec la proposition 1, nous obtenons la définition suivante :

Définition 5 **(ordre fortement contractile) :** Soit $|X|$ un ordre. L'ordre $|X|$ est *fortement contractile* si X peut être réduit en un seul point par la répétition jusqu'a stabilité des deux itérations suivantes : $i)$ enlever en parallèle tous les points α-unipolaires, $ii)$ enlever en parallèle tous les points β-unipolaires.

$\rightarrow$ Retrouvons notre recherche d'une définition récursive par la notion d'ordre 1-contractile :

Définition 6 **(ordre** 1**-contractile) :** Soit $|X|$ un ordre. L'ordre $|X|$ est 1-*contractile* si X peut être réduit en un seul point par la répétition jusqu'a stabilité des deux itérations suivantes : $i)$ enlever en parallèle tous les points α_1-simples, $ii)$ enlever en parallèle tous les points β_1-simples.

Dans $|H^1|$, nous avons alors la propriété suivante (qui découle des propriétés 1, 6 et 7 et des définitions 4, 5 et 6) :

Propriété 8 **:** Soit $S \subseteq H^1$. L'ordre $|S|$ est 1-*contractile* ssi $|S|$ est (fortement) contractile.

8.4.1.1 Récapitulatif

Soit $S \subseteq H^1$.

Un singleton n'est jamais α-unipolaire ni α_1-simple. Un intervalle n'est jamais β-unipolaire ni β_1-simple. Un singleton s de S est β_S-unipolaire ssi s est β_1-simple pour $|S|$ ssi $\#[\beta_S^{\square}(i)] = 1$. Un intervalle i de S est α_S-unipolaire ssi i est α_1-simple pour $|S|$ ssi $\#[\alpha_S^{\square}(i)] = 1$.

$\rightarrow$ Retenons que, dans $|H^1|$, l'α_1-simplicité équivaut à l'α-unipolarité.

8.4.2 Étude de la simplicité dans $|H^2|$

Nous allons mettre en évidence que certaines propriétés proposées dans $|H^1|$ ne sont plus valables dans $|H^2|$. Nous allons donner les statuts des différents points de $|H^2|$ (du point de vue de l'α-unipolarité, de l'α-simplicité, etc). Cela nous permettra de retrouver la définition d'un point α_2-simple (donnée à la section 7.5) et de prouver que nous pouvons supprimer en parallèle de tels points.

8.4.2.1 Exemple de points α-unipolaires dans $|H^2|$

Considérons le sous-ensemble S de la figure 8.1. Les éléments A_1, A_2 ne contiennent que l'élément P_1 et sont α-unipolaires. L'élément F_1 n'est pas α-unipolaire car $\alpha^{\bullet}(\{F_1\}) = \{A_1, A_2\}$. L'élément P_1 est contenu par les éléments A_1 ou A_2, qui sont, à leur tour, contenus par les éléments F_1 ou F_2, donc l'élément P_1 ne peut appartenir ni à $\alpha^{\bullet}(\{F_1\})$, ni à $\alpha^{\bullet}(\{F_2\})$ (cela illustre la condition que le point y doit satisfaire pour appartenir à $\alpha^{\bullet}(x)$, en effet, il ne doit pas y avoir de point "intermédiaire" contenu par le point x et contenant le point y). Le point F_2 est α-unipolaire car $\alpha^{\bullet}(\{F_2\}) = \{A_1\}$. Si l'on supprime le point A_1, alors le point F_1 devient à son tour α-unipolaire (pour l'ordre $|X \setminus \{A_1\}|$), il est donc α-libre (voir la définition dans la section 7.3). Puis, par les suppressions des points F_1, F_2 et A_2, on obtient l'unique élément P_1, c'est l'α-noyau de S (voir la définition dans la section 7.3).

La recherche de l'α-noyau conduit directement à un processus d'amincissement.

8.4.2.2 Statuts des points de $|H^2|$

Donnons d'abord quelques remarques.
Dans la suite, nous considérons un sous-ensemble $S \subseteq H^2$.

rmq_1 **:** Cas de l'α-unipolarité et de l'α_1-simplicité d'un intervalle i de S.
Remarquons que $\alpha^{\square}_{H^2}(i) = \alpha^{\square}_{H^1}(i)$. Nous avons alors les mêmes résultats que dans $|H^1|$, à savoir : i est α_1-simple pour $|S| \Leftrightarrow i$ est α_S-unipolaire $\Leftrightarrow \#[\alpha^{\square}_S(i)] = 1$ (*i.e.* ssi l'intervalle i contient un seul singleton de S).

rmq_2 **:** Cas de la β-unipolarité et de la β_1-simplicité d'un intervalle i de S.
Rappelons que $\beta^{\square}_{H^1}(i) = \emptyset$ et $\beta^{\square}_{H^2}(i) \neq \beta^{\square}_{H^1}(i)$.
Remarquons que $\beta^{\square}_{H^2}(i)$ est de la même forme que $\alpha^{\square}_{H^2}(i)$: deux composantes connexes constituées d'un seul élément, étant soit un singleton dans le cas $\alpha^{\square}_{H^2}(i)$ (cf. la remarque **rmq_**1), soit un carré ici. Nous avons alors les mêmes résultats que dans la remarque **rmq_**1, en changeant α par β et le "rôle joué" par un singleton par celui d'un carré.
Nous détaillons néanmoins les résultats :
Dans $|S|$, i peut être contenu par aucun, un ou deux carré(s) c_j :
$\diamond$ soit $\beta^{\square}_S(i) = \emptyset$, alors $\beta^{\bullet}_S(i) = \emptyset$,
$\diamond$ soit $\beta^{\square}_S(i) = \{c_1\}$, alors $\beta^{\square}_S(i) \cap \alpha^{\square}_S(c_1) = \emptyset$ et $\beta^{\bullet}_S(i) = \{c_1\}$,
$\diamond$ soit $\beta^{\square}_S(i) = \{c_1, c_2\}$, alors $\forall j \in \{1,2\}, \beta^{\square}_S(i) \cap \alpha^{\square}_S(c_j) = \emptyset$ et $\beta^{\bullet}_S(i) = \{c_1, c_2\}$.
Nous avons $\beta^{\square}_S(i) = \beta^{\bullet}_S(i)$. Par conséquent : i est β_S-unipolaire $\Leftrightarrow \#[\beta^{\square}_S(i)] = 1 \Leftrightarrow |\beta^{\square}_S(i)|$ est 0-contractile $\Leftrightarrow i$ est β_1-simple pour $|S|$ (*i.e.* ssi l'intervalle i est contenu par un seul carré de S).

rmq_3 **:** Cas de l'α-unipolarité et de l'α_1-simplicité d'un singleton s de S.
Nous avons $\alpha^{\square}(s) = \emptyset \Rightarrow \begin{cases} |\alpha^{\square}_S(s)| \text{ non } 0\text{-contractile} \Rightarrow s \text{ non } \alpha_1\text{-simple pour } |S|, \\ \alpha^{\bullet}_S(s) = \emptyset \Rightarrow s \text{ non } \alpha_S\text{-unipolaire.} \end{cases}$
Par conséquent, un singleton s n'est jamais α_1-simple pour $|S|$, ni α_S-unipolaire.

rmq_4 **:** Cas de la β-unipolarité et de la β_1-simplicité d'un carré c de S.
Nous avons $\beta^{\square}(c) = \emptyset \Rightarrow \begin{cases} |\beta^{\square}_S(c)| \text{ non } 0\text{-contractile} \Rightarrow c \text{ non } \beta_1\text{-simple pour } |S|, \\ \beta^{\bullet}_S(c) = \emptyset \Rightarrow c \text{ non } \beta_S\text{-unipolaire.} \end{cases}$
Par conséquent, un carré c n'est jamais β_1-simple pour $|S|$, ni β_S-unipolaire.

rmq_5 **:** Statuts d'un intervalle i relativement à $\alpha^{\square}(c)$ d'un carré c contenant i.
Soient i un intervalle et c un carré tels que $i \in S$, $c \in S$ et $i \in \alpha^{\square}(c)$. Soient $T = \alpha^{\square}(c)$ et $U = T \cap S$.
$\diamond$ Nous avons $\beta^{\square}_T(i) = \emptyset$, en fait $\beta^{\square}_T(i) = \beta^{\square}_{H^1}(i)$; par conséquent i est non β_1-simple pour $|U|$ et i est non β_U-unipolaire.
$\diamond$ $\alpha^{\square}_T(i) = \alpha^{\square}_{H^2}(i)$. Nous avons alors les mêmes résultats que dans la remarque **rmq_**1, *i.e.* i est α_1-simple pour $|U| \Leftrightarrow i$ est α_U-unipolaire $\Leftrightarrow \#[\alpha^{\square}_U(i)] = 1$. De plus $\alpha^{\square}_U(i) = \alpha^{\square}_S(i)$, ainsi i est α_1-simple pour $|U| \Leftrightarrow i$ est α_1-simple pour $|S| \Leftrightarrow i$ est α_S-unipolaire $\Leftrightarrow \#[\alpha^{\square}_S(i)] = 1$ (*i.e.* ssi l'intervalle i contient un seul singleton de S).

rmq_6 : Statuts d'un intervalle i relativement à $\beta^{\square}(s)$ d'un singleton s contenu par i.

Soient i un intervalle et s un singleton tels que $i \in S$, $s \in S$ et $i \in \beta^{\square}(s)$. Soient $T = \beta^{\square}(s)$ et $U = T \cap S$.

$\diamond$ Nous avons $\alpha_T^{\square}(i) = \emptyset \Rightarrow \begin{cases} |\alpha_U^{\square}(i)| \text{ non } 0\text{-contractile} \Rightarrow i \text{ non } \alpha_1\text{-simple pour } |U| \\ \alpha_U^{\bullet}(i) = \emptyset \Rightarrow i \text{ non } \alpha_U\text{-unipolaire} \end{cases}$

$\diamond$ $\beta_T^{\square}(i) = \beta_{H^2}^{\square}(i)$. Nous avons alors les mêmes résultats que dans la remarque **rmq_**2, *i.e.* i est β_1-simple pour $|U| \Leftrightarrow i$ est β_U-unipolaire $\Leftrightarrow \#[\beta_U^{\square}(i)] = 1$. De plus $\beta_U^{\square}(i) = \beta_S^{\square}(i)$, ainsi i est β_1-simple pour $|U| \Leftrightarrow i$ est β_1-simple pour $|S| \Leftrightarrow i$ est β_S-unipolaire $\Leftrightarrow \#[\beta_S^{\square}(i)] = 1$ (*i.e.* ssi l'intervalle i est contenu par un seul carré de S).

rmq_7 **:** Statuts d'un singleton s relativement à $\alpha^{\square}(c)$ d'un carré c contenant s.

Soient s un singleton et c un carré tels que $s \in S$, $c \in S$ et $s \in \alpha^{\square}(c)$. Soient $T = \alpha^{\square}(c)$ et $U = T \cap S$.

$\diamond$ Remarquons que $\beta_T^{\square}(s)$ est de la même forme que $\beta_{H^1}^{\square}(s)$. Par conséquent, s est β_U-unipolaire $\Leftrightarrow \#[\beta_U^{\square}(s)] = 1 \Leftrightarrow |\beta_U^{\square}(s)|$ est 0-contractile $\Leftrightarrow s$ est β_1-simple dans $|U|$.

$\diamond$ Rappelons que s n'est jamais α-unipolaire, ni α_1-simple dans $|H^2|$ (**rmq_**3), nous avons alors les mêmes résultats dans $|U|$.

rmq_8 **:** Statuts d'un carré c relativement à $\beta^{\square}(s)$ d'un singleton s contenu par c.

Soient c un carré et s un singleton tels que $c \in S$, $s \in S$ et $c \in \beta^{\square}(s)$. Soient $T = \beta^{\square}(s)$ et $U = T \cap S$.

$\diamond$ Remarquons que $\alpha_T^{\square}(c)$ est de la même forme que $\beta_{H^1}^{\square}(s)$, avec s un singleton. Par conséquent, c est α_U-unipolaire $\Leftrightarrow \#[\alpha_U^{\square}(c)] = 1 \Leftrightarrow |\alpha_U^{\square}(c)|$ est 0-contractile $\Leftrightarrow c$ est α_1-simple dans $|U|$.

$\diamond$ Rappelons que c n'est jamais β-unipolaire, ni β_1-simple dans $|H^2|$ (**rmq_**4), nous avons alors les mêmes résultats dans $|U|$.

Propriété 9 **:** Soit $S \subseteq H^2$. Soit c un carré de S. Dans $\alpha^{\square}(c)$, seuls les intervalles peuvent être α-unipolaires (ssi ils sont α_1-simples) ; seuls les singletons peuvent être β-unipolaires (ssi ils sont β_1-simples). De la même façon, soit s un singleton de S. Dans $\beta^{\square}(s)$, seuls les intervalles peuvent être β-unipolaires (ssi ils sont β_1-simples) ; seuls les carrés peuvent être α-unipolaires (ssi ils sont α_1-simples).

Preuve :

- Soit c un carré de S.
  Soient $T = \alpha^{\square}(c)$ et $U = T \cap S$.
  Rappelons que $\alpha^{\square}(c)$ est composé uniquement d'intervalles et de singletons. Nous avons :
  - un intervalle i est α_U-unipolaire $\Leftrightarrow i$ est α_1-simple pour $|U| \Leftrightarrow \#[\alpha_U^{\square}(i)] = 1$ (**rmq_**1 et **rmq_**5),
  - un singleton n'est ni α-unipolaire, ni α_1-simple (**rmq_**3),
  - dans T, un intervalle n'est ni β_U-unipolaire, ni β_1-simple pour $|U|$ (**rmq_**5),
  - dans T, un singleton s est β_U-unipolaire $\Leftrightarrow s$ est β_1-simple pour $|U| \Leftrightarrow \#|\beta_U^{\square}(i)| = 1$ (**rmq_**7).

– Soit s un singleton de S.
Soient $T = \beta^{\square}(s)$ et $U = T \cap S$.
Rappelons que $\beta^{\square}(s)$ est composé uniquement d'intervalles et de carrés. Nous avons :
  – un intervalle i est β_U-unipolaire $\Leftrightarrow$ i est β_1-simple pour $|U| \Leftrightarrow \#[\beta_U^{\square}(i)] = 1$ (**rmq_**2 et **rmq_**6),
  – un carré n'est ni β-unipolaire, ni β_1-simple (**rmq_**4),
  – dans T, un intervalle n'est ni α_U-unipolaire, ni α_1-simple pour $|U|$ (**rmq_**6),
  – dans T, un carré c est α_U-unipolaire $\Leftrightarrow$ c est α_1-simple pour $|U| \Leftrightarrow \#|\alpha_U^{\square}(c)| = 1$ (**rmq_**8).

□

rmq_9 **:** Cas d'α-unipolarité et d'α_1-simplicité d'un carré c.
Soit $c \in S$.
◇ c est α_1-simple pour $|S| \Leftrightarrow \#[\alpha_S^{\square}(c)] = 1 \Leftrightarrow c$ contient un unique élément e de S.
Dans ce cas, $\alpha_S^{\square}(c) \cap \beta_S^{\square}(e) = \emptyset$ et $\alpha_S^{\bullet}(c) = \{e\}$, par conséquent c est également α_S-unipolaire.
◇ Soit un intervalle $i \in \alpha_S^{\square}(c)$. Alors i est tel que $\alpha_S^{\square}(c) \cap \beta_S^{\square}(i) = \emptyset$ et $i \in \alpha_S^{\bullet}(c)$. Une condition nécessaire pour que c soit α_S-unipolaire est que c contienne au plus un intervalle de S.
◇ Soit un singleton $s \in \alpha_S^{\square}(c)$. Soit $T = \alpha_S^{\square}(c) \cap \beta_S^{\square}(s)$, T peut être vide ou contenir un ou deux intervalle(s) i_j de S :
- soit $T = \emptyset$, alors $s \in \alpha_S^{\bullet}(c)$,
- soit $T = \{i_1\}$, alors $s \notin \alpha_S^{\bullet}(c)$ et $i_1 \in \alpha_S^{\bullet}(c)$ (cf. avant),
- soit $T = \{i_1, i_2\}$, alors $s \notin \alpha_S^{\bullet}(c)$ et $i_1 \in \alpha_S^{\bullet}(c)$ et $i_2 \in \alpha_S^{\bullet}(c)$ (cf. avant).

◇ Par conséquent, c est α_S-unipolaire $\Leftrightarrow \exists! z \in \alpha_S^{\square}(c)$ tel que $\alpha_S^{\square}(c) \cap \beta_S^{\square}(z) = \emptyset$; deux cas sont possibles :
- soit $\exists!$ intervalle $i \in \alpha_S^{\square}(c)$ et dans ce cas $\forall$ singleton $s \in \alpha_S^{\square}(c)$ alors $s \in \alpha_S^{\square}(i)$ (et $z = i$) (si $\alpha_S^{\square}(i) = \emptyset$ alors $\alpha_S^{\square}(c) = \{i\}$ et c est α_1-simple pour $|S|$),
- soit $\nexists$ intervalle $i \in \alpha_S^{\square}(c)$ et dans ce cas $\exists!$ singleton $s \in \alpha_S^{\square}(c)$ (*i.e.* $\alpha_S^{\square}(c) = \{s\}$) (et $z = s$) et dans ce cas c est α_1-simple pour $|S|$.

En résumé,
– c est α_1-simple pour $|S|$ si $\alpha_S^{\square}(c)$ est soit un singleton, soit un intervalle ; et dans ces cas, c est α_S-unipolaire,
– c est α_S-unipolaire et non α_1-simple pour $|S|$ s'il contient un unique intervalle et si celui-là contient un ou deux singletons de S.
→ Cela montre qu'il n'y a pas équivalence entre l'α-unipolarité et l'α_1-simplicité dans $|H^2|$ (contrairement au cas $|H^1|$, cf. la propriété 6, page 169).

rmq_10 **:** Cas de β-unipolarité et de β_1-simplicité d'un singleton s.
Ici, un intervalle (resp. un carré) "joue le même rôle" que celui d'un intervalle (resp. un singleton) lors du test de l'α-unipolarité ou de l'α_1-simplicité d'un carré (cf. **rmq_**9).
En détaillant, cela donne :

Soit $s \in S$.

$\diamond$ s est β_1-simple pour $|S| \Leftrightarrow \#[\beta_S^{\square}(s)] = 1 \Leftrightarrow s$ est contenu par un unique élément e de S.

Dans ce cas, $\beta_S^{\square}(s) \cap \alpha_S^{\square}(e) = \emptyset$ et $\beta_S^{\bullet}(s) = \{e\}$, par conséquent s est également β_S-unipolaire.

$\diamond$ Soit un intervalle $i \in \beta_S^{\square}(s)$. Alors i est tel que $\beta_S^{\square}(s) \cap \alpha_S^{\square}(i) = \emptyset$ et $i \in \beta_S^{\bullet}(s)$. Une condition nécessaire pour que s soit β_S-unipolaire est que s soit contenu par au plus un intervalle.

$\diamond$ Soit un carré $c \in \beta_S^{\square}(s)$. Soit $T = \beta_S^{\square}(s) \cap \alpha_S^{\square}(c)$, T peut être vide ou contenir un ou deux intervalle(s) i_j de S :

- soit $T = \emptyset$, alors $c \in \beta_S^{\bullet}(s)$,
- soit $T = \{i_1\}$, alors $c \notin \beta_S^{\bullet}(s)$ et $i_1 \in \beta_S^{\bullet}(s)$ (cf. avant),
- soit $T = \{i_1, i_2\}$, alors $c \notin \beta_S^{\bullet}(s)$ et $i_1 \in \beta_S^{\bullet}(s)$ et $i_2 \in \beta_S^{\bullet}(s)$ (cf. avant).

$\diamond$ Par conséquent, s est β_S-unipolaire $\Leftrightarrow \exists! z \in \beta_S^{\square}(s)$ tel que $\beta_S^{\square}(s) \cap \alpha_S^{\square}(z) = \emptyset$; deux cas sont possibles :

- soit $\exists!$ intervalle $i \in \beta_S^{\square}(s)$ et dans ce cas $\forall$ carré $c \in \beta_S^{\square}(s)$ alors $c \in \beta_S^{\square}(i)$ (et $z = i$) (si $\beta_S^{\square}(i) = \emptyset$ alors $\beta_S^{\square}(s) = \{i\}$ et s est β_1-simple pour $|S|$),
- soit $\nexists$ intervalle $i \in \beta_S^{\square}(s)$ et dans ce cas $\exists!$ carré $c \in \beta_S^{\square}(s)$ (*i.e.* $\beta_S^{\square}(s) = \{c\}$) (et $z = c$) et dans ce cas s est β_1-simple pour $|S|$.

En résumé,

- s est β_1-simple pour $|S|$ si $\beta_S^{\square}(s)$ est soit un carré, soit un intervalle ; et dans ces cas, s est β_S-unipolaire,
- s est β_S-unipolaire et non β_1-simple pour $|S|$ s'il est contenu par un unique intervalle et si celui-là est contenu par un ou deux carrés de S.

$\rightarrow$ Cela montre qu'il n'y a pas équivalence entre la β-unipolarité et la β_1-simplicité dans $|H^2|$ (contrairement au cas $|H^1|$, cf. la propriété 6).

Des remarques **rmq_**1**, rmq_**2**, rmq_**9 **et rmq_**10, nous avons :

Propriété 10 **:** Soit $S \subseteq H^2$; tout élément α_1-simple (resp. β_1-simple) pour $|S|$ est α_S-unipolaire (resp. β_S-unipolaire). La réciproque est vraie pour les intervalles, elle est fausse pour les carrés et les singletons.

Considérons la figure 8.6(a). Dans la figure de gauche, les seuls points α-unipolaires sont P_1 et P_2. Un processus consistant en la suppression (itérative ou parallèle) des points α-unipolaires produit l'objet dessiné à droite dans la figure 8.6(a). Ce processus est bloqué par les deux points P_3 et P_4, qui eux sont β-unipolaires.

Par ailleurs, la suppression parallèle à la fois de points α- ou β-unipolaires peut ne pas préserver la topologie. En effet, dans la figure 8.6(b), le point P_1 est β-unipolaire et le point P_2 est α-unipolaire. L'objet disparaît par la suppression parallèle de points α- ou β-unipolaires. Nous devons alors considérer un processus consistant en la répétition alternée des deux étapes suivantes : l'une supprime en parallèle les points α-unipolaires, et l'autre supprime en parallèle les points β-unipolaires. Ce processus parallèle de suppression préserve la topologie (cf. la propriété 1 page 167).

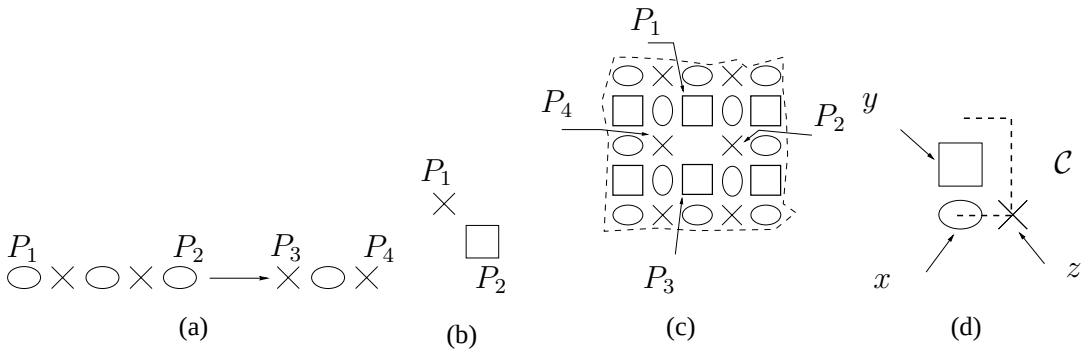


FIG. 8.6 – *Figures utilisées dans la discussion sur la simplicité dans $|H^2|$.*

Toutefois, nous pouvons remarquer que l'objet S constitué de l'ordre $|H^2|$ privé d'un intervalle (objet dessiné partiellement, figure 8.6(c)), ne peut pas être aminci avec ce processus. En effet, aucun des points de cet objet n'est α-unipolaire, ou β-unipolaire. Remarquons que les carrés de la figure 8.6(c), voisins de l'intervalle absent, sont tels que $|\alpha_S^{\square}(x)|$ est contractile, ou que les singletons voisins de l'intervalle absent sont tels que $|\beta_S^{\square}(x)|$ est contractile (cf. la définition 4, page 169).

Nous proposons alors la définition suivante :
Définition 7 **(point α-simple) :** Soient $S \subseteq H^2$ et x un point de S. Le point x est α-*simple* pour $|S|$ si $|\alpha_S^{\square}(x)|$ est contractile.

Dans l'exemple de la figure 8.6(c), les points P_1 et P_3 sont α-simples, et les points P_2 et P_4 sont β-simples. Un processus consistant en la suppression parallèle et alternée de points α- ou β-simples permet maintenant d'amincir cet objet.

rmq_11 **:** Cas de contractilité de $|\alpha^{\square}(c)|$ d'un carré c.
La propriété 9 montre que $|\alpha^{\square}(c)|$ (fortement) contractile $\Leftrightarrow$ $\alpha^{\square}(c)$ peut être réduit en un point par la répétition d'itérations de suppression (parallèle) de points α-unipolaires (ici, seuls les intervalles peuvent l'être) alternées avec des itérations de suppression (parallèle) de points β-unipolaires (ici, seuls les singletons peuvent l'être).

Nous avons la propriété suivante :
Propriété 11 **:** Soient $S \subseteq H^2$ et c un carré de S. Alors $|\alpha_S^{\square}(c)|$ est (fortement) contractile $\Leftrightarrow$ $\alpha_S^{\square}(c)$ est une courbe ouverte simple.
Preuve :
On suppose que $\alpha_S^{\square}(c)$ est une courbe ouverte simple. Si elle est de longueur 1, alors c'est un ensemble (fortement) contractile. Sinon, tous les points appartenant à $\alpha_S^{\square}(c)$ ont deux voisins sauf les extrémités qui n'en ont qu'un. Les points extrémités sont les seuls points α_S- ou β_S-unipolaires. Par la suppression itérative des points α_S- ou β_S-unipolaires (*c.-à-d.* les extrémités

de la courbe), on transforme la courbe ouverte simple en une autre courbe ouverte simple (de longueur inférieure). En répétant cela jusqu'à stabilité, on aboutit finalement à une courbe ouverte simple de longueur 1. Ainsi $\alpha_S^{\square}(c)$ est contractile, et le point c est α-simple.

De la forme même de $\alpha_S^{\square}(c)$ (*i.e.* alternance de singletons et d'intervalles), nous voyons que la suppression des points unipolaires (les extrémités) peut s'effectuer en parallèle. Ainsi $|\alpha_S^{\square}(c)|$ est fortement contractile.

$\diamond$ Nous pouvons voir de la même façon que la condition est nécessaire.
En effet, $\alpha_S^{\square}(c)$ peut être vide, être une courbe fermée simple, ou être une ou l'union de plusieurs courbes ouvertes simples. La suppression de points α_S-unipolaires ou β_S-unipolaires peut amincir chacune des (ou de la) courbe(s) ouverte(s) simple(s) en un point. Par conséquent, $\alpha_S^{\square}(c)$ peut être réduit en un point si $\alpha_S^{\square}(c)$ est une unique courbe ouverte simple. $\square$

rmq_12 **:** Cas de contractilité de $|\beta^{\square}(s)|$ d'un singleton s.

De la même façon que dans la remarque **rmq_**11 et avec la propriété 9, nous avons la propriété suivante :

Propriété 12 **:** Soient $S \subseteq H^2$ et s un singleton de S. Alors $|\beta_S^{\square}(s)|$ est (fortement) contractile $\Leftrightarrow \beta_S^{\square}(c)$ est une courbe ouverte simple.

Nous en déduisons les deux propriétés suivantes :

Propriété 13 **:** Soient $S \subseteq H^2$ et $x \in S$; x est α-simple pour $|S|$ ssi $\alpha_S^{\square}(x)$ est une courbe ouverte simple ssi $|\alpha_S^{\square}(x)|$ est 1-contractile. De la même façon, x est β-simple pour $|S|$ ssi $\beta_S^{\square}(x)$ est une courbe ouverte simple.
Preuve : Pour le cas x α-simple :
- si x est un carré,
  voir la propriété 11 concernant le processus de suppression de points unipolaires à partir d'une courbe ouverte simple,
  et la propriété 9 signifiant l'équivalence entre les points α-unipolaires (resp. β-unipolaires) et les points α_1-simples (resp. β_1-simples) dans $\alpha_S^{\square}(x)$,
- si x est un intervalle, nous avons : x est simple pour $|S|$
  $\Leftrightarrow |\alpha_S^{\square}(x)|$ est contractile
  $\Leftrightarrow \#[\alpha_S^{\square}(x)] = 1 \Leftrightarrow |\alpha_S^{\square}(x)|$ est 1-contractile,
  $\Leftrightarrow \alpha_S^{\square}(x)$ est une courbe ouverte simple de longueur 1,
- si x est un singleton, alors $\alpha^{\square}(x) = \emptyset$.

$\square$

Propriété 14 **:** *Soient $S \subseteq H^2$, x un point α-simple pour $|S|$ et y un point de S avec $y \neq x$. Si y est α-simple pour $|S|$, alors y est α-simple pour $|T|$, avec $T = S \setminus \{x\}$.*

Preuve (voir également la figure 8.6(d)) :
Soient x et y deux points α-simples pour $|S|$. Comme l'α-simplicité de y pour $|S|$ ne dépend que de $\alpha_S^{\square}(y)$, le seul cas à considérer est : $x \in \alpha_S^{\square}(y)$; x ne peut être un singleton car celui-ci n'est jamais α-simple, on a alors uniquement le cas où x est un intervalle et y, un élément carré. Comme x est α-simple pour $|S|$, il doit contenir un unique singleton z de S. Or y étant α-simple pour $|S|$, $\alpha_S^{\square}(y)$ est une courbe ouverte simple de la forme $\mathcal{C} = x_0, \ldots, x_{n-1}, x_n$ avec $x_{n-1} = z$,

$x_n = x$ et $n > 0$ (courbe dessinée en pointillé sur la figure 8.6(d)). Le fait de supprimer x transforme $\mathcal{C}$ en une autre courbe ouverte simple (l'existence est prouvée par la présence de z), et y est α_T-simple pour $T = S \setminus \{x\}$. Notons que si la courbe $\mathcal{C}$ est telle que x_0 est également un intervalle y α-simple (le seul autre possible) alors le raisonnement tient également pour la suppression parallèle de ces deux extrémités de courbe que sont les deux seuls points α-simples de $\mathcal{C}$ pour $|S|$. □

Nous pouvons donc supprimer en parallèle des points α-simples. Par dualité, on peut supprimer en parallèle des points β-simples. C'est cette propriété que nous allons exploiter dans la prochaine section, pour proposer des algorithmes de squelettisation. Nous définissons alors la notion de point α_2-simple (équivalente ici à celle de point α-simple) et la notion d'ordre 2-contractile (afin de retrouver la récursivité des définitions données dans la section 7.5).

Définitions 8 **(point α_2-simple, ordre 2-contractile) :** Soit $|X| = (X, \alpha, \beta)$ un ordre dénombrable, localement fini et non vide. Un point x est α_2-*simple* pour $|X|$ si $|\alpha^{\square}(x)|$ est 1-contractile. L'ensemble composé de tous les points de X qui ne sont pas α_2-simples pour $|X|$ est noté par X_{α_2}. L'ordre $|X|$ est 2-*contractile*, s'il existe une séquence $X^0, \ldots, X^k$, avec $X^0 = X$, $X^i = X^{i-1}_{\alpha_2}$ si i est impair, $X^i = X^{i-1}_{\beta_2}$ si i est pair, pour $i = 1, \ldots, k$ et tel que $X_k = \{a\}, a \in X$.

8.4.2.3 Récapitulatif

Soit $S \subseteq H^2$ (voir la figure 8.7).

- Statuts d'un intervalle. Soit i un intervalle de S.
  - i est α_S-unipolaire $\Leftrightarrow$ i est α_1-simple pour $|S| \Leftrightarrow \#[\alpha_S^{\square}(i)] = 1$
    $\Leftrightarrow \alpha_S^{\square}(i)$ est un singleton $\Leftrightarrow$ i est α_2-simple pour $|S|$ (Fig. 8.7 (a)),
  - si $\alpha_S^{\square}(i)$ n'est pas un singleton ***i.e.*** i ne contient aucun singleton ou i contient deux singletons alors i n'est ni α_S-unipolaire, ni α_1-simple pour $|S|$, ni α_2-simple pour $|S|$ (Fig. 8.7 (b)),
  - i est β_S-unipolaire $\Leftrightarrow$ i est β_1-simple pour $|S| \Leftrightarrow \#[\beta_S^{\square}(i)] = 1$
    $\Leftrightarrow \beta_S^{\square}(i)$ est un carré $\Leftrightarrow$ i est β_2-simple pour $|S|$ (Fig. 8.7 (c)),
  - si $\beta_S^{\square}(i)$ n'est pas un carré ***i.e.*** i n'est contenu par aucun carré ou i est contenu par deux carrés alors i n'est ni β_S-unipolaire, ni β_1-simple pour $|S|$, ni β_2-simple pour $|S|$ (Fig. 8.7 (d)).
- Statuts d'un carré. Soit c un carré de S.
  - c est α_1-simple pour $|S|$ si $\alpha_S^{\square}(c)$ est soit un singleton soit un intervalle ; dans ces cas, il est α_S-unipolaire et α_2-simple pour $|S|$ (Fig. 8.7 (e)),
  - c est α_S-unipolaire et non α_1-simple pour $|S|$ s'il contient un unique intervalle et que celui-là contient un ou deux singletons ; dans ces cas, il est α_2-simple pour $|S|$ (Fig. 8.7 (f)),
  - c est α_2-simple pour $|S|$ et non α_S-unipolaire (par conséquent non α_1-simple pour $|S|$) s'il contient une courbe ouverte simple contenant au moins 2 intervalles de S (Fig. 8.7 (g)),

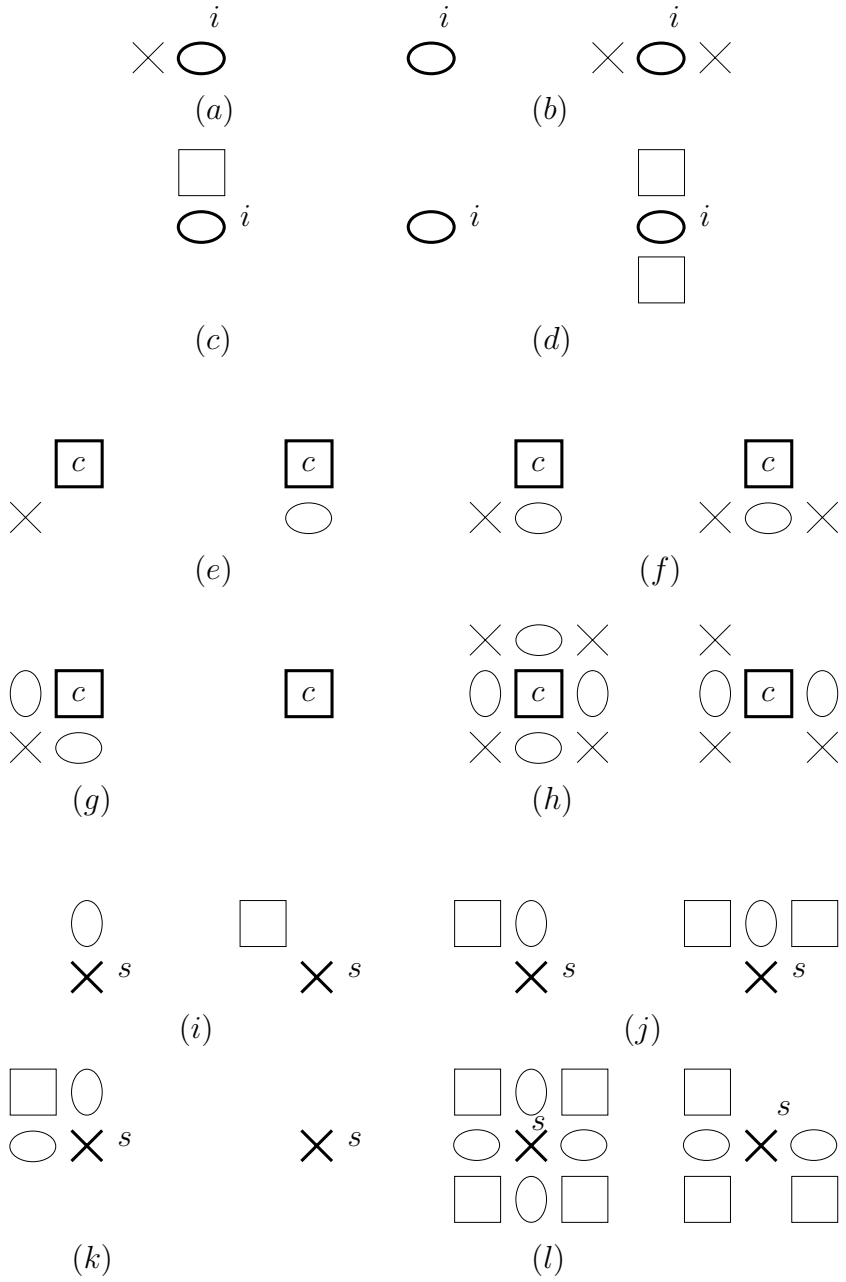


FIG. 8.7 – *Récapitulatif des statuts des points de H^2. Exemples pour un intervalle i : (a) i α-unipolaire et α_1-simple ; (b) i non α-unipolaire (donc ni α_1-simple, ni α_2-simple) ; (c) i β-unipolaire et β_1-simple ; (d) i non β-unipolaire (donc ni β_1-simple, ni β_2-simple). Exemples pour un carré c : (e) c α_1-simple, α-unipolaire, α_2-simple ; (f) c α-unipolaire, non α_1-simple, α_2-simple ; (g) c α_2-simple non α-unipolaire ; (h) c non α_2-simple. Exemples pour un singleton s : (i) s β_1-simple, β-unipolaire, β_2-simple ; (j) s β-unipolaire, non β_1-simple, β_2-simple ; (k) s β_2-simple non β-unipolaire ; (l) s non β_2-simple.*

  - c n'est pas α_2-simple pour $|S|$ si $\alpha_S^{\square}(c) = \emptyset$, si $\alpha_S^{\square}(c)$ est une courbe fermée, si $\alpha_S^{\square}(c)$ est constitué de plusieurs courbes ouvertes simples (Fig. 8.7 (h)),
  - c jamais β_1-simple, ni β_2-simple, ni β-unipolaire car $\beta^{\square}(c) = \emptyset$.
- Statuts d'un singleton. Soit s un singleton de S.
  - s est β_1-simple pour $|S|$ si $\beta_S^{\square}(c)$ est soit un carré soit un intervalle ; dans ces cas, il est β_S-unipolaire et β_2-simple pour $|S|$ (Fig. 8.7 (i)),
  - s est β_S-unipolaire et non β_1-simple pour $|S|$ s'il est contenu par un unique intervalle et que celui-là est contenu par un ou deux carrés ; dans ces cas, il est β_2-simple pour $|S|$ (Fig. 8.7 (j)),
  - s est β_2-simple pour $|S|$ et non β_S-unipolaire (par conséquent non β_1-simple pour $|S|$) si $\beta_S^{\square}(s)$ est une courbe ouverte simple contenant au moins 2 intervalles de S (Fig. 8.7 (k)),
  - s n'est pas β_2-simple pour $|S|$ si $\beta_S^{\square}(s) = \emptyset$, si $\beta_S^{\square}(s)$ est une courbe fermée, si $\beta_S^{\square}(s)$ est constitué de plusieurs courbes ouvertes simples (Fig. 8.7 (l)),
  - s jamais α_1-simple, ni α_2-simple, ni α-unipolaire car $\alpha^{\square}(s) = \emptyset$.

Introduisons maintenant la notion d'ordre σ-équivalent (utilisée au chapitre 10) :

Définition (ordre σ-équivalent) : Soient $|X|$ et $|Y|$ deux ordres. L'ordre $|Y|$ est *(sous) σ-équivalent* à $|X|$ si $X = Y$ ou si Y peut être obtenu à partir de X par suppressions itératives de points α- ou β-simples.

Considérons un sous-ensemble S de H^2, et le sous-ensemble T de H^2 obtenu à partir de S après la suppression en parallèle de tous les points α-simples de S (rappelons que dans $|H^2|$ les points α-simples sont les points α_2-simples). La propriété 14 implique que $|T|$ est σ-équivalent à $|S|$.

8.4.2.4 Conclusion

À partir de la seule notion de point α-unipolaire et de la propriété de pouvoir supprimer en parallèle de tels points, nous avons défini une notion plus puissante (pour un processus de suppression de points) : celle de point α_2-simple ; et avons montré que nous pouvons supprimer en parallèle de tels points.

8.5 Algorithmes de squelettisation pour images binaires

8.5.1 Algorithme de squelettisation ultime

Soit S un sous-ensemble de H^2. Soit $S_\alpha = \{x \in S;\ x$ est α_2-simple pour $|S|\}$, l'ensemble des points α_2-simples pour $|S|$. Dans la suite, nous notons S^0 le sous-ensemble initial S.

Nous proposons un processus d'amincissement consistant à retirer en parallèle des points α_2-simples, puis à retirer en parallèle des points β_2-simples, et cela jusqu'à stabilité. Ce processus est décrit par l'algorithme 9.

$S^0 = S$
Répéter

$$\left\{ \begin{array}{lcll} S^{2i+1} & = & S^{2i} & \setminus \; S^{2i}_{\alpha}, \\ S^{2i+2} & = & S^{2i+1} & \setminus \; S^{2i+1}_{\beta}. \end{array} \right.$$

Jusqu'à ce qu'il n'y ait plus de suppression durant 2 sous-itérations successives.

Algorithme 9: *Algorithme de squelettisation ultime dans* $|H^2|$.

Dans le cadre de topologie digitale, un algorithme d'amincissement produit parfois des résultats pouvant comporter des points redondants, *i.e.* des points simples. Ce n'est pas le cas avec notre processus d'amincissement : le résultat ne comporte ni point α_2-simple, ni point β_2-simple. Nous pouvons alors dire que le résultat est *mince i.e.* qu'il ne comporte plus de point que l'on puisse supprimer tout en préservant les propriétés topologiques de l'objet.

C'est pour cette raison, que nous introduisons une nouvelle terminologie, qui sera adoptée dans la suite de cette partie.

Un processus décrit par l'algorithme 9 s'appelle *algorithme de squelettisation ultime* et le résultat est appelé *squelette ultime* (voir la terminologie dans la section 8.5.2).

8.5.2 Terminologie [LB00a]

Dans cette section, un point simple est à prendre ici au sens de point α_2- ou point β_2-simple.

Nous disons qu'un objet Y est un *squelette* d'un objet X si Y peut être obtenu à partir de X par suppression itérative de points simples. Tout algorithme qui fournit un squelette est appelé *algorithme de squelettisation*. Un *squelette ultime* est un squelette qui ne contient plus de point simple. Un algorithme de squelettisation qui produit un squelette ultime est appelé *algorithme de squelettisation ultime*. Un autre cas spécial de squelette est celui de *squelette curviligne* dans lequel tous les points sont soit non simples soit extrémités. Un tel squelette préserve des propriétés géométriques de l'objet initial. Un algorithme de squelettisation qui produit un squelette curviligne à partir d'un objet est appelé *algorithme de squelettisation curviligne*.

8.5.3 Algorithme de squelettisation curviligne

Si nous voulons obtenir un *squelette curviligne*, les points extrémités de courbe doivent être préservés durant le processus décrit précédemment. Un point x est un *point extrémité dans l'ordre* $|S|$ si $\theta^{\square}_S(x)$ est composé d'un unique point. Notons qu'un point extrémité est soit α_2-simple, soit β_2-simple. Soit S_{ext}, l'ensemble des points extrémités de $|S|$.

Nous proposons un processus de squelettisation consistant à retirer en parallèle des points α_2-simples et non extrémités, puis à retirer en parallèle des points β_2-simples et non extrémités, et cela jusqu'à stabilité. Ce processus est décrit par l'algorithme 10.

Dans la suite, une itération de suppression parallèle de points α_2-simples (resp. β_2-simples) sera appelée une itération PHASE_α (resp. PHASE_β) (sans précision supplémentaire, pour l'ins-

$S^0 = S$
Répéter

$$\begin{cases} S^{2i+1} = S^{2i} \setminus [S^{2i}_{\alpha} \cap \overline{S^{2i}_{ext}}], \\ S^{2i+2} = S^{2i+1} \setminus [S^{2i+1}_{\beta} \cap \overline{S^{2i+1}_{ext}}]. \end{cases}$$

Jusqu'à ce qu'il n'y ait plus de suppression durant 2 sous-itérations successives.

Algorithme 10: *Algorithme de squelettisation curviligne dans $|H^2|$.*

tant, concernant les points extrémités).

Considérons l'objet initial $S(= S^0)$ représenté dans la figure 8.8 (a). Les singletons sont ici représentés par des cercles plutôt que par des croix, et les intervalles sont représentés par des rectangles plutôt que par des ovales. Nous voulons d'abord obtenir un squelette curviligne de S. La première sous-itération supprime la couche externe de S, car elle est composée par tous les points α_2-simples de S (qui sont non extrémités) ; ainsi nous obtenons S^1 (Fig. 8.8 (b)). La deuxième sous-itération supprime la couche externe de S^1 qui est constituée par tous les points β_2-simples de S^1 (ils sont également non extrémités) ; ainsi nous obtenons S^2 (Fig. 8.8 (c)). Notons que les points x_1 et x_2 sont des points extrémités. Il n'y a plus de point α_2- ou β_2-simple non extrémité. Le processus de squelettisation curviligne s'arrête et S^2 est un squelette curviligne de S. Si nous voulons un squelette ultime de S, après les deux premières sous-itérations, nous obtenons également le sous-ensemble S^2 (aucun point extrémité n'apparaît dans S^0 ou S^1). Dans la troisième sous-itération de suppression, les points x_1 et x_2 sont supprimés puisqu'ils sont α_2-simples. Nous obtenons S^3 (Fig. 8.8 (d)), et ainsi de suite. Finalement, S^4 est un squelette ultime de S, puisqu'un seul point reste et que celui-ci n'est ni α_2-simple, ni β_2-simple (Fig. 8.8 (e)).

8.6 Tests de la simplicité

Dans cette section, nous remarquons qu'il est inutile de considérer tous les points comme candidats éventuels aux tests de simplicité (première remarque). Puis nous proposons une caractérisation rapide des points α_2-simples et des points extrémités en minimisant le nombre d'accès aux voisins (deuxième remarque).

Remarque : Un élément carré n'est jamais β_2-simple. Il n'a donc pas à être examiné lors d'une sous-itération PHASE_β. De la même façon, un élément croix n'est jamais α_2-simple. Il n'a pas à être examiné lors d'une sous-itération PHASE_α. Nous examinons donc trois types d'éléments sur quatre, durant chaque sous-itération.

Remarque : Soit x, un point pour lequel l'α_2-simplicité est testée. Les points dans le $\alpha^{\square}$-voisinage de x sont nommés selon la figure 8.9. Soit S un sous-ensemble de H^2. Supposons que les points de S (resp. $\overline{S}$) prennent la valeur "1" (resp. "0"). Nous avons vu qu'un point x est α_2-simple si $\alpha^{\square}_S(x)$ est une courbe ouverte simple. Nous donnons les remarques suivantes, selon le type de point x testé (voir également la figure 8.3).

• Un intervalle horizontal x est α_2-simple si et seulement si $x_1 + x_2 = 1$. Si, de plus, $y_1 + y_2 = 0$,

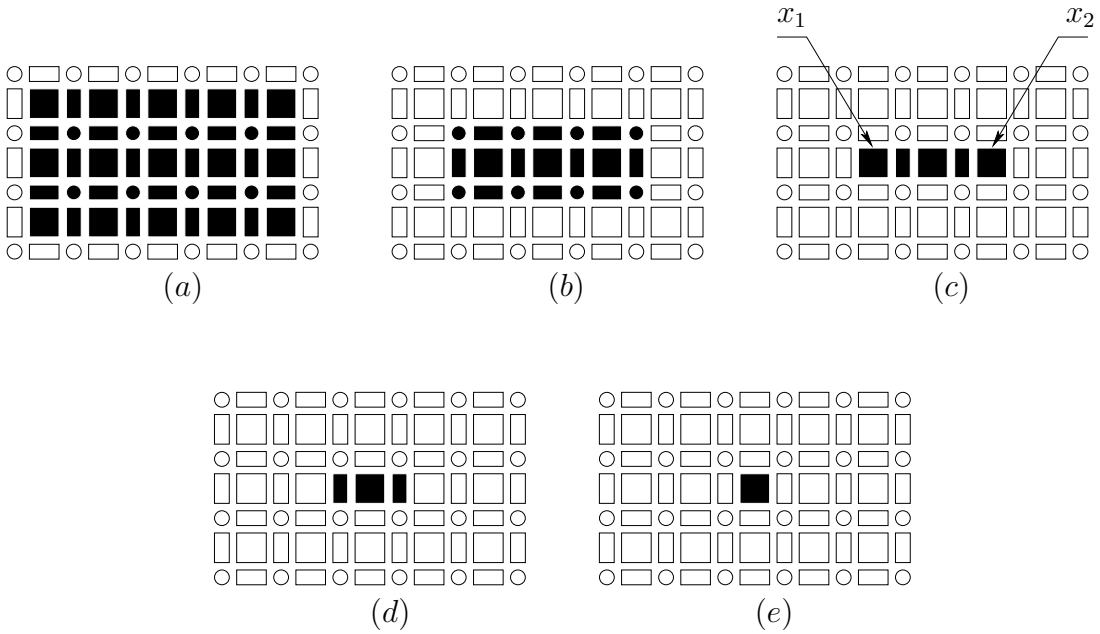


FIG. 8.8 –
(a) L'objet initial $S = S^0$, (b) S^1, (c) S^2 : un squelette curviligne de S, (d) S^3, (e) S^4 : un squelette ultime de S.

le point x est extrémité et doit être gardé lors du calcul du squelette curviligne. Nous obtenons les mêmes résultats pour l'intervalle vertical (à rotation près). Pour le cas de la β_2-simplicité, il suffit d'échanger le rôle de x_i et de y_i.

• Nous voulons maintenant déterminer si un élément carré x est α_2-simple (ou par dualité, si un élément singleton est β_2-simple). Soit le nombre $\mathcal{R}(x) = \sum_{i=1}^{8} |z_{i+1} - z_i|$, avec $z_9 = z_1$ (Fig. 8.9). Cette valeur comptabilise le nombre de transitions de 0 à 1 et de 1 à 0, lors d'un parcours de $\alpha^{\square}(x)$, dans le sens horaire. En fait $\mathcal{R}(x)$ est l'équivalent, dans un ordre, du "crossing-number" de Rutovitz [Rut66] (voir section 2.5.2). Si $\mathcal{R}(x) = 2$, alors il existe une unique courbe ouverte simple dans $\alpha_S^{\square}(x)$, et le point x est α_2-simple pour $|S|$. Afin d'éviter d'effectuer le calcul de ce nombre dans l'algorithme, nous pouvons créer et utiliser une table de consultation ("lookup-table"), qui indique pour chaque configuration $\alpha^{\square}(x)$ (indice dans la table) si elle est α_2-simple ou non pour $|S|$. En fait, nous avons utilisé un graphe de décision binaire (BDD) [Bry86], [RM96], qui nous indique si un point est α_2-simple pour $|S|$, au pire en huit tests (présence ou non des huit voisins dans S) (cf. l'annexe C).

Notons $\#X$ le nombre d'éléments (ou cardinal) de l'ensemble X. Le carré x est extrémité de courbe si $\#\alpha_S^{\square}(x) = 1$. Dans le cas du squelette curviligne, nous pouvons supprimer un carré unité x si $\mathcal{R}(x) = 2$ et si $\#\alpha_S^{\square}(x) > 1$. De façon duale, soit x un singleton, x est β_2-simple pour $|S|$ ssi $R(x) = 2$ et x est extrémité de courbe si $\#\beta_S^{\square}(x) = 1$.

La figure 8.10 récapitule les conditions qu'un point doit vérifier pour être supprimé dans le cas du squelette ultime ou du squelette curviligne. En résumé, nous observons trois types d'éléments (sur quatre) pour toute sous-itération. Dans le cas du squelette ultime, nous accédons

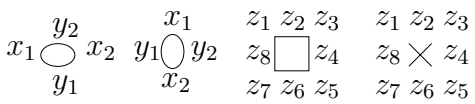


FIG. 8.9 – *Notations utilisées dans les voisinages.*

Squelette ultime	Squelette curviligne
PHASE_α	
On retire en parallèle les carrés x	
si $\mathcal{R}(x) = 2$	si $\mathcal{R}(x) = 2$ et $\#\alpha_S^{\square}(x) > 1$
et les ovales verticaux ou horizontaux	
si $x_1 + x_2 = 1$	si $x_1 + x_2 = 1$ et $y_1 + y_2 \geq 1$
PHASE_β	
On retire en parallèle les croix x	
si $\mathcal{R}(x) = 2$	si $\mathcal{R}(x) = 2$ et $\#\beta_S^{\square}(x) > 1$
et les ovales verticaux ou horizontaux	
si $y_1 + y_2 = 1$	si $y_1 + y_2 = 1$ et $x_1 + x_2 \geq 1$

FIG. 8.10 – *Détails pour les algorithmes d'amincissement et de squelettisation.*

à deux voisins pour les éléments ovales et au pire à huit voisins pour les carrés ou les singletons. Dans le cas du squelette curviligne, nous accédons au pire à quatre des voisins des éléments ovales et au pire à huit voisins pour les carrés ou singletons.

Remarque : Au chapitre 10 (section 10.3), nous introduirons des nombres appelés *nombres de connexité*, permettant une autre caractérisation des points α_2- ou β_2-simples.

8.7 Résultats et comparaisons

Nous rappelons d'abord différentes classes d'algorithmes parallèles de squelettisation (par suppression de points), en indiquant les noms de ceux choisis (dans le récapitulatif de Hall [Hal96], voir également l'annexe A), que nous comparons avec les nôtres. Nous appelons *support*, l'ensemble minimal de points auxquels un algorithme accède pour savoir si un point peut être détruit ou non (par convention, ce point appartient au support). Notre algorithme de squelet-

tisation curviligne est noté LB_Φ_1 ou LB_Φ_3, selon qu'il utilise la transformation Φ_1 ou Φ_3 (cf. section 8.2.2).

Les algorithmes parallèles de squelettisation dans $\mathcal{Z}^2$ peuvent être classés en plusieurs catégories :
- algorithmes fortement parallèles : en tout point est appliqué un même opérateur de réduction, ici d'un support de 11 points (GH_AFP2 [GH92], GH_AFP3 [GH92]) ;
- algorithmes à base de sous-itérations directionnelles : en tout point est appliqué un opérateur dépendant de l'itération courante, ici d'un support de 9 points (en 4 sous-itérations : ROS [RK82], en 2 sous-itérations : HSCPN [HSCP87],[Hal89], LW [ZS84],[LW86], GH_A1 [GH89], TSIN [Hal96]) ;
- algorithmes à base de sous-mailles : un seul opérateur est appliqué aux points d'une même sous-maille, ici d'un support de 9 points (avec 2 sous-mailles : GH_A2 [GH89]).

Les algorithmes mis en œuvre ici pour les comparaisons ont été détaillés dans l'annexe A (à l'exception de TSIN, voir [Hal96]).

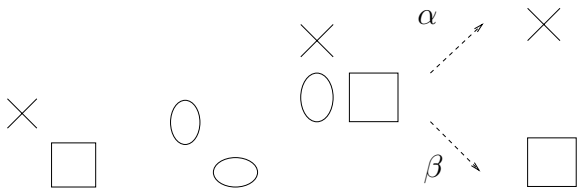
Notre algorithme de squelettisation curviligne est parallèle, de type deux sous-itérations, avec un support de 5 points (resp. 9 points) pour un ovale (resp. un carré (PHASE_α) ou un singleton (PHASE_β)). Dans la section 8.6, nous avons remarqué que 3 types de points sur 4 sont examinés lors de chaque sous-itération.

La figure 8.12 montre les résultats obtenus avec les différents algorithmes de squelettisation appliqués à deux images binaires. Pour chaque figure, nous avons indiqué l'algorithme utilisé ainsi que le numéro de la sous-itération de la dernière suppression. Les sous-itérations successives sont représentées par différents niveaux de gris.

8.7.1 Discussion sur Φ_1

Même si la transformation Φ_1, utilisée par l'algorithme LB_Φ_1, nécessite 4 fois plus de points que les autres algorithmes, l'algorithme LB_Φ_1 présente une complexité du même ordre que celle des autres algorithmes (en effet, le nombre de sous-itérations est proche de celui obtenu par les autres algorithmes ; il est de l'ordre du rayon de la boule maximale incluse dans l'objet [CM91]).

Les algorithmes classiques de squelettisation dépendent fortement de la première direction de suppression, ou de la première sous-maille choisie, ou d'une direction de préservation (par exemple, la bande supérieure d'un ruban horizontal d'épaisseur 2 est préservée), ou, dans le cadre de la morphologie mathématique, de l'ordre de rotation d'un élément structurant [JC90]. En revanche, notre algorithme n'accorde aucune place à l'arbitraire. En effet, avec la transformation Φ_1, l'objet initial ne comporte pas de point α_2-simple ; le résultat est alors indépendant de l'ordre des sous-itérations (de même avec l'utilisation de Φ_2, pour laquelle il n'y a pas initialement de point β_2-simple). Les propriétés intrinsèques de l'algorithme LB_Φ_1 ont pour conséquence une squelettisation homogène et la production de squelettes curvilignes minces et bien centrés. Remarquons également l'apparition de petites branches (partie supérieure de la lettre "A") qui proviennent de l'apparition rapide de points extrémités, cela étant dû à l'homogénéité de la suppression. Par ailleurs, elles témoignent de la forme convexe d'une région d'un objet.



(a) à la translation (b) à l'ordre de suppression

FIG. 8.11 – *Sensibilité de la transformation* Φ_3.

Considérons un objet de $\mathcal{Z}^2$, constitué de deux pixels partageant une même arête. Les algorithmes classiques d'amincissement doivent effectuer un choix afin de pouvoir supprimer l'un des deux points. Un algorithme "symétrique" (*c.-à-d.* qui réalise une squelettisation homogène) chercherait à garder une information entre ces deux pixels ; il faut alors faire intervenir un point intermédiaire (qui peut symboliser l'arête commune, par exemple). Un raisonnement similaire dans le cas du carré de 2×2 pixels, nous amène à garder un point "central". Nous comprenons ainsi qu'il est nécessaire de multiplier par 4 la taille de l'image initiale afin de pouvoir envisager une squelettisation homogène. Notre algorithme est donc optimal du point de vue de la taille mémoire de l'image, son objectif étant justement de produire des squelettes minces et bien centrés.

8.7.2 Discussion sur Φ_3

La transformation Φ_3 est sensible à la translation. En effet, considérons l'objet constitué de deux points en diagonale (Fig. 8.11(a)). Les points sont voisins dans la configuration de gauche, tandis qu'ils ne le sont pas dans la configuration de droite (translatée de celle de gauche), et seule la configuration de gauche sera modifiée par l'algorithme de squelettisation ultime (et réduite en un point).

Contrairement à l'algorithme précédent utilisant Φ_1, l'algorithme de squelettisation proposé ici est sensible à l'ordre des sous-itérations de suppression. En effet, considérons l'objet de la figure 8.11(b). Si la première sous-itération de suppression est de type PHASE_α, l'objet se réduit au singleton ; si elle est de type PHASE_β, l'objet se réduit au carré. Nous n'obtenons donc pas le même résultat selon la première sous-itération de suppression choisie. Remarquons que le nombre de sous-itérations utilisées par l'algorithme de squelettisation curviligne LB_Φ_3 est inférieur à celui des autres algorithmes sur les exemples présentés (Fig. 8.12). Néanmoins le squelette présente plus de petites branches parasites.

8.8 Conclusions et perspectives

Suite à l'étude détaillée de l'ordre $(H^2, \supseteq)$ associé à $\mathcal{Z}^2$, nous avons proposé une caractérisation simple et efficace des points α_2-simples, nous permettant de proposer des algorithmes parallèles de squelettisation ultime et curviligne. Ces algorithmes parallèles sont de type deux sous-itérations et ne nécessitent l'accès qu'à 3, 5 ou 9 points pour décider de la simplicité d'un point.

Par ailleurs, s'il utilise une certaine transformation (Φ_1), notre algorithme de squelettisation curviligne a la propriété remarquable de produire des squelettes minces et bien définis.

À notre connaissance, le seul algorithme permettant d'avoir un squelette (discret) mince et bien défini, est celui utilisant la notion de mailles dérivées [Ber84]. En fait, la structure d'ordre peut être vue comme un cadre permettant de formaliser et d'enrichir la notion de maille dérivée, et permettant notamment l'obtention d'un squelette ultime mince et bien défini.

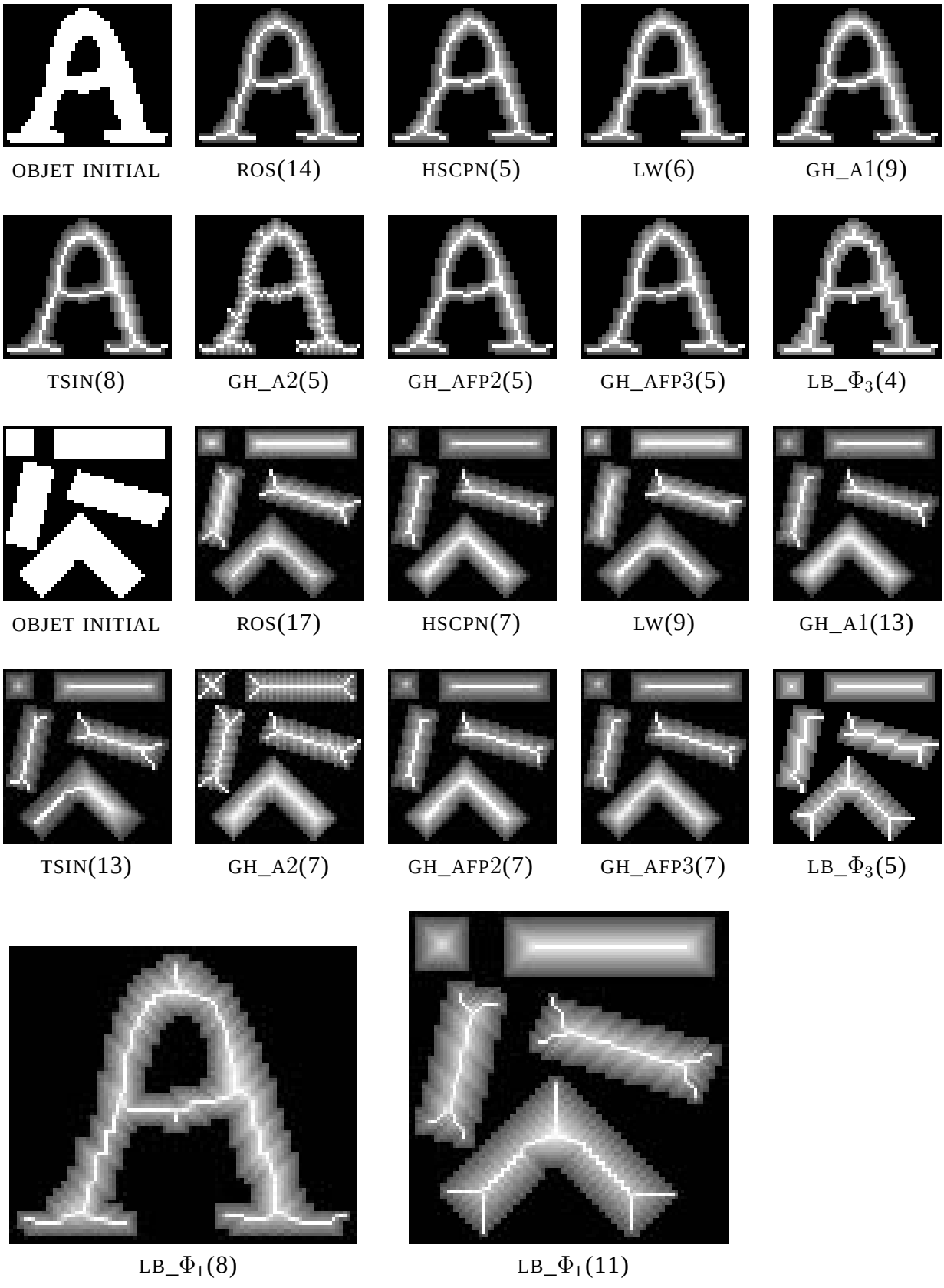


FIG. 8.12 – *Squelettes obtenus par différents algorithmes, le numéro de la dernière sous-itération de suppression est indiqué entre parenthèses.*

Chapitre 9

Ordre pour images binaires 3D

9.1 Introduction

Nous présentons l'étude d'algorithmes de squelettisation dans l'ordre de base associé à $\mathcal{Z}^3$ (voir également [LB99]).

9.2 Ordre associé à $\mathcal{Z}^3$

9.2.1 L'ordre $|H^3|$

Soit $\mathcal{Z}$ l'ensemble des entiers relatifs. Nous considérons les familles d'ensembles H_0^1, H_1^1 et H^1, telles que $H_0^1 = \{\{a\};\ \ a \in \mathcal{Z}\}$, $H_1^1 = \{\{a, a+1\};\ a \in \mathcal{Z}\}$ et $H^1 = H_0^1 \cup H_1^1$. Soit $m \in \{0, 1, 2, 3\}$; un sous-ensemble S de $\mathcal{Z}^3$ qui est le produit cartésien d'exactement m éléments de H_1^1 et de $(3-m)$ éléments de H_0^1 est appelé un m-*cube* de $\mathcal{Z}^3$. Soit H^3 l'ensemble composé de tous les m-cubes de $\mathcal{Z}^3$, $m \in \{0, \ldots, 3\}$. Un m-cube de $\mathcal{Z}^3$ est appelé un *singleton*,

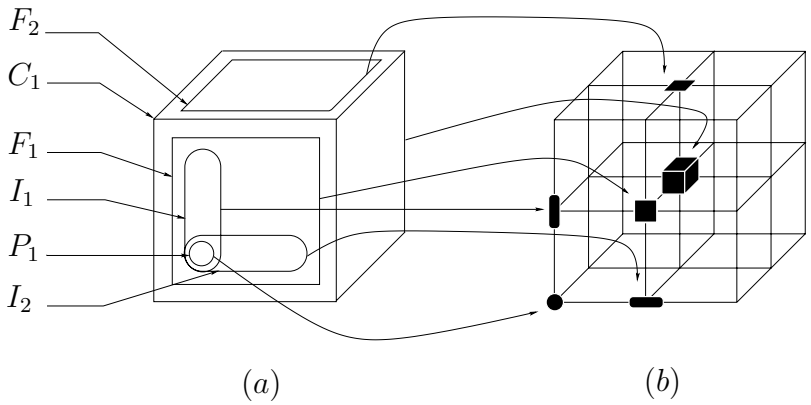


FIG. 9.1 – (a) *Un sous-ensemble* S *de* H^3 *et* (b) *sa représentation tableau.*

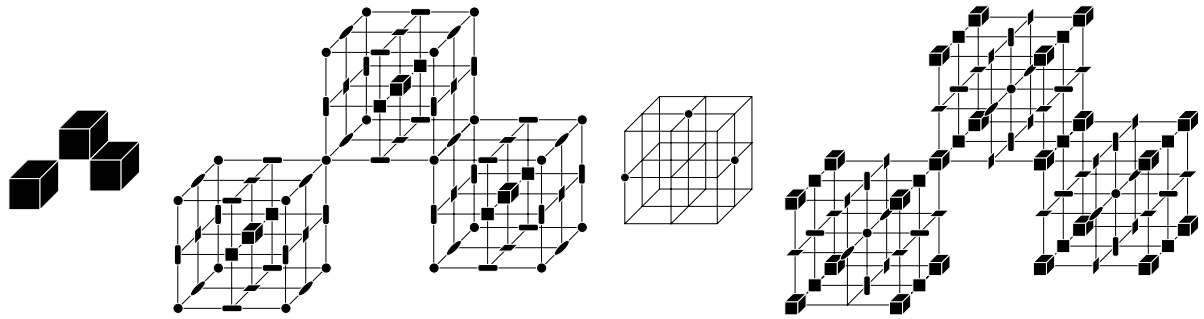


FIG. 9.2 – (a) *Un ensemble* S *de voxels,* (b) $\Phi_1(S)$*,* (c) *un sous-ensemble* S *de* $\mathcal{Z}^3$*,* (d) $\Phi_2(S)$*.*

un *intervalle unité,* un *carré unité* ou un *cube unité* pour $m = 0, 1, 2$ ou 3, respectivement. En fait, H^3 est une structure 3D qui est un complexe cellulaire [Kov89].

Notons $*\mathcal{Z} = \{a + 1/2;\ a \in \mathcal{Z}\}$ et $\mathcal{Z}_+ = \mathcal{Z} \cup *\mathcal{Z}$. Soit l'application ψ_1 de H^1 dans $\mathcal{Z}_+$ telle que pour tout x de H^1, $\psi_1(x) = a$ si $x = \{a\}$, $a \in \mathcal{Z}$ et $\psi_1(x) = a + 1/2$ si $x = \{a, a + 1\}$, $a \in \mathcal{Z}$. Considérons également l'application ψ_3 de H^3 dans $\mathcal{Z}_+^3$ telle que pour tout x de H^3, la i-ième coordonnée de $\psi_3(x)$ est l'image de la i-ième coordonnée de x par ψ_1, *i.e.* $[\psi_3(x)](i) = \psi_1[x(i)]$, avec $i \in \{1, 2, 3\}$. Si $S \subseteq H^3$, alors $\psi_3(S) = \{\psi_3(x), x \in S\}$ peut être vu comme une représentation de S, appelée *représentation tableau* [Ber99]. En fait, $\psi_3(H^3)$ est la grille digitale hétérogène qui est le produit de trois copies de la ligne de Khalimsky [KKM90] [KKM91] [Kop94].

Considérons que l'ordre associé à $\mathcal{Z}^3$ est l'ordre $|H^3| = (H^3, \alpha)$, avec $\alpha = \supseteq$, ainsi $y \in \alpha(x)$ si $x \supseteq y$.

La figure 9.1 (a) montre un sous-ensemble S de H^3. Le sous-ensemble S est composé d'un cube unité, de deux carrés unités, d'un intervalle unité horizontal, d'un intervalle unité vertical et d'un singleton. La figure 9.1 (b) montre le même sous-ensemble dans sa représentation tableau, avec les conventions suivantes, soit $x \in S$: si $\psi_3(x)$ a ses trois coordonnées dans $\mathcal{Z}$ (*i.e.* si x est un singleton), alors x sera représenté par un élément rond ; si $\psi_3(x)$ a ses trois coordonnées dans $*\mathcal{Z}$ (*i.e.* si x est un cube unité), alors x sera représenté par un élément cubique ; si $\psi_3(x)$ a exactement deux de ses trois coordonnées dans $\mathcal{Z}$ (*i.e.* si x est un intervalle unité), alors x sera représenté par un élément ovale et si $\psi_3(x)$ a exactement deux de ses trois coordonnées dans $*\mathcal{Z}$ (*i.e.* si x est un carré unité), alors x sera représenté par un élément carré.

9.2.2 Objets 3D

Nous allons maintenant proposer différentes façons de considérer des objets binaires 3D dans $|H^3|$. En analyse d'images 3D, un objet est souvent défini comme un ensemble de voxels. Rappelons [Kon97] qu'un voxel dans $\mathcal{R}^3$ est le produit cartésien de la forme $[i_1, i_1 + 1] \times [i_2, i_2 + 1] \times [i_3, i_3 + 1]$, avec $i_1,\ i_2,\ i_3 \in \mathcal{Z}$. Une image binaire tridimensionnelle, ou 3-image, est définie comme un ensemble fini de voxels de $\mathcal{R}^3$.

À chaque 3-image I est associée un sous-ensemble S de H^3 comme suit : à chaque voxel de I est associé le cube unité de H^3 qui est le produit cartésien de la forme $\{i_1, i_1+1\} \times \{i_2, i_2+1\} \times \{i_3, i_3+1\}$, ainsi que les éléments de H^3 qu'il contient (voir figures 9.2 (a) et (b)). Soit Φ_1 une telle transformation.

Une autre approche est de considérer un objet 3D comme un sous-ensemble de $\mathcal{Z}^3$. À chaque sous-ensemble I de $\mathcal{Z}^3$ est associé un sous-ensemble S de H^3 comme suit : à chaque point de I de coordonnées (i, j, k) dans $\mathcal{Z}^3$ est associé le singleton $\{(i, j, k)\}$ de H^3 ainsi que les éléments de H^3 qui contiennent ce singleton. (voir figures 9.2 (c) et (d)). Soit Φ_2 une telle transformation.

9.3 Simplicité et processus parallèle d'amincissement

Les définitions de point α_n-simple et de point β_n-simple sont récursives et croisées avec celle d'ordre n-contractile (voir section 7.5). Elles peuvent être implémentées par des algorithmes récursifs.

L'examen de l'α_1-simplicité d'un point x est directe, il est suffisant de tester si $\alpha^{\square}(x)$ est composé d'un unique point. Dans le but de vérifier l'α_n-simplicité d'un point x, avec $n > 1$, nous considérons $\alpha^{\square}(x)$ et nous procédons jusqu'à stabilité avec des étapes alternées de suppression parallèle de points α_{n-1}-simples (une telle étape est appelée une α-*étape*), et d'étapes de suppression parallèle de points β_{n-1}-simples (une telle étape est appelée une β-*étape*). Si, après cette procédure, $\alpha^{\square}(x)$ est constitué d'un seul élément, alors le point x est α_n-simple.

9.3.1 Caractérisation des points α_n-simples dans $|H^3|$

Dans cette section, nous donnons quelques remarques qui permettent de tester plus efficacement l'α_n-simplicité dans $|H^3|$. En fait, ces remarques permettent d'avoir une caractérisation non récursive des points α_n-simples dans $|H^3|$.

Tout d'abord, nous proposons deux propriétés obtenues directement de la définition récursive d'un point α_n-simple et de celle d'ordre n-contractile (section 7.5).

Propriétés : Si un point est α_n-simple, alors il est α_p-simple, avec $p \geq n > 0$. Si un ordre est n-contractile, alors il est p-contractile, avec $p \geq n \geq 0$.

Dans $|H^3|$, il est suffisant de tester l'α_3-simplicité d'un point. Supposons qu'un point x est α_p-simple, les propriétés précédentes nous amènent à chercher le nombre n minimal à partir duquel x est α_n-simple, avec $0 < n \leq p \leq 3$.

Soit $S \subseteq H^3$. En d'autres termes, pour un type donné de point x de S, nous voulons connaître le nombre n pour lequel nous pouvons décider si x est α_3-simple pour $|S|$, $n \leq 3$. Dans la suite, nous émettons quelques remarques qui permettent de déterminer un tel nombre et d'améliorer ainsi le test de l'α_3-simplicité. Par dualité, nous avons les mêmes remarques pour la β_3-simplicité avec les éléments appropriés, par exemple nous avons les mêmes remarques pour l'α_3-simplicité d'un carré unité x et pour la β_3-simplicité d'un intervalle unité y, car $\alpha^{\square}_{H^3}(x)$ a la même forme que $\beta^{\square}_{H^3}(y)$.

Soit $S \subseteq H^3$.

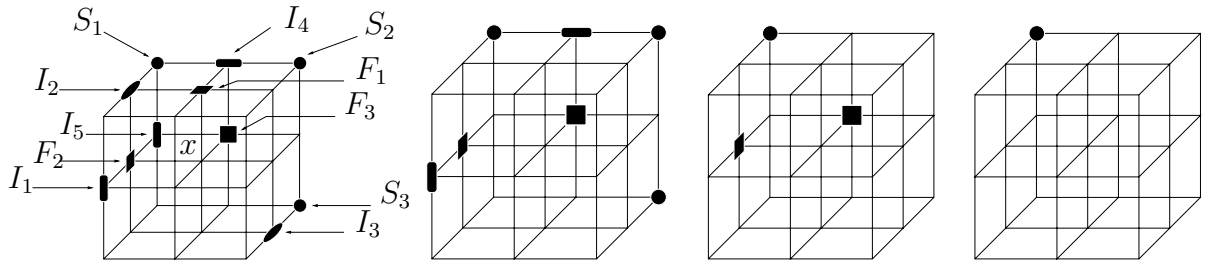
1. Un singleton x est tel que $\alpha^{\square}(x) = \emptyset$. Cet élément n'est jamais α_n-simple, $\forall n > 0$.
2. Un intervalle unité x est α_3-simple pour $|S|$ si et seulement si x est α_1-simple pour $|S|$, *i.e.* si $\alpha_S^{\square}(x)$ est composé d'un seul élément. Nous devons seulement tester l'α_1-simplicité de ce type d'élément. De plus $\alpha_{H^3}^{\square}(x) = \alpha_{H^2}^{\square}(x) = \alpha_{H^1}^{\square}(x)$ et x est α_1-simple pour $|S|$ ssi $\alpha_S^{\square}(x)$ est composé d'un unique singleton.
3. Soit x un carré unité de S. Si $\alpha_S^{\square}(x)$ n'est composé que d'un seul élément alors x est α_1-simple pour $|S|$. Autrement, nous devons tester l'α_2-simplicité de x. En fait, si x est un carré unité, nous avons la propriété suivante : x est α_2-simple si et seulement si $\alpha_S^{\square}(x)$ est une courbe ouverte simple (nous rappelons qu'une *courbe ouverte simple* est un ensemble connexe de points X, tel que tous les points x dans X ont exactement deux éléments dans $\theta^{\square}(x)$ (les *points de courbe*), sauf deux points x dans X qui n'ont qu'un seul élément dans $\theta^{\square}(x)$ (les *points terminaux de courbe*)). En effet, nous pouvons montrer que dans ce cas, et uniquement dans celui-ci, nous pouvons supprimer itérativement les intervalles unités α_1-simples pour $|T|$ et les singletons β_1-simples pour $|T|$ avec $T = \alpha_S^{\square}(x)$, et obtenir un point isolé de T (cf. la propriété 11 dans la section 8.4.2). Notons que les résultats obtenus dans $|H^2|$ sont valables ici car nous avons $\alpha_{H^2}^{\square}(x) = \alpha_{H^3}^{\square}(x)$ pour un carré unité x.
4. Soit x un cube unité de S. De la même façon, si $\alpha_S^{\square}(x)$ n'est composé que d'un seul élément alors x est α_1-simple pour $|S|$. Autrement, nous devons tester l'α_3-simplicité du cube unité. Dans ce cas, nous devons tester l'α_2-simplicité du ou des carrés unités existants qui doivent être supprimés de $\alpha^{\square}(x)$ lors d'une α-étape ; tester l'α_1-simplicité ou la β_1-simplicité du ou des intervalles unités lors d'une α-étape ou d'une β-étape respectivement, s'il y en a ; et la β_2-simplicité du ou des singletons lors d'une β-étape, s'il y en a.

Le test de l'α_3-simplicité d'un intervalle unité ou d'un carré unité est directe. Seul le test d'un cube unité est plus compliqué. Les remarques précédentes permettent de tester l'α_3-simplicité d'un cube unité, plus rapidement. De plus, nous pouvons stocker toutes les configurations d'un voisinage produisant des points α_3-simples. Comme le nombre de ces configurations est important, nous utilisons les graphes de décision binaire (BDD) pour les stocker [Bry86, BRB90] (cf. l'annexe C). Nous pouvons alors décider en au plus 26 tests (correspondant au nombre de points nécessaires pour décrire une configuration) [RM96], si un point est α_3-simple ou non (voir l'annexe C et la section 9.6).

9.3.2 Illustration de l'α_3-simplicité

Nous vérifions l'α_3-simplicité d'un cube unité x appartenant au sous-ensemble S de H^3, représenté dans la figure 9.3. Nous avons représenté $\alpha_S^{\square}(x)$ dans la figure 9.3(a), les éléments de S sont représentés en noir. Nous utilisons les remarques précédentes lors du test de l'α_3-simplicité de x pour $|S|$.

À la première étape, nous supprimons les points α_2-simples pour $|S|$. Les éléments I_2, I_3 et I_5 contiennent un unique élément : S_1, S_3 et S_1, respectivement. Ainsi, ils sont α_1-simples pour $|S|$ (donc α_2-simples, comme nous l'avons dit précédemment). L'élément F_1 est α_2-simple



(a) $\alpha_S^{\square}(x)$

(b) Après la première α-étape.

(c) Puis, après la première β-étape.

(d) Puis, après la seconde α-étape.

FIG. 9.3 – *Test de l'α_3-simplicité de x.*

pour $|S|$ car $\alpha_S^{\square}(F_1)$ est composé d'une courbe ouverte simple que nous pouvons décrire par la séquence : $I_2,\ S_1,\ I_4,\ S_2$. Notons que F_3 n'est pas α_2-simple pour $|S|$ car $\alpha_S^{\square}(F_3)$ n'est pas une courbe ouverte simple. Nous pouvons voir que les autres éléments ne sont pas α_2-simples pour $|S|$. Ainsi, les éléments I_2, I_3, I_5 et F_1 sont supprimés en parallèle lors de la première α-étape (Fig. 9.3(b)). Soit $|T|$, l'ordre relatif à $S \setminus \{I_2, I_3, I_5, F_1\}$. Lors de la deuxième sous-itération (*i.e.* la première β-étape), nous supprimons les points β_2-simples pour $|T|$. Les éléments I_1, I_4 et S_3 sont β_1-simples pour $|T|$. L'élément S_2 est β_2-simple pour $|T|$ car $\beta_T^{\square}(S_2)$ est une courbe ouverte simple (la séquence $I_4,\ F_3$). Nous remarquons que les autres éléments ne sont pas β_2-simples pour $|T|$. Ainsi, nous supprimons en parallèle les éléments I_1, I_4, S_3 et S_2 et obtenons l'ensemble de la figure 9.3(c). Après la deuxième α-étape, il ne reste qu'un seul élément S_1 (Fig. 9.3(d)). L'ordre $|\alpha_S^{\square}(x)|$ est ainsi 2-contractile ; le point x est alors α_3-simple pour $|S|$.

9.4 Raisonnement menant à la notion de point α_n-simple

Nous rappelons d'abord la notion de point α-unipolaire (proposée dans un cadre général à la section 7.3). Un point α-unipolaire peut être vu comme un "point inessentiel pour la topologie". Nous proposons un processus d'amincissement basé sur la suppression de points α-unipolaires. Nous montrons qu'un tel processus "n'amincit pas suffisamment" certaines images. C'est la raison pour laquelle les points α_n-simples ont été introduits [Ber99]. Cette section nécessite quelques propriétés 2D (données à la section 8.4) afin de mieux comprendre la récursivité mise en jeu dans la définition de point α_n-simple.

Définition : Soit $|X| = (X, \alpha)$ un ordre, soient x et y deux points de X. Soit $\alpha^{\bullet}$, la relation sur X telle que $y \in \alpha^{\bullet}(x)$ si et seulement si $y \in \alpha^{\square}(x)$ et $\alpha^{\square}(x) \cap \beta^{\square}(y) = \emptyset$. Un point x de X est α-*unipolaire* si $\alpha^{\bullet}(x)$ consiste en un seul point. Un élément x est α-*libre* si x est α-unipolaire ou s'il existe une séquence $x_0, \ldots, x_k$, avec $x_k = x$ telle que x_0 est α-unipolaire et x_i est α_i-unipolaire avec $\alpha_i = \alpha \cap (S_i \times S_i),\ S_i = X \setminus \{x_0, \ldots, x_{i-1}\}$, $i = 1, \ldots, k$. Un point qui n'est pas α-libre est appelé un α-*lien*. L'α-*noyau* de X est le sous-ensemble de X composé de tous les

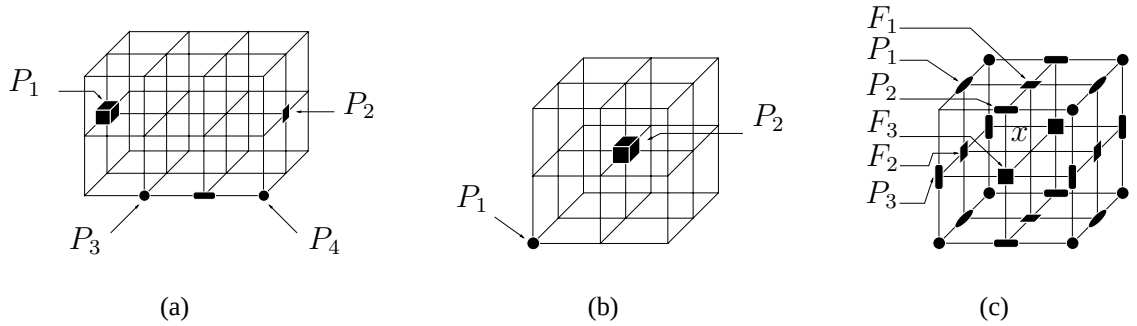


FIG. 9.4 – *Figures utilisées dans la section 9.4.*

α-liens de X.

Rappelons que l'ordre de base associé à $\mathcal{Z}^3$ est l'ordre $|H^3| = (H^3, \alpha)$, avec $\alpha = \supseteq$. Soit $S \subseteq H^3$, et soient x et y deux points de l'ordre (S, α). Soulignons que $y \in \alpha^\bullet(x)$ si et seulement si $y \subset x$ et s'il n'y a pas de point z de S tel que $y \subset z \subset x$. Considérons le sous-ensemble S décrit à la figure 9.1. Les éléments F_2 et P_1 ne contiennent aucun élément ($\alpha^\square(F_2) = \alpha^\square(P_1) = \emptyset$), ainsi F_2 et P_1 ne sont pas α-unipolaires, F_2 et P_1 sont α-terminaux. L'élément I_1 contient un unique élément P_1, ainsi I_1 est α-unipolaire, de la même façon I_2 est α-unipolaire.

L'élément F_1 n'est pas α-unipolaire car $\alpha^\bullet(F_1) = \{I_1, I_2\}$. Nous observons que P_1 n'appartient pas à $\alpha^\bullet(F_1)$ à cause de l'élément I_1, car par exemple, $P_1 \subset I_1 \subset F_1$. Nous avons $\alpha^\bullet_T(F_1) = \{I_2\}$ avec $T = S \setminus \{I_1\}$, ainsi F_1 est α_T-unipolaire ; en d'autres termes "F_1 est α-unipolaire après la suppression de I_1". Comme I_1 est α-unipolaire, cela signifie que F_1 est un point α-libre. Nous pouvons voir que seuls I_1, I_2 et F_1 sont des points α-libres. Ainsi, l'α-noyau de S est composé des éléments F_2, C_1 et P_1.

Observons que la définition d'un point α-libre conduit à un processus d'amincissement dans lequel les points α-unipolaires sont retirés itérativement. Considérons la figure 9.4(a). Les points P_1 et P_2 sont α-unipolaires. Si nous les supprimons, il n'y a plus de point α-unipolaire. L'unique composante connexe n'est pas réduite en un seul point avec un tel processus d'amincissement consistant à supprimer des points α-unipolaires. Les points P_3 et P_4 sont β-unipolaires. Considérons maintenant la suppression des points α- et β-unipolaires. Cela nous amène à la définition suivante. Soit $|X| = (X, \alpha)$ un ordre. Nous disons que $|X|$ est *contractile* s'il peut être réduit en un seul point par suppression itérative de points α- ou β-unipolaires (cf. la définition 4, page 169). Notons que nous devons tester toutes les possibilités d'arranger les suppressions de points α- ou β-unipolaires, afin de décider si un ordre est contractile ou non.

Rappelons la propriété donnée dans [Ber99] (voir section 7.3) concernant la suppression parallèle de points α-unipolaires.

Propriété : Soient $|X| = (X, \alpha)$ un ordre, y un point α-unipolaire, et x un point de X, avec $x \neq y$. Si x est α-unipolaire, alors x est α_S-unipolaire avec $S = X \setminus \{y\}$.

Nous avons alors la propriété remarquable de pouvoir supprimer en parallèle les points α-

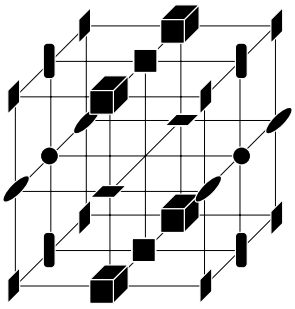


FIG. 9.5 – *Représentation partielle de H^3 moins un intervalle.*

unipolaires d'un ordre (sous-entendu, en préservant ses propriétés topologiques). Par dualité, nous pouvons supprimer en parallèle les points β-unipolaires d'un ordre. Notons également que la suppression simultanée de points α- et β-unipolaires ne garantit pas la preservation de la topologie de l'objet. En effet, à la figure 9.4(b), le point P_1 est β-unipolaire, le point P_2 est α-unipolaire ; la suppression parallèle des points α- et β-unipolaires supprime l'objet composé de ces deux points. Afin de préserver la topologie, nous devons alterner une étape de suppression parallèle de points α-unipolaires et une autre de suppression parallèle de points β-unipolaires.

Cependant, dans l'ordre relatif à H^3 moins un élément intervalle (représenté partiellement Fig. 9.5, en fait, cet objet comporte une cavité), il n'y a pas de point α- ou β-unipolaire. Ainsi, il n'est pas possible d'amincir cet objet avec un processus supprimant des points α- ou β-unipolaires : la notion de point α-unipolaire est limitée pour l'amincissement. Par conséquent, nous allons chercher une notion plus puissante de simplicité. Nous avions déjà tenu un tel raisonnement dans le cas 2D, voir Fig. 8.6(c).

Une première idée pour surmonter ce problème est de considérer un point x d'un ordre $|X|$ comme α-*simple* pour $|X|$ si $|\alpha^{\square}(x)|$ est contractile. Dans le dernier exemple (H^3 moins un élément intervalle), dans le voisinage de l'élément intervalle absent, les carrés sont α-simples et les singletons sont β-simples ; ainsi nous pouvons amincir cet objet. Nous pouvons les supprimer, et le processus de suppression continue. Mais, même cette notion de simplicité n'est pas suffisamment puissante (alors qu'elle l'est dans le cas 2D, voir la section 8.4.2).

Soient un élément cube unité x et un ensemble S tel que $\alpha_S^{\square}(x)$ est représenté à la figure 9.4(c) ; en outre, nous considérons que $S \setminus \{x\}$ est constitué seulement des points de cette même figure (en fait, S est $\alpha_{H^3}(x)$ moins un singleton). Nous testons si x est α-simple pour $|S|$. Nous pouvons supprimer les trois points α-unipolaires P_1, P_2 et P_3. Après ces suppressions, il n'y a plus de point α- ou β-unipolaire. Ainsi, le point x n'est pas α-simple selon la définition proposée ci-avant. Mais $\alpha_S^{\square}(x)$ est comme une surface ouverte et x devrait être considéré comme inessentiel du point de vue de la topologie.

Dans la figure 9.4(c), nous pouvons constater que les carrés F_1, F_2 et F_3 sont α_2-simples pour $|S|$; de plus $\alpha_S^{\square}(x)$ se réduit en un point par la répétition jusqu'à stabilité de suppressions alternées de points α_2-simples ou β_2-simples : $|\alpha_S^{\square}(x)|$ est 2-contractile (voir la définition 8 dans la section 8.4.2). Notons que x n'est pas α_2-simple pour $|S|$ car $|\alpha_S^{\square}(x)|$ n'est pas 1-contractile. Il nous faut une notion plus puissante pour la suppression de points.

De même que nous l'avions fait dans la section 8.4.2, nous pouvons alors définir la notion de point α_3-simple :

Définition (point α_3-simple) : Soit $|X| = (X, \alpha, \beta)$ un ordre dénombrable, localement fini et non vide. Un point x est α_3-*simple* pour $|X|$ si $|\alpha^{\square}(x)|$ est 2-contractile.

Le cube unité x pour lequel $\alpha_S^{\square}(x)$ est représenté à la figure 9.4(c) est α_3-simple pour $|S|$.

9.5 Algorithmes de squelettisation

Nous pouvons supprimer en parallèle les points α_3-simples d'un sous-ensemble de H^3 tout en préservant la topologie (cf. [Ber99]). Ainsi, nous proposons un processus de squelettisation ultime (voir la terminologie dans la section 8.5.2) qui consiste en la répétition, jusqu'à stabilité, d'une suppression en parallèle de points α_3-simples (une telle étape est appelée une α-*sous-itération*) suivie d'une suppression en parallèle des points β_3-simples (une telle étape est appelée une β-*sous-itération*). Dans un processus de squelettisation, nous voulons préserver certaines propriétés géométriques. Certains points simples doivent être gardés durant le processus de suppression. Ces points sont appelés *points terminaux du squelette*.

Si durant le processus, nous gardons des points terminaux de courbe, alors nous obtenons un *squelette curviligne*. Si, de plus, nous gardons les points terminaux de surface (définis ci-après), alors nous obtenons un *squelette surfacique*.

Soit S un sous-ensemble de H^3. Soit S_α, l'ensemble composé de tous les points α_3-simples pour $|S|$, *i.e.* $S_\alpha = \{x \in S; x$ est un point α_3-simple pour $|S|\}$. Dans la suite, nous utilisons S^0 pour noter le sous-ensemble initial S.

9.5.1 Squelette ultime

Nous proposons un algorithme de squelettisation ultime consistant à retirer en parallèle des points α_3-simples, puis à retirer en parallèle des points β_3-simples, et cela jusqu'à stabilité. Ce processus est décrit par l'algorithme 11.

$S^0 = S$

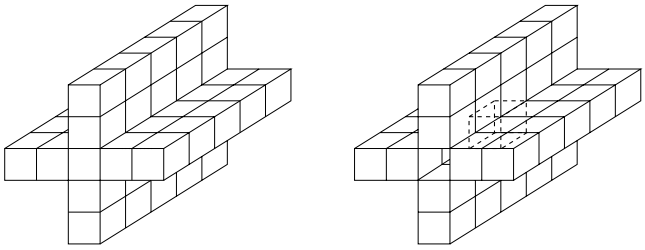
Répéter

$$\begin{cases} S^{2i+1} &= S^{2i} \setminus S_\alpha^{2i}, \\ S^{2i+2} &= S^{2i+1} \setminus S_\beta^{2i+1}. \end{cases}$$

Jusqu'à ce qu'il n'y ait plus de suppression durant 2 sous-itérations successives.

Algorithme 11: *Algorithme de squelettisation ultime dans $|H^3|$.*

Nous obtenons un *squelette ultime*, *i.e.* un ensemble qui ne contient aucun point α_3- ou β_3-simple.



(a) (b)

FIG. 9.6 – *Deux surfaces s'intersectant.*

9.5.2 Squelette curviligne

Dans la section 9.3, nous avons rappelé la définition d'une courbe ouverte simple. Les points simples d'une courbe ouverte simple sont les points extrémités de cette courbe. Si nous préservons ces points durant le processus de suppression, nous obtenons un squelette curviligne. Soit S un sous-ensemble de H^3. Un point x de S est un *point terminal de courbe* si $\theta_S^{\square}(x)$ est composé d'un unique point. Soit S_{ptc}, l'ensemble des points terminaux de courbe du sous-ensemble S.

Nous proposons un algorithme de squelettisation curviligne consistant à retirer en parallèle des points α_3-simples et non terminaux de courbe, puis à retirer en parallèle des points β_3-simples et non terminaux de courbe, et cela jusqu'à stabilité. Ce processus est décrit par l'algorithme 12.

$S^0 = S$

Répéter

$$\begin{cases} S^{2i+1} &= S^{2i} \setminus [S^{2i}_{\alpha} \cap \overline{S^{2i}_{ptc}}], \\ S^{2i+2} &= S^{2i+1} \setminus [S^{2i+1}_{\beta} \cap \overline{S^{2i+1}_{ptc}}]. \end{cases}$$

Jusqu'à ce qu'il n'y ait plus de suppression durant 2 sous-itérations successives.

Algorithme 12: *Algorithme de squelettisation curviligne dans* $|H^3|$.

Nous obtenons ainsi un *squelette curviligne*, *i.e.* un ensemble qui ne contient aucun point α_3-simple ou β_3-simple non extrémité de courbe.

9.5.3 Squelette surfacique

Nous voulons maintenant préserver des courbes et des surfaces durant le processus d'amincissement. Différentes caractérisations de points terminaux de surface ont déjà été proposées dans $\mathcal{Z}^3$: l'une d'entre elles utilise une formule booléenne et compte le nombre de points dans un octant [GB90], une autre teste de plus la suppressibilité dans des plans d'intersection [TF81]. Il existe des problèmes lorsqu'apparaissent des jonctions de surface, comme l'a montré C. Pudney [Pud98]. Par exemple, considérons l'objet (Fig. 9.6 (a)), composé de deux plans s'intersectant.

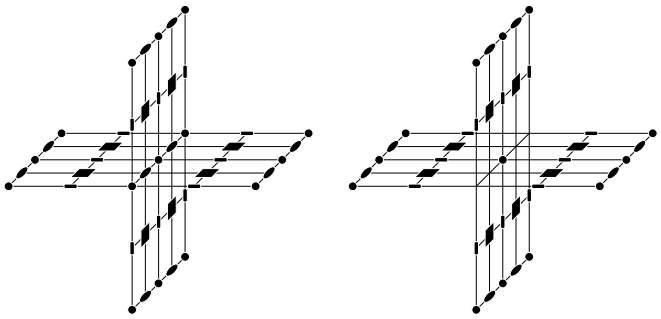


FIG. 9.7 – *Un contre-exemple pour notre première caractérisation de point terminal de surface.*

Les points simples qui ne sont pas détectés comme points terminaux de surface avec les caractérisations précédentes sont les deux points terminaux de la ligne d'intersection des deux plans. Si nous les supprimons, les nouveaux points simples qui ne sont pas terminaux de surface sont les deux voisins des deux points supprimés précédemment. Nous les supprimons et nous répétons ce processus. Finalement, la ligne d'intersection est supprimée sauf un de ses points (dans son centre, le cube en pointillé de la figure 9.6 (b)) qui n'est pas simple (ou deux points, selon la parité de la longueur des plans). La jonction est partiellement détruite. Il est parfois souhaitable de préserver de telles jonctions de surface. Une caractérisation plus récente [Pud98] cherche une correspondance du voisinage d'un point testé avec différentes intersections de plans. Mais un nouveau problème apparaît : des coins ne sont pas supprimés, ainsi le résultat doit être filtré [Pud98].

Nous proposons dans ce cadre d'ordre, une première approche pour un point terminal de surface.

Définition : Soit $|X| = (X, \alpha)$ un ordre. Un point x de X est dit être un *point de bord de surface* si $\theta_X^{\square}(x)$ est une courbe ouverte simple dont la longueur est strictement plus grande que 1.

Si nous considérons un point terminal de surface comme étant soit un point terminal de courbe soit un point de bord de surface, nous ne préservons pas certaines jonctions de surfaces, comme pour les caractérisations ci-avant. La figure 9.7 (a) montre l'objet initial (qui peut être considéré comme un équivalent de H^3 de l'objet montré à la figure 9.6 (a)), et la figure 9.7 (b) montre le squelette surfacique selon la caractérisation ci-avant (il est équivalent à l'objet montré à la figure 9.6 (b)). La jonction est partiellement détruite.

Un point de bord de surface x est tel que $\theta_X^{\square}(x)$ est une courbe ouverte simple. Ainsi, si un point x est une extrémité de courbe à l'intersection de surfaces, alors on peut s'attendre à ce que $\theta_X^{\square}(x)$ soit composé de plusieurs courbes ouvertes simples, ce qui constitue un "arbre".

Nous proposons les caractérisations suivantes d'un arbre et d'un point terminal de surface.

Définition : Soit $|X| = (X, \alpha)$ un ordre. L'ensemble X est dit être un *arbre* s'il peut être réduit en un seul point par suppression itérative de points terminaux de courbe. Soit x un point

de X. Le point x est dit être un *point terminal de surface* si $\theta_X^{\square}(x)$ est un arbre. Notons que les points terminaux de courbe et les points de bord de surface sont des points terminaux de surface.

Soit S_{pts} l'ensemble composé de tous les points terminaux de surface du sous-ensemble S, *i.e.* $S_{pts} = \{x \in S;\ \theta_S^{\square}(x)$ est un arbre$\}$.

Nous proposons un algorithme de squelettisation surfacique consistant à retirer en parallèle des points α_3-simples et non terminaux de surface, puis à retirer en parallèle des points β_3-simples et non terminaux de surface, et cela jusqu'à stabilité. Ce processus est décrit par l'algorithme 13.

$S^0 = S$

Répéter

$$\begin{cases} S^{2i+1} &= S^{2i} \setminus [S_\alpha^{2i} \cap \overline{S_{pts}^{2i}}], \\ S^{2i+2} &= S^{2i+1} \setminus [S_\beta^{2i+1} \cap \overline{S_{pts}^{2i+1}}]. \end{cases}$$

Jusqu'à ce qu'il n'y ait plus de suppression durant 2 sous-itérations successives.

Algorithme 13: *Algorithme de squelettisation surfacique dans* $|H^3|$.

Nous obtenons ici un *squelette surfacique, i.e.* un ensemble qui ne contient aucun point α_3-simple ou β_3-simple non terminal de surface.

Avec cette caractérisation, la jonction du précédent objet décrit à la figure 9.7 (a) est entièrement préservée dans le squelette surfacique.

9.6 Comparaisons et résultats

Dans $\mathcal{Z}^3$, différentes stratégies d'algorithmes de squelettisation ont été proposées. Si nous supprimons itérativement des points simples selon un balayage vidéo de l'image, nous obtenons un squelette mal centré. Dans $\mathcal{Z}^2$, la plupart des algorithmes parallèles utilise une stratégie directionnelle ($Nord$, Sud, Est, $Ouest$), consistant à supprimer les points simples et ayant le voisin selon la direction considérée appartenant au complémentaire (voir section A.3.4). Mais, de tels algorithmes parallèles directionnels, étendus dans $\mathcal{Z}^3$, ne garantissent pas la préservation de la topologie. Considérons un ruban de deux voxels d'épaisseur. Tous les points d'un tel objet sont simples, et il existe une direction selon laquelle tous ces points ont leur voisin selon cette direction, qui appartient au complémentaire ; le ruban disparaît lors d'une suppression en parallèle des points simples voisins d'un point du complémentaire le long de cette direction (cf. section 4.3). Il faut alors supprimer un sous-ensemble de l'ensemble des points simples d'un objet (cf. section B.2). Une autre stratégie utilise des sous-mailles, mais produit souvent des squelettes dentelés (cf. section B.3.1). Une stratégie hybride qui utilise à la fois les approches directionnelles et sous-mailles a aussi été proposée [PK98b] (cf. section B.3.2). Une autre approche utilise la transformation de distance, permettant un processus parallèle et l'obtention de résultats bien centrés [Pud98]. Certains auteurs ont étudié les ensembles minimaux de points simples interconnectés,

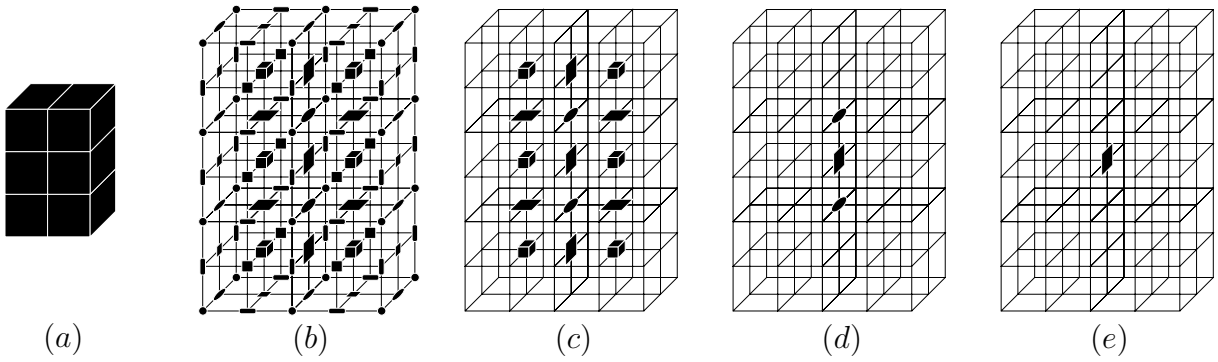


FIG. 9.8 – *Suppressions successives dans un ruban de* 2×3 *éléments.*

pour lesquels la suppression parallèle ne préservait pas la topologie. Ces ensembles sont appelés ensembles minimaux non simples (MNS) [Hal92] [Kon95] [Ron86] [Ron88] [Ma94] [Ma93]. Dans le but d'obtenir un squelette d'épaisseur 1, il faut supprimer certains points d'un MNS et imposer des contraintes directionnelles additionnelles dans le but de choisir les points simples d'un MNS qui doivent être éliminés [Ma95] [MS96]. Ce qui amène à considérer des conditions directionnelles à l'intérieur même d'un algorithme fortement parallèle (cf. sections B.4.1 et B.4.2). Une autre technique consiste à considérer un ensemble P de points qui sont candidats à une suppression parallèle. C'est la stratégie des points P-simples [Ber95a]. L'utilisation de cette technique nous a permis de construire des algorithmes utilisant la stratégie directionnelle ; les résultats dépendent alors de l'ordre choisi des directions selon lequel les points vont être éliminés (cf. chapitre 6).

En fait, il est très difficile d'avoir un algorithme simple parallèle qui préserve la topologie, qui produit des squelettes minces et bien centrés, obtenus par un processus homogène de suppression sans condition directionnelle, et n'utilisant qu'un support de petite taille afin de déterminer si un point est supprimable ou non. Avec l'approche topologie discrète, nous avons proposé de tels algorithmes, simples, parallèles, qui préservent la topologie, utilisant au plus un voisinage $3 \times 3 \times 3$ pour déterminer si un point est supprimable ou non. Les squelettes sont minces (l'algorithme se termine lorsqu'il n'y a plus de point simple et non extrémité). Ils sont bien centrés, à condition d'adopter une transformation multipliant la taille de l'image initiale.

Illustrons nos algorithmes de squelettisation sur un simple exemple. Considérons l'objet composé par six voxels décrivant un ruban 2×3 (Fig. 9.8 (a)). Nous avons représenté l'ensemble associé dans H^3 (Fig. 9.8 (b)), selon la première transformation Φ_1 décrite dans la section 9.2.2. À la première α-sous-itération, aucun point n'est supprimé. À la première β-sous-itération, la couche entière externe est supprimée (Fig. 9.8 (c)). Les points simples restants ont une courbe ouverte simple dans leur voisinage. Ainsi, nous avons obtenu un squelette surfacique. Après une nouvelle α-sous-itération, nous obtenons un squelette curviligne, montré à la figure 9.8 (d). Après une nouvelle β-sous-itération, nous obtenons un squelette ultime, montré à la figure 9.8 (e). Le résultat peut être considéré comme le "centre" du ruban initial, *i.e.* la face entre les deux "voxels du milieu" du ruban. Nous pouvons voir sur cet exemple, que le processus de suppression

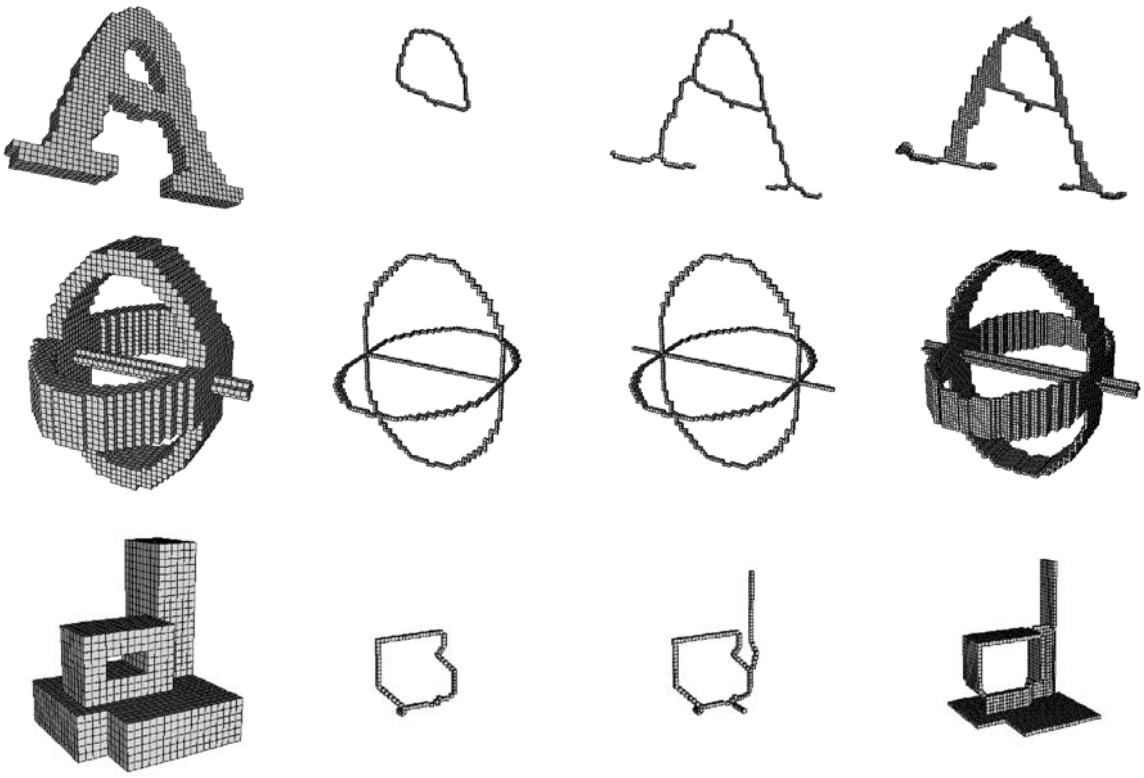
est homogène et que les résultats sont bien centrés.

Nous donnons quelques résultats sur d'autres objets. Un des objets de la figure 9.9 est inspiré de [SCM97]. La figure 9.10 montre un objet inspiré de [Pud98]. Nous pouvons voir de "jolis" morceaux de surface dans les squelettes surfaciques pour les objets des figures 9.9 et 9.10. Nous pouvons comparer avec ceux obtenus dans le cadre utilisant l'approche topologie digitale, à l'annexe B. La figure 9.11 montre des objets inspirés de [PK97a] [PK98a]. Sauf pour le dernier exemple dans cette figure, le squelette ultime, le squelette curviligne et le squelette surfacique sont les mêmes. Nous pouvons voir que tous les résultats sont bien centrés.

Remarques (à propos de la complexité des algorithmes) : Nous avons indiqué que nous avions une caractérisation rapide de points α_3- ou β_3-simples, avec l'utilisation de diagrammes de décision binaire (BDD). De plus, le nombre de points à considérer durant le test de la simplicité d'un point diffère d'un point à un autre. Par exemple, le test de l'α_3-simplicité d'un carré unité x exige l'accès à au plus huit points dans $\alpha^{\square}(x)$, le test de la β_3-simplicité d'un carré unité x exige l'accès à au plus deux points dans $\beta^{\square}(x)$. En fait, nous pouvons seulement stocker les configurations α_3-simples d'un cube unité dans un BDD (cf. section C.3.4.4), et nous testons l'α_3-simplicité des autres types d'éléments à l'aide des remarques données dans la section 9.3. De plus, dans une α-sous-itération, il est inutile de tester l'α_3-simplicité d'un singleton car cet élément n'est jamais α_3-simple. Par dualité, nous avons la même remarque pour un cube unité pendant une β-sous-itération.

9.7 Conclusion

Nous avons présenté des algorithmes parallèles de squelettisation ultime, curviligne et surfacique. Ces algorithmes utilisent un support de $3 \times 3 \times 3$ points, et sont de type deux sous-itérations ; celles-là n'utilisent pas de direction ni de sous-mailles. Nous avons proposé une nouvelle caractérisation de point terminal de surface. Les squelettes surfaciques obtenus sont minces et bien centrés.



Objet initial | Squelette ultime | Squelette curviligne | Squelette surfacique

FIG. 9.9 – *Différentes squelettisations.*

Objet initial | Squelette curviligne | Squelette surfacique

FIG. 9.10 – *Squelettisations sur un objet composé de trois “L”.*

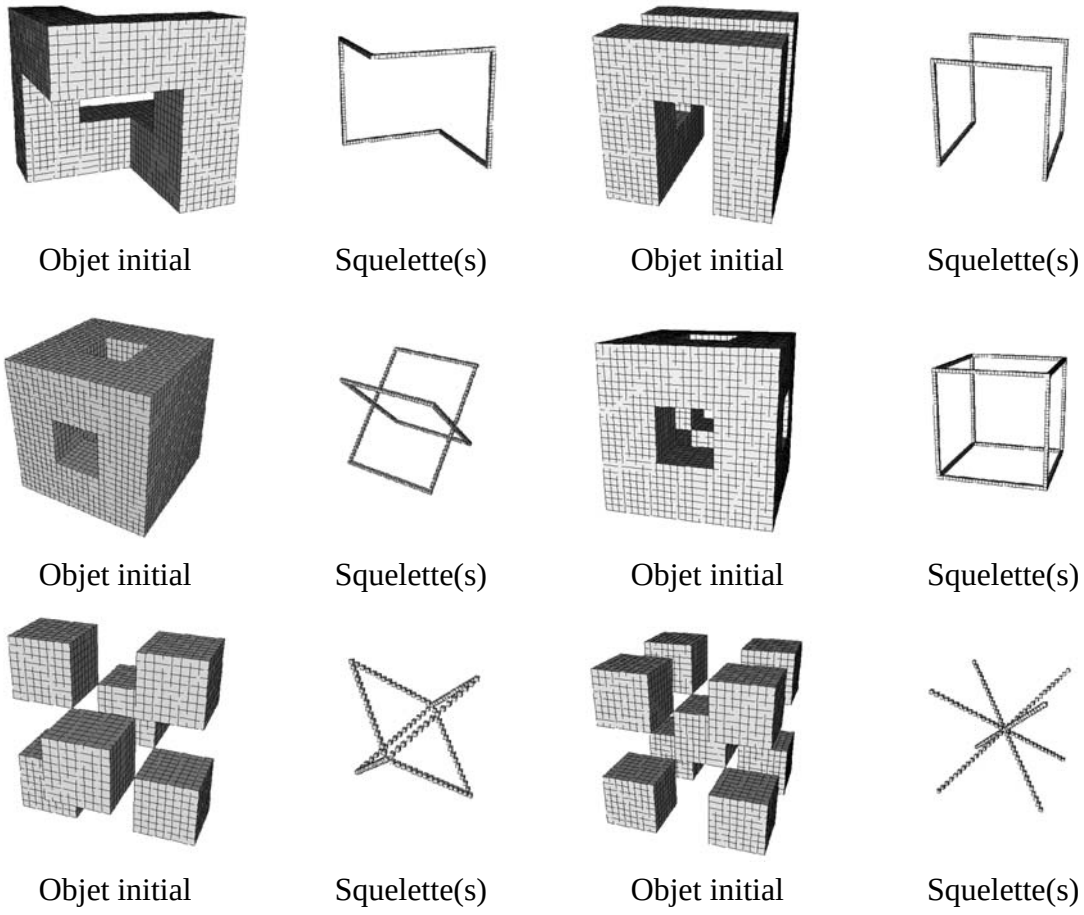


FIG. 9.11 – *Squelettisations sur quelques objets.*

Chapitre 10

Ordre pour images 2D en niveaux de gris

Peu d'algorithmes de squelettisation pour images 2D en niveaux de gris ont été proposés [DR79] [Ros84] [Pal89] [KCM91] [AR96]. Les plus récents ont été développés en adaptant ceux déjà proposés dans le cadre de la topologie digitale pour images 2D binaires, et en exploitant l'information du niveau de gris à travers des notions proposées dans le cadre de la topologie des coupes [AR95] [AS98] [BC00]. Notons que d'autres opérateurs utilisant la topologie des coupes ont également été proposés afin de segmenter des images 2D en niveaux de gris [GBC96] [EB97] [EBC97] [Eve97] [Arc99] [CBB99].

Dans un premier temps, nous rappelons succinctement quelques notions de base de topologie digitale pour les images 2D binaires et montrons comment elles ont été étendues aux images 2D en niveaux de gris menant à la topologie des coupes. C'est de cette façon que nous avons étendu les notions présentées pour l'ordre de base associé aux images 2D binaires (chapitre 8) à un ordre de base associé aux images 2D en niveaux de gris (voir également [LB00a]).

10.1 Rappels sur la topologie des images 2D binaires

Nous rappelons ici quelques notions de base de topologie digitale pour les images 2D binaires [KR89].

Soit $\mathcal{Z}$ l'ensemble des entiers relatifs. Un point $x \in \mathcal{Z}^2$ est défini par le couple (x_1, x_2), avec $x_i \in \mathcal{Z}$ et $i = 1, 2$. On considère les deux relations de voisinage Γ_4 et Γ_8, définies de la façon suivante : pour tout point $x \in \mathcal{Z}^2$, $\Gamma_4(x) = \{y \in \mathcal{Z}^2;\ |y_1 - x_1| + |y_2 - x_2| \leq 1\}$, et $\Gamma_8(x) = \{y \in \mathcal{Z}^2;\ max(|y_1 - x_1|, |y_2 - x_2|) \leq 1\}$. Dans la suite, n désigne un nombre pouvant prendre la valeur 8 ou 4. On définit $\Gamma_n^*(x) = \Gamma_n(x) \setminus \{x\}$. Le point y est *n-adjacent* au point $x \in \mathcal{Z}^2$ si $y \in \Gamma_n^*(x)$. Un *n-chemin* est une suite de points $x_0, \ldots, x_k$ avec x_i n-adjacent à x_{i-1} pour $i = 1, \ldots, k$.

Soit $X \subset \mathcal{Z}^2$, on note $\overline{X}$ l'ensemble complémentaire de X. Deux points x et y de $\mathcal{Z}^2$ sont dits *n-connectés dans X* s'il existe un n-chemin dans X entre ces deux points. Cela définit une relation d'équivalence. Les classes d'équivalence pour cette relation sont les *composantes n-connexes* de X. Un objet $X \subset \mathcal{Z}^2$ est dit *n-connexe* s'il est constitué exactement d'une seule composante n-connexe. L'ensemble composé de toutes les composantes n-connexes de X et

n-adjacentes au point x est noté par $C_n[x, X]$.

Afin d'avoir une correspondance entre la topologie de X et la topologie de $\overline{X}$, nous devons considérer deux sortes différentes d'adjacence pour X et pour $\overline{X}$, si nous utilisons la n-adjacence pour X, nous devons utiliser la $\overline{n}$-adjacence pour $\overline{X}$, avec $(n, \overline{n}) = (8, 4)$ ou $(4, 8)$.

Soient $X \subset \mathcal{Z}^2$ et $x \in \mathcal{Z}^2$, on définit les deux *nombres topologiques* suivants (avec $\#X$, le cardinal de l'ensemble X) : $T_n(x, X) = \#C_n[x, \Gamma_8^*(x) \cap X]$, et $\overline{T}_{\overline{n}}(x, X) = \#C_{\overline{n}}[x, \Gamma_8^*(x) \cap \overline{X}]$. On note $T = T_n(x, X)$ et $\overline{T} = \overline{T}_{\overline{n}}(x, X)$. Un point $x \in X$ est un *point isolé* si $T = 0$, un *point de bord* si $\overline{T} > 0$, un *point intérieur* si $\overline{T} = 0$.

Le point $x \in X$ est *simple* (pour X) s'il existe une correspondance une-à-une entre les n-composantes de X et celles de $X \setminus \{x\}$, et entre les $\overline{n}$-composantes de $\overline{X}$ et celles de $\overline{X} \cup \{x\}$. La propriété suivante permet de caractériser de façon locale les points simples [Ber94] : $x \in \mathcal{Z}^2$ est simple pour $X \subset \mathcal{Z}^2 \Leftrightarrow T = 1$ et $\overline{T} = 1$.

Soient $X,\ Y \subset \mathcal{Z}^2$. L'ensemble Y est *sous-homotope à* X si $Y = X$ ou si Y peut être obtenu à partir de X par une suppression itérative de points simples. L'ensemble Y est un *noyau (sous-) homotopique de* X si Y est sous-homotope à X et s'il n'existe plus de point simple pour Y.

Soient $X \subset \mathcal{Z}^2$ et $x \in X$, x est un *point extrémité (pour X)* si $\#(\Gamma_n^*(x) \cap X) = 1$. Si $X \subset \mathcal{Z}^2$ représente une image binaire, $S \subset \mathcal{Z}^2$ est un *squelette de* X si S peut être obtenu à partir de X par une suppression itérative de points simples et non extrémités de X, et en répétant cette suppression jusqu'à stabilité.

Notons que certains algorithmes d'amincissement ou de squelettisation ne produisent pas toujours des résultats satisfaisant les définitions précédentes, cela par l'approche qu'ils utilisent pour sélectionner et/ou supprimer les points simples. Il arrive qu'il reste des points simples dans le "noyau homotopique" ou le "squelette" ; nous obtenons alors des résultats non minces, avec des *points redondants*, *c.-à-d.* des points simples dans le cas du "noyau homotopique" ou des points simples et non extrémités dans le cas du "squelette". Avec notre approche utilisant les ordres, les résultats obtenus par nos algorithmes sont minces. C'est la raison pour laquelle nous avons utilisé une nouvelle terminologie, présentée dans la section 8.5.2, afin d'éviter toute ambiguïté.

10.2 Topologie des coupes pour les images 2D en niveaux de gris

Dans cette section, nous rappelons les définitions de base et les propriétés de la topologie des coupes [BEC97]. Une image 2D en niveaux de gris peut être vue comme une application F de $\mathcal{Z}^2$ dans $\mathcal{Z}$. Pour tout point $x \in \mathcal{Z}^2$, $F(x)$ est la valeur en niveaux de gris du point x. On note $\mathcal{F}$, l'ensemble composé de toutes les applications de $\mathcal{Z}^2$ dans $\mathcal{Z}$.

Soit $F \in \mathcal{F}$, la *coupe de F au niveau k* est l'ensemble F_k composé de tous les points $x \in \mathcal{Z}^2$ tel que $F(x) \geq k$. Notons qu'une coupe est un ensemble de points, *i.e.* une image binaire. Comme dans le cas binaire, si nous utilisons la n-adjacence pour les coupes F_k de F, nous devons utiliser la $\overline{n}$-adjacence pour les ensembles complémentaires $\overline{F}_k$, avec $(n, \overline{n}) = (4, 8)$ ou $(8, 4)$. Considérons l'application $-F$, appelée *l'application complémentaire* de F. Notons que les ensembles

complémentaires des coupes de F sont les coupes de $-F$. Dans la suite, nous avons choisi $n = 8$ pour les coupes de F, ainsi nous devons utiliser $\overline{n} = 4$ pour les coupes de $-F$. Une composante connexe non vide X d'une coupe F_k de F est un *maximum (régional) pour* F si $X \cap F_{k+1} = \emptyset$. Un ensemble $X \subset \mathcal{Z}^2$ est un *minimum (régional) pour* F si c'est un maximum (régional) pour $-F$.

Intuitivement, nous disons qu'une transformation sur F "préserve la topologie" si la topologie de toutes les coupes de F est préservée. Ainsi, la topologie des coupes pour les images en niveaux de gris peut être dérivée directement de la topologie des images binaires.

Les notions suivantes généralisent la notion de point simple aux images en niveaux de gris.

Soit $F \in \mathcal{F}$, le point $x \in \mathcal{Z}^2$ est *destructible (pour F)*, si x est simple pour F_k, avec $k = F(x)$. Nous constatons que si la valeur en niveaux de gris d'un point destructible est abaissée de 1, la topologie de F est préservée. Soient $F \in \mathcal{F}$ et $G \in \mathcal{F}$. L'application G est *sous-homotope à* F si $G = F$ ou si G peut être obtenue à partir de F par une sélection itérative de points destructibles et en les abaissant d'une valeur de 1.

Soient $F \in \mathcal{F}$ et $x \in \mathcal{Z}^2$. Nous définissons les deux voisinages suivants : $\Gamma^+(x, F) = \{y \in \Gamma_8(x),\ F(y) \geq F(x)\}$ et $\Gamma^{--}(x, F) = \{y \in \Gamma_8(x),\ F(y) < F(x)\}$. Nous définissons les deux *nombres topologiques* suivants : $T^+(x, F) = \#C_n[x, \Gamma^+(x, F)]$ et $T^{--}(x, F) = \#C_{\overline{n}}[x, \Gamma^{--}(x, F)]$. Nous notons $T^+ = T^+(x, F)$, et $T^{--} = T^{--}(x, F)$. Ces nombres topologiques nous permettent de caractériser de façon locale un point destructible. Soient $F \in \mathcal{F}$ et $x \in \mathcal{Z}^2$. Le point x est destructible pour $F \Leftrightarrow T^+ = 1$ et $T^{--} = 1$.

Notons que d'autres voisinages et nombres topologiques ont été définis ; ils permettent de classer les points d'une image 2D en niveaux de gris selon 11 classes (voir [GBC96]).

Soit $val^-(x, F) = \begin{cases} max\{F(y), \forall y \in \Gamma^{--}(x, F)\} \text{ si } \Gamma^{--}(x, F) \neq \emptyset, \\ F(x) \text{ sinon.} \end{cases}$

Il est montré que la valeur d'un point destructible peut être abaissée à la valeur $val^-(x, F)$ tout en préservant la topologie (dans le cas d'un point destructible, $\Gamma^{--}(x, F) \neq \emptyset$).

Soient $F \in \mathcal{F}$ et $G \in \mathcal{F}$. On dit que G est un *noyau (sous-) homotopique* de F si l'application G est sous-homotope à F et s'il n'existe plus de point destructible pour G. Soient $F \in \mathcal{F}$ et $x \in \mathcal{Z}^2$, x est un *point extrémité (pour F)* s'il est point extrémité pour l'ensemble F_k avec $k = F(x)$.

Soit $F \in \mathcal{F}$, on dit que $G \in \mathcal{F}$ est un *squelette de* F si G peut être obtenu à partir de F par une sélection itérative de points destructibles et non extrémités puis en les abaissant à la valeur $val^-(x, F)$, et en réitèrant ce processus jusqu'à stabilité.

Soulignons que la plupart des algorithmes de squelettisation proposés dans le cadre des images 2D en niveaux de gris sont une adaptation de ceux employés dans le cadre des images 2D binaires, ce qui engendre au minimum les mêmes inconvénients, à savoir que les résultats obtenus peuvent présenter des points redondants, qu'ils peuvent dépendre de la première sous-maille choisie, de l'ordre des directions considérées [KCM91] [AR95] [AS98] ou utiliser un voisinage étendu [AR95]. De plus, l'ordre selon lequel sont sélectionnés les points destructibles à abaisser a également une influence sur les résultats (voir [Eve97] pour plus de détails). Dans [BC00] a été réalisée une étude concernant le choix des points destructibles à abaisser, afin de mieux centrer les squelettes d'images 2D en niveaux de gris.

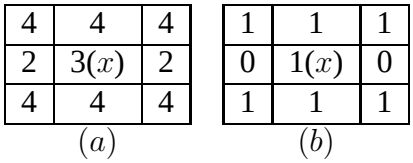


FIG. 10.1 – *Exemple de point non destructible.*

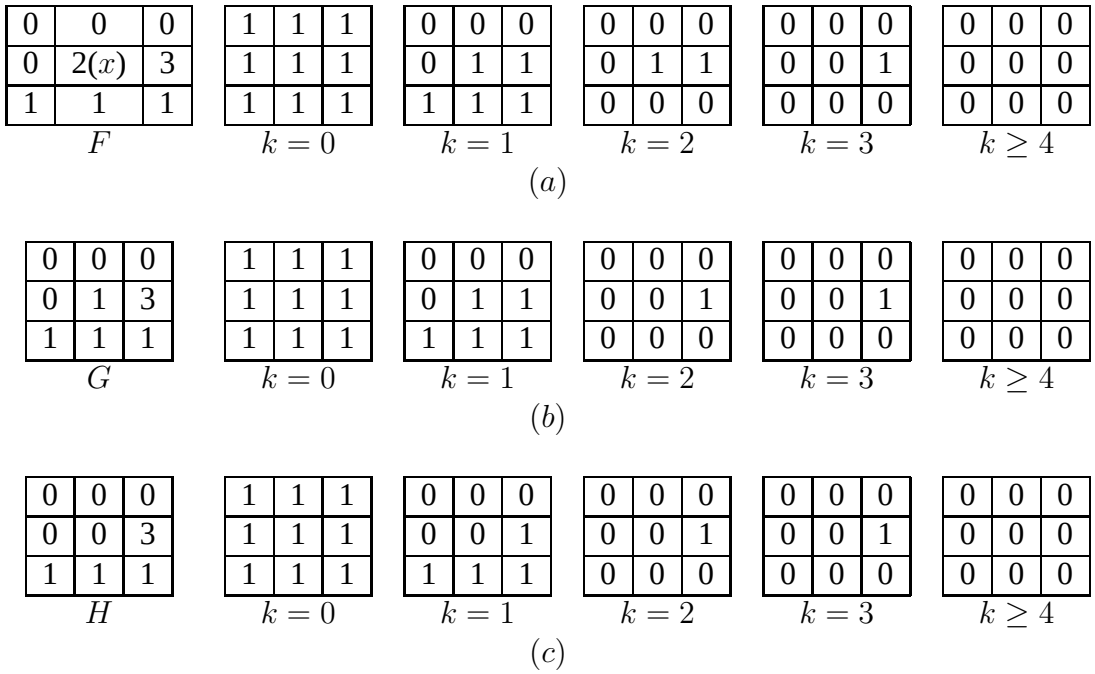


FIG. 10.2 – *Exemple de point destructible et discussion à propos de la valeur d'abaissement.*

10.2.1 Discussion à propos de la valeur d'abaissement d'un point destructible

Nous donnons maintenant quelques exemples de configurations décrivant $\Gamma_8(x)$ et correspondant à des points x destructibles ou non, puis nous commentons les résultats obtenus selon la valeur à laquelle nous essayons d'abaisser un point destructible.

Considérons F représenté à la figure 10.1 (a). Le point x n'est pas destructible pour F, car x n'est pas simple dans sa coupe F_3 (Fig. 10.1 (b)) ou plus directement à partir de la figure 10.1 (a), nous avons $T^{+}(x, F) = 2$ et $T^{--}(x, F) = 2$.

À la figure 10.2 (a) sont représentées F, puis différentes coupes. Le point x est destructible. Nous constatons que si la valeur de x est abaissée de 1 (Fig. 10.2 (b)), x prenant alors la valeur $1 = val^{-}(x, F)$, alors la topologie de chaque coupe est préservée. En fait, si la valeur de x est abaissée de 2 (Fig. 10.2 (c)) alors la topologie de chaque coupe est encore préservée, mais

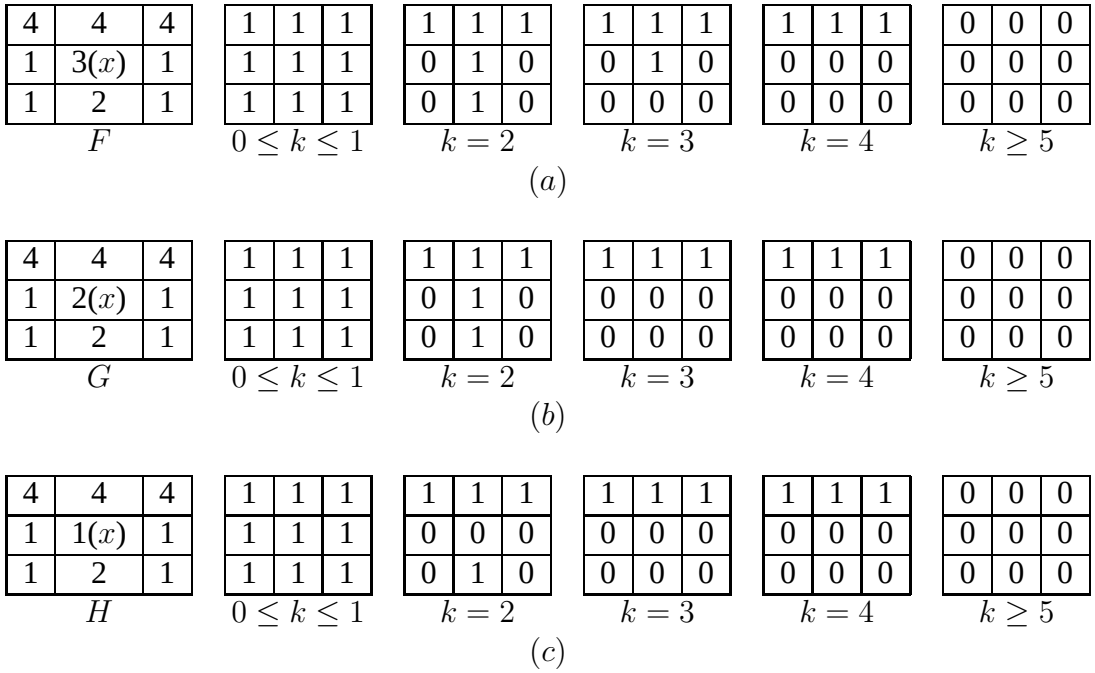


FIG. 10.3 – *Autre exemple de point destructible et discussion à propos de la valeur d'abaissement.*

cela n'est généralement pas vrai. En effet, considérons la configuration de la figure 10.3 (a) dans laquelle sont représentées F, puis différentes coupes. Le point x est destructible. Nous constatons que si la valeur de x est abaissée de 1 (Fig. 10.3 (b)), x prenant alors la valeur $2 = val^-(x, F)$, alors la topologie de chaque coupe est préservée (voir sur la même ligne que la figure 10.3 (b)). Si la valeur de x est abaissée de 2 (Fig. 10.2 (c)) alors la topologie de la coupe F_2 n'est pas préservée ; en effet, F_2 est composée d'une composante connexe de X et de deux composantes connexes de $\overline{X}$, et après l'abaissement de 2 de la valeur de x, la coupe obtenue H_2 comporte deux composantes connexes de X et une composante connexe de $\overline{X}$.

Intuitivement, un algorithme "efficace" consisterait alors à abaisser un point destructible le plus bas possible, dans le but de réduire le nombre total d'abaissements à effectuer. En fait, il est difficile de généraliser cela, étant donné que les résultats dépendent de l'ordre de sélection des points à détruire [Eve97] et de l'approche adoptée par l'algorithme : directionnelle, sous-mailles, fortement parallèle. Il serait possible, par exemple, d'abaisser itérativement en priorité les points destructibles les plus hauts, et pour un point destructible x (avec $k = F(x)$), de l'abaisser à un niveau l tel que soit $l = 0$, soit $l > 0$ et pour chaque coupe F_i avec $i = l, \ldots, k$, x est simple pour F_i, mais x n'est pas simple pour F_{l-1}. Ce processus, consistant à réaliser le test précédent pour chaque point abaissé, est coûteux ; en revanche la valeur $val^-(x, F)$ à laquelle on peut abaisser un point tout en préservant la topologie peut être déterminée rapidement [GBC96].

À la figure 10.4 est proposée une configuration pour laquelle les points x et y sont destructibles ; si la valeur de x est abaissé de 1 (*i.e* à 0) alors la topologie des coupes est préservée, néanmoins y n'est plus destructible une fois que x est abaissé (de la même façon, x devient non

0	0	0	0
0	1	1	0
0	1(y)	1(x)	0
0	3	3	0
0	0	0	0

(a)

FIG. 10.4 – *Problème pour un abaissement en parallèle de points destructibles.*

destructible si y est abaissé à 0 d'abord). En fait, dans cet exemple, nous avons un MNS dans la coupe de x et y, constitué des deux points simples x et y.

Cela indique la difficulté de mettre en œuvre un algorithme d'abaissement parallèle de points destructibles, simple et efficace. Retenons que la difficulté est double pour les images 2D en niveaux de gris : il faut une stratégie de sélection de points, puis une stratégie d'abaissement. Étant donné les résultats satisfaisants obtenus avec les algorithmes parallèles de squelettisation pour les images 2D binaires, proposés dans le cadre des ordres, il nous a semblé intéressant de proposer des algorithmes parallèles de squelettisation pour les images 2D en niveaux de gris, dans le cadre des ordres adaptés aux images 2D en niveaux de gris.

10.3 Nombre de connexité dans $|H^2|$

Au chapitre 8, nous avons traité des points α_2-simples pour les images 2D binaires ; dans la suite, nous écrivons point α-simple à la place de point α_2-simple. Nous définissons au préalable deux nombres de connexité nous permettant de donner une autre caractérisation d'un point α-simple dans l'ordre de base $(H^2, \supseteq)$ associé aux images 2D binaires. Nous adapterons ces nombres au cas des images 2D en niveaux de gris dans la section 10.5.1.

Soient $\#S$ le nombre d'éléments d'un sous-ensemble S de H^2, et $C[S]$ l'ensemble des composantes connexes de $|S|$. Nous définissons les deux *nombres de connexité* suivants : $T_\alpha(x, S) = \#C[\alpha^\square(x) \cap S]$ et $\overline{T}_\alpha(x, S) = \#C[\alpha^\square(x) \cap \overline{S}]$.

Nous avons vu qu'un point est α-simple pour $|S|$ si et seulement si $\alpha^\square(x) \cap S$ est une courbe ouverte simple ou un point isolé (cf. la propriété 13, page 176). Cela implique que $\alpha^\square(x) \cap \overline{S}$ est également une courbe ouverte simple ou un point isolé (nous rappelons qu'un point isolé est un cas particulier de courbe simple ouverte). Cela explique que nous pouvons donner une autre caractérisation d'un point α-simple, utilisant ces deux nombres de connexité précédents.

Proposition : Soient S un sous-ensemble de H^2 et x un point de S. Le point x est α-simple pour $|S|$ si et seulement si $T_\alpha(x, S) = 1$ et $\overline{T}_\alpha(x, S) = 1$.

Nous donnons quelques exemples de points qui sont α- ou β-simples ou non, selon les différents types d'éléments de H^2 (*i.e.*, un carré unité, un intervalle unité, un singleton). Considérons le sous-ensemble S décrit à la figure 10.5. Nous écrivons T_α pour $T_\alpha(x, S)$ et $\overline{T}_\alpha$ pour $\overline{T}_\alpha(x, S)$. Les points a et e sont α-simples ($T_\alpha = \overline{T}_\alpha = 1$), les autres points ne sont pas α-simples. Le point b est tel que $T_\alpha = 3$ et $\overline{T}_\alpha = 3$; c est tel que $T_\alpha = 1$ et $\overline{T}_\alpha = 0$; d est tel que $T_\alpha = 0$ et $\overline{T}_\alpha = 1$;

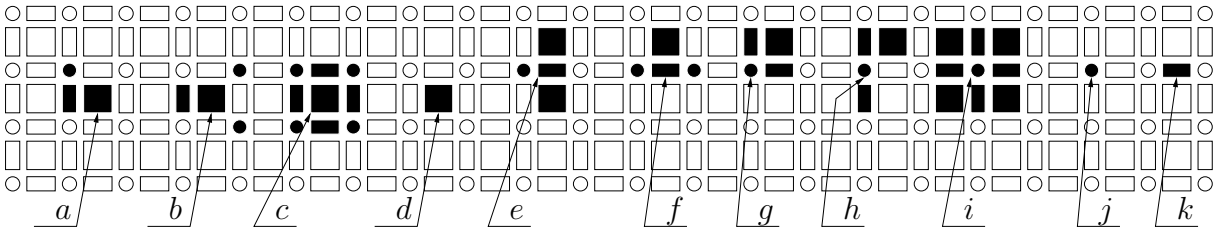


FIG. 10.5 – *Quelques points α-simples (a, e), non α-simples (b à d, f à k), β-simples (f, g), non β-simples (a à e, h à k).*

f est tel que $T_\alpha = 2$ et $\overline{T}_\alpha = 0$; k est tel que $T_\alpha = 0$ et $\overline{T}_\alpha = 2$. Chaque point de l'ensemble $\{g, h, i, j\}$ est tel que $\alpha^\square(x) = \emptyset$, ainsi $T_\alpha = 0$ et $\overline{T}_\alpha = 0$. Les points f et g sont β-simples ($T_\beta = \overline{T}_\beta = 1$), les autres points ne sont pas β-simples. Le point e est tel que $T_\beta = 2$ et $\overline{T}_\beta = 0$; h est tel que $T_\beta = 2$ et $\overline{T}_\beta = 2$; i est tel que $T_\beta = 1$ et $\overline{T}_\beta = 0$; j est tel que $T_\beta = 0$ et $\overline{T}_\beta = 1$; k est tel que $T_\beta = 0$ et $\overline{T}_\beta = 2$. Chaque point de l'ensemble $\{a, b, c, d\}$ est tel que $\beta^\square(x) = \emptyset$, ainsi $T_\beta = 0$ et $\overline{T}_\beta = 0$.

10.4 Ordre valué associé aux images 2D en niveaux de gris

Nous introduisons un ordre valué afin de traiter les images 2D en niveaux de gris, selon deux façons différentes de considérer des objets 2D en niveaux de gris.

Définition (ordre valué) : Soit $|X| = (X, \alpha)$ un ordre. Soit F une application de X dans $\mathcal{Z}$. Le triplet (X, α, F) est appelé un *ordre valué*.

Nous choisissons $|X| = |H^2|$ pour traiter les images 2D en niveaux de gris. Maintenant, nous spécifions F selon deux façons différentes de considérer des objets 2D en niveaux de gris dans un ordre valué (nous adaptons l'approche utilisée pour les images 2D binaires, décrites dans la section 8.2.2). Une image 2D en niveaux de gris est un couple (I, f) dans lequel I est un ensemble de pixels (voir section 8.2.2), et f une application de I dans $\mathcal{Z}$. Pour tout élément p de I, $f(p)$ est la valeur en niveaux de gris du pixel p.

À chaque image 2D en niveaux de gris (I, f), nous associons l'ordre valué $(X, \supseteq, F)$ de la façon suivante : à chaque pixel $p = [i, i+1] \times [j, j+1]$, nous associons le carré unité p' de H^2 : $\{(i, j), (i, j+1), (i+1, j), (i+1, j+1)\}$ et valué par $F(p') = f(p)$, ainsi que tous les éléments y de H^2 inclus dans ce carré unité (les valeurs de ces éléments y doivent être précisées). Dans notre étude, les éléments y prennent la valeur $F(y) = max\{F(z);\ z \in \beta^\square(y)$ et z est un carré unité $\}$, *i.e.* un élément y (singleton ou intervalle) prend la plus grande valeur des carrés le contenant (voir Fig. 10.6 (a) et (b)). Soit Φ'_1 une telle transformation (voir également [Kov89] [Kov94]).

Une autre façon consiste à associer à chaque image en niveaux de gris (I, f), l'ordre valué $(X, \supseteq, F)$ comme suit : à chaque pixel $p = [i, i+1] \times [j, j+1]$, est associé l'élément p' de H^2 dont les coordonnées sont $(i/2, j/2)$ dans $\mathcal{Z}^2_+$ et $F(p') = f(p)$ (voir Fig. 10.6 (a) et (c)). Soit Φ'_3 une telle transformation.

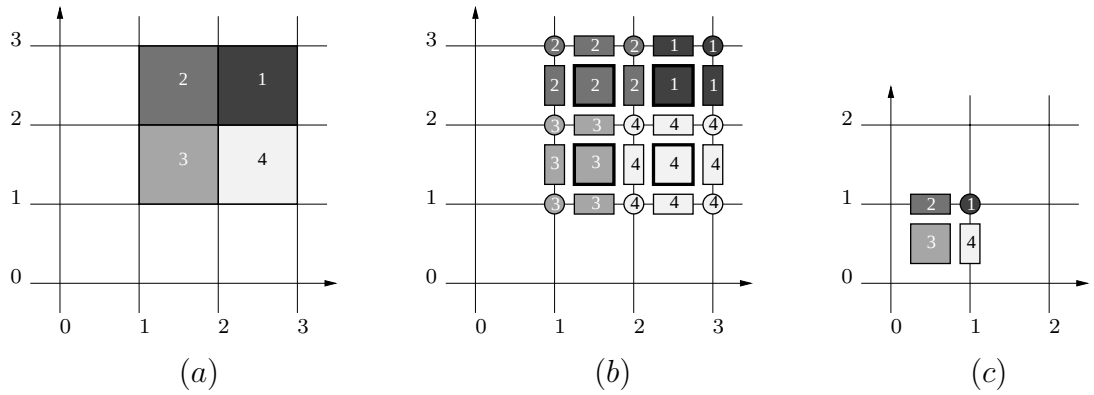


FIG. 10.6 – (a) Un objet 2D en niveaux de gris (I, f), (b) $\Phi'_1((I, f))$, (c) $\Phi'_3((I, f))$.

10.5 Destructibilité pour les images en niveaux de gris dans $(H^2, \supseteq, F)$

Une image 2D en niveaux de gris peut être vue comme un relief topographique tel que la valeur en niveaux de gris d'un point corresponde à sa hauteur dans ce relief. Une *coupe* à une hauteur donnée est l'ensemble constitué de tous les points dont la valeur en niveaux de gris est plus grande ou égale à cette hauteur donnée (en fait, une coupe est une image binaire). Une transformation sur une image 2D en niveaux de gris "préserve la topologie" si la topologie de chaque coupe est préservée. Cette approche est celle adoptée dans le cadre de la topologie digitale étendue aux images 2D en niveaux de gris, approche connue sous le nom de "topologie des coupes", et que nous avons rappelée dans la section 10.2.

Dans le cadre des ordres, nous bénéficions du fait que l'ordre obtenu après la suppression parallèle de points α- ou β-simples d'une coupe, est σ-équivalent à l'ordre initial (voir section 8.4.2.3). De cette façon, nous allons proposer un algorithme parallèle de squelettisation ultime et un algorithme parallèle de squelettisation curviligne, pour images 2D en niveaux de gris, basés sur l'abaissement parallèle de 1 des valeurs des points α-destructibles (qui sont les points α-simples dans leur coupe). Dans cette section, après une définition formelle d'un point α-destructible, nous donnons une caractérisation de tels points, en utilisant les nombres de connexité (préalablement introduits dans le cas binaire dans la section 10.3).

10.5.1 Destructibilité dans $(H^2, \supseteq, F)$

Nous introduisons d'abord la définition d'une coupe pour un ordre valué, puis la définition d'un point α-destructible ou β-destructible.

Soit X un sous-ensemble de H^2.

Définition (coupe) : Soit $(X, \supseteq, F)$ un ordre valué. La *coupe de F au niveau k sur X* est l'ensemble F_k composé de tous les points dans X dont la valeur en niveaux de gris est supérieure ou égale à k, *i.e.* $F_k = \{x \in X;\ F(x) \geq k\}$ avec $k \in \mathcal{Z}$.

Nous remarquons qu'une coupe est un ensemble de points, *i.e.* une image binaire. Ainsi, nous pouvons considérer l'ordre $|F_k|$ relatif à la coupe F_k : nous avons l'ordre "binaire" $|F_k|$ associé à l'image binaire F_k.

La notion suivante généralise celle d'un point α-simple pour des images en niveaux de gris, dans des ordres valués.

Définition (point α-destructible, point β-destructible) : Soient $(X, \supseteq, F)$ un ordre valué, et $x \in X$. Le point x est *α-destructible pour l'ordre valué* $(X, \supseteq, F)$ (resp. *β-destructible*), si x est α-simple (resp. β-simple) pour l'ordre $|F_k|$ (l'ordre relatif à la coupe F_k), avec $k = F(x)$. En d'autres termes, un point x est α-destructible si et seulement si $\alpha^{\square}(x)$ est constitué d'une seule composante connexe de points plus hauts que x et d'une seule composante connexe de points plus bas que x.

Définition (nombre de connexité) : Nous définissons les deux ensembles suivants : $\alpha^{+}(x, F) = \{y \in \alpha^{\square}(x); F(y) \geq F(x)\}$ et $\alpha^{--}(x, F) = \{y \in \alpha^{\square}(x); F(y) < F(x)\}$. Nous avons $\alpha^{\square}(x) \cap F_k = \{y \in \alpha^{\square}(x); F(y) \geq k\}$. En prenant $k = F(x)$, nous avons $\alpha^{\square}(x) \cap F_k = \alpha^{+}(x, F)$. De la même façon, $\alpha^{\square}(x) \cap \overline{F}_k = \{y \in \alpha^{\square}(x); F(y) < k\}$. En prenant $k = F(x)$, nous avons $\alpha^{\square}(x) \cap \overline{F}_k = \alpha^{--}(x, F)$.
Nous définissons les deux *nombres de connexité* suivants :
$T_{\alpha}^{+}(x, F) = \#C[\alpha^{+}(x, F)]$ et $T_{\alpha}^{--}(x, F) = \#C[\alpha^{--}(x, F)]$. Un point x est α-destructible pour $(X, \supseteq, F) \Leftrightarrow T_{\alpha}(x, F_k) = 1$ et $\overline{T}_{\alpha}(x, F_k) = 1$. En prenant $k = F(x)$, nous avons $T_{\alpha}(x, F_k) = \#C[\alpha^{\square}(x) \cap F_k] = \#C[\alpha^{+}(x, F)] = T_{\alpha}^{+}(x, F)$. De la même façon, $\overline{T}_{\alpha}(x, F_k) = T_{\alpha}^{--}(x, F)$.

Ainsi, nous proposons la caractérisation suivante d'un point α-destructible par l'utilisation de ces nombres de connexité.

Proposition : Soient $(X, \supseteq, F)$ un ordre valué, et x un point de X. Le point x est α-destructible pour $(X, \supseteq, F)$ si et seulement si $T_{\alpha}^{+}(x, F) = 1$ et $T_{\alpha}^{--}(x, F) = 1$.

Définition (point extrémité) : Soient $(X, \supseteq, F)$ un ordre valué, et x un point de X. Le point x est un *point extrémité pour* $(X, \supseteq, F)$ s'il est point extrémité pour l'ordre $|F_k|$ avec $k = F(x)$, *i.e.* si $\#[\theta^{\square}(x) \cap F_k] = 1$.

Nous avons $\theta^{\square}(x) \cap F_k = \{y \in \theta^{\square}(x); F(y) \geq k\}$. Soit $\theta^{+}(x, F) = \{y \in \theta^{\square}(x); F(y) \geq F(x)\}$. En prenant $k = F(x)$, nous avons $\theta^{\square}(x) \cap F_k = \theta^{+}(x, F)$. Ainsi x est un point extrémité pour $(X, \supseteq, F) \Leftrightarrow \#[\theta^{+}(x, F)] = 1$. Or un point extrémité pour $|X|$ est soit un point α-simple, soit un point β-simple pour $|X|$ (cf. section 8.5.3). Par conséquent, si x est un point extrémité pour $|F_k|$, alors il est soit un point α-simple soit un point β-simple pour $|F_k|$, ainsi il est soit un point α-destructible, soit un point β-destructible pour $(X, \supseteq, F)$.

La figure 10.7 montre quelques exemples de points qui sont α- ou β-destructibles ou non (exemples donnés uniquement pour les cas de carré unité ou de singleton). Nous écrivons T_{α}^{+} pour $T_{\alpha}^{+}(x, F)$, et T_{α}^{--} pour $T_{\alpha}^{--}(x, F)$. La figure 10.7 (a) montre un point α-destructible et non extrémité. La figure 10.7 (b) montre $\alpha^{+}(x, F) \cup \{x\} = (\alpha^{\square}(x) \cap F_2) \cup \{x\}$ pour F de la figure 10.7 (a). Le point x est α-destructible pour $(X, \supseteq, F)$ car il est α-simple pour $|F_2|$ (Fig. 10.7 (b)), ou directement à partir de la figure 10.7 (a), $T_{\alpha}^{+} = 1$ et $T_{\alpha}^{--} = 1$. De plus, $\#[\theta^{+}(x, F)] = 2$, ainsi x n'est pas un point extrémité. La figure 10.7 (c) montre un point α-destructible et extrémité ($\#[\theta^{+}(x, F)] = 1$). La figure 10.7 (e) montre un point non α-destructible ($T_{\alpha}^{+} = 3$ et $T_{\alpha}^{--} = 3$). De la même façon, les figures 10.7 (h) (j) (l) montrent $\beta^{+}(x, F) \cup \{x\}$ respectivement pour F

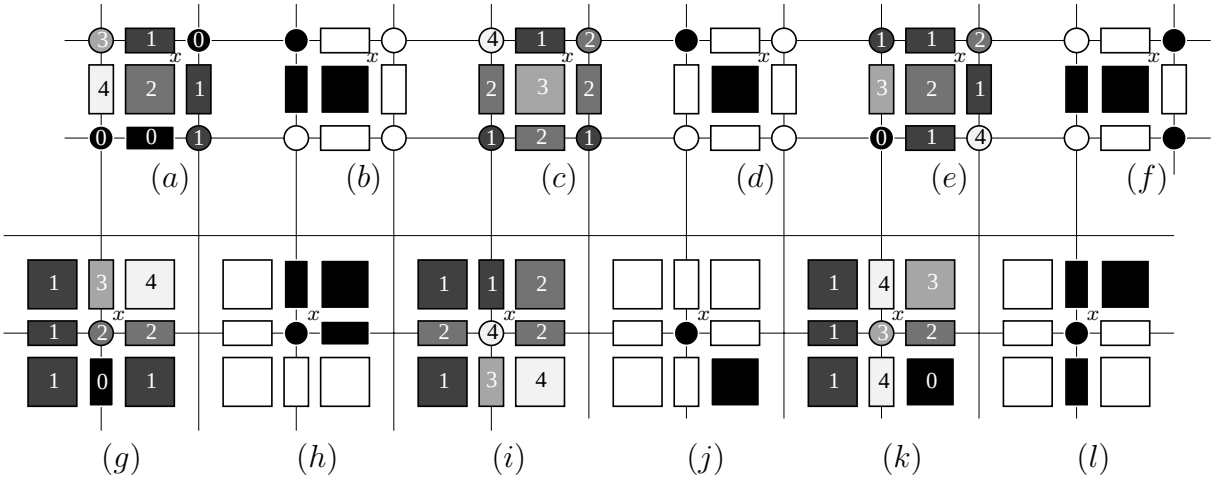


FIG. 10.7 – *Exemples de points α- ou β-destructibles ou non, extrémités ou non : (a) un point α-destructible et non extrémité, (c) un point α-destructible et extrémité, (e) un point non α-destructible, (b) (d) (f) $\alpha^+(x,F) \cup \{x\}$ (en noir) de (a) (c) (e) respectivement. Exemples de : (g) un point β-destructible et non extrémité, (i) un point β-destructible et extrémité, (k) un point non β-destructible, (h) (j) (l) $\beta^+(x,F) \cup \{x\}$ (en noir) de (g) (i) (k) respectivement.*

des figures 10.7 (g) (i) (k) ; et x est un point β-destructible et non extrémité dans la figure 10.7 (g) ($\#[\theta^+(x,F)] = 3$) ; x est un point β-destructible et extrémité dans la figure 10.7 (i) ; et x est un point non β-destructible dans la figure 10.7 (k) ($T_\beta^+ = 2$ et $T_\beta^{--} = 2$).

10.5.2 Équivalence entre les ordres valués dans $(H^2, \supseteq, F)$

Nous introduisons d'abord les notions d'ordres valués σ-équivalents. Puis, nous donnons une proposition qui énonce que l'ordre valué, obtenu après les abaissements en parallèle de 1 des valeurs des points α-destructibles, est σ-équivalent à l'ordre initial. Dans la section suivante, nous donnerons un algorithme de squelettisation ultime et un squelette de squelettisation curviligne pour des images 2D en niveaux de gris, basés sur les abaissements parallèles des points α- ou β-destructibles.

Définition (ordres valués σ-équivalents) : Soient $(X, \supseteq, F)$ et $(X, \supseteq, G)$ deux ordres valués, X étant un sous-ensemble de H^2. L'ordre $(X, \supseteq, G)$ est *(bas) σ-équivalent* de $(X, \supseteq, F)$ si $F = G$ ou si G peut être obtenu à partir de F par la sélection itérative d'un point α- ou β-destructible et par l'abaissement de sa valeur de 1.

Nous donnons la proposition qui indique qu'un ordre valué, obtenu après les abaissements en parallèle de 1 des valeurs de tous les points α-destructibles, est σ-équivalent à l'ordre initial.

Proposition : Soit $(X, \supseteq, F)$ un ordre valué, X étant un sous-ensemble de H^2. Soit $x \in X$ un point α-destructible pour $(X, \supseteq, F)$. Soit y un point de X avec $y \neq x$. Si y est α-destructible pour $(X, \supseteq, F)$, alors y est α-destructible pour $(X, \supseteq, G)$; $(X, \supseteq, G)$ étant l'ordre valué obtenu après l'abaissement de 1 de la valeur de x.

Preuve :
Remarquons d'abord que lorsque la valeur d'un point α-destructible est abaissée de 1, seule la coupe de ce point est modifiée. Considérons dans un premier temps que x et y n'appartiennent pas à la même coupe. L'abaissement de x n'altère pas la coupe de y ; ainsi y est α-destructible pour $(X, \supseteq, G)$. Considérons maintenant que x et y appartiennent à la même coupe. Comme x et y sont α-destructibles pour $(X, \supseteq, F)$, alors ils sont α-simples pour $|F_k|$, avec $k = F(x) = F(y)$. La propriété 14, page 176, garantit que y est α-simple pour $|G_k|$, obtenu par la suppression de x de $|F_k|$ (réalisé par l'abaissement de 1 de x) ; en fait, l'ordre $|G_k|$ est σ-équivalent à $|F_k|$. Par conséquent, le point y est α-destructible pour $(X, \supseteq, G)$. □

10.6 Algorithmes de squelettisation pour images 2D en niveaux de gris dans $(H^2, \supseteq, F)$

Nous proposons un algorithme parallèle de squelettisation ultime qui consiste en la répétition jusqu'à stabilité d'une sous-itération d'abaissements en parallèle de 1 des valeurs de tous les points α-destructibles, suivie d'une sous-itération d'abaissements en parallèle de 1 des valeurs de tous les points β-destructibles. Ce processus est décrit par l'algorithme 14.

Soit $(X, \supseteq, F)$ un ordre valué, X étant un sous-ensemble de H^2. Notons "$\backslash_\alpha\ F$" l'opérateur qui consiste en l'abaissement en parallèle de 1 des valeurs de tous les points α-destructibles pour $(X, \supseteq, F)$. Nous écrivons $F^{\,0} = F$.

$F^{\,0} = F$
Répéter

$$\left\{ \begin{array}{lcl} F^{\,2i+1} & = & \backslash_\alpha \quad F^{\,2i}, \\ F^{\,2i+2} & = & \backslash_\beta \quad F^{\,2i+1}. \end{array} \right.$$

Jusqu'à ce qu'il n'y ait plus d'abaissement durant 2 sous-itérations successives.

Algorithme 14: *Algorithme de squelettisation ultime dans* $(H^2, \supseteq, F)$.

Le résultat est un *squelette ultime* (en niveaux de gris) de F, *c.-à-d.* un ordre valué n'incluant plus de point α- ou β-destructible (voir Fig. 10.8 et 10.10 pour des exemples de squelettes ultimes).

De la même façon, nous proposons un algorithme parallèle de squelettisation curviligne qui abaisse les points α-destructibles, puis les points β-destructibles, à l'exception des points destructibles qui sont des points terminaux. Ce processus est décrit par l'algorithme 15.

Notons "$\backslash_{\alpha \cap \overline{pe}}\ F$" l'opérateur qui consiste en l'abaissement en parallèle de 1 des valeurs de tous les points α-destructibles pour $(X, \supseteq, F)$, excepté les points destructibles qui sont des points extrémités pour $(X, \supseteq, F)$.

$F^0 = F$
Répéter

$$\begin{cases} F^{2i+1} = \setminus_{\alpha \cap \overline{pe}} F^{2i}, \\ F^{2i+2} = \setminus_{\beta \cap \overline{pe}} F^{2i+1}. \end{cases}$$

Jusqu'à ce qu'il n'y ait plus d'abaissement durant 2 sous-itérations successives.

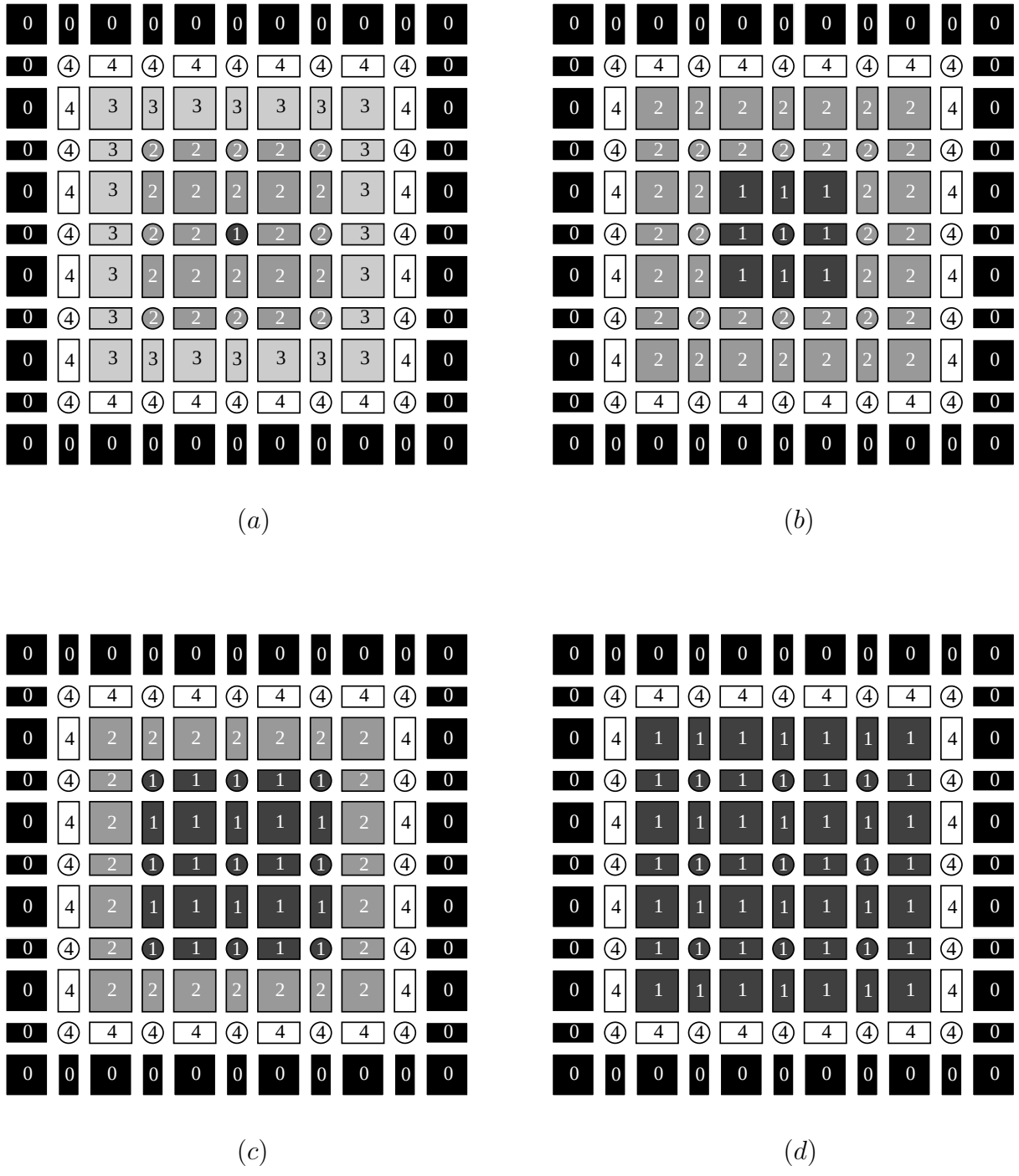
Algorithme 15: *Algorithme de squelettisation curviligne dans* $(H^2, \supseteq, F)$.

Le résultat est un *squelette curviligne* de F (en niveaux de gris) (voir Fig. 10.9 pour des exemples de squelettes curvilignes).

La figure 10.8 décrit les sous-itérations successives d'abaissement durant l'algorithme de squelettisation ultime pour l'image initiale en niveaux de gris $F(= F^0)$ (Fig. 10.8 (a)). La première sous-itération abaisse les points α-destructibles de F^0, ce sont les points α-simples de $|F_3^0|$ (tous les points de $|F_3^0|$) qui sont abaissés à la valeur 2 ; et les points α-simples de $|F_2^0|$ qui sont abaissés à la valeur 1 ; nous obtenons ainsi F^1 (Fig. 10.8 (b)). La seconde sous-itération abaisse les points β-destructibles de F^1 ; ce sont les points β-simples de $|F_2^1|$ qui sont abaissés à la valeur 1, nous obtenons ainsi F^2 (Fig. 10.8 (c)). La troisième sous-itération abaisse les points α-destructibles de F^2 ; ce sont les points α-simples de $|F_2^2|$ qui sont abaissés à la valeur 1, nous obtenons ainsi F^3 (Fig. 10.8 (c)). Finalement, il n'y a plus de point α-, ou β-destructible ; F^3 est alors un squelette ultime de F.

En fait, l'objet initial peut être vu comme un cratère, dont la ligne de crête a la valeur 4, entouré par un plateau. Le plateau a la valeur constante 0 et la pente du bassin à l'intérieur du cratère décroît de 3 à 1. Le squelette ultime obtenu est tel que la ligne de crête est gardée ; en effet, tout point de cette ligne de crête n'est pas destructible. Nous pouvons voir que le bassin est transformé en une région plate de valeur 1. Intuitivement, nous avons aminci symétriquement le bassin en un plateau.

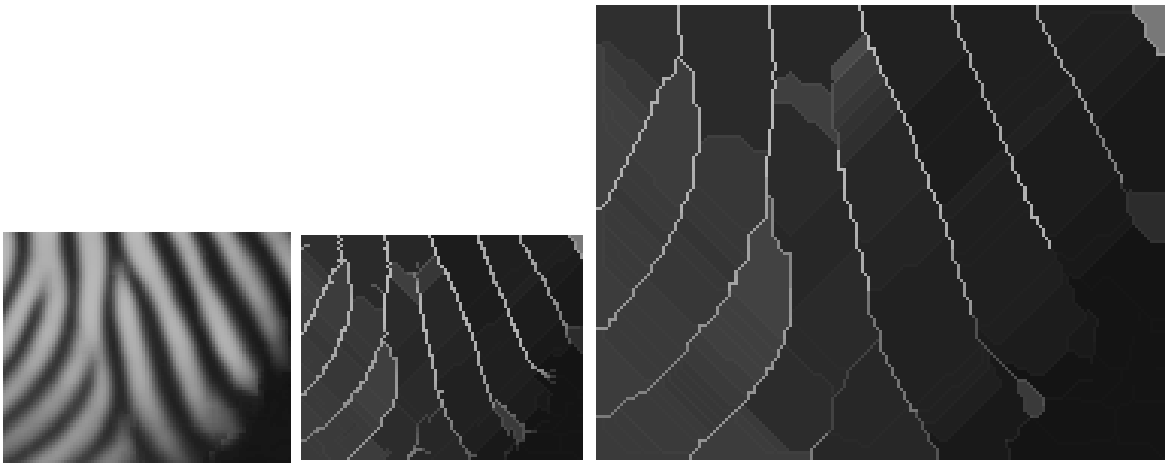
Remarque : Considérons l'ordre valué $(X, \supseteq, F)$ obtenu à partir d'une image 2D en niveaux de gris f, après l'utilisation de Φ'_1 (voir un exemple à la figure 10.6 (b)). Tout élément x de X est tel que $\alpha^{--}(x, F) = \emptyset$, ainsi $T_\alpha^{--}(x, F) = 0$, par conséquent il n'y a pas de point α-destructible dans X. Les résultats obtenus par nos algorithmes (après l'utilisation de Φ'_1) ne dépendent alors pas de la première sous-itération choisie (comme dans le cas binaire dans les ordres, cf. section 8.7.1), contrairement aux approches classiques utilisant la topologie des coupes pour lesquelles les résultats dépendent d'un ordre selon lequel les points sont sélectionnés pour être abaissés (voir la discussion dans la section 10.2).



(a) (b)

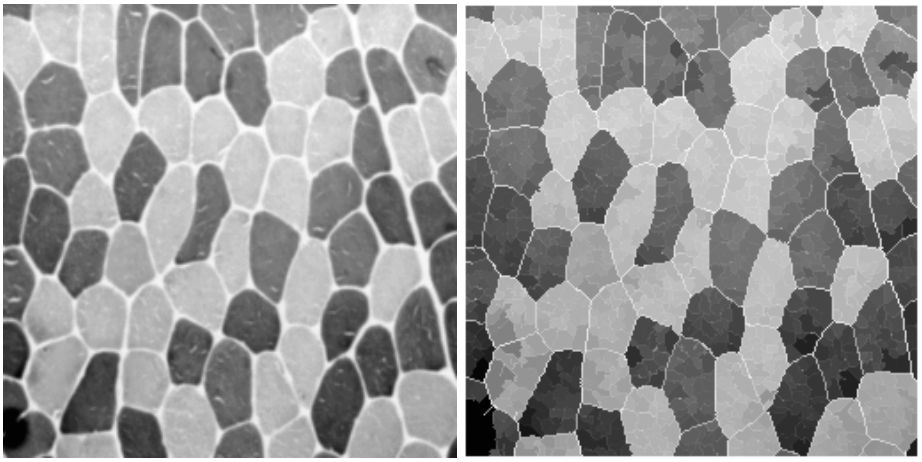
(c) (d)

FIG. 10.8 – (a) *L'image initiale en niveaux de gris* $F = F^0$, (b) F^1, (c) F^2, (d) F^3 *: un squelette ultime de* F.

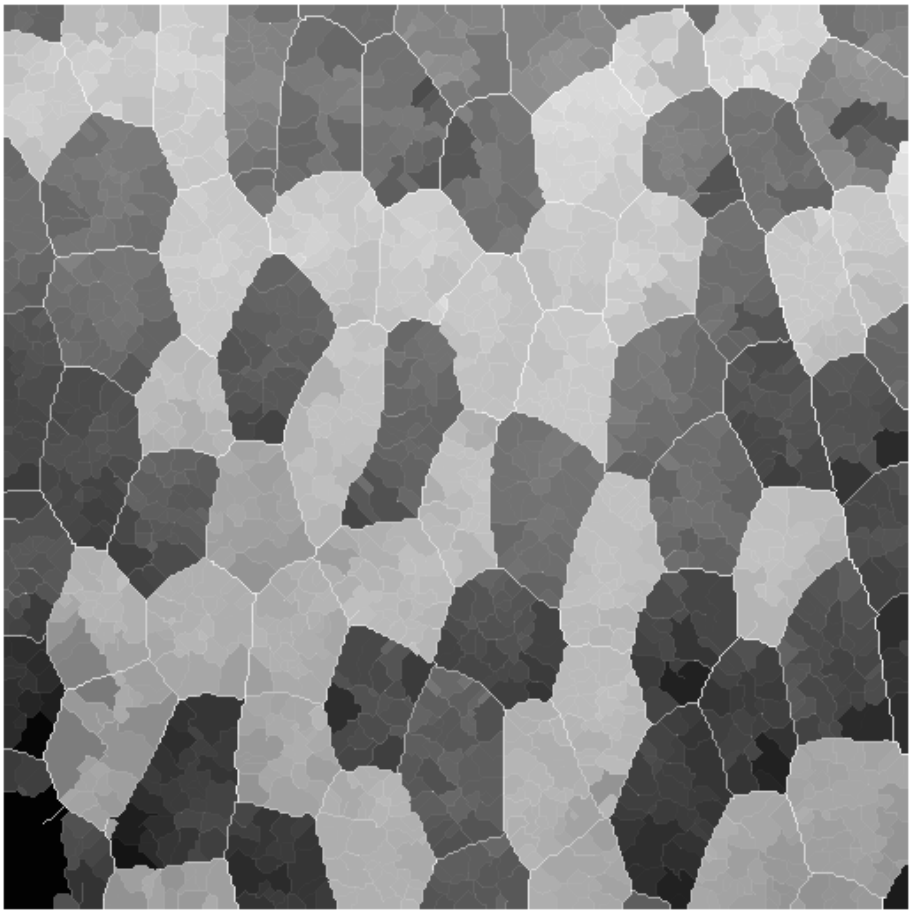


(a) (b) (c)

FIG. 10.9 – *(a) Objet initial (empreinte digitale partielle), (b) (c) un squelette curviligne de (a), respectivement avec* Φ'_3 *et* Φ'_1.



(a) (b)



(c)

FIG. 10.10 – *(a) Image de section de muscle, (b) (c) un squelette ultime de (a), respectivement avec* Φ'_3 *et* Φ'_1.

10.7 Discussion à propos de la valeur d'abaissement d'un point α-destructible

Nous avons vu dans la section 10.2 que selon l'approche topologie des coupes, la topologie était préservée lors de l'abaissement de 1 des valeurs des points destructibles, et cela de façon itérative. Nous avons rappelé que la topologie était également préservée si nous abaissions un point x destructible pour F à la valeur $val^-(x,F) = max\{F(y), \forall y \in \{z \in \Gamma_8(x),\ F(z) < F(x)\}\}$. Dans [Arc99], il est montré que certains points destructibles (points crêtes) peuvent être abaissés itérativement à une valeur inférieure à $val^-(x,F)$, tout en préservant la topologie.

En ce qui concerne notre étude dans un ordre valué, des recherches sont en cours afin de savoir si l'abaissement d'un point x α-destructible pour l'ordre valué $(X, \supseteq, F)$ à la valeur $val^-_\alpha(x,F)$ ou à $val^{--}_\alpha(x,F)$ (valeurs définies ci-après) préserve la topologie, et cela dans le cas d'un abaissement itératif ou parallèle, avec les valeurs suivantes : $val^-_\alpha(x,F) = max\{F(y), \forall y \in \alpha^{--}(x,F)\}$ et $val^{--}_\alpha(x,F) = max\{F(y), \forall y \in \alpha^{--}(x,F)$ et y non α-destructible $\}$.

Notons que si x est α-destructible, alors $\alpha^{--}(x,F) \neq \emptyset$. Remarquons que si x est un intervalle α-destructible, alors $\alpha^{--}(x,F)$ est composé d'un unique singleton z (non α-destructible), et $val^-_\alpha(x,F) = val^{--}_\alpha(x,F) = F(z)$. Nous constatons également que si x est un carré α-destructible, alors il existe au moins y appartenant à $\alpha^{--}(x,F)$ et y non α-destructible.

Nous pouvons remarquer que cette nouvelle proposition, appliquée sans modification, ne fonctionne pas selon l'approche topologie des coupes avec tous les algorithmes. En effet, considérons l'algorithme de type 2 sous-mailles, proposé par Guo et Hall (GH_A2) [GH89], étudié dans la section A.4.1 dans le cas binaire, et étendu ici au cas des images 2D en niveaux de gris. Dans l'image représentée dans le tableau 10.1, les points x et y sont destructibles. Lors de la première sous-itération de cet algorithme, les points de la première sous-maille et destructibles sont abaissés en parallèle. Si x est dans la première sous-maille, alors la variante consistant à abaisser les points destructibles x à $val^{--}(x,F)$ (= $max\{F(y), \forall y \in \{z \in \Gamma_8(x),\ F(z) < F(x)\}$ et y non destructible $\}$ se bloque ici ; en effet, seul le point y est dessous x mais y est destructible.

255	255	255	255
108	102(x)	96(y)	93
102	102	102	93
75	77	93	89

TAB. 10.1 – *Contre-exemple pour la variante d'abaissement selon l'approche topologie des coupes.*

Avec la version consistant à abaisser des points x destructibles à la valeur $val^-_\alpha(x,F)$, il se peut que des points destructibles deviennent extrémités, ce qui peut entraîner la création de nombreuses barbules *c.-à-d.* des blocs minces de points non significatifs (l'équivalent selon la topologie digitale pour images 2D binaires est : (parties de) courbe ouverte simple, voir le chapitre 11). Cela n'arrive apparamment pas si les points x destructibles sont abaissés à la valeur

$val^{--}_{\alpha}(x, F)$, mais nous n'avons pas encore montré que cela préservait la topologie, lorsque de tels points étaient abaissés en parallèle. Il semble que ce phénomène se produise également avec l'approche topologie des coupes.

Considérons les figures 10.11 et 10.12. En haut de ces figures, un même objet 2D en niveaux de gris (I, f) ; $\Phi'_1((I, f))$ est représenté aux figures 10.11 (a) et 10.12 (a) (les points non représentés sont à 0).

À la figure 10.11 (resp. 10.12) sont représentées les différentes sous-itérations de squelettisation si nous imposons à l'algorithme de squelettisation curviligne, proposé dans la section 10.6, d'effectuer la variante consistant à abaisser en parallèle les points α-destructibles x pour l'objet courant F, à la valeur $val^{-}_{\alpha}(x, F)$ (resp. à la valeur $val^{--}_{\alpha}(x, F)$).

Avec ces deux algorithmes, comme nous l'avions remarqué à la fin de la section 10.6, la première sous-itération n'abaisse aucun point et n'est pas représentée de nouveau. Nous avons les mêmes trois premières sous-itérations. Lors de la quatrième sous-itération, qui consiste à abaisser en parallèle les points β-destructibles des figures 10.11 (c) et 10.12 (c), le point fléché est β-destructible et est abaissé à 3 dans la figure 10.11 (d), tandis qu'il est abaissé à 2 dans la figure 10.12 (d), car le point donnant la valeur 3 avec le premier algorithme est lui-même β-destructible. Le raisonnement est le même en ce qui concerne les figures 10.11 (h) et (j), et la figure 10.12 (f). Dans la figure 10.11 (f), le point fléché est un point β-destructible, mais n'est pas abaissé car il est extrémité. C'est d'ailleurs ce point qui va donner naissance à une barbule, tandis qu'avec la variante, ce point va être abaissé, puis ne sera pas dans le squelette résultant, il disparaît à la figure 10.12 (h). En fait, cet exemple provient d'essais sur des images réelles d'empreintes digitales, pour lesquelles la première variante créait un grand nombre de barbules, non présentes avec la seconde variante. L'aspect en était alors presque flou.

Il semble intéressant d'étudier un autre type d'algorithme de squelettisation dans le cadre des images 2D en niveaux de gris, consistant à remplacer la sous-itération d'abaissement en parallèle de points α-destructibles dans l'algorithme actuellement proposé, en une suite de sous-itérations d'abaissement en parallèle de points α-destructibles x à la valeur $val^{-}_{\alpha}(x, F^i)$, F^i étant l'objet courant, sous-itérations répétées jusqu'à stabilité (*c.-à-d.* jusqu'à ce qu'il n'y ait plus de point α-destructible) ; nous procédons de la même façon pour l'abaissement des points β-destructibles. En pratique, cette modification entraîne moins de barbules sur les quelques exemples testés. Il reste à prouver que la topologie est préservée avec ce processus.

10.8 Conclusion

Nous avons étendu nos travaux réalisés dans le cadre des ordres pour images 2D binaires aux images 2D en niveaux de gris, définissant ainsi la notion d'ordre valué. En utilisant l'approche adoptée dans le cadre de la topologie des coupes, nous avons défini les notions de point α- ou β-destructible, et avons proposé des algorithmes de squelettisation ultime ou curviligne, en abaissant de 1 la valeur des points α- ou β-destructibles. Des recherches sont actuellement menées sur les valeurs d'abaissement de points α-destructibles, lorsque l'abaissement est itératif ou parallèle, afin de proposer des algorithmes plus efficaces encore. L'intérêt est réel si l'on souhaite étendre ces algorithmes parallèles de squelettisation aux images 3D en niveaux de gris.

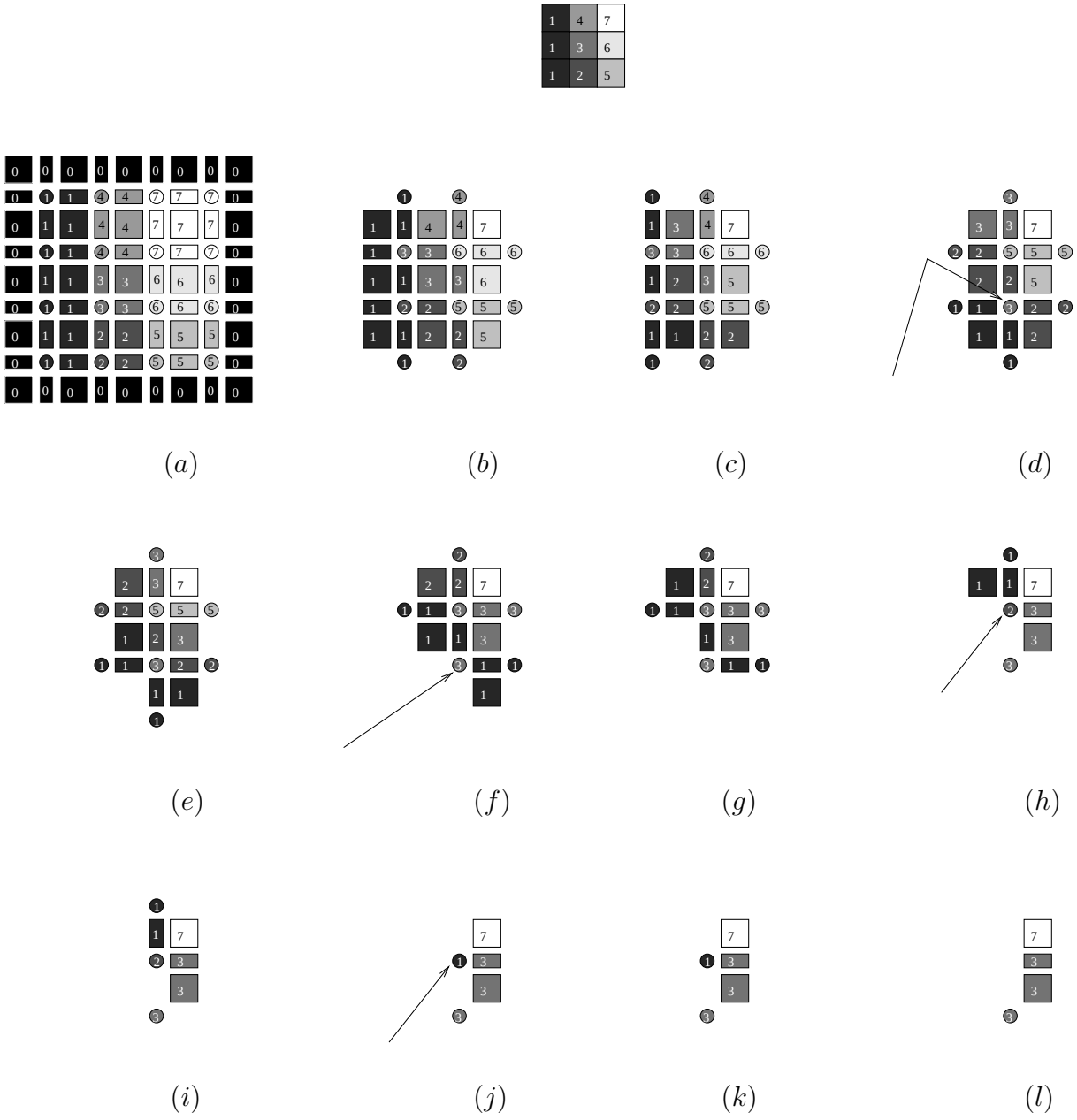


FIG. 10.11 – *Création d'une branche qui peut donner lieu à des barbules lors de l'utilisation de la première variante.*

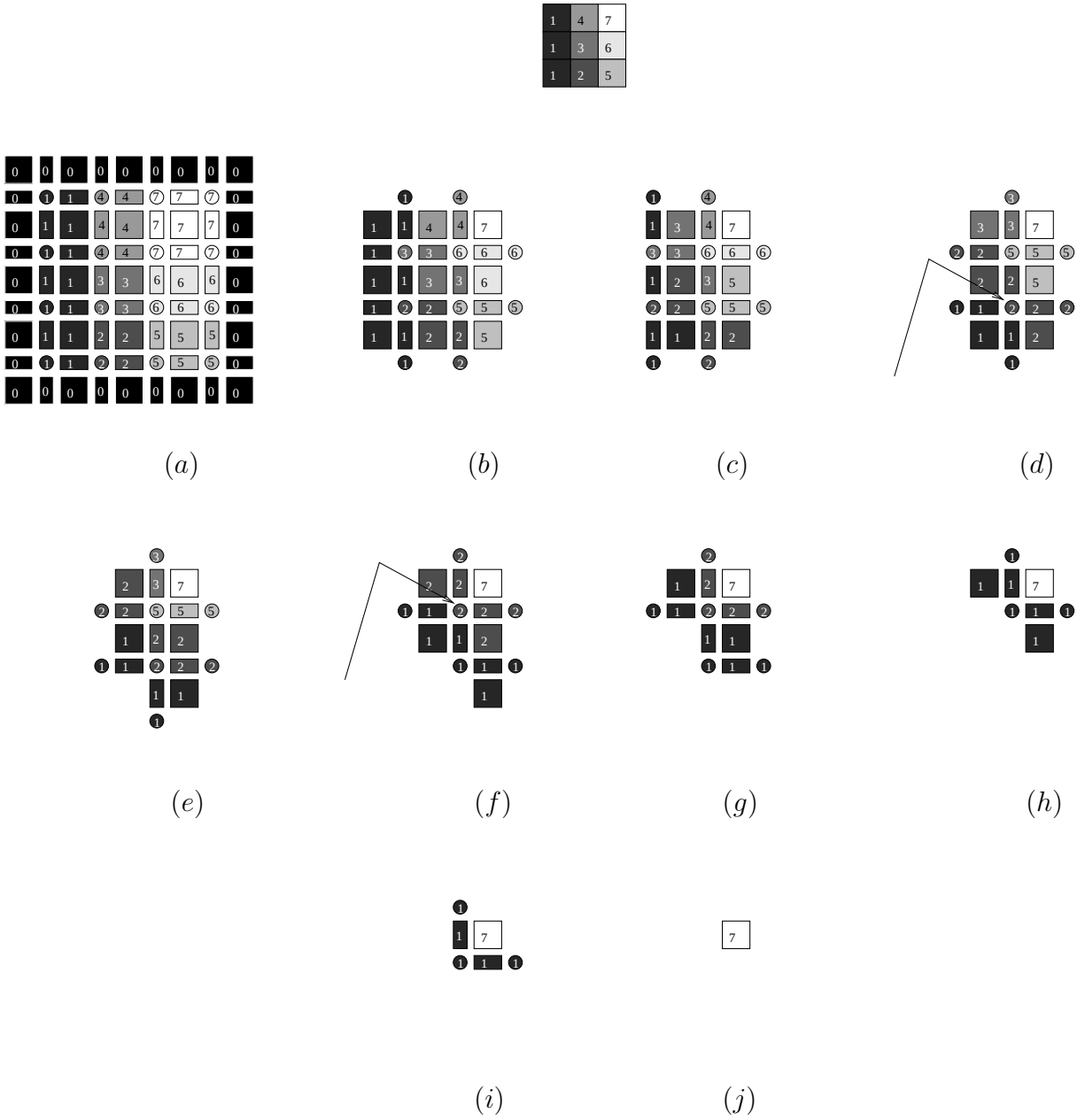


FIG. 10.12 – *À partir de la même image que celle de la figure 10.11, ici il n'y a pas de branche pouvant donner lieu à des barbules.*

Chapitre 11

Filtrage topologique d'images binaires 2D

11.1 Introduction

Dans ce chapitre, nous étudions différents moyens d'améliorer l'aspect visuel de squelettes binaires 2D. Certains squelettes comportent des petites branches non significatives de par leur longueur. Ces branches, que nous souhaitons retirer d'un squelette, sont appelées *barbules*. Dans le cadre des ordres, nous pouvons supprimer en parallèle des points simples, tout en préservant la topologie. Nous voulons exploiter cette propriété afin de réaliser un filtrage parallèle des squelettes. En proposant des solutions aux différents problèmes soulevés, nous allons imposer des contraintes permettant l'élaboration d'un algorithme de filtrage parallèle d'images binaires 2D. Nous évoquerons ensuite son extension directe au cas des images binaires 3D.

Dans la suite, nous appelons point α-simple un point α_2-simple (resp. α_3-simple) s'il est considéré pour le cas 2D (resp. 3D).

11.2 Principe général du filtrage

Nous émettons l'hypothèse de disposer d'un squelette curviligne (voir terminologie à la section 8.5.2), en entrée de l'algorithme cherché ; le filtrage d'un squelette ultime n'a pas de sens si nous souhaitons préserver la topologie puisqu'un squelette ultime ne comporte plus de point α-, ou β-simple (tout point est nécessaire afin de conserver la topologie). Nous souhaitons éliminer les barbules de ce squelette curviligne. En fait, les barbules peuvent correspondre à des parties de courbe ouverte simples, ou à des ramifications de telles courbes, leur jonction formant ce que l'on appelle un "*pâté*". Nous devons alors fournir un paramètre k fixant la longueur maximale des branches à éliminer. Afin de faciliter la discussion, nous considérons que l'image n'est formée que d'une seule composante connexe. En effet, le choix d'un paramètre peut être plus compliqué si l'on considère plusieurs composantes ; des branches de longueur k pouvant être considérées comme significatives ou non, selon la composante à laquelle elles appartiennent.

Le principe, initialement (et naturellement) proposé, est constitué de deux étapes :

• la première étape consiste à effectuer k sous-itérations de l'algorithme de squelettisation ultime à partir du squelette curviligne initial : avec $k = 2p + m$ ($m = 0$ si k est pair, $m = 1$ si k

est impair), il s'agit alors de $p + m$ sous-itérations de suppression parallèle de points α-simples, alternées avec p sous-itérations de suppression parallèle de points β-simples. Cette étape est appelée l'*étape de squelettisation conditionnée* (de paramètre k) ; le résultat de cette première étape sera appelé *le squelette conditionné*,

• la seconde étape a pour objectif de reconstruire les branches significatives (disparues lors de la première étape) en ajoutant des points de façon parallèle, tout en évitant la reconstruction de branches non significatives. Cette étape est alors constituée au plus de k sous-itérations parallèles d'ajout de points. Bien évidemment, ce processus devra respecter la topologie. Cette étape est appelée l'*étape de reconstruction*. À noter que cela ne correspond pas au sens classique de reconstruction, à savoir retrouver l'objet initial à partir du squelette et d'autres informations.

Le principe de notre reconstruction va exploiter le fait suivant : les branches non significatives vont être éliminées en premier du squelette initial lors de l'étape de squelettisation conditionnée, car leur longueur est inférieure aux branches significatives. En gardant comme information le numéro d'étape à laquelle un point a été supprimé, nous voulons déterminer quelles sont les branches significatives parmi toutes les branches (ou parties de branches) supprimées lors de la première étape, et les reconstruire par un processus de propagation selon cette information et à partir du squelette conditionné.

11.3 Considérations préliminaires pour le filtrage

11.3.1 Contrainte sur k

Nous imposons à la k-ième sous-itération de suppression de l'étape de squelettisation conditionnée d'être effective, *c.-à-d* qu'il y ait réellement suppression lors de cette sous-itération, cela afin d'avoir des points étiquetés à k pour amorcer la propagation dans l'étape de reconstruction (voir les détails à la section 11.3.2).

En pratique, si ce n'est pas le cas, l'algorithme de reconstruction affecte à k le numéro de la dernière sous-itération effective de suppression.

11.3.2 Points graines et processus de propagation pour la reconstruction

Considérons une courbe ouverte simple et k un entier strictement positif. À la fin de k sous-itérations de squelettisation conditionnée ; le squelette conditionné obtenu est une courbe ouverte simple (éventuellement réduite en un point) incluse dans celle initiale.

La seule information dont nous disposons est le numéro de la sous-itération de squelettisation conditionnée (en fait, k) durant laquelle ont été éliminés les points voisins des extrémités de la courbe (ou du point isolé) formant le squelette conditionné.

Nous souhaitons recouvrer la courbe initiale (même si cela n'avait pas de sens de filtrer une courbe ouverte simple). Il faut alors propager le processus de reconstruction en ajoutant les points voisins des extrémités du squelette conditionné (ou du point isolé le constituant), puis en ajoutant les voisins des extrémités de la courbe obtenue à la suite de la première sous-itération de reconstruction (voisins non encore ajoutés pendant la reconstruction), et ainsi de suite.

Nous avons alors besoin de points afin d'amorcer le processus de reconstruction. Nous appelons de tels points des *points graines*. D'après ce qui a été dit auparavant, initialement les points graines sont les points extrémités du squelette conditionné ou le seul point isolé le constituant (nous rappelons que le squelette curviligne en entrée du filtrage n'est composé que d'une seule composante connexe). Lors d'une sous-itération donnée de reconstruction, les points ajoutés seront les points graines de la sous-itération suivante (si sous-itération, il doit y avoir).

11.3.3 Liens entre les sous-itérations de squelettisation conditionnée et les sous-itérations de reconstruction

Les points supprimés lors de la i-ième sous-itération de suppression de l'étape de squelettisation conditionnée sont affectés de la valeur i. Lors de l'étape de reconstruction, la propagation se fait dans le sens contraire : à un instant donné, nous ajoutons certains points d'étiquette i, puis nous ajoutons certains points d'étiquette $i-1$, ... En fait, l'ensemble des points ajoutés lors de la j-ième sous-itération de reconstruction est inclus dans celui des points supprimés lors de la i-ième sous-itération de squelettisation conditionnée ; nous avons ainsi l'invariant $i+j=k+1$; par conséquent $j=k-i+1$ et j prend alors les valeurs successives de 1 à k lorsque i prend les valeurs successives de k à 1.

Considérons un point d'étiquette i ajouté lors de la j-ième sous-itération de la reconstruction. Si j et k sont pairs, ou si j et k sont impairs (resp. si j est pair et k est impair, ou si j est impair et k est pair) alors i est impair (resp. pair), ce qui signifie que le point était α-simple (resp. β-simple) lorsqu'il a été retiré lors de la i-ième sous-itération de squelettisation conditionnée.

Cette remarque n'intervient en rien actuellement dans le fait d'ajouter des points d'étiquette i lors de la j-ième étape de reconstruction. Néanmoins, elle aura son importance par la suite, comme nous le verrons dans la section 11.7.6.

11.3.4 Différents tests de la valeur k en pratique

Il existe deux possibilités immédiates : soit nous donnons un paramètre k qui sert à la fois à la squelettisation conditionnée et à la reconstruction (VAR_1), soit nous l'utilisons uniquement pour l'étape de reconstruction, l'étape de squelettisation conditionnée étant alors remplacée par une étape de squelettisation ultime (*c.-à-d.* produisant un squelette ultime) (VAR_2).

Comme nous l'avons remarqué auparavant, suite à l'étape de squelettisation conditionnée, une initialisation fixe des *points graines* par lequel le processus de reconstruction va se propager. Ces graines sont les points extrémités (ou point isolé) du squelette conditionné (pour VAR_1). Avec la variante VAR_2, nous disposons d'un squelette ultime (contenu dans le squelette conditionné, par construction), qui ne comporte aucun point extrémité (par le processus même de squelettisation ultime). Il suffit alors de considérer les points d'étiquette supérieure à k (dans l'image étiquetée par les sous-itérations de suppression lors du calcul du squelette ultime) comme des points du squelette conditionné et de se reporter au cas VAR_1. L'avantage d'adopter VAR_2 est de n'exécuter qu'une seule fois la première étape de squelettisation et de pouvoir tester plusieurs valeurs pour le paramètre k pour la reconstruction (bien sûr, après avoir mis à jour les points

graines, comme expliqué précédemment).

Dans la suite, nous ne considérons que la variante VAR_1.

11.4 Conventions de représentation

Nous avons adopté les conventions suivantes. Si une figure représente les différentes étapes de la squelettisation conditionnée, les points de l'objet final sont représentés en trait plein et les points supprimés sont représentés en pointillé avec l'indication (si nécessaire) du numéro de sous-itération lors de laquelle ils ont été supprimés (Repr1). Si une figure représente les différentes étapes de reconstruction, les points initiaux sont en trait plein et les points ajoutés sont en pointillé avec l'indication (si nécessaire) du numéro de sous-itération de reconstruction lors de laquelle ils ont été ajoutés (Repr2). Une autre représentation consiste à afficher uniquement le résultat final d'une ou de plusieurs étapes (Repr3). Parfois des couleurs sont utilisées à la place du pointillé pour illustrer la propagation des suppressions ou les ajouts parallèles de points, en complément des numéros d'itération. Ces couleurs sont liées aux points et à la sous-itération de la squelettisation conditionnée durant laquelle ils ont été supprimés, et permettent de visualiser plus rapidement quels points ne sont pas ajoutés lors de la reconstruction parmi ceux supprimés lors de l'étape de squelettisation conditionnée correspondante.

11.5 Discussion à propos des points supprimés lors de la première étape

En entrée de l'algorithme de filtrage, est fourni un squelette curviligne. Nous pouvons nous demander ce qui se passerait si on effectuait k sous-itérations de suppression parallèle de points α- ou β-simples et extrémités. Avec $k = 2p + m$ ($m = 0$ si k est pair, $m = 1$ si k est impair), il s'agit alors de $p + m$ sous-itérations de suppression parallèle de points α-simples et extrémités, alternées avec p sous-itérations de suppression parallèle de points β-simples et extrémités.

Considérons l'objet de la figure 11.1 (a). À la figure 11.1 (b) est montré le résultat à la fin de la première étape si celle-là effectue k sous-itérations alternées de suppression parallèle de points α-simples et extrémités, ou β-simples et extrémités ($k = 5$). Le processus se propage le long des courbes ouvertes simples. En revanche, il est bloqué au niveau de la jonction initiale des courbes ouvertes simples. En effet, à ce niveau les points sont α- ou β-simples mais non extrémités. Le processus ne se propage alors pas plus loin dans la barbule. Le résultat de la reconstruction est représenté à la figure 11.1 (c).

À la figure 11.1 (d), est montré le résultat à la fin de la première étape si celle-là effectue k sous-itérations de squelettisation ultime, comme nous l'avons proposé initialement. À la figure 11.1 (e), est représenté le résultat de la reconstruction : la barbule a été éliminée. Nous nous apercevons que la barbule éliminée dans ce cas, n'est pas simplement une partie d'une courbe ouverte simple, elle comporte une jonction de courbes ouvertes simples.

Cet exemple a motivé notre choix de prendre comme première étape k sous-itérations de squelettisation ultime.

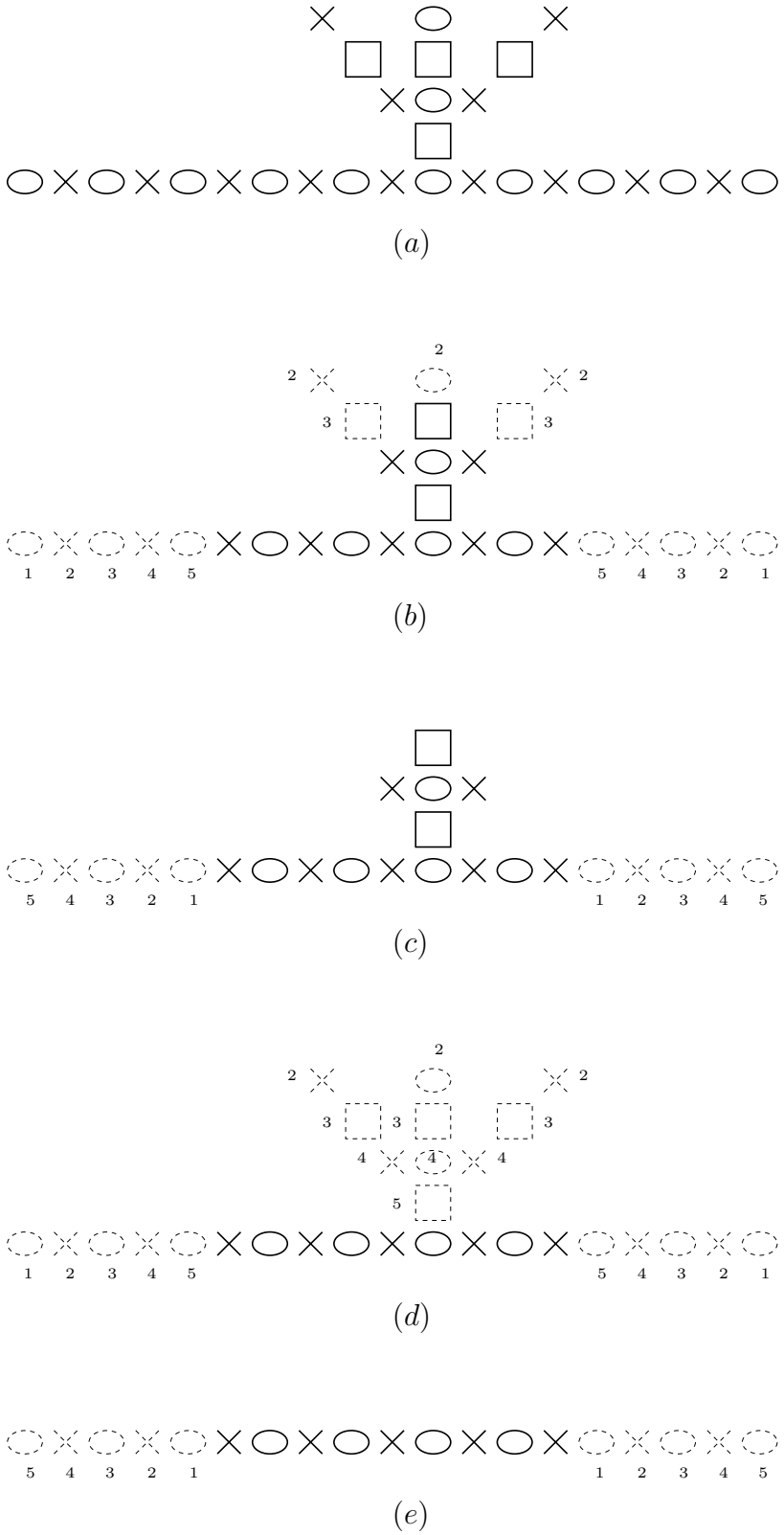


FIG. 11.1 – *Illustration des choix réalisés durant la première étape du filtrage.* (a) *Objet initial,* (b) *résultat en supprimant des points* α*-simples ou* β*-simples et extrémités (*Repr1*),* (c) *reconstruction à partir de* (b) *(*Repr2*),* (d) *squelette conditionné (*Repr1*),* (e) *reconstruction à partir de* (d) *(*Repr2*).*

11.6 Discussion à propos des points graines lors de l'étape de reconstruction

Dans la section 11.3.2, les points graines considérés initialement sont les points extrémités ou le point isolé du squelette conditionné. Lors de la j-ième sous-itération de reconstruction donnée, nous ajoutons les points d'étiquette $i = k - j + 1$ (voir section 11.3.3), voisins des points graines et n'appartenant pas encore à l'objet reconstruit ; ces points deviendront les points graines de la prochaine sous-itération de reconstruction (si $j < k$).

Nous pouvons nous demander ce qui se passerait si plutôt que de fixer les points graines pour la future sous-itération de reconstruction, nous imposons aux points graines d'être les points extrémités (ou isolé) de l'objet reconstruit tel qu'il est au début de la sous-itération en cours.

Considérons l'objet de la figure 11.2. L'étape de squelettisation conditionnée ($k = 6$) aboutit à ne préserver que le carré central, qui devient graine pour la première sous-itération de reconstruction. Sont alors ajoutés les 8 éléments de H^2 que le carré contient ; ces points sont non extrémités ; et le processus de reconstruction s'arrête ici, si les graines que l'on considère sont les points isolés ou extrémités de la sous-itération de la reconstruction courante.

Si nous considérons initialement comme graines le(s) point(s) isolé ou extrémité(s) du squelette conditionné, et si nous ajoutons les points voisins des graines avec la bonne étiquette, et si ces points deviennent à leur tour points graines pour la sous-itération suivante de reconstruction, nous retrouvons alors l'objet initial de la figure 11.2 (notons tout de même que de fixer le paramètre $k = 6$ sur cet exemple n'est sûrement pas judicieux, ici, toutes les branches peuvent être considérées comme significatives). Cet exemple nous a motivé pour garder ce choix concernant les points graines lors de l'étape de reconstruction. En fait, ce processus, bien que naturel, n'est pas correct. Nous le montrerons un peu plus tard à la section 11.7.6.

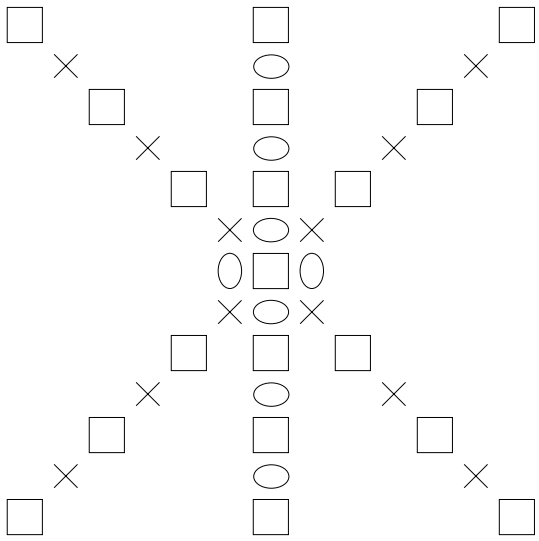


FIG. 11.2 – *Illustration des choix réalisés durant la seconde étape du filtrage.*

11.7 Élaboration de l'algorithme

Nous observons le comportement de l'algorithme actuellement proposé. Un défaut majeur (la non-préservation de la topologie) sera mis en évidence puis corrigé.

11.7.1 Absence de point graine pour l'étape de reconstruction

FIG. 11.3 – (a) *Squelette conditionné (*$k = 4$*) (*Repr1*),* (b) *reconstruction (*Repr2*).*

Considérons l'objet de la figure 11.3 (a). Le paramètre k utilisé a la valeur 4, c'est la longueur de la branche parasite que nous souhaitons éliminer. À la fin de l'étape de squelettisation conditionnée, il n'y a pas de point isolé ou extrémité. Il n'y a alors pas de point graine, et aucun point ne peut être ajouté lors de l'étape de reconstruction (Fig. 11.3 (b)).

La partie reconstruction de l'algorithme doit alors vérifier la présence de points graines avant de pouvoir se dérouler. L'étape de reconstruction effectue donc au plus k sous-itérations de reconstruction, même s'il y a réellement eu k sous-itérations effectives de suppression durant l'étape de squelettisation conditionnée (c'est le cas sur l'exemple de la figure 11.3).

11.7.2 Élimination d'une barbule et reconstruction

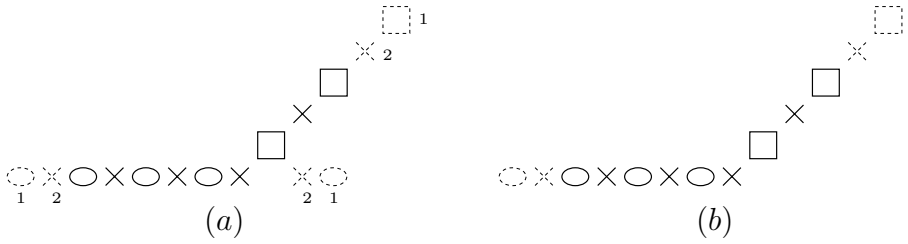


FIG. 11.4 – (a) *Squelette conditionné (*$k = 2$*) (*Repr1*),* (b) *reconstruction (*Repr2*).*

Considérons l'exemple de la figure 11.4 (a) ($k = 2$). La petite barbule de longueur 2 (dans la partie inférieure droite de l'objet) n'a pas été reconstruite car elle n'est pas voisine d'un point graine (Fig. 11.4 (b)). Les deux autres parties de longueur 2, éliminées lors de la phase de squelettisation conditionnée, ont été reconstruites car elles sont voisines d'un point graine. Notons sur cet exemple, que nous retrouvons la branche de plus grande longueur du squelette initial.

11.7.3 Reconstruction d'une ramification

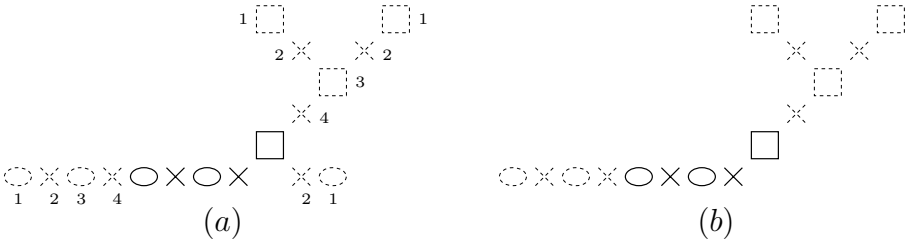


FIG. 11.5 – (a) *Squelette conditionné (*$k = 4$*) (*R*epr*1*), *(b) *reconstruction (*R*epr*2*).*

Considérons l'exemple de la figure 11.5 (a) ($k = 4$). Nous avons éliminé la barbule de longueur 2, et reconstruit deux branches de longueur 4 dont une se termine par une ramification composée de deux petites branches de longueur 2 (Fig. 11.5 (b)). En fait, les deux branches de longueur 2 formant la ramification ont été reconstruites car elles sont voisines d'un point graine lors de la deuxième itération de reconstruction.

11.7.4 Problème au niveau des jonctions de courbes

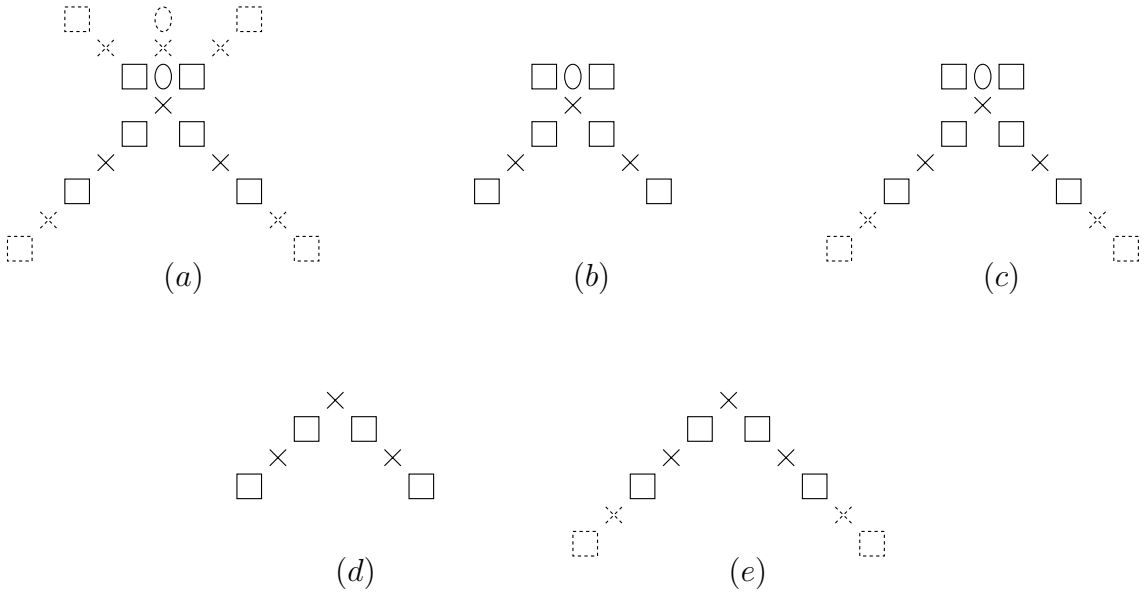


FIG. 11.6 – *Élimination des barbules issues d'un pâté.* (a) *Squelette conditionné (*$k = 2$*) (*R*epr*1*),* (b) *squelette curviligne (*R*epr*3*),* (c) *reconstruction (*R*epr*2*),* (d) *squelette curviligne du squelette conditionné* $(k = 2)$*,* (e) *reconstruction à partir de* (d)*.*

Considérons la figure 11.6 (a) ($k = 2$). L'objet obtenu (Fig. 11.6 (b)) à la fin la squelettisation conditionnée comporte des points simples non extrémités (ce n'est ni un squelette curviligne, ni

un squelette ultime). Le résultat de la reconstruction à partir de cet objet est représenté à la figure 11.6 (c), et contient encore ces points simples non extrémités. Nous choisissons alors d'effectuer une étape de squelettisation curviligne à la fin de la squelettisation conditionnée puis de faire la reconstruction.

L'étape de squelettisation curviligne va éliminer les deux carrés ainsi que l'intervalle, formant le pâté, car ils sont tous trois α-simples et non extrémités (Fig. 11.6 (d)). Finalement, aucune branche du pâté ne va être reconstruite (Fig. 11.6 (e)). En fait, sur cet exemple, les barbules de longueur $k = 3$ ont été éliminées. Nous aurions obtenu le même résultat pour k prenant les valeurs 3, 4 ou 5.

Sur ce même objet, le résultat serait également le même, si on déplaçait l'étape de squelettisation curviligne à la fin de celle de reconstruction.

11.7.5 "Effet de bord"

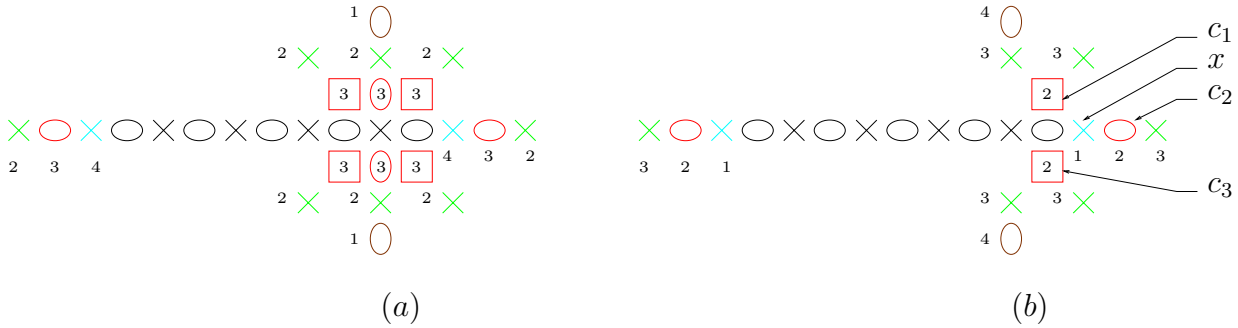


FIG. 11.7 – *"Effet de bord". (a) Squelette conditionné ($k = 4$) (*Repr1*), (b) squelette curviligne et reconstruction (*Repr2*).*

Considérons la figure 11.7 (a) ($k = 4$). Nous reconstruisons une partie du "pâté" (11.7 (b)) car après l'ajout du singleton x d'étiquette $i = 4$, il y a 3 points candidats c_1, c_2, et c_3 à être ajoutés (de valeur $i = 3$ et dans le voisinage du singleton x). En fait, ce problème d'"effet de bord" peut ne pas préserver la topologie. C'est ce que nous constatons dans la section suivante. Tout cela montre que les problèmes liés aux pâtés sont délicats ; il semble alors difficile d'obtenir une solution générale s'adaptant à tous les cas possibles.

11.7.6 Préservation de la topologie lors de la reconstruction

Considérons l'objet de la figure 11.8 (a) ($k = 4$). Une partie du pâté va être éliminée à la suite de la squelettisation conditionnée. Après la phase de squelettisation (inutile ici, car il n'y a pas de point redondant dans le squelette conditionné), l'étape de reconstruction produit l'objet dessiné à la figure 11.8 (b). Nous pouvons constater la création d'une composante connexe du complémentaire, non présente à l'itération précédente, et non présente dans le squelette curviligne initial de la figure 11.8 (a). Ainsi, la topologie n'est pas préservée. En fait, nous constatons la présence

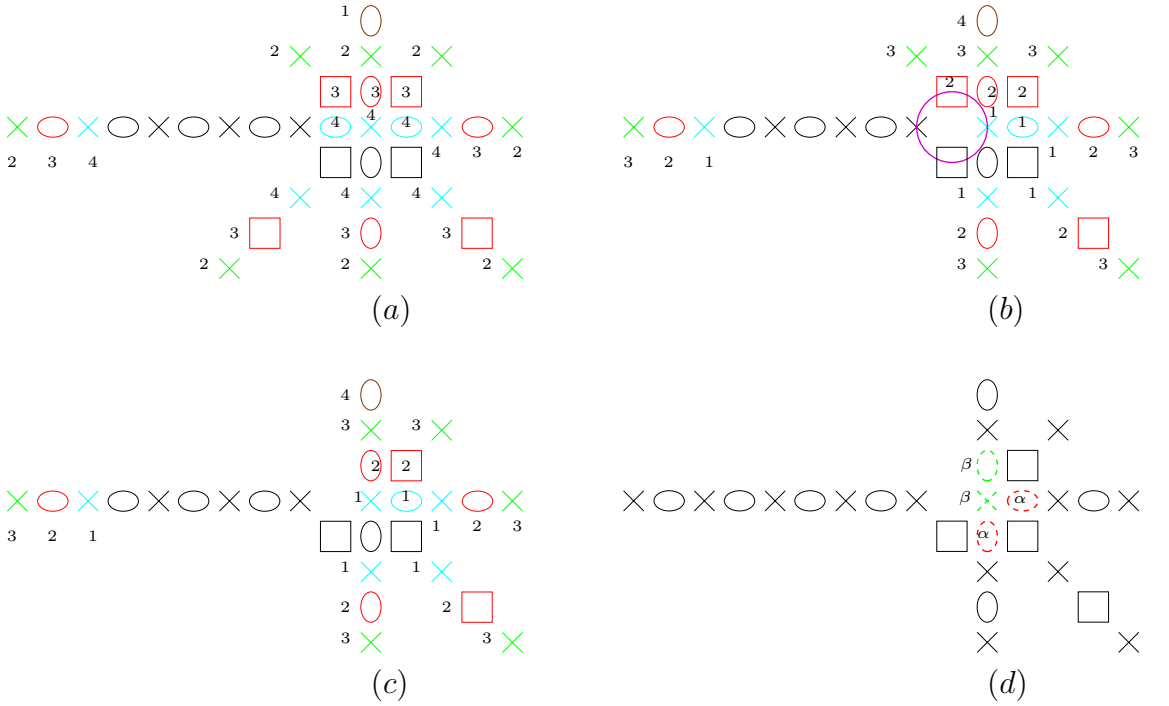


FIG. 11.8 – *Non préservation de la topologie.* (a) *Squelette conditionné (*$k = 4$*)* (Repr1)*,* (b) *squelette curviligne et reconstruction (*Repr2*),* (c) *reconstruction par ajout de points simples,* (d) *squelette curviligne (sans les points en pointillés).*

d'une courbe fermée matérialisée par le cercle violet, délimitant cette nouvelle composante du complémentaire ; c'est l'analogue d'un trou 2D selon l'approche topologie digitale.

Cet exemple montre qu'en ajoutant des points uniquement selon leur étiquette, la topologie peut ne pas être préservée. Une solution est de vérifier si les points ajoutés sont α- ou β-simples, selon l'itération courante (voir la section 11.3.3). En faisant cela, nous obtenons l'objet de la figure 11.8 (c). Nous remarquons, de nouveau, la présence de points redondants. Nous effectuons alors une nouvelle étape de squelettisation curviligne et obtenons l'objet de la figure 11.8 (d) (privé des points en pointillé notés α ou β, indiquant la phase lors de laquelle ces points ont été supprimés, durant 3 sous-itérations successives de suppression).

Remarque : Dans la section 8.4 était énoncée la propriété suivante : la suppression en parallèle de points α-simples (ou β-simples) préserve la topologie. Ici nous utilisons la propriété duale, à savoir, l'ajout en parallèle de points α-simples (ou β-simples) préserve la topologie (propriété obtenue par passage par le complémentaire). Cela confirme la validité de notre algorithme de filtrage en raison des opérations mises en œuvre et préservant la topologie (squelettisation curviligne, squelettisation conditionnée, reconstruction par ajout parallèle de points α- ou β-simples).

11.8 Algorithme de filtrage

L'algorithme de filtrage est décrit au schéma 16.

11.9 Résultats 2D

Nous présentons les résultats du filtrage sur l'image lettre "A" obtenus pour différentes valeurs du paramètre k.

Sur la figure 11.9, sont représentés l'image originale (lettre "A"), un squelette ultime, un squelette curviligne (après utilisation de la transformation Φ_1, voir section 8.2.2), ainsi que les résultats du filtrage selon différentes valeurs de k. La figure 11.10 est la représentation tableau de l'objet initial (singleton en bleu, intervalle en gris, carré en vert). Cette figure récapitule les différentes parties qui sont éliminées, ainsi que le paramètre utilisé pour leur suppression.

Les résultats semblent satisfaisants, nous observons la reconstruction de certaines branches pour différentes valeurs de k (communes pour la squelettisation conditionnée et la reconstruction). Nous observons également que le filtrage (squelettisation conditionnée et reconstruction) pour le paramètre $k \geq 56$ donne le squelette ultime du squelette curviligne donné en entrée, c'est normal ici car il n'y a plus de point graine à ce stade, pour cette image ; cela n'est pas valable par exemple pour une courbe ouverte simple (il resterait un point isolé qui serait graine).

11.10 Conclusion pour le cas 2D

Tous les cas étudiés précédemment n'apparaissent peut-être pas, si le squelette curviligne en entrée de l'algorithme est le résultat d'une squelettisation curviligne à partir d'un objet initial après les transformations Φ_1 ou Φ_2 (cf. section 8.2.2).

Le principal intérêt de la méthode repose sur une suppression parallèle de points simples et une reconstruction par ajout en parallèle de points simples, guidée par les étiquettes ayant pour valeur le numéro de sous-itération à laquelle ils ont été retirés.

À la section 11.11, nous cherchons à étendre notre étude au cas des images 3D binaires.

11.11 Extension au cas des images binaires 3D

Nous souhaitons maintenant étendre l'étude précédente au cas des images binaires 3D.

Nous supposons disposer en entrée de l'algorithme de filtrage d'un squelette curviligne ou surfacique (dans ce dernier cas tous les points simples sont terminaux de surface). Le schéma de l'algorithme de filtrage reste inchangé, la seule modification concerne les points graines : nous considérons comme points graines initialement, les points terminaux de courbe pour les squelettes curvilignes, et les points terminaux de surface pour les squelettes surfaciques.

Considérons l'objet initial de la figure 11.11 (a), qui est un plan d'épaisseur 1 et de largeur 5 ; cet objet est un squelette surfacique. Même si l'objet ne présente pas de barbules, nous nous

Filtrage(X, k) :

Y = Squelette conditionné(X, k) (voir ci-dessous) puis
Résultat = Reconstruction(Y, k) (voir ci-dessous)
(avec des phases intermédiaires de squelettisation curviligne,
voir sections 11.7.4 et 11.7.6).

Squelette conditionné(X, k) : (k adéquat - voir la section 11.3.1)

$S^0 = X$
Pour i allant de 1 à k **faire**
  Si i est impair **alors** Y=$\{y \in S^{i-1}$ tel que y est α-simple $\}$
    Sinon Y=$\{y \in S^{i-1}$ tel que y est β-simple $\}$
  Étiqueter les points de Y à i
  S^i=$S^{i-1} \setminus Y$
Fin Pour
Tout point de l'objet initial non étiqueté prend la valeur $PT\_SQUELETTE$

Reconstruction(Y, k) :

En // Détection des points graines (point isolé ou point(s) extrémité(s))
Si nombre de points graines > 0 **alors** continuer=$VRAI$ **sinon** continuer =$FAUX$
itération=k
Tant que itération> 0 et continuer$== VRAI$
  continuer = $FAUX$
  En // **Pour** tout point x tel que étiquette$(x) == PT_GRAINE$
    En // **Pour** tout $y \in \theta^{\square}(x)$ tel que étiquette(y) ==itération
      et y α- ou β-simple pour l'objet au début
      de la sous-itération, voir section 11.3.3
      Faire étiquette(y)=PT_FUTURE_GRAINE et continuer=$VRAI$
  En // PT_GRAINE = $PT\_SQUELETTE$
  Si itération$== 1$ **alors** En // PT_FUTURE_GRAINE = $PT\_SQUELETTE$
    Sinon En // PT_FUTURE_GRAINE = PT_GRAINE
  itération = itération-1
Fin Tant que
Le résultat est l'ensemble des points étiquetés à $PT\_SQUELETTE$

Notations : "=" pour l'affectation et "==" pour le test de l'égalité.

Algorithme 16: *Algorithme de filtrage.*

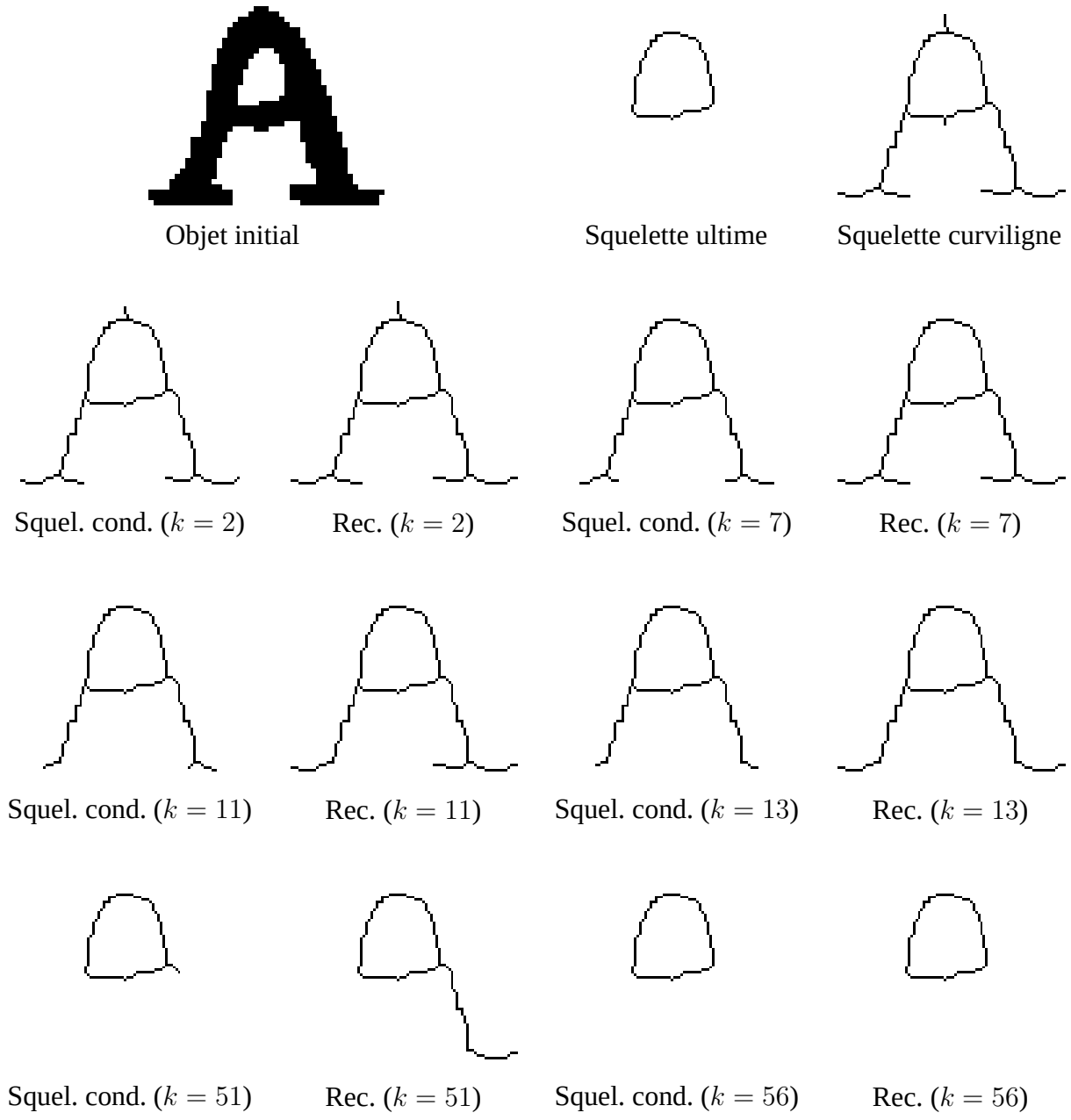


FIG. 11.9 – *Filtrages pour différentes valeurs de paramètre k à partir du squelette curviligne 2D de l'objet "lettre A" (images inverses).*

attendons à ce qu'il soit recouvré par l'algorithme de filtrage. Après la squelettisation conditionnée $(k = 3)$, l'objet obtenu est un squelette curviligne (Fig. 11.11 (b)). Si nous prenons comme graines les points terminaux de surface, seules les extrémités du squelette curviligne seront considérées, et l'étape de reconstruction ajoutera deux petits bouts de surface autour des points extrémités (Fig. 11.11 (c)). La surface initiale n'est pas entièrement reconstruite.

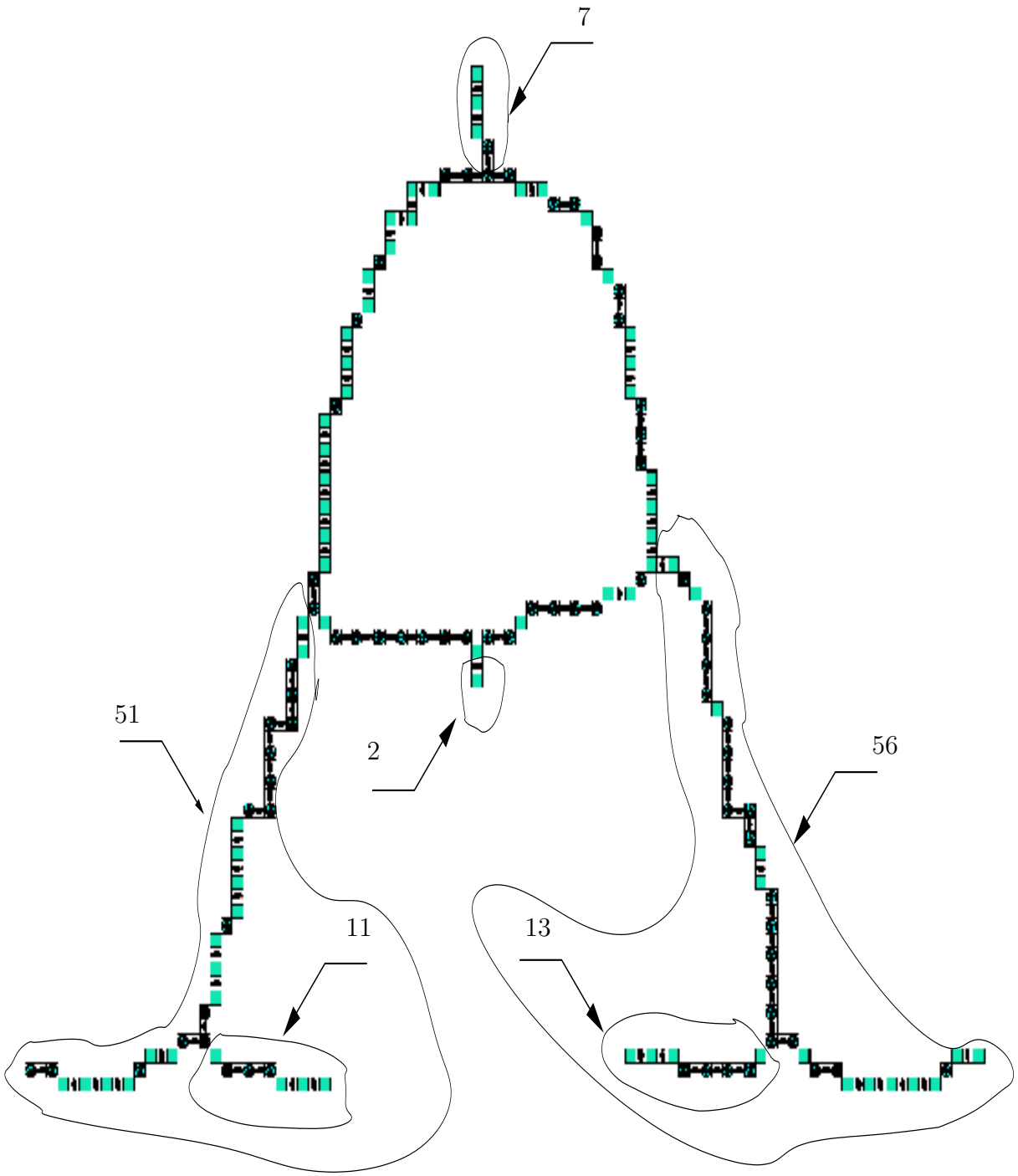
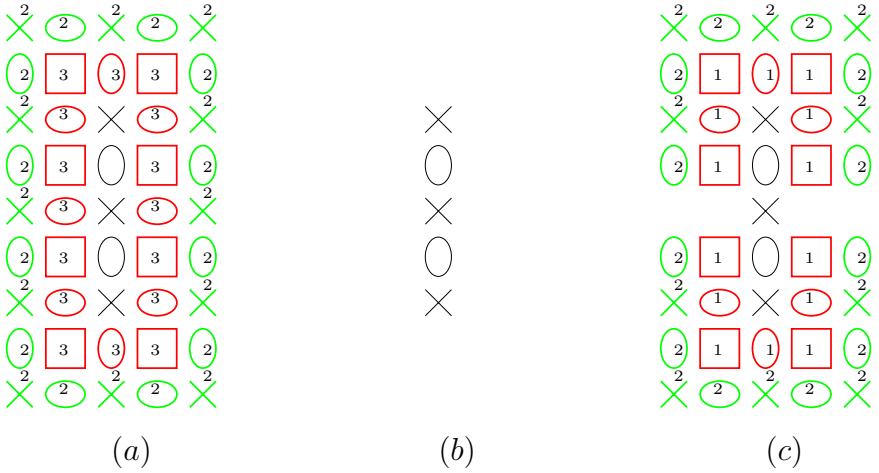


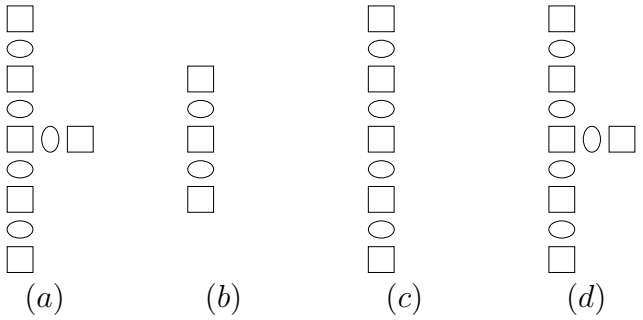
FIG. 11.10 – *Filtrages pour différentes valeurs du paramètre k à partir du squelette curviligne 2D de l'objet "lettre A", et paramètres pour la suppression des parties entourées.*



(a) (b) (c)

FIG. 11.11 – *Effet de bord.* (a) *Squelette conditionné (*$k = 3$*) (*Repr1*),* (b) *(*Repr3*) et* (c) *reconstruction (les graines étant les points isolés ou terminaux de surface) (*Repr2*).*

Si l'algorithme est modifié de façon à considérer comme points graines tous les points restants à la suite de l'étape de noyau conditionné (*c.-à-d.* les points du squelette conditionné), dans ce cas la surface initiale sera entièrement reconstruite. Mais cette approche n'est plus cohérente avec le travail réalisé en 2D. En effet, considérons l'objet 3D de la figure 11.12 (a). Dans l'étape de reconstruction, la barbule n'est pas reconstruite si nous ne considérons que les points extrémités comme points graines de notre reconstruction (Fig. 11.12 (c)) (c'est également le résultat obtenu pour cet objet considéré dans le cas 2D). En revanche, si nous considérons tous les points du squelette conditionné comme des points graines, la barbule est reconstruite (Fig. 11.12 (d)) (et on retrouve l'objet initial), et l'objet initial comportant la barbule est retrouvé.



(a) (b) (c) (d)

FIG. 11.12 – (a) *Objet initial,* (b) *squelette conditionné (*$k = 2$*) (*Repr3*),* (c) *reconstruction (les graines étant les points isolés ou terminaux de surface) (*Repr3*) et* (d) *reconstruction (les graines étant les points du squelette conditionné) (*Repr3*).*

En fait, dans l'exemple de la figure 11.11, nous pouvons considérer que le filtrage d'un bout de plan n'avait pas de sens, ou que le paramètre n'était pas judicieux ; l'effet de bord étant plus impressionnant dans le cas 3D que dans le cas 2D (voir Fig. 11.7).

À la figure 11.13, sont représentés les résultats des filtrages pour différentes valeurs du paramètre k, sur un squelette curviligne 3D. Pour $k = 2$, une barbule est éliminée, et pour $k \geq 4$ une autre barbule est éliminée ; le résultat est ici une courbe ouverte simple.

Aux figures 11.14, 11.15 et 11.16 sont représentés les résultats des filtrages pour différentes valeurs du paramètre k, sur un squelette surfacique 3D.

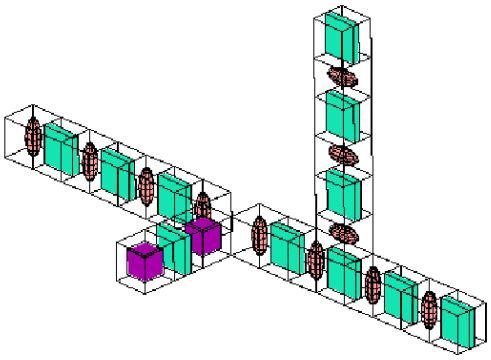
Nous avons également proposé une représentation tableau des objets initiaux des figures 11.13 à 11.16 (singleton en bleu, intervalle en orange, carré en vert, cube en violet).

À la figure 11.14, pour $k \geq 4$, un effet de bord apparaît (tout comme dans l'exemple de la figure 11.11) ; pour $k \leq 3$, l'objet initial est retrouvé (ce qui serait le cas de l'objet de la figure 11.11 pour $k \leq 2$. À la figure 11.15, l'effet de bord n'est pas présent pour $k \leq 6$, et est présent pour $k \geq 7$. Dans le cas $k \leq 6$, l'objet initial est retrouvé ; rien n'est filtré, nous pouvons considérer que toutes les surfaces sont significatives. À la figure 11.16 est représenté l'objet "visage" composé d'une face, de deux yeux, d'un nez et d'une bouche. Pour $k \leq 2$, l'objet est inchangé, pour $3 \leq k \leq 4$, les yeux disparaissent, pour $5 \leq k \leq 8$, la bouche et le nez disparaissent à leur tour ; nous obtenons alors uniquement un plan "propre", correspondant à la face. Pour $k \geq 9$, nous retrouvons l'effet de bord.

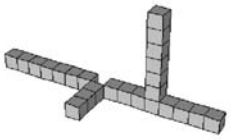
11.12 Conclusion pour le cas 3D

Il n'y a pas de problème majeur pour le filtrage des squelettes curvilignes 3D, pas plus que dans le cas 2D.

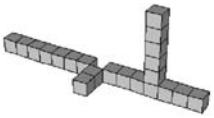
Dans le cas des squelettes 3D surfaciques, nos résultats présentent des effets de bord plus importants que dans le cas des filtrages de squelettes 2D. En fait, cela est dû à un mauvais choix du paramètre k ou à des dimensions non négligeables des parties que nous souhaitons filtrer par rapport à celles que nous souhaitons garder ; les problèmes se situent généralement au niveau des jonctions de surface. Néanmoins, les résultats semblent satisfaisants.



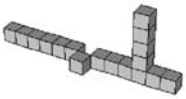
Objet initial



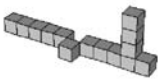
Objet initial



Squel. cond. ($k = 1$)



Squel. cond. ($k = 2$)



Squel. cond. ($k = 3$)

Squel. cond. ($k = 4$)

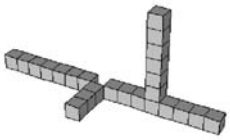
Squel. cond. ($k = 5$)

Squel. cond. ($k = 6$)

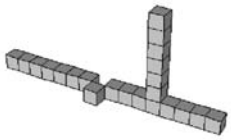
Squel. cond. ($k = 7$)

Squel. cond. ($k = 8$)

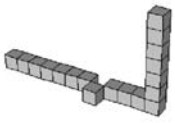
Squel. cond. ($k = 9$)



Rec. ($k = 1$)

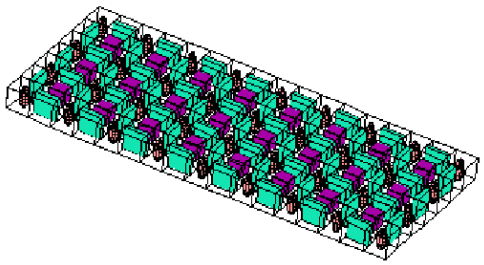


Rec. ($k = 2$)

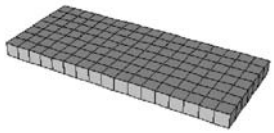


Rec. ($k \geq 4$)

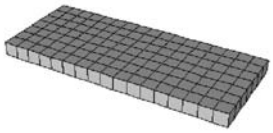
FIG. 11.13 – *Filtrages et reconstructions pour différentes valeurs du paramètre k, objet "filaire".*



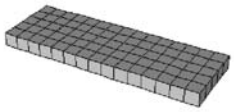
Objet initial



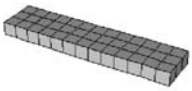
Objet initial



Squel. cond. ($k = 1$)



Squel. cond. ($k = 2$)



Squel. cond. ($k = 3$)

Squel. cond. ($k = 4$)

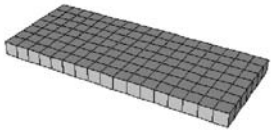
Squel. cond. ($k = 5$)

Squel. cond. ($k = 6$)

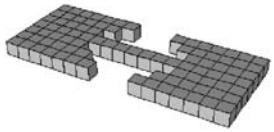
Squel. cond. ($k = 7$)

Squel. cond. ($k = 8$)

Squel. cond. ($k = 9$)

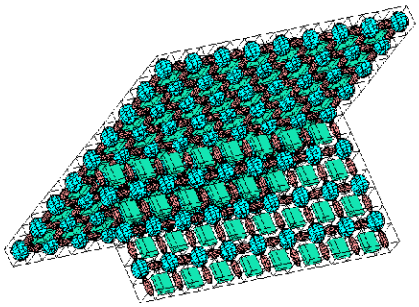


Rec. ($k \leq 3$)

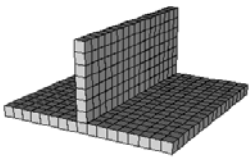


Rec. ($k \geq 4$)

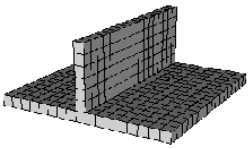
FIG. 11.14 – *Filtrages et reconstructions pour différentes valeurs du paramètre k, objet "plan".*



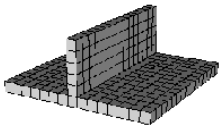
Objet initial



Objet initial



Squel. cond. ($k = 1$)



Squel. cond. ($k = 2$)

Squel. cond. ($k = 3$)

Squel. cond. ($k = 4$)

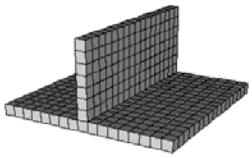
Squel. cond. ($k = 5$)

Squel. cond. ($k = 6$)

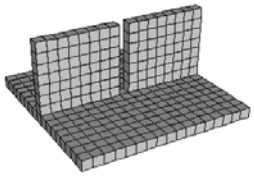
Squel. cond. ($k = 7$)

Squel. cond. ($k = 8$)

Squel. cond. ($k = 9$)



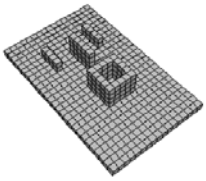
Rec. ($k \leq 6$)



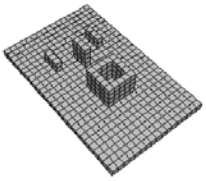
Rec. ($k \geq 7$)

FIG. 11.15 – *Filtrages et reconstructions pour différentes valeurs du paramètre k, objet "ping-pong".*

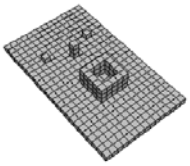
Objet initial



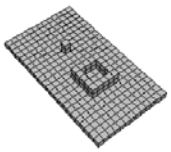
Objet initial



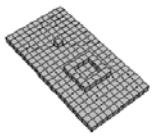
Squel. cond. ($k = 1$)



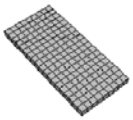
Squel. cond. ($k = 2$)



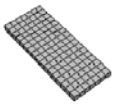
Squel. cond. ($k = 3$)



Squel. cond. ($k = 4$)



Squel. cond. ($k = 5$)

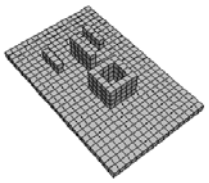


Squel. cond. ($k = 6$)

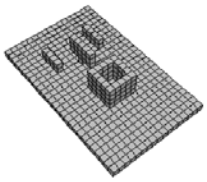
Squel. cond. ($k = 7$)

Squel. cond. ($k = 8$)

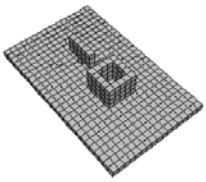
Squel. cond. ($k = 9$)



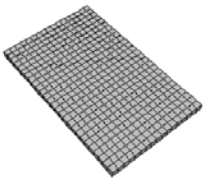
Rec. ($k = 1$)



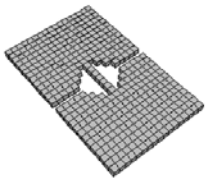
Rec. ($k = 2$)



Rec. ($3 \leq k \leq 4$)



Rec. ($5 \leq k \leq 8$)



Rec. ($k \geq 9$)

FIG. 11.16 – *Filtrages et reconstructions pour différentes valeurs du paramètre k, objet "visage".*

11.13 Conclusion générale pour le filtrage d'images 2D ou 3D binaires

Nous avons proposé des filtrages parallèles de squelettes curvilignes d'images 2D ou 3D binaires, et donné quelques résultats de squelettes surfaciques d'images 3D binaires.

Dans le cas des squelettes curvilignes (2D ou 3D), les problèmes mis en évidence se situent au niveau des jonctions de courbe. En fait, nous éliminons des parties de courbes ouvertes simples, avec un paramètre approprié. Les "pâtés" ne sont pas tous éliminés, auquel cas ils témoignent de la géométrie de l'objet et semblent devoir être conservés. En général, nos résultats semblent satisfaisants.

Dans le cas des squelettes 3D surfaciques, nos résultats semblent présenter des effets de bord plus importants que dans le cas de filtrages de squelettes 2D. En fait, cela est dû à un mauvais choix du paramètre k ou à des dimensions non négligeables des parties que nous souhaitons filtrer par rapport à celles que nous souhaitons garder : les problèmes se situent généralement au niveau des jonctions de surface. Néanmoins, les résultats semblent également satisfaisants.

Une étude portant sur une segmentation topologique de squelettes curvilignes d'images 3D binaires [DLPB99a] [DLPB99b] [Dok00] (plus précisément, du réseau vasculaire du foie) a été récemment réalisée. Elle considérait l'ajout de points de façon itérative ; des problèmes pouvaient apparaître concernant les choix des points à ajouter. Elle pourrait peut-être bénéficier de notre étude afin d'ajouter des points de façon parallèle, tout en préservant la topologie, et afin de s'affranchir de certains problèmes comme la reconstruction de petites branches parasites, etc.

Cette étude confirme que le cadre des ordres semble prometteur pour la conception d'opérateurs parallèles de traitement d'images.

Conclusion et perspectives

Notre contribution principale est divisée en deux parties.

La première est consacrée à la proposition de nouveaux algorithmes de squelettisation selon l'approche topologie digitale.

Grâce aux notions d'ensemble P^x et de point P-simple, nous avons proposé des algorithmes supprimant des points P^x-simples, et supprimant plus de points que certains algorithmes déjà existants. Cette démarche novatrice nous permet d'affirmer que nos algorithmes, et ceux existants desquels ils dérivent, préservent bien la topologie ; aucune preuve supplémentaire n'est exigée contrairement à ceux existants. De plus, contrairement aux algorithmes déjà existants supprimant des points P-simples, nous n'avons besoin ni de phase préliminaire d'étiquetage, ni de l'examen d'un voisinage étendu. Nous avons proposé deux algorithmes de type sous-itérations définissant clairement les points à supprimer, évitant également l'utilisation de templates généralement obtenues de façon empirique.

Nous avons également montré, avec la seule notion de point P-simple, que sur trois algorithmes classiques fortement parallèles de squelettisation d'images 3D binaires, un seul préservait réellement la topologie. Actuellement, nous travaillons en collaboration avec l'auteur des deux autres, afin de proposer une version corrigée de ses deux algorithmes préservant la topologie.

Un travail important a également été réalisé afin d'étudier en détail, et d'implémenter, quelques algorithmes de squelettisation pour images 2D binaires et une grande partie de ceux proposés pour la squelettisation d'images 3D binaires. Cela nous a permis de mieux comprendre leur fonctionnement, d'en repérer également les limites. En effets, les résultats obtenus par les algorithmes "classiques" dépendent de la première direction choisie, de la première sous-maille choisie, d'une préférence pour une direction de préservation (pour le cas des rubans d'épaisseur 2, par exemple, dans les algorithmes fortement parallèles). De plus ces algorithmes sont plutôt difficiles à mettre en œuvre : examen d'un voisinage étendu (algorithmes fortement parallèles), "lourdeur" de l'approche directionnelle (templates, isométries). En outre, les résultats ne sont pas forcément minces et bien centrés, ou présentent un aspect dentelé (c'est le cas des squelettes obtenus par les algorithmes utilisant des sous-mailles). Notons enfin la définition empirique de point terminal de surface qui semble non satisfaisante.

Dans la seconde partie, à partir de l'étude de l'homotopie dans les ordres récemment réalisée par Gilles Bertrand, nous avons dérivé des schémas simples et parallèles de squelettisation, et de façon à pouvoir décider dans un voisinage non étendu de la suppressibilité d'un point. À l'aide d'une transformation, multipliant néanmoins la taille de l'image, nous obtenons des squelettes minces, bien centrés. Nos algorithmes exigent un nombre d'itérations de suppression du même ordre que celui requis par les algorithmes classiques (selon l'approche topologie digitale). Nous donnons également une définition plus formelle de la notion de point terminal de surface, utilisant le concept d'arbre de la théorie des graphes. De plus, ces schémas simples de squelettisation parallèle ont été déclinés pour traiter les images 2D, 3D binaires ou 2D en niveaux de gris.

Insistons sur le fait que nos algorithmes de squelettisation proposés selon l'approche topo-

logie discrète évitent ainsi les problèmes rencontrés au cours de notre étude selon l'approche topologie digitale.

De nombreux algorithmes de squelettisation, dans le cadre de la topologie digitale, utilisent la notion de distance et produisent des squelettes bien centrés, mais pas forcément minces. Il est peut-être possible d'obtenir des résultats minces et bien centrés de façon parallèle avec une distance dans les ordres et de comparer avec les résultats obtenus jusqu'à présent. Cette notion de distance pourrait également nous permettre d'approfondir l'étude de filtrage que nous avons entreprise.

Une étude approfondie concernant les images 2D en niveaux de gris selon l'approche topologie discrète semble une perspective prometteuse, d'une part afin de mieux cerner les problèmes concernant la sélection en parallèle des points et la valeur à laquelle ils doivent être abaissés en parallèle d'une façon efficace ; et d'autre part, en vue de pouvoir étendre certains de ces résultats aux images 3D en niveaux de gris.

Quatrième partie

Annexes

Annexe A

Algorithmes de squelettisation 2D

A.1 Introduction

Dans ce chapitre, nous proposons d'étudier quelques algorithmes de squelettisation d'images binaires 2D.

Pour chacune des classes d'algorithmes mises en évidence au chapitre 4, utilisant une stratégie sous-itérations directionnelles, sous-mailles ou fortement parallèle, nous étudions un algorithme (ou plusieurs) y appartenant. Des comparaisons seront faites entre algorithmes de même classe ou de classes différentes. Nous mettrons en évidence les avantages et inconvénients de chacune de ces classes. Il est parfois proposé des explications permettant de comprendre la conception d'un algorithme B à partir d'un autre A d'une même classe ; B résolvant les problèmes de A. Enfin, nous proposons une liste de critères permettant d'"évaluer" un algorithme de squelettisation. Nous avons retenu un certain nombre des algorithmes présentés ici, et nous donnons les squelettes obtenus par ceux-ci, pour une série d'images à la fin du chapitre.

Notons que la compréhension du fonctionnement des algorithmes pour images 2D permet de comprendre plus aisément celle des algorithmes d'images binaires 3D, présentés au chapitre suivant. Retenons que les algorithmes utilisés dans le cadre des images 2D en niveaux de gris sont une adaptation des algorithmes pour images 2D binaires.

Dans la troisième partie de ce mémoire (section 8.5) a été proposé un algorithme de squelettisation d'images 2D binaires selon une approche topologie discrète ; les résultats obtenus par cet algorithme ont été comparés avec ceux retenus à la fin de ce chapitre.

A.2 Notations et représentations

A.2.1 Notations

Les voisins d'un point central P d'une configuration sont désignés suivant la figure A.1. Le nombre $A(P)$ est le nombre de motifs 01 apparaissant dans l'ensemble ordonné $P_1, P_2, \ldots, P_8$, ensemble constitué des points du 8-voisinage de P. C'est le nombre de composantes 4-connexes dans le 8-voisinage de P. Le nombre de voisins de P dans l'objet est noté par $B(P)$; nous avons

P_1	P_2	P_3
P_8	P	P_4
P_7	P_6	P_5

FIG. A.1 – *Désignation des points voisins de P.*

1	0	1
0	P	1
1	1	1

(a)

0	1	0
1	P	0
0	0	0

(b)

FIG. A.2 – *Illustration de la non-nécessité de $A(P) = 1$ pour caractériser des points simples.*

$B(P) = P_1 + P_2 + \ldots + P_8$, avec $P_i = 1$ s'il s'agit d'un point de l'objet, ou 0 s'il s'agit d'un point du complémentaire. Le nombre de composantes 8-connexes dans le 8-voisinage de P est dénoté par $C(P)$. La condition "$C(P) = 1$" et la condition "P est 4-adjacent au complémentaire de l'objet" caractérisent les points 8-simples.

Nous avons $A(P) = 1$ si et seulement si $N_8^*(P)$ a exactement une composante 4-connexe de X et une composante 4-connexe de $\overline{X}$. Puisque qu'une composante 4-connexe est 8-connexe, nous avons :

- si $A(P) = 1$ et P est 4-adjacent à X alors P est 4-simple ; la réciproque est fausse (cf. Fig. A.2 (a), P est 4-simple et $A(P) = 2$),
- si $A(P) = 1$ et P est 4-adjacent à $\overline{X}$ alors P est 8-simple ; la réciproque est fausse (cf. Fig. A.2 (b), P est 8-simple et $A(P) = 2$).

Notons $dir(P)$, l'un des voisins de P selon les 8 directions possibles : $Nord-Ouest$, $Nord$, $Nord - Est$, $Ouest$, Est, $Sud - Ouest$, Sud ou $Sud - Est$. Considérons dir, une direction parmi $Nord$, Sud, Est, $Ouest$, un point P tel que $dir(P)$ appartient au complémentaire est appelé *point de bord* dir. Considérons dir, une direction parmi $Nord - Ouest$, $Nord - Est$, $Sud - Ouest$ ou $Sud - Est$, s'écrivant sous la forme $dir_1 - dir_2$, un point P tel que $dir(P)$, $dir_1(P)$ et $dir_2(P)$ appartiennent au complémentaire est dit *coin* dir.

A.2.2 Différentes représentations adoptées

Afin d'illustrer les résultats obtenus par certains algorithmes, nous adoptons les représentations suivantes :

- REPR1 (image) : objet initial et le squelette obtenu avec l'algorithme considéré. En légende, au dessous du squelette, se trouvent le nom de l'algorithme, et entre parenthèses successivement le nombre total de points retirés, le numéro de la dernière (sous-)itération effective de suppression et le nombre moyen de points supprimés par (sous-)itération,
- REPR2 (image) : objet initial et certaines ou toutes les (sous-)itérations. Nous représentons en gris, les points supprimés jusqu'à l'itération (ou sous-itération) considérée,

– REPR3 (image) : objet initial et certaines ou toutes les (sous-)itérations. Nous représentons en gris, les points supprimés par rapport à l'itération précédente. Sous chacune des figures sont indiqués le numéro de la (sous-)itération considérée ($ite = i$, pour la i-ième (sous-)itération) et généralement, la dernière figure apparaissant, correspond à la dernière (sous-)itération effective de suppression,
– REPR4 (texte) : objet initial (éventuellement) et le squelette obtenu avec l'algorithme considéré ; une `*` représente un point de l'objet ou du squelette, un chiffre désigne l'itération durant laquelle le point, à cette position dans l'objet initial, a été supprimé.

A.3 Algorithmes de squelettisation de type sous-itérations directionnelles

Nous examinons d'abord trois algorithmes de type 2 sous-itérations, puis un algorithme de type 4 sous-itérations.

A.3.1 Algorithme de Zhang et Suen - 2 sous-itérations (ZS) [ZS84]

C'est un algorithme parallèle de type deux sous-itérations. Le schéma est décrit par l'algorithme 17.

Répéter

Supprimer en parallèle les points P de l'objet vérifiant les conditions suivantes :

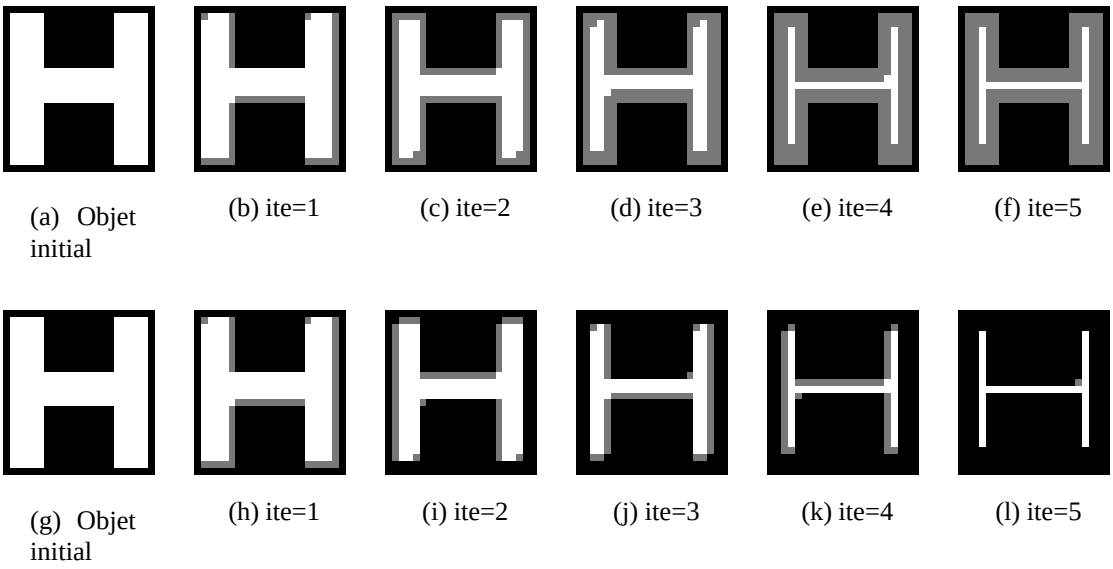
$$2 \leq B(P) \leq 6 \quad \text{(A.1)}$$
$$A(P) = 1 \quad \text{(A.2)}$$
$$P_4 = 0 \text{ ou } P_6 = 0 \text{ ou } P_2 = P_8 = 0 \text{ (itérations impaires)} \quad \text{(A.3)}$$
$$P_2 = 0 \text{ ou } P_8 = 0 \text{ ou } P_4 = P_6 = 0 \text{ (itérations paires)} \quad \text{(A.4)}$$

Jusqu'à ce qu'il n'y ait plus de point supprimé durant 2 sous-itérations successives.

Algorithme 17: *Algorithme de Zhang et Suen (*ZS*).*

La condition (A.1) permet la préservation des points extrémités (points 8-adjacents à un point de l'objet). Elle est telle que les points supprimés sont 4-adjacents au complémentaire et 4-adjacents à l'objet. La condition (A.2) jointe à la condition (A.1) cible alors des points à la fois 4-simples et 8-simples. La condition (A.3) s'attache à retirer les points de bord Est ou Sud et les coins $Nord - Ouest$. La condition (A.4) s'attache à retirer les points de bord $Nord$ ou $Ouest$ et les coins $Sud - Est$.

Sur la figure A.3, sont représentés les résultats intermédiaires des différentes itérations de suppression sur l'objet lettre "H", la cinquième sous-itération étant la dernière durant laquelle une suppression a eu lieu. Nous pouvons vérifier que lors des sous-itérations impaires, seuls des points de bord Est ou Sud et des coins $Nord - Ouest$ ont été supprimés (Fig. A.3(h), A.3(j),



(a) Objet initial (b) ite=1 (c) ite=2 (d) ite=3 (e) ite=4 (f) ite=5

(g) Objet initial (h) ite=1 (i) ite=2 (j) ite=3 (k) ite=4 (l) ite=5

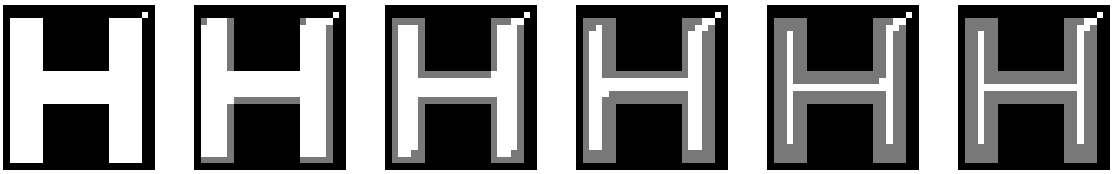
FIG. A.3 – *Lettre "H" et étapes de la squelettisation (algorithme* ZS*) (*REPR2 *en haut et* REPR3 *en bas).*

A.3(l)) ; et lors des itérations paires, seuls des points $Nord$ ou $Ouest$ et des coins $Sud - Est$ ont été supprimés (Fig. A.3(i), A.3(k)).

Malheureusement, cet algorithme comporte des imperfections, relevées par H.E. Lü et P.S.P. Wang [LW86].

- **PROB**1 : si on considère un petit bruit (Fig. A.4(a)), celui-là peut empêcher une squelettisation "homogène" et être la cause de la préservation de branches non désirées (Fig. A.4(f)),
- **PROB**2 : un deuxième problème est la réduction excessive des lignes obliques d'épaisseur 2, en 2 points (Fig. A.5),
- **PROB**3 : le troisième problème, et non le moindre, est la disparition de certains motifs : par exemple, le carré de 2×2 points disparaît au bout d'une itération. Cette remarque intervient également pour des objets de plus grande taille se réduisant, avec cet algorithme, en un carré de 2×2 points à une certaine itération ; et disparaissant alors lors de l'itération suivante. Rappelons que cet objet est un 8-MNS (voir section 2.6.2 au chapitre 2).

Remarquons que rien n'est dit à propos de la connexité de l'image à considérer. Par rapport à la lettre "H" bruitée (Fig. A.4), il faudrait utiliser la $(8, 4)$-connexité, sinon l'objet serait déconnecté (voir le point le plus en haut et à droite) ; néanmoins le problème **PROB**3 persiste. Nous avons représenté le résultat de la squelettisation de composantes composées de trois points mutuellement 8-adjacents (qui sont également des 8-MNS) (Fig. A.6).



(a) Objet initial | (b) ite=1 | (c) ite=2 | (d) ite=3 | (e) ite=4 | (f) ite=5

FIG. A.4 – *Lettre "H" bruitée et étapes de la squelettisation (algorithme* ZS*) (*REPR2*).*

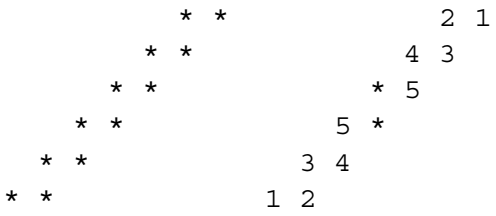


FIG. A.5 – *Érosion excessive (algorithme* ZS*) : cette ligne oblique d'épaisseur 2 se réduit en 2 points (*REPR4*).*

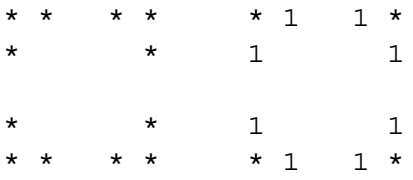


FIG. A.6 – *Différents coins et leurs squelettes (algorithme* ZS*) (*REPR4*).*

A.3.2 Algorithme de Lü et Wang - 2 sous-itérations (LW) [LW86]

La condition (A.1) de l'algorithme de Zhang et Suen (ZS) (cf. section A.3.1) est remplacée par la condition (A.5) (voir l'algorithme 18) ; on obtient ainsi l'algorithme de Lü et Wang (LW).

$$3 \leq B(P) \leq 6 \tag{A.5}$$

Algorithme 18: *Algorithme (fragment) de Lü et Wang (*LW*) (cette condition remplace la condition (A.1) de l'algorithme de Zhang et Suen (*ZS*)).*

Cet algorithme propose une solution possible au problème **PROB**2 relevé auparavant. Les points extrémités sont maintenant les points voisins seulement d'un ou de deux points 4-adjacents. Avec cette variante, aucun point des objets des figures A.5 et A.6 n'est éliminé.
Mais pour certains, le fait de ne pas amincir du tout les bandes obliques d'épaisseur 2 est également un problème (**PROB**4). En fait, une solution acceptable (selon certains auteurs) consiste à réduire un ruban oblique d'épaisseur 2 en une ligne oblique. Les deux autres problèmes (**PROB**1 et **PROB**3) persistent avec cet algorithme.

A.3.3 Algorithme de Guo et Hall - 2 sous-itérations (GH_A1) [GH89]

C'est une adaptation de l'algorithme de Zhang et Suen (ZS) (ou de sa variante par Lü et Wang (LW), afin de préserver la topologie (le problème **PROB**3 est alors résolu) et d'obtenir des courbes plus minces (**PROB**4) sans érosion excessive (**PROB**2).

Répéter

Supprimer en parallèle les points P de l'objet vérifiant les conditions suivantes :

$$C(P) = 1 \quad \text{(A.6)}$$

$$2 \leq N(P) \leq 3 \quad \text{(A.7)}$$

$$(P_2 \vee P_3 \vee \overline{P_5}) \wedge P_4 = 0 \text{ (itérations impaires)} \quad \text{(A.8)}$$

$$(P_6 \vee P_7 \vee \overline{P_1}) \wedge P_8 = 0 \text{ (itérations paires)} \quad \text{(A.9)}$$

avec :

$$N(P) = Min[N_1(P), N_2(P)] \quad \text{(A.10)}$$

$$N_1(P) = (P_1 \vee P_2) + (P_3 \vee P_4) + (P_5 \vee P_6) + (P_7 \vee P_8) \quad \text{(A.11)}$$

$$N_2(P) = (P_2 \vee P_3) + (P_4 \vee P_5) + (P_6 \vee P_7) + (P_8 \vee P_1) \quad \text{(A.12)}$$

Jusqu'à ce qu'il n'y ait plus de point supprimé durant 2 sous-itérations successives.

Algorithme 19: *Algorithme de Guo et Hall (*GH_A1*).*

Le schéma est décrit par l'algorithme 19. Le nombre $N(P)$ (condition (A.10)) est tel que les points supprimés sont 4-adjacents à un point du complémentaire. Les points voisins d'un seul point de l'objet ou de deux points de l'objet 4-adjacents ne peuvent alors être supprimés, car $N(P) = 1$ et ils ne vérifient pas la condition (A.7). Les conditions (A.6) et (A.7) ciblent les points 8-simples. La condition (A.10) permet également, par exemple, la suppression de points dans les rubans inclinés d'épaisseur 2 (Fig. A.7), contrairement à l'algorithme LW. La condition (A.8) cible certains points de bord $Nord$ ou Est. La condition (A.9) cible certains points de bord Sud ou $Ouest$.

FIG. A.7 – *Ruban oblique d'épaisseur* 2 *et son squelette (algorithme* GH_A1*) (*REPR4*).*

A.3.4 Algorithme de Rosenfeld - 4 sous-itérations (ROS) [Ros75] [RK82]

C'est un algorithme parallèle de type quatre sous-itérations. Le schéma est décrit par l'algorithme 20.

Répéter
Supprimer en parallèle les points P de l'objet vérifiant les conditions suivantes :

$$C(P) = 1 \quad \text{(A.13)}$$

P n'est pas un point extrémité (A.14)

$$P_i = 0 \quad \text{(A.15)}$$

i prend les valeurs $2, 6, 4, 8, 2, 6, 4, \ldots$ pour les sous-itérations successives
Jusqu'à ce qu'il n'y ait plus de point supprimé durant 4 sous-itérations successives.

Algorithme 20: *Algorithme de Rosenfeld (*ROS*).*

La condition (A.13) associée à la condition (A.15) permet de cibler les points 8-simples. Cet algorithme supprime en parallèle les points 8-simples non extrémités, qui sont également points de bord selon la direction $Nord$ (resp. Sud, Est, $Ouest$) pour la j-ième sous-itération, avec $j[mod4]$ valant 1 (resp. 2, 3, 0). Les conditions (A.14) et (A.15) impliquent : $2 \leq B(P) \leq 7$. Se reporter aux figures A.33 à A.40 pour des squelettisations sur différents objets. Intuitivement, un meilleur centrage des squelettes est obtenu par l'utilisation de quatre sous-itérations plutôt que deux.

A.4 Algorithmes de squelettisation de type sous-mailles

Dans cette section, nous présentons un seul algorithme de type sous-mailles, celui de Guo et Hall (algorithme GH_A2) (section A.4.1). Cet algorithme utilise deux sous-mailles.

A.4.1 Algorithme (GH_A2) [GH89]

L'image est divisée en sous-ensembles distincts appelés *sous-mailles*. Les auteurs ont proposé un amincissement parallèle n'utilisant que deux sous-mailles. L'image est divisée en deux sous-mailles selon le motif d'un échiquier (voir Fig. A.8). Les sous-mailles sont activées selon l'ordre $1, 2, 1, 2, \ldots$. Seuls les points d'une même sous-maille active peuvent changer de statut. Le schéma est décrit par l'algorithme 21.

Les conditions (A.16) et (A.17) ciblent précisément les points 8-simples et la condition (A.18) préserve les points extrémités de la suppression (points 8-adjacents à un seul autre point de l'objet).

Remarque : Notons que l'on peut ne pas obtenir le même résultat si l'on permute l'ordre d'activation des mailles ou lorsque l'on considère un translaté de l'objet (**PROB_SM**) (voir ci-après).

Répéter

Pour chaque sous-maille **faire**

Supprimer en parallèle les points P de l'objet et de la sous-maille active vérifiant les conditions suivantes :

$$C(P) = 1 \tag{A.16}$$

$$P \text{ est } 4\text{-adjacent à } \overline{X} \tag{A.17}$$

$$B(p) > 1 \tag{A.18}$$

Jusqu'à ce qu'il n'y ait plus de point supprimé durant 2 sous-itérations successives.

Algorithme 21: *Algorithme de Guo et Hall (*GH_A2*).*

Conformément à ce que nous avons vu à la section 2.6.2.2, cet algorithme préserve la topologie car les seuls MNS constitués de points diagonaux (seule possibilité pour un ensemble connexe composé de deux points d'une même sous-maille) sont les composantes connexes constituées de deux points, et celles-ci sont préservées par la condition (A.18).

```
1   2   1   2   .   .   .
2   1   2   1   .   .   .
1   2   1   2   .   .   .
.   .   .   .
```

FIG. A.8 – *Motif des sous-mailles utilisées par l'algorithme* GH_A2.

A.4.2 Comparaisons

Quelques comparaisons entre les algorithmes ZS, LW et GH_A2 sur des rubans horizontaux ou diagonaux nous permettent de dresser certains avantages et inconvénients de l'algorithme GH_A2.

A.4.2.1 Cas des rubans horizontaux

Soit un ruban horizontal de taille $m \times n$ $(m > n)$. Les algorithmes ZS ou LW utilisent $n - 1$ itérations pour squelettiser ce ruban. Une approche complètement parallèle nécessiterait $\lfloor n/2 \rfloor$ itérations ($\lfloor q \rfloor$ désigne la partie entière de q), en essayant d'obtenir un résultat mince. L'algorithme GH_A2 nécessite un nombre inférieur ou égal à $\lfloor n/2 \rfloor + 1$ itérations ; l'inégalité large provient de la parité de la largeur n du ruban considéré (voir exemples ci-après).

Considérons d'abord un ruban de largeur 9 et de hauteur 5 $(m = 9, n = 5)$. Si le coin supérieur est dans la sous-maille 1 (resp. 2), le squelette obtenu est représenté à la figure A.9 (a) (resp. A.9 (b)) (**PROB_SM**) (voir également la figure A.12). Le squelette est obtenu à la suite de $3 (= \lfloor n/2 \rfloor + 1)$ itérations effectives de suppression.

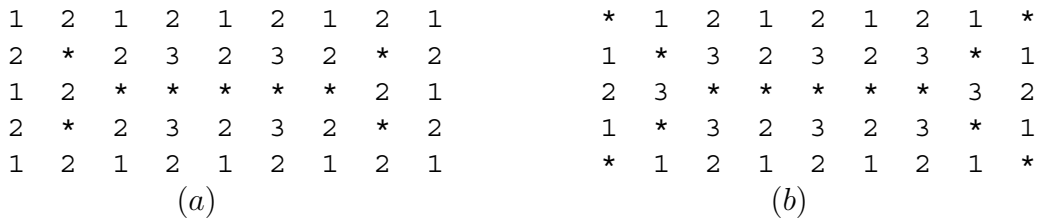


FIG. A.9 – *Ruban de largeur* 9 *et de hauteur* 5 *(*n *impair) (algorithme* GH_A2*) (*REPR4*).*

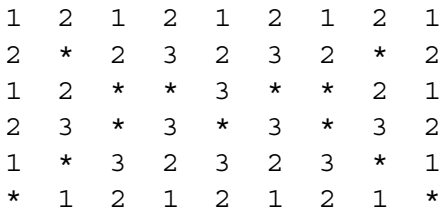


FIG. A.10 – *Ruban de largeur* 9 *et de hauteur* 6 *(*n *pair) (algorithme* GH_A2*) (*REPR4*).*

Considérons maintenant un ruban de largeur 9 et de hauteur 6 $(m = 9, n = 6)$. Si le coin supérieur est dans la sous-maille 1, le squelette obtenu est celui de la figure A.10. Si le coin supérieur est dans la sous-maille 2, le squelette est le symétrique par rapport à l'axe horizontal de celui de la figure A.10 (**PROB_SM**) (voir également la figure A.12). Le squelette est obtenu à la suite de $3(= n/2)$ itérations effectives de suppression.

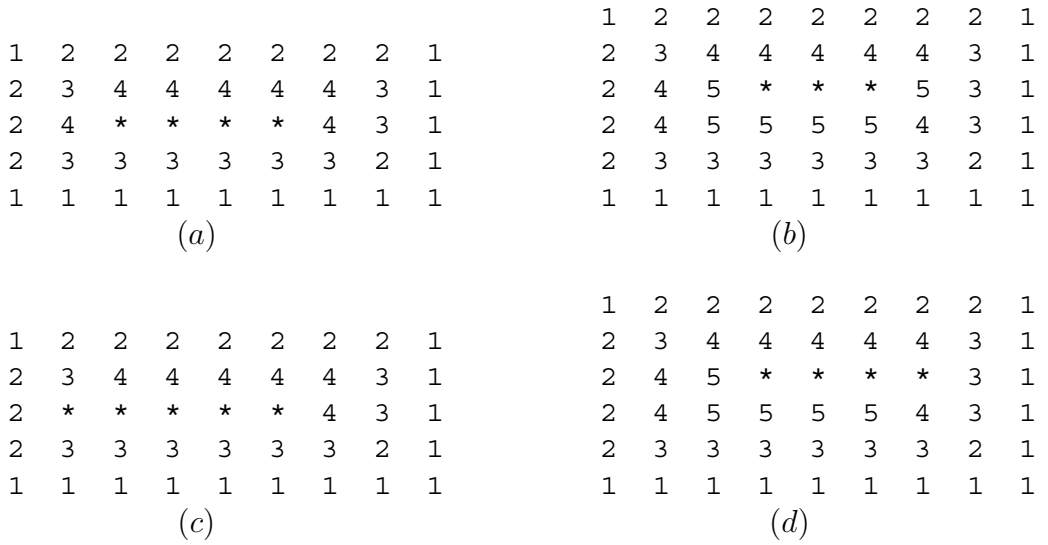
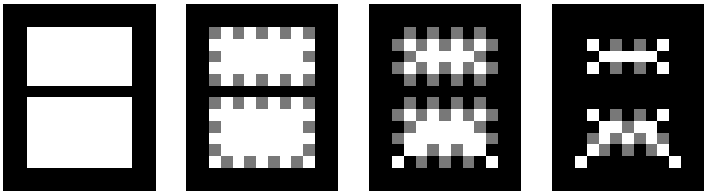
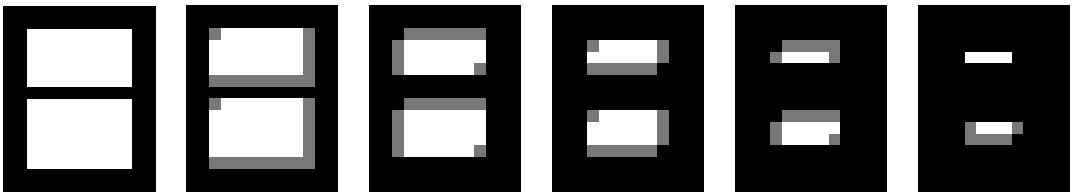


FIG. A.11 – *Ruban de largeur* 9 *et de hauteur* 5 *(resp.* 6*) avec l'algorithme* ZS (a) *(resp.* (b)*) ou avec l'algorithme* LW (c) *(resp.* (d)*) (*REPR4*).*



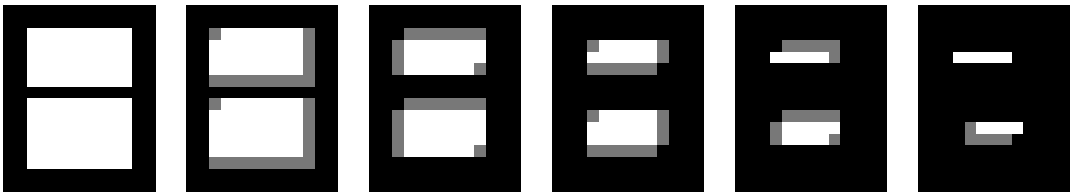
(a) Objet initial (b) ite=1 (c) ite=2 (d) ite=3

FIG. A.12 – *Rubans de largeur* 9 *et de hauteur* 5 *(dessus) et* 6 *(dessous) et étapes de la squelettisation (algorithme* GH_A2*) (*REPR3*).*



(a) Objet initial (b) ite=1 (c) ite=2 (d) ite=3 (e) ite=4 (f) ite=5

FIG. A.13 – *Rubans de largeur* 9 *et de hauteur* 5 *(dessus) et* 6 *(dessous) et étapes de la squelettisation (algorithme* ZS*) (*REPR3*).*



(a) Objet initial (b) ite=1 (c) ite=2 (d) ite=3 (e) ite=4 (f) ite=5

FIG. A.14 – *Rubans de largeur* 9 *et de hauteur* 5 *(dessus) et* 6 *(dessous) et étapes de la squelettisation (algorithme* LW*) (*REPR3*).*

Avec les algorithmes ZS et LW, $4(=(n-1))$ (resp. $5(=(n-1))$) itérations sont requises pour obtenir un squelette de l'objet de hauteur impaire ($n=5$), Fig. A.11 (a) ou (c) (resp. de largeur paire ($n=6$), Fig. A.11 (b) ou (d)) (voir également les figures A.13 et A.14).

A.4.2.2 Cas des rubans diagonaux

Nous donnons maintenant quelques squelettes de rubans inclinés (d'épaisseur de parité différente) obtenus avec les algorithmes ZS (Fig. A.16 (a) et (b) et Fig. A.18), LW (Fig. A.16 (c) et (d) et Fig. A.19) et GH_A2 (Fig. A.15 et A.17).

Remarque : Sur ces quelques figures, les algorithmes ZS et LW nécessitent moins de sous-itérations pour obtenir un squelette que l'algorithme GH_A2 ; néanmoins le squelette obtenu n'est pas toujours aussi mince que celui obtenu avec l'algorithme GH_A2.

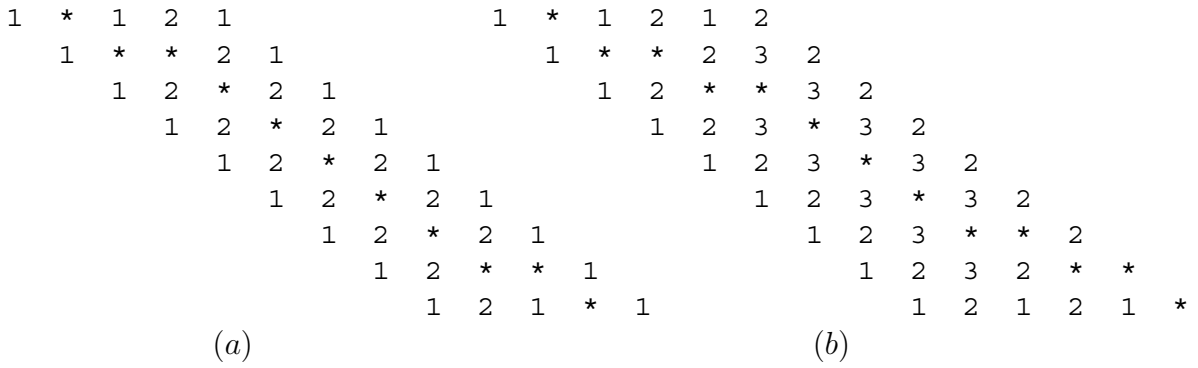


FIG. A.15 – *Rubans inclinés de "largeur"* (a) 5, (b) 6 *(algorithme* GH_A2*) (*REPR4*).*

A.4.2.3 Cas d'un autre ruban incliné

Considérons le ruban incliné de la figure A.20. L'inclinaison du ruban (135 degrés) est telle que tous les points du bord sont dans une même sous-maille. S'ils appartiennent à la sous-maille 1, on obtient les trois mêmes squelettes avec les algorithmes ZS, LW et GH_A2 (Fig. A.20), et 5 itérations sont nécessaires à l'obtention du squelette (voir également la figure A.21). Si ces points de bord appartiennent à la sous-maille 2, l'algorithme GH_A2 requiert une itération supplémentaire.

Nous pouvons noter l'efficacité (du point de vue du nombre de sous-itérations de suppression) des algorithmes ZS et LW sur cet objet. En effet, lors d'une itération impaire de l'un de ces algorithmes, les points Est ou Sud et les coins $Nord - Ouest$ sont retirés ; les points $Nord$ d'un tel ruban sont également des points Est ou Sud ou $Nord - Ouest$, les points $Ouest$ sont également des points Est ou Sud ou $Nord - Ouest$; dans une même itération des points de bord des quatre directions peuvent alors être retirés simultanément ; le même raisonnement est valable pour des itérations paires. Sur un tel objet, la squelettisation opére par suppression homogène de couches de points. Le bon centrage du squelette en est une conséquence immédiate.

A.4.3 Conclusion

En conclusion, nous pouvons retenir que les squelettes obtenus présentent généralement un aspect dentelé (voir les figures A.12 et A.33 à A.40). En revanche, et cela est vrai également en

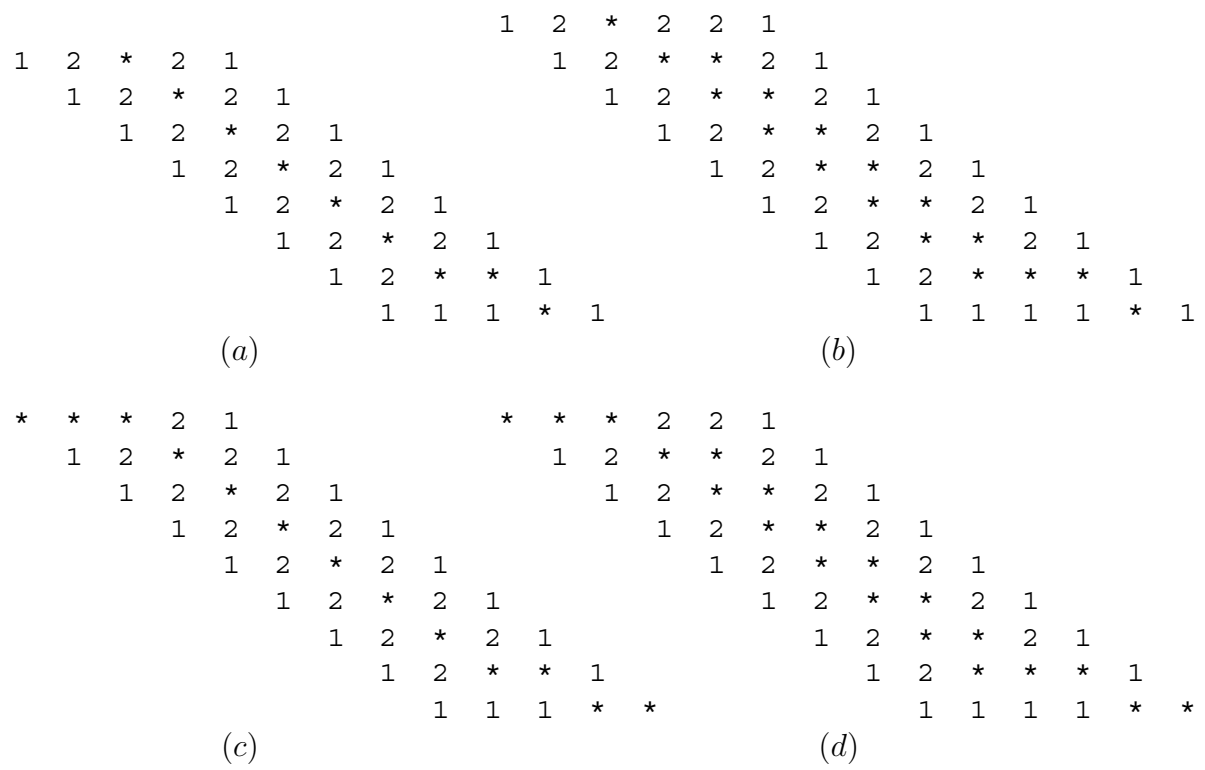
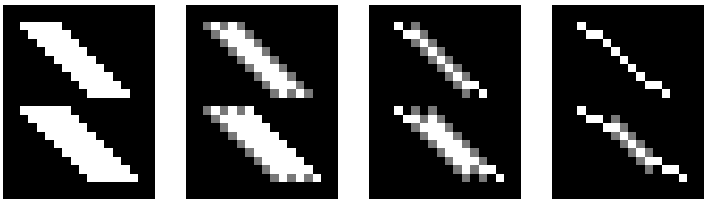


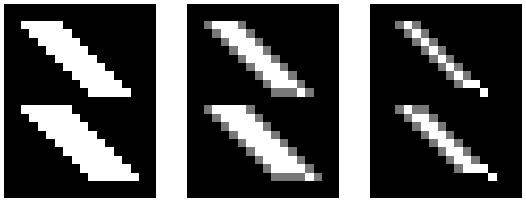
FIG. A.16 – *Rubans inclinés de "largeur"* 5 *(resp.* 6*) avec l'algorithme* ZS (a) *(resp.* (b)*) ou avec l'algorithme* LW (c) *(resp.* (d)*) (*REPR4*).*

3D, d'une manière générale, il est plus facile de montrer qu'un algorithme opérant par suppression de points simples préserve la topologie s'il utilise la technique des sous-mailles (pour un nombre adéquat de sous-mailles, bien définies), plutôt que s'il utilise la stratégie directionnelle ou s'il est fortement parallèle.



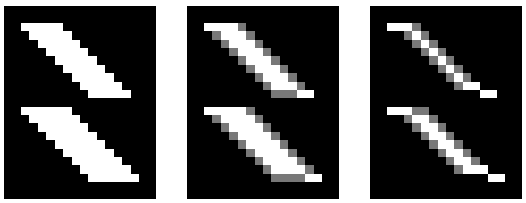
(a) Objet initial (b) ite=1 (c) ite=2 (d) ite=3

FIG. A.17 – *Rubans inclinés de "largeur"* 5 *(dessus) et* 6 *(dessous) et étapes de la squelettisation (algorithme* GH_A2*) (*REPR3*).*



(a) Objet initial (b) ite=1 (c) ite=2

FIG. A.18 – *Rubans inclinés de "largeur"* 5 *(dessus) et* 6 *(dessous) et étapes de la squelettisation (algorithme* ZS*) (*REPR3*).*

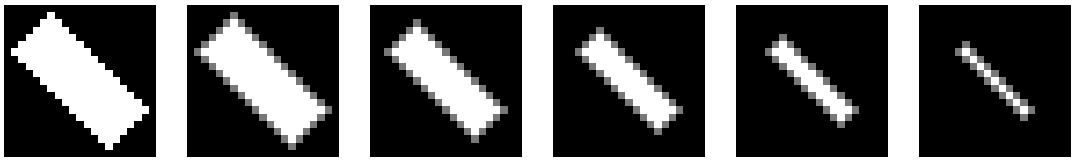


(a) Objet initial (b) ite=1 (c) ite=2

FIG. A.19 – *Rubans inclinés de "largeur"* 5 *(dessus) et* 6 *(dessous) et étapes de la squelettisation (algorithme* LW*) (*REPR3*).*

```
                    1
                  1 2 1
                1 2 3 2 1
              1 2 3 4 3 2 1
            1 2 3 4 5 4 3 2 1
          1 2 3 4 5 * 5 4 3 2 1
            1 2 3 4 5 * 5 4 3 2 1
              1 2 3 4 5 * 5 4 3 2 1
                1 2 3 4 5 * 5 4 3 2 1
                  1 2 3 4 5 * 5 4 3 2 1
                    1 2 3 4 5 * 5 4 3 2 1
                      1 2 3 4 5 * 5 4 3 2 1
                        1 2 3 4 5 * 5 4 3 2 1
                          1 2 3 4 5 * 5 4 3 2 1
                            1 2 3 4 5 4 3 2 1
                              1 2 3 4 3 2 1
                                1 2 3 2 1
                                  1 2 1
                                    1
```

FIG. A.20 – *Ruban incliné (algorithmes* ZS, LW, GH_A2*) (*REPR4*).*



(a) Objet initial (b) ite=1 (c) ite=2 (d) ite=3 (e) ite=4 (f) ite=5

FIG. A.21 – *Ruban incliné avec les étapes de squelettisation (algorithmes* ZS, LW, GH_A2*) (*REPR3*).*

A.5 Algorithmes fortement parallèles

A.5.1 Algorithme de Holt, Stewart, Clint et Perrott (HSCP) [HSCP87]

C'est un algorithme de type fortement parallèle, qui nécessite un support de taille 16 points.

Le schéma de squelettisation est décrit par l'algorithme 22. Un point est retiré s'il vérifie la condition $a)$ et si ses voisins ne vérifient aucune des trois conditions de la condition $b)$. Certains des voisins testés dans la condition $b)$ utilisent la condition $a)$, ce qui explique l'utilisation d'un support de taille 4×4.

Notons que les conditions (A.19) et (A.20) sont les mêmes que les conditions (A.1) et (A.2) de l'algorithme (ZS), cf. section A.3.1. La condition (A.21) préserve certains points de bord $Ouest$ dans une bande verticale d'épaisseur 2 (Fig. A.22). La condition (A.22) préserve certains points de bord $Nord$ dans une bande horizontale d'épaisseur 2 (Fig. A.22). La condition (A.23) préserve au moins un point dans un carré 2×2, ce qui est une solution au problème **PROB**3. En revanche, le problème **PROB**2 persiste. En effet, pour l'objet initial ruban oblique d'épaisseur 2 (objet à gauche, Fig. A.5), le squelette obtenu par cet algorithme est le même que celui obtenu avec l'algorithme de Zhang et Suen (Fig. A.5, à droite).

Condition $a)$:

$$2 \leq B(P) \leq 6 \quad \text{(A.19)}$$

$$A(P) = 1 \quad \text{(A.20)}$$

Condition $b)$:

$P_2 = P_6 = 1$ et P_4 a été retenu dans l'étape $a)$ (A.21)

$P_4 = P_8 = 1$ et P_6 a été retenu dans l'étape $a)$ (A.22)

P_4, P_5, P_6 ont été retenus dans l'étape a) (A.23)

Répéter

Supprimer en parallèle tout point de l'objet vérifiant la condition $a)$

sauf ceux qui vérifient au moins l'une des trois conditions de la condition $b)$

Jusqu'à ce qu'il n'y ait plus de point supprimé durant une itération.

Algorithme 22: *Algorithme de Holt, Stewart, Clint et Perrott (*HSCP*).*

Les auteurs proposent également une amélioration de l'algorithme afin d'éliminer des motifs spéciaux comme celui, par exemple, consistant en un point central entouré de 7 points et 4-voisin d'un point du complémentaire de l'objet (dans ce cas $B(P) = 7$, $A(P) = 1$).

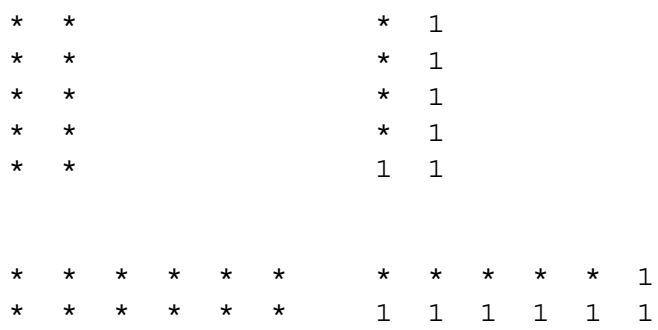


FIG. A.22 – *Rubans horizontaux et verticaux d'épaisseur 2 et leurs squelettes (algorithme* HSCP*) (*REPR4*).*

Une variante consiste à retirer l'un des deux points de la ligne centrale du motif
```
1
1 1
  1,
```
ce qui constitue un "filtrage" du squelette, intégré lors du processus de squelettisation.

A.5.2 Algorithme de Hall (HSCPN) [Hal89]

La condition (A.19) de l'algorithme de Holt, et al. (HSCP) est remplacée par la condition (A.24). Cet algorithme est noté par HSCPN (voir l'algorithme 23).

$$3 \leq B(P) \leq 6 \tag{A.24}$$

Algorithme 23: *Algorithme (fragment) de Hall (*HSCPN*) (cette condition remplace la condition (A.19) de l'algorithme (*HSCP*)).*

Cette variante (condition (A.24)) est la même que celle apportée par Lü et Wang (condition (A.5)) à l'algorithme de Zhang et Suen (condition (A.1)). Elle préserve totalement le ruban oblique de la figure A.5. Cette variante évite le problème **PROB**2, c'est alors le problème **PROB**4 qui survient, voir section A.3.2.

A.5.3 Algorithmes de Guo et Hall (GH_AFP) [GH92]

Z. Guo et R.W. Hall ont proposé dans [GH92] trois algorithmes fortement parallèles. Nous les décrivons ici, et montrons succinctement l'amélioration apportée par le troisième par rapport aux deux premiers.

A.5.3.1 Algorithme GH_AFP1

C'est un algorithme fortement parallèle utilisant un support de 11 points. Le schéma est décrit par l'algorithme 24.

Répéter

Supprimer en parallèle les points P de l'objet vérifiant les conditions suivantes :

$$A(P) = 1 \tag{A.25}$$

$$P \text{ est 4-adjacent à } \overline{X} \tag{A.26}$$

$$B(P) > 2 \tag{A.27}$$

$$L(P) = 0 \tag{A.28}$$

avec $L(P) = [(\overline{P_2} \wedge P_6 \wedge \overline{Q_2}) \wedge (P_3 \vee P_4 \vee P_5) \wedge (P_7 \vee P_8 \vee P_1)] \vee (\overline{P_4} \wedge P_8 \wedge \overline{Q_1})$
et les points Q_1 et Q_2 sont tels que :

	P_1	P_2	P_3
Q_1	P_8	P	P_4
	P_7	P_6	P_5
		Q_2	

Jusqu'à ce qu'il n'y ait plus de point supprimé durant une itération.

Algorithme 24: *Algorithme de Guo et Hall (*GH_AFP1*).*

Soit $L(P) = D(P) \vee E(P)$ avec $D(P) = [(\overline{P_2} \wedge P_6 \wedge \overline{Q_2}) \wedge (P_3 \vee P_4 \vee P_5) \wedge (P_7 \vee P_8 \vee P_1)]$ et $E(P) = \overline{P_4} \wedge P_8 \wedge \overline{Q_1}$. Si P est un point de bord Est dans un ruban vertical d'épaisseur 2, alors $E(P) = 1$ et P est préservé ; si P est un point de bord $Nord$ dans un ruban horizontal, et s'il existe un voisin appartenant à l'objet parmi P_3, P_4 et P_5 et un voisin appartenant à l'objet parmi P_7, P_8 et P_1 alors $D(P) = 1$ et P est préservé. Cela entraîne que la bande $Ouest$ d'un ruban vertical d'épaisseur 2 est éliminée et que la bande Sud d'un ruban horizontal d'épaisseur 2 est supprimée (voir Fig. A.23). Les rubans diagonaux d'épaisseur 2 ne sont pas amincis (**PROB**4) ; cela est dû aux conditions (A.25) et (A.27) (Fig. A.23).

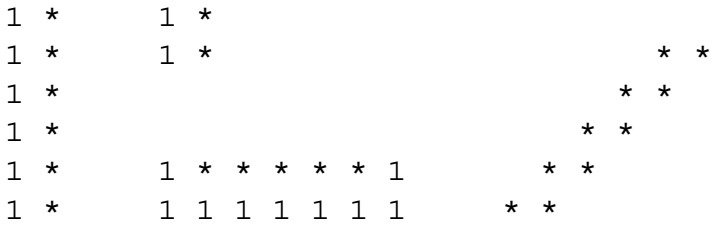


FIG. A.23 – *Squelettisations avec l'algorithme* GH_AFP1 *(rubans) (*REPR4*).*

A.5.3.2 Algorithme GH_AFP2

C'est un algorithme fortement parallèle utilisant un support de 11 points. Le schéma est décrit par l'algorithme 25.

Répéter

Supprimer en parallèle les points P de l'objet vérifiant les conditions suivantes :

$$A(P) = 1 \tag{A.29}$$

$$P \text{ est } 4\text{-adjacent à } \overline{X} \tag{A.30}$$

$$B(P) > 2 \tag{A.31}$$

et le voisinage de P ne satisfait aucun des motifs suivants :

```
   0          1 1          0 0 0
1  P 1  ,  0  1 P 0  ,  0  1 P 0
1  1 1        1 1          1 1 0
   0                       0
```

Jusqu'à ce qu'il n'y ait plus de point supprimé durant une itération.

Algorithme 25: *Algorithme de Guo et Hall (*GH_AFP2*).*

Les motifs permettent respectivement de ne pas perdre les rubans horizontaux ou verticaux d'épaisseur 2, ou un carré 2×2. Ici, s'exprime le compromis entre pleinement parallèle et le bon centrage du squelette. En effet, une direction de sauvegarde est imposée afin de préserver des points ; par exemple, une ligne de points de bord $Nord$ sera préservée dans les rubans horizontaux d'épaisseur 2 (premier motif de la seconde condition) (tout comme dans l'algorithme GH_AFP1). Cet algorithme n'amincit pas non plus les rubans diagonaux d'épaisseur 2 (**PROB**4) (Fig. A.24).

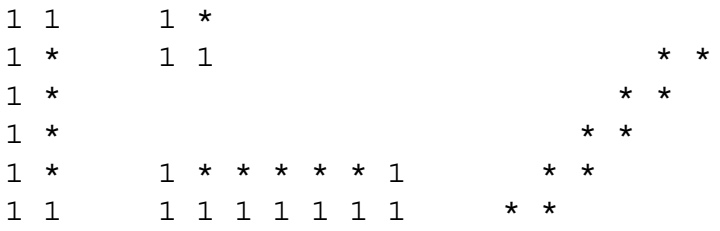


FIG. A.24 – *Squelettisations avec l'algorithme* GH_AFP2 *(rubans) (*REPR4*).*

A.5.3.3 Algorithme GH_AFP3

C'est un algorithme fortement parallèle utilisant un support de 11 points. Le schéma est décrit par l'algorithme 26.

Répéter

Supprimer en parallèle les points P de l'objet vérifiant les conditions de GH_AFP2 ou si le voisinage de P satisfait l'un des motifs suivants :

```
   0  0     0  0
1  P  0  ,  0  P  1
0  1           1  0
```

Jusqu'à ce qu'il n'y ait plus de point supprimé durant une itération.

Algorithme 26: *Algorithme de Guo et Hall (*GH_AFP3*).*

Cet algorithme permet l'obtention de résultats plus minces qu'avec l'algorithme GH_AFP2. Les rubans obliques d'épaisseur 2 sont maintenant amincis (Fig. A.25).

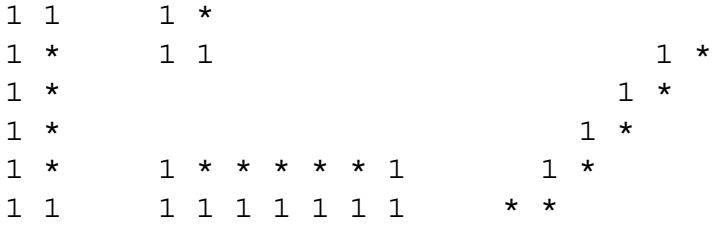


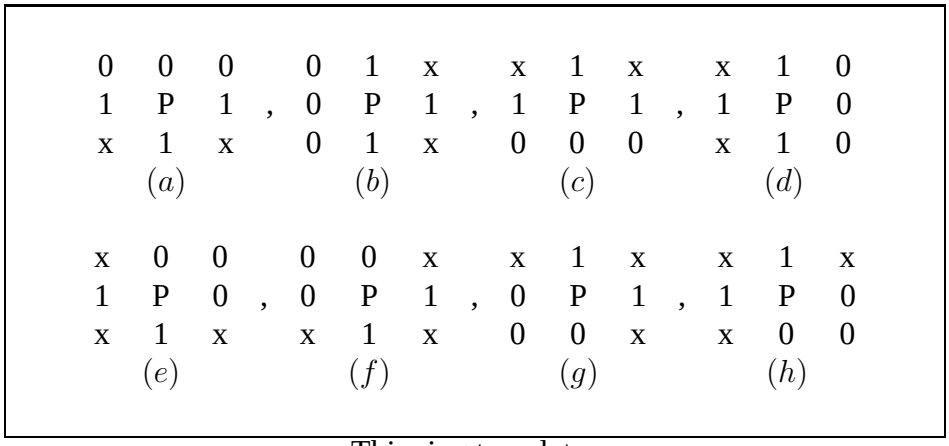
FIG. A.25 – *Squelettisations avec l'algorithme* GH_AFP3 *(rubans) (*REPR4*).*

A.5.4 Algorithme de Chin, Wan, Stover et Iverson (CWSI) [CWSI87]

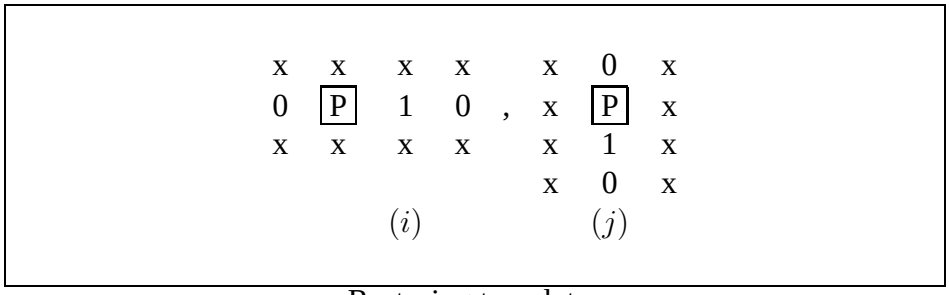
C'est un algorithme de type fortement parallèle d'un support de 15 points. Le schéma est décrit par l'algorithme 27. Il est composé de thinning templates ciblant des candidats à la suppression auxquelles est nécessaire d'adjoindre des restoring templates, afin de ne pas supprimer des rubans horizontaux ou verticaux d'épaisseur 2 (y compris le carré 2×2) (Fig. A.26). Cet algorithme propose également deux série de trimming templates, qui, en quelque sorte, réalisent un filtrage pendant le processus de suppression de points. La première série consiste à retirer soit des points au-dessus ou au-dessous de segments de droites horizontales, soit des points à gauche ou à droite de segments de droites verticales. La seconde série retire des points extrémités de droites diagonales. Les auteurs prétendent que de telles lignes se réduisent en un point et qu'il est impossible de déterminer localement s'il s'agit de points extrémités de lignes droites ou de points de lignes inesthétiques. En fait, nous avons remarqué que ces derniers motifs font qu'une composante constituée uniquement de deux points diagonaux (ou un objet qui se réduit en une telle composante, voir la figure A.27) disparaît ; la topologie n'est alors pas préservée (**PROB**3).

Répéter

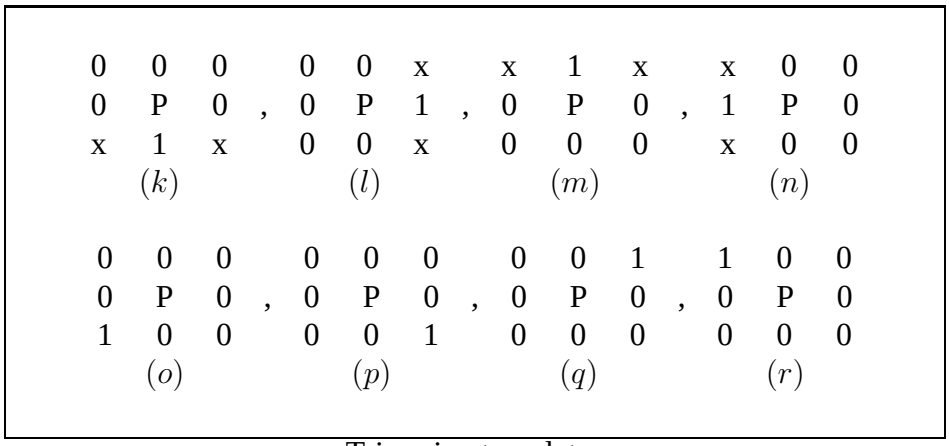
Supprimer en parallèle les points P de l'objet qui vérifient
au moins une des thinning ou trimming templates,
et qui ne vérifient pas une restoring template :



Thinning templates



Restoring templates



Trimming templates

Le symbole "x" signifie "appartient ou non à l'objet".

Jusqu'à ce qu'il n'y ait plus de point supprimé durant une itération.

Algorithme 27: *Algorithme de Chin, Wan, Stover et Iverson (*CWSI*).*

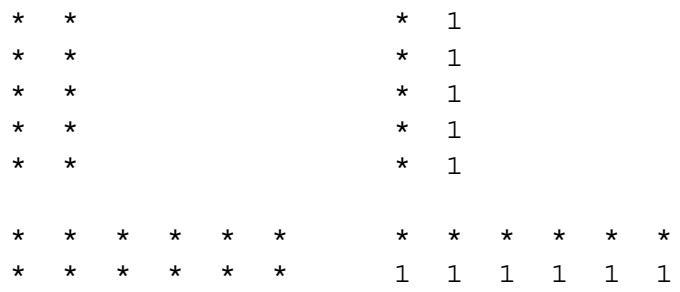


FIG. A.26 – *Rubans horizontal et vertical d'épaisseur 2 et leurs squelettes (*CWSI*) (*REPR4*).*

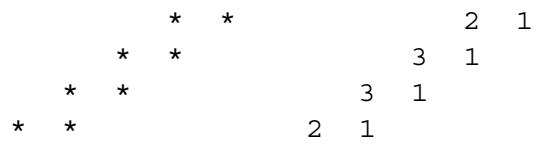


FIG. A.27 – *Ruban oblique d'épaisseur 2 et les différentes itérations pour l'obtention du squelette, ici l'objet disparaît et la topologie n'est pas préservée (*CWSI*) (*REPR4*).*

A.5.5 Algorithme de Wu et Tsai (WT) [WT92]

C'est un algorithme de type fortement parallèle d'un support de 15 points. Le schéma est décrit par l'algorithme 28. Cet algorithme utilise trois séries de templates : la première permet l'amincissement des rubans horizontaux ou verticaux, la deuxième permet celui des rubans diagonaux, la dernière réalise un filtrage intégré, et permet l'élimination de bruit apparaissant au voisinage de morceaux de lignes droites.

Par la suite, nous notons WT, l'algorithme pour lequel ont été mises en œuvre seules les deux premières séries de templates ; et WT2, la version utilisant également la troisième série de templates. Aux figures A.30 et A.31 sont représentés les résultats des algorithmes WT et WT2 sur l'image du ruban avec du bruit et sur l'image lettre "A" avec du bruit. Malheureusement, le carré 2×2 n'est pas préservé (**PROB**3), comme l'a remarqué A. Stewart [Ste94] (cela provient des motifs (e), (f), (h), (i)).

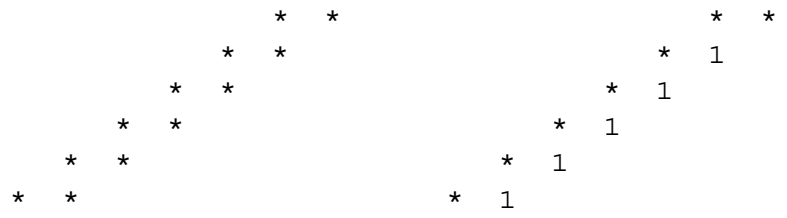
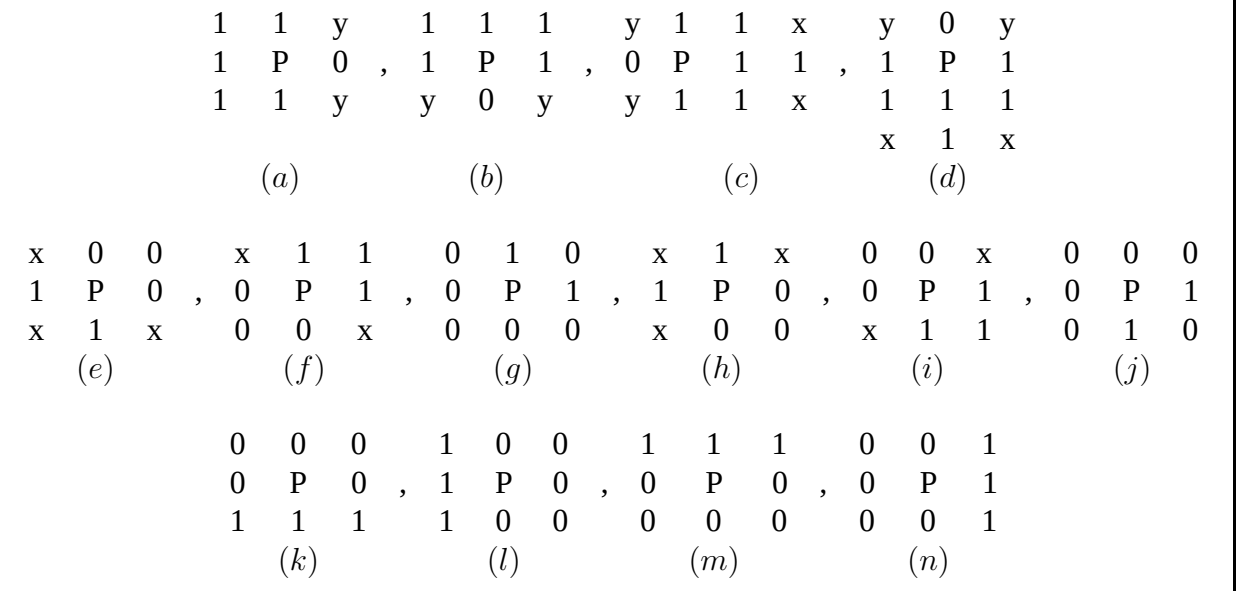


FIG. A.28 – *Ruban oblique d'épaisseur 2 et son squelette (algorithme* WT*) (*REPR4*).*

Répéter

Supprimer en parallèle les points P de l'objet qui vérifient
au moins l'un des motifs suivants :



Le symbole "x" signifie "appartient ou non à l'objet", au moins l'un des deux symboles "y" apparaissant dans un motif doit être un point du complémentaire de l'objet.

Jusqu'à ce qu'il n'y ait plus de point supprimé durant une itération.

Algorithme 28: *Algorithme de Wu et Tsai (*WT*).*

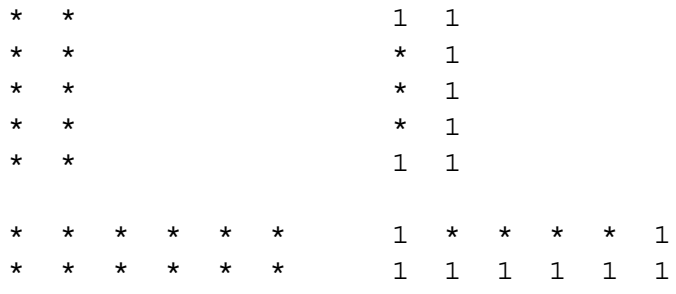
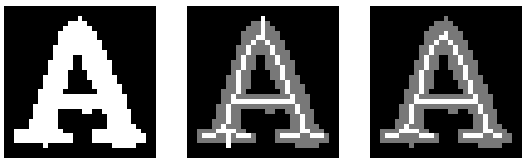


FIG. A.29 – *Rubans horizontal ou vertical d'épaisseur 2 et leurs squelettes (algorithme* WT*) (*REPR4*).*

(a) Objet initial (b) WT (c) WT2

FIG. A.30 – *Ruban bruité. Résultats, sans et avec les trimming templates (resp. algorithmes* WT *et* WT2*) (*REPR2*).*



(a) Objet initial (b) WT (c) WT2

FIG. A.31 – *Lettre "A" bruitée. Résultats, sans et avec les trimming templates (resp. algorithmes* WT *et* WT2*) (*REPR2*).*

A.5.6 Algorithme de Manzanera et Bernard (MB_2D) [BM99][MBPL99c]

C'est un algorithme de type fortement parallèle. Le schéma est décrit par l'algorithme 29.

Le support de l'algorithme est de 21 points (en comptant le point central) : il s'agit des points d'un carré 5×5 points, excepté les coins.

À la section 6.7.4, nous avons montré que cet algorithme préservait bien la topologie dans le cas 3D (nous l'avons également vérifié dans le cas 2D). Si P vérifie α_0 et non β ou si P vérifie α_1 (à ce moment, il ne peut vérifier β), alors P est simple (4- ou 8-simple) et est 4-adjacent (cas α_0) ou 8-adjacent (cas α_1) à un point intérieur (intérieur selon la connexité $(8, 4)$). La première version de l'algorithme [MBPL99b] ne tenait pas compte du motif α_1, et ne retirait pas les points simples 8-adjacents à un point intérieur s'ils n'étaient pas également 4-adjacents à un point intérieur (non forcément le même).

Les rubans d'épaisseur 2 horizontaux, verticaux ou obliques ne peuvent être amincis (**PROB**4). Il en est de même des composantes connexes dans un carré unité 2×2 points. À la section B.4.3 est étudiée la version 3D de cet algorithme.

Considérons un ruban de dimensions m fois n ($m > n$). Les quatre coins du ruban vérifient le motif α_1, les autres points de bord vont vérifier le motif α_0 (si $n > 2$). La couche extérieure va alors être retirée à la première itération, et ainsi de suite. Nous pouvons constater que $\lfloor \frac{n-1}{2} \rfloor$ itérations sont requises pour amincir un tel objet, le squelette obtenu est de largeur 1 (resp. 2) si le ruban initial est de largeur impaire (resp. paire). Ce processus de squelettisation est isotrope : une conséquence immédiate est le bon centrage des squelettes.

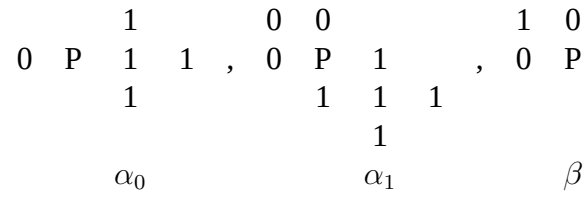
Ici encore, le bon centrage est obtenu au détriment de l'épaisseur du squelette, celui-là pré-

Répéter

Supprimer en parallèle les points P de l'objet vérifiant :

le motif α_0 ou le motif α_1, et non le motif β

(motifs à considérer à rotations de 90, 180 ou 270 degrés près) :



Jusqu'à ce qu'il n'y ait plus de point supprimé durant une itération.

Algorithme 29: *Algorithme de Manzanera et Bernard (*MB_2D*).*

sente des points redondants. C'est le cas par exemple des rubans d'épaisseur 2 qui ne sont pas amincis, comme nous l'avons dit ci-avant. Les résultats obtenus avec cet algorithme sont donnés aux figures A.33 à A.40.

A.5.7 Remarques concernant les voisinages étendus mis en jeu par les algorithmes fortement parallèles

A.5.7.1 Utilisation d'un voisinage étendu

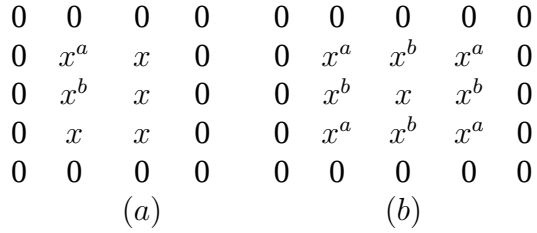


FIG. A.32 – *Cette figure explique la raison pour laquelle un algorithme fortement parallèle utilise un voisinage étendu afin de supprimer des points, tout en préservant la topologie.*

Considérons la figure A.32. La figure A.32 (a) représente un ruban d'épaisseur 2, tous les points sont simples. La figure A.32 (b) représente un ruban 3×3 points. Si un algorithme supprime les points simples de la figure A.32 (a), alors le ruban disparaît et la topologie n'est pas préservée. Mais s'il ne s'autorise pas à retirer de tels points (notés x^a ou x^b) alors l'objet de la figure (b) ne peut être aminci étant donné que $N_8^*(x^a)$ et $N_8^*(x^b)$ sont les mêmes pour les objets des figures (a) et (b).

Afin de distinguer ces deux cas, il faut alors considérer un voisinage étendu.

A.5.7.2 Terminologie employée : distinction entre algorithmes fortement parallèles symétrique et non symétrique

La classe des algorithmes fortement parallèle est distinguée par rapport à celle des algorithmes de type sous-itérations (directionnelles et sous-mailles). Cependant, comme nous l'avons vu auparavant, certains algorithmes fortement parallèles privilégient des directions de préservation dans les rubans horizontaux ou verticaux, de façon à pouvoir les amincir tout en préservant la topologie. La squelettisation n'est alors pas isotrope. Quant à lui, l'algorithme de Manzanera ne privilégie pas de direction, le processus de squelettisation est parfaitement symétrique ; l'inconvénient principal est la grande taille du support utilisé.

A.6 Résultats et comparaisons

A.6.1 Critères de comparaisons

Nous donnons quelques critères de comparaisons, recensés le long des articles traitant des algorithmes de squelettisation [CCS95], [GH89], [LS95] :

- la préservation de la topologie,
- le nombre d'itérations (*parallel speed*) : c'est le nombre total d'itérations afin d'obtenir le squelette, avec les conventions suivantes : le nombre d'itérations étant soit le nombre d'itérations exécutées par un algorithme fortement parallèle, soit le nombre de sous-itérations pour un algorithmes de type sous-itérations directionnelles, soit le nombre d'activations de sous-maille pour les algorithmes de type sous-mailles [GH89],
- la taille du support : nombre de points à examiner pour décider de la suppressibilité d'un point (pour les algorithmes utilisant des templates, c'est le nombre de points mis en jeu dans ces templates),
- le taux de compression : $(S_o - S_{squel})/S_o$, avec S_o le nombre de points de l'objet initial, S_{squel} le nombre de points dans le squelette,
- le nombre de points redondants divisé par le nombre de points dans le squelette. Un point *redondant* est un point du squelette qui n'est pas extrémité et dont la suppression ne déconnecte pas le squelette. Le *nombre de points redondants* est défini comme le nombre maximal de points pouvant être supprimés simultanément sans déconnecter le squelette [GH89], c'est une quantité difficile à évaluer,
- le fait de pouvoir recouvrer l'objet initial à partir du squelette et d'une information supplémentaire, par exemple, les numéros d'itération durant lesquelles les points voisins du squelette, ont été supprimés. Cette possibilité est connue sous le nom de *reconstruction*,
- le nombre de points retirés à la première itération,
- les comparaisons avec des squelettes de référence, les déviations entre les squelettes idéaux et obtenus de rubans synthétiques sans bruit ou avec bruit,
- la préservation de l'allure générale (géométrie) : pas d'érosion excessive, squelette d'épaisseur 1, bon centrage du squelette, immunité aux bruits.

Finalement, les critères de qualité que nous retenons sont par ordre de priorité :

- la préservation de la topologie,

– le nombre d'itérations effectives de suppression,
– le nombre total de points supprimés,
– le nombre moyen de points supprimés par itération effective.

Il s'agit également des critères que nous retenons pour le cas des algorithmes 3D.

A.6.2 Récapitulatif

Nous avons d'abord étudié quelques algorithmes de type sous-itérations directionnelles :
– en 2 sous-itérations : Zhang et Suen (ZS), Lü et Wang (LW) (ces deux algorithmes ne préservent pas la topologie (**PROB**3)), Guo et Hall (GH_A1),
– en 4 sous-itérations : Rosenfeld (ROS). Ce dernier est un des algorithmes les plus connus ; on rappelle qu'il met en œuvre la technique directionnelle $Nord$, Sud, Est, $Ouest$.

Puis, nous avons étudié un algorithme en 2 sous-mailles de Guo et Hall (GH_A2), que nous avons comparé avec quelques algorithmes de type sous-itérations directionnelles pour quelques objets. Nous avons constaté que les résultats dépendaient de l'ordre d'activation des sous-mailles et n'étaient pas indépendants par translation de l'objet (**PROB_SM**). De plus, les squelettes obtenus présentent un aspect dentelé.

Enfin, nous avons traité des algorithmes fortement parallèles :
– Holt, Stewart, Clint et Perrott (HSCP) d'un support de 16 points. Les rubans obliques d'épaisseur 2 sont réduits de façon excessive (**PROB**2) ; la notion de filtrage intégré au processus de squelettisation est abordée,
– la modification par R.W. Hall (HSCPN) d'un support de 16 points. Les bandes obliques ne sont pas réduites (**PROB**4),
– trois algorithmes par Z. Guo et R.W. Hall :
  – (GH_AFP1) d'un support de 11 points. Les bandes obliques ne sont pas réduites (**PROB**4),
  – (GH_AFP2) d'un support de 11 points. Les bandes obliques ne sont pas réduites (**PROB**4),
  – (GH_AFP3) d'un support de 11 points. Les bandes obliques sont réduites.
– Chin, Wan, Stover et Iverson (CWSI) d'un support de 15 points. La topologie n'est pas préservée (**PROB**3),
– Wu et Tsai (WT) d'un support de 15 points. La topologie n'est pas préservée (**PROB**3),
– Manzanera et Bernard (MB_2D) support de 21 points. Les rubans d'épaisseur 2 horizontaux, verticaux et obliques ne sont pas amincis (**PROB**4).

Dans la suite, afin de faciliter les comparaisons, nous ne retenons que les algorithmes GH_A1, ROS, GH_A2, HSCPN, GH_AFP2, GH_AFP3 et MB_2D. Les figures A.33 à A.40 présentent les squelettes obtenus en utilisant ces 7 algorithmes.

A.6.3 Comparaisons

Il est difficile de pouvoir comparer les résultats de ces algorithmes. Par exemple, l'algorithme ROS et l'algorithme MB_2D retirent tous les deux 222 points de la lettre "H" (Fig. A.37). Néanmoins, l'algorithme ROS semble moins efficace : il utilise 12 sous-itérations effectives de suppression tandis que l'algorithme MB_2D n'en exige que 3. Considérons maintenant les résultats de ces deux algorithmes sur la lettre A (Fig. A.33). L'algorithme ROS exige 14

sous-itérations effectives, tandis que l'algorithme MB_2D n'en exige que 4. Mais ce dernier retire moins de points (464) que le premier (508) ; le squelette obtenu avec MB_2D présente de nombreux points redondants (*i.e.* points simples non terminaux). Il semble plus difficile de parler d'efficacité d'un algorithme sans prendre en compte le nombre de points supprimés.

Sur tous les exemples, on peut s'apercevoir de l'aspect dentelé des squelettes avec l'algorithme GH_A2 en 2 sous-mailles (par exemples, Fig. A.36, Fig. A.39 et Fig. A.40).

A.7 Conclusion

Dans ce chapitre, quelques algorithmes parallèles de squelettisation d'images 2D binaires ont été étudiés.

Nous avons laissé de côté les algorithmes séquentiels (suppression séquentielle de point simple, selon un balayage vidéo par exemple), qui ne sont guère utilisés du fait de leur lenteur par rapport aux algorithmes parallèles, ou du mauvais centrage des squelettes obtenus. D'autres algorithmes utilisent des notions de distance ou la morphologie mathématique. Bien que très intéressants, nous n'en parlons pas dans ce mémoire.

Nous avons proposé un ou plusieurs algorithmes pour chacune des trois grandes classes d'algorithmes :

- les algorithmes de type sous-itérations directionnelles,
- les algorithmes de type sous-mailles,
- les algorithmes fortement parallèles.

De cette étude, retenons que :

- les squelettes obtenus par des algorithmes de type sous-mailles présentent un aspect dentelé,
- les squelettes obtenus par des algorithmes de type sous-itérations directionnelles sont d'autant mieux centrés que le nombre de directions est élevé,
- le centrage des squelettes est meilleur avec des algorithmes fortement parallèles (étant donné une squelettisation homogène), en contrepartie ces algorithmes utilisent un plus grand support pour examiner si un point peut être supprimé ou non (cela afin de préserver la topologie),
- la plupart des algorithmes fortement parallèles utilise des directions et le processus de suppression de points n'opère pas de façon symétrique. En contrepartie, l'algorithme de Manzanera opère de façon isotrope, le squelette obtenu est bien centré mais non forcément mince.

Retenons également que c'est à partir de petits objets synthétiques (et simples, par exemple, un carré 2×2 points, un ruban d'épaisseur 2) qu'a été mise en évidence la plupart des critères permettant d'"évaluer" un algorithme de squelettisation.

Le chapitre suivant traite des algorithmes de squelettisation 3D. Les mêmes classes d'algorithmes y seront vues. Nous pourrons constater la différence de difficulté pour comparer ou étudier les algorithmes 3D, par rapport aux algorithmes 2D. Soulignons qu'il existe plusieurs centaines d'algorithmes de squelettisation pour images 2D, et moins d'une trentaine d'algorithmes pour images 3D.

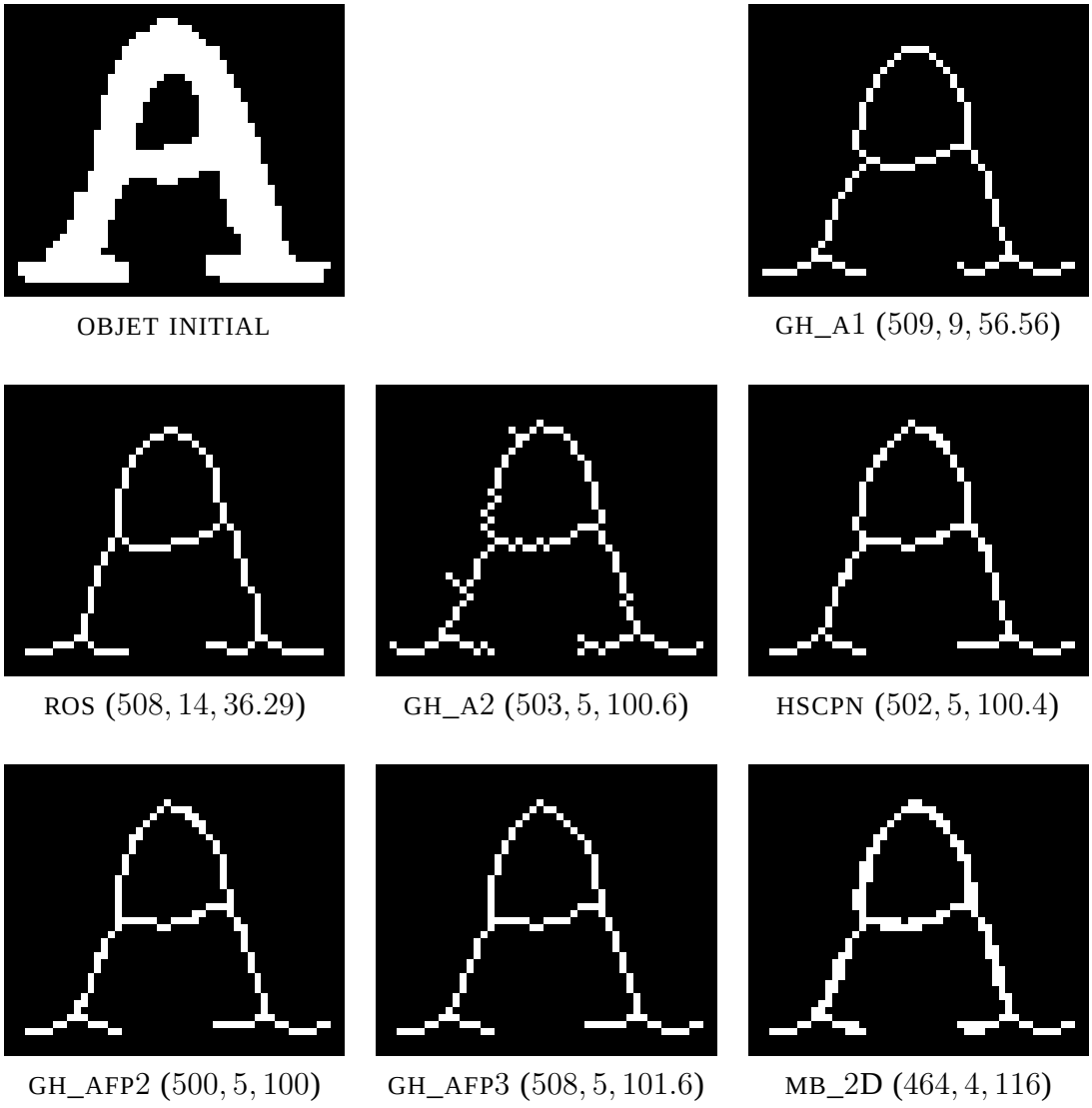


FIG. A.33 – *Squelettisations avec différents algorithmes (lettre "A") (*REPR1).

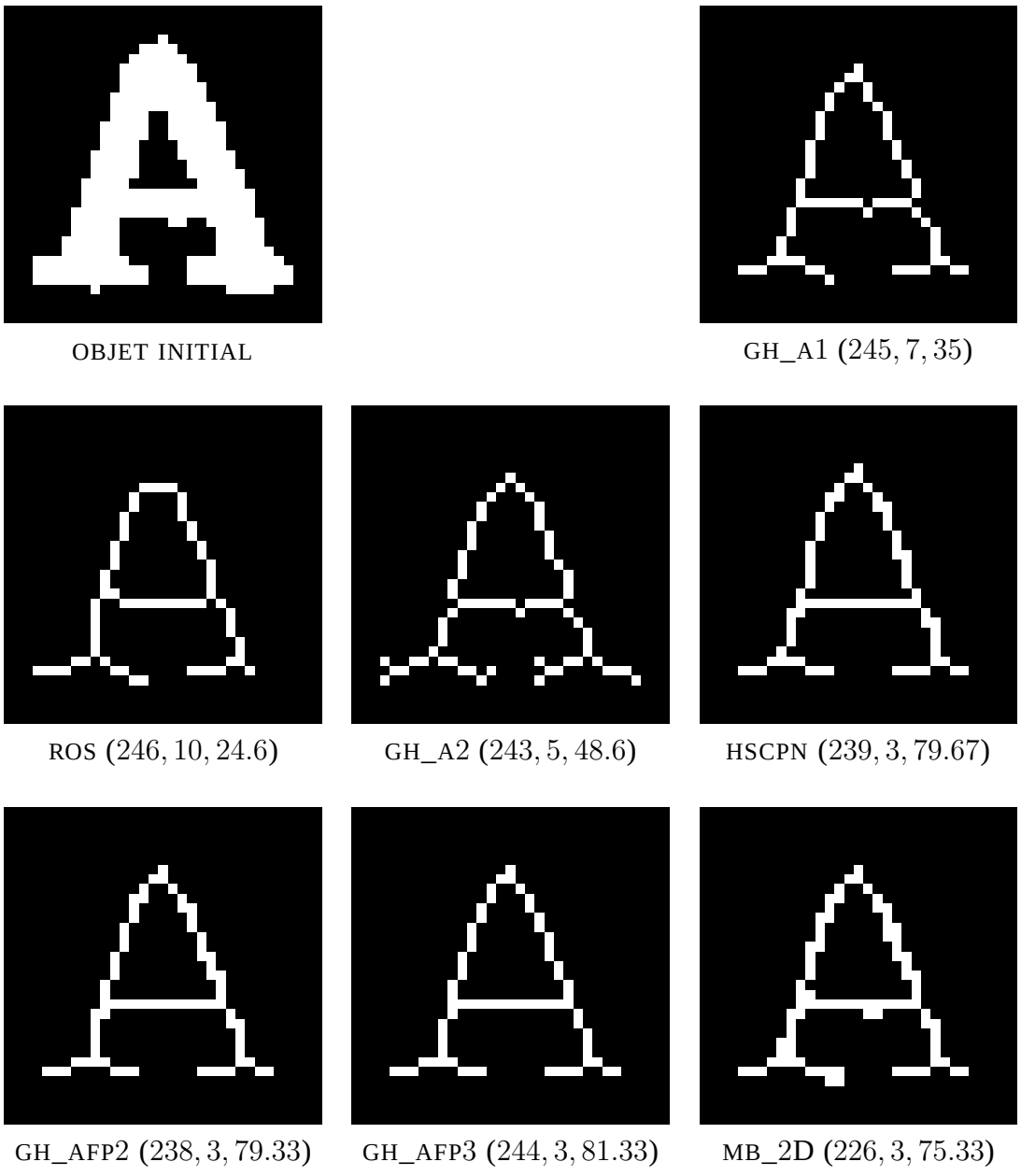


FIG. A.34 – *Squelettisations avec différents algorithmes (lettre "A" bruitée) (*REPR1).

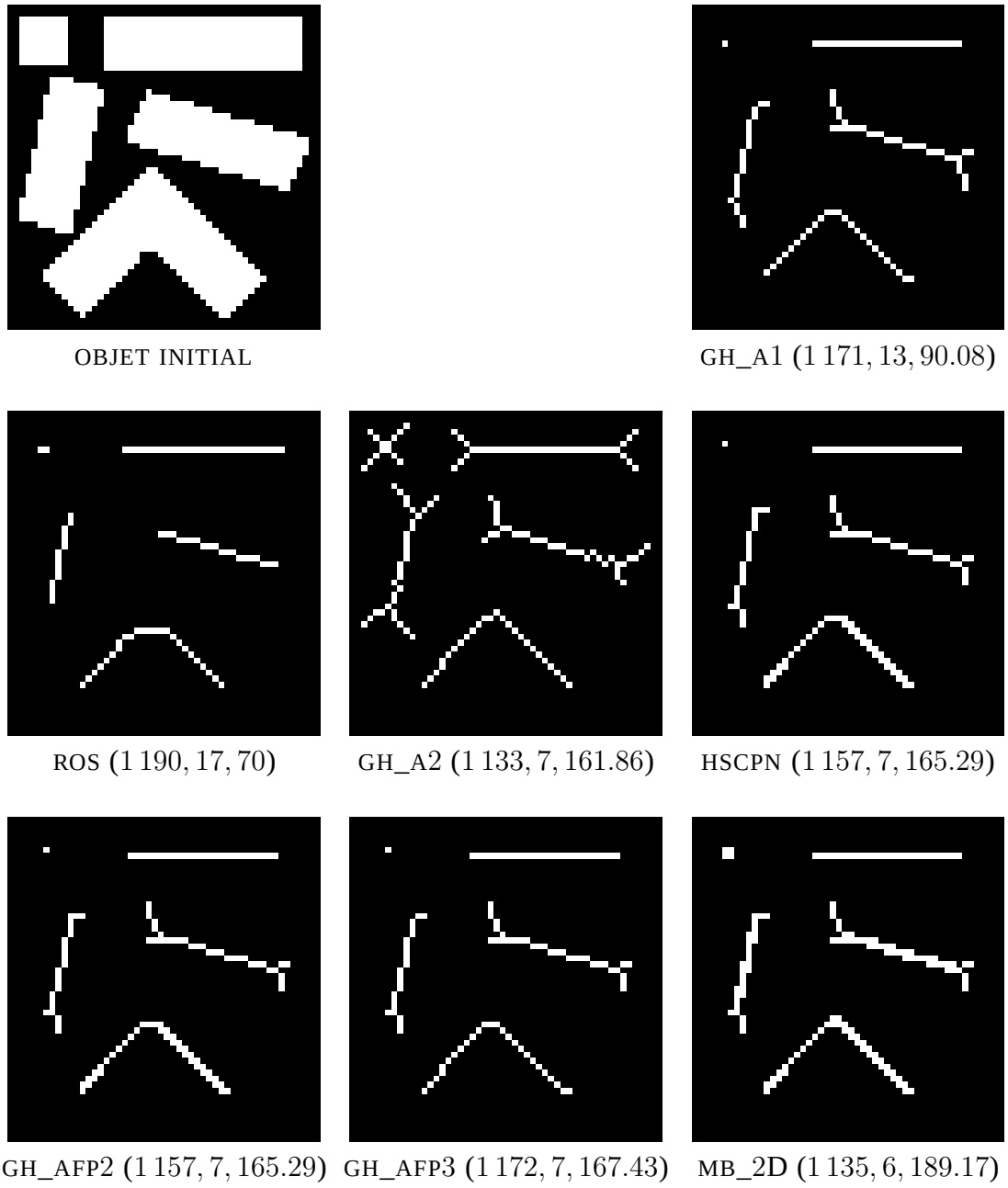
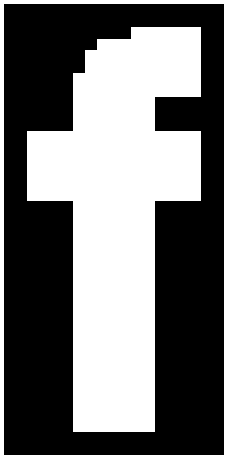
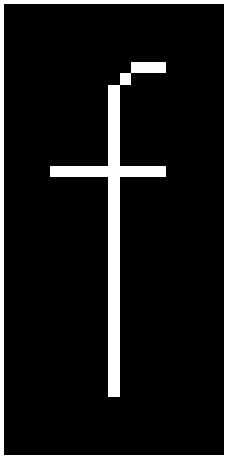


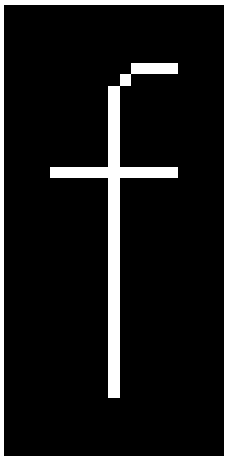
FIG. A.35 – *Squelettisations avec différents algorithmes (divers rubans) (*REPR1).



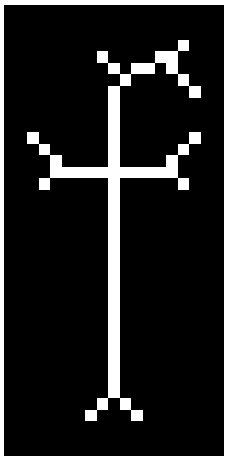
OBJET INITIAL



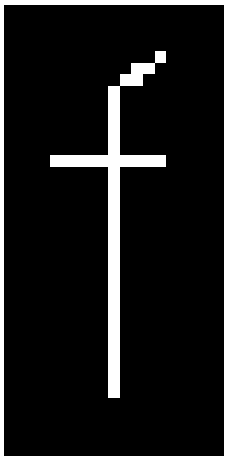
GH_A1 $(268, 12, 22.33)$



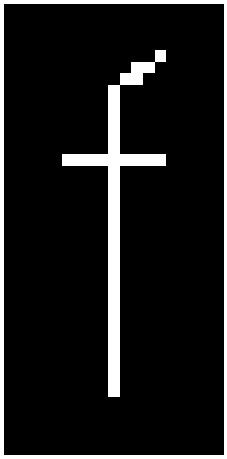
ROS $(266, 12, 22.17)$



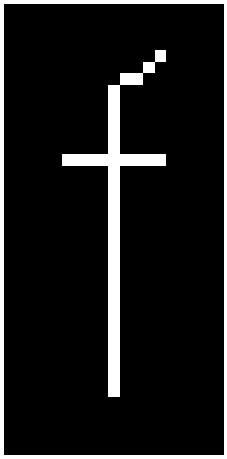
GH_A2 $(248, 6, 41.33)$



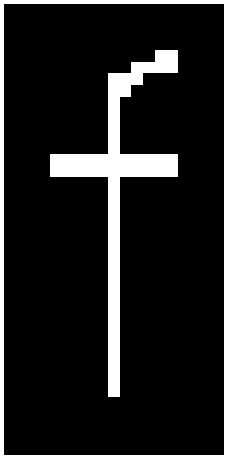
HSCPN $(267, 6, 44.5)$



GH_AFP2 $(268, 6, 44.67)$



GH_AFP3 $(269, 6, 44.83)$



MB_2D $(251, 5, 50.2)$

FIG. A.36 – *Squelettisations avec différents algorithmes (lettre "F") (*REPR1).

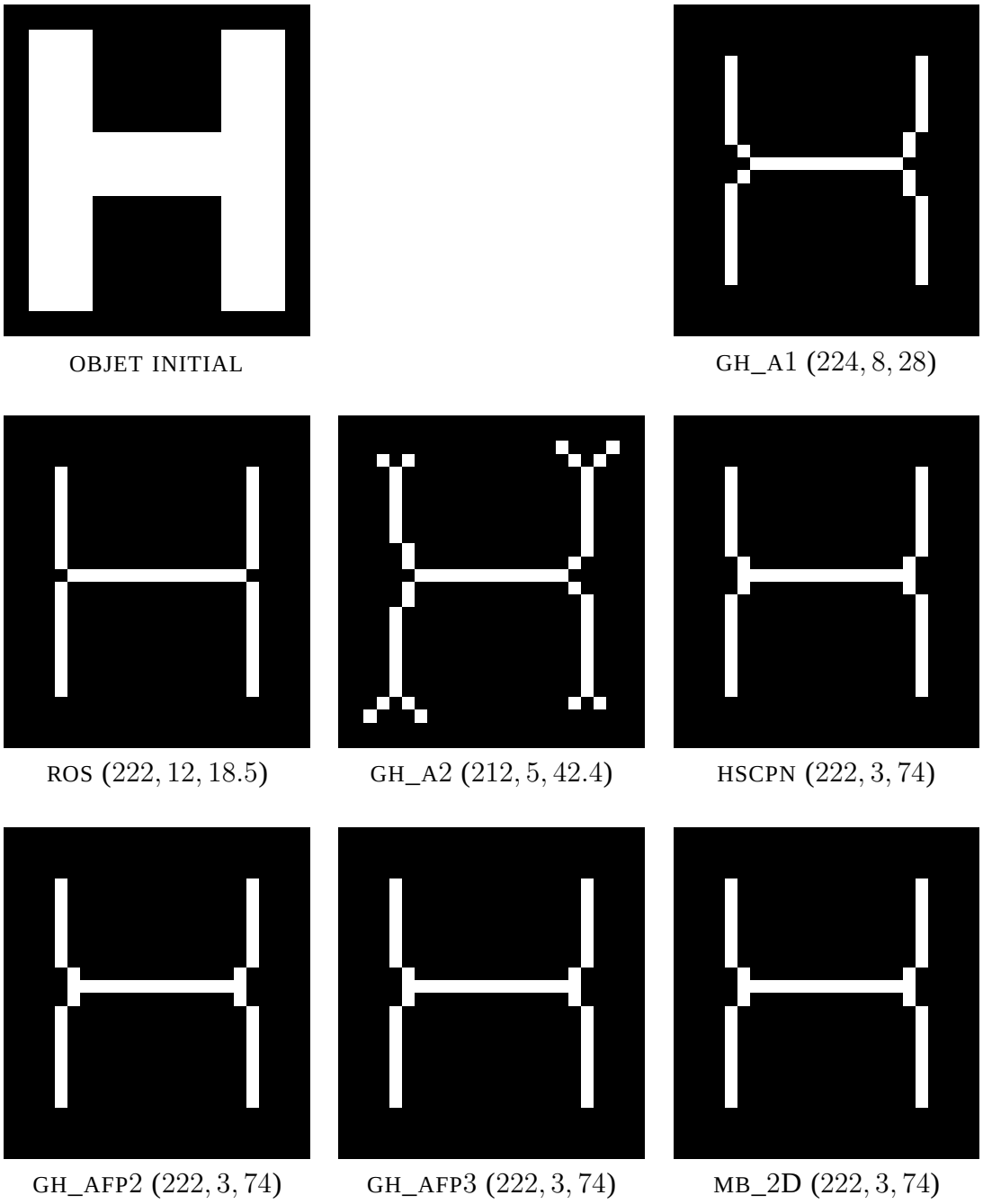


FIG. A.37 – *Squelettisations avec différents algorithmes (lettre "H") (*REPR1).

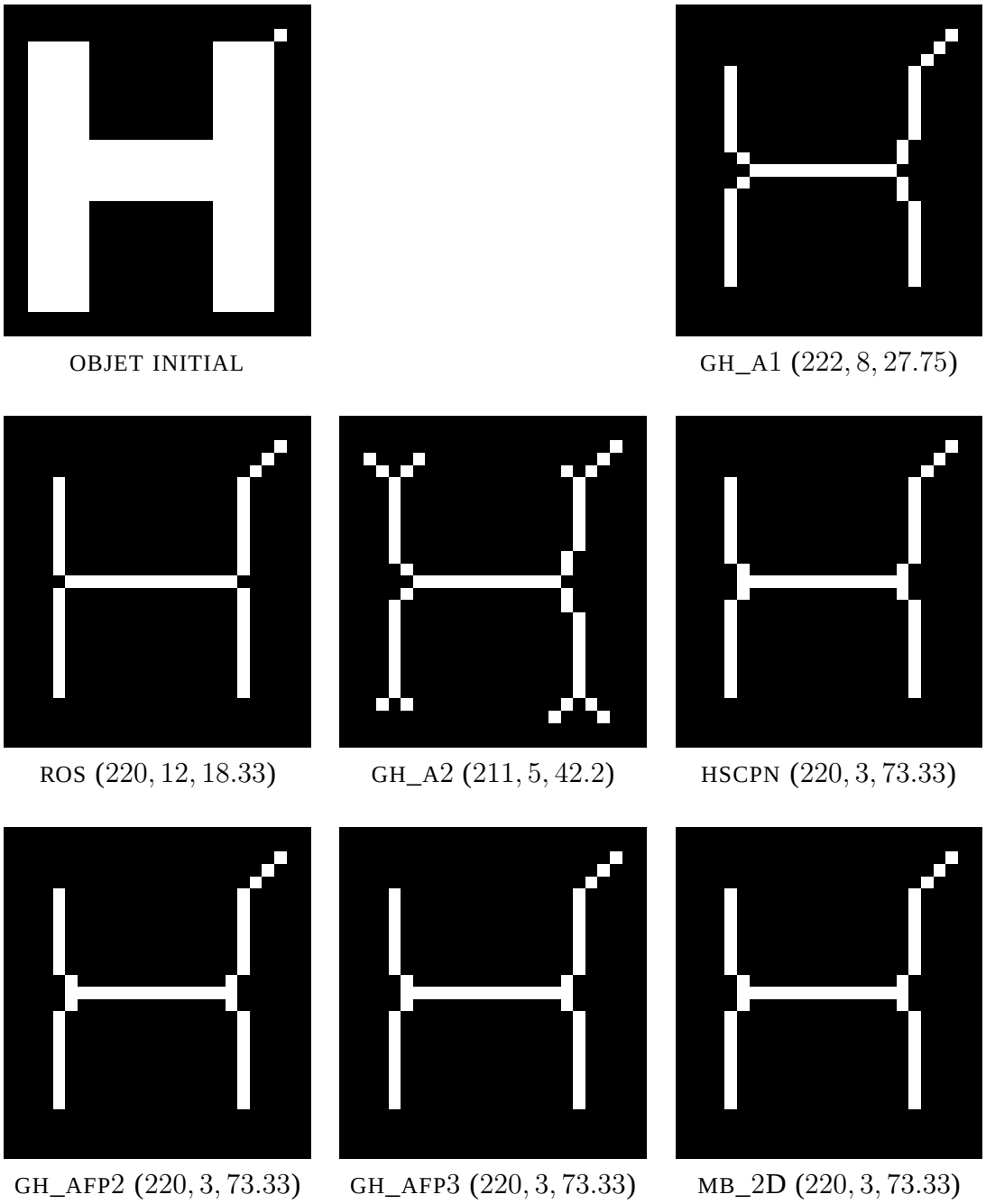


FIG. A.38 – *Squelettisations avec différents algorithmes (lettre "H" bruitée) (*REPR1).

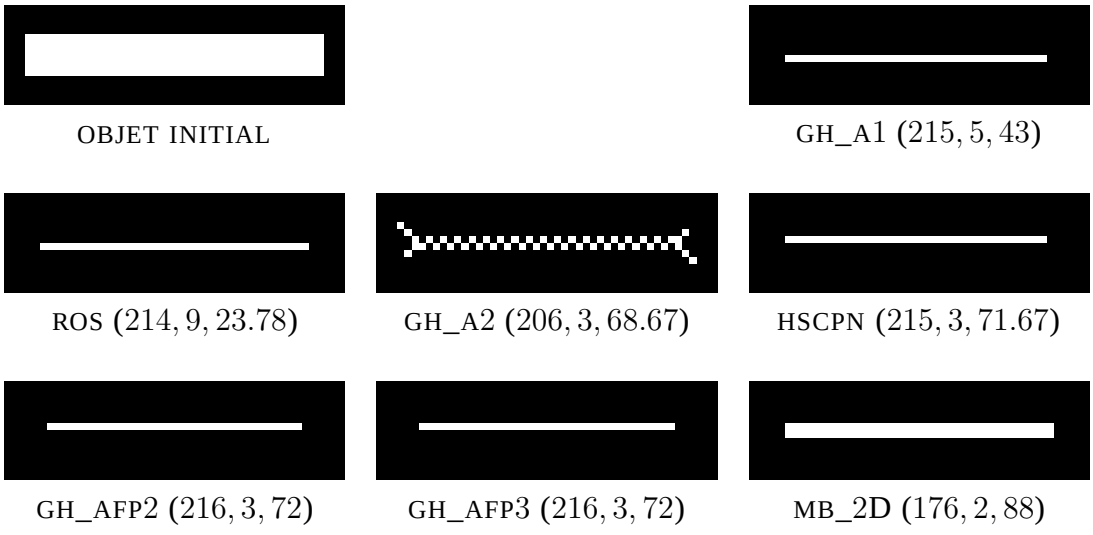


FIG. A.39 – *Squelettisations avec différents algorithmes (ruban épais) (*REPR1).

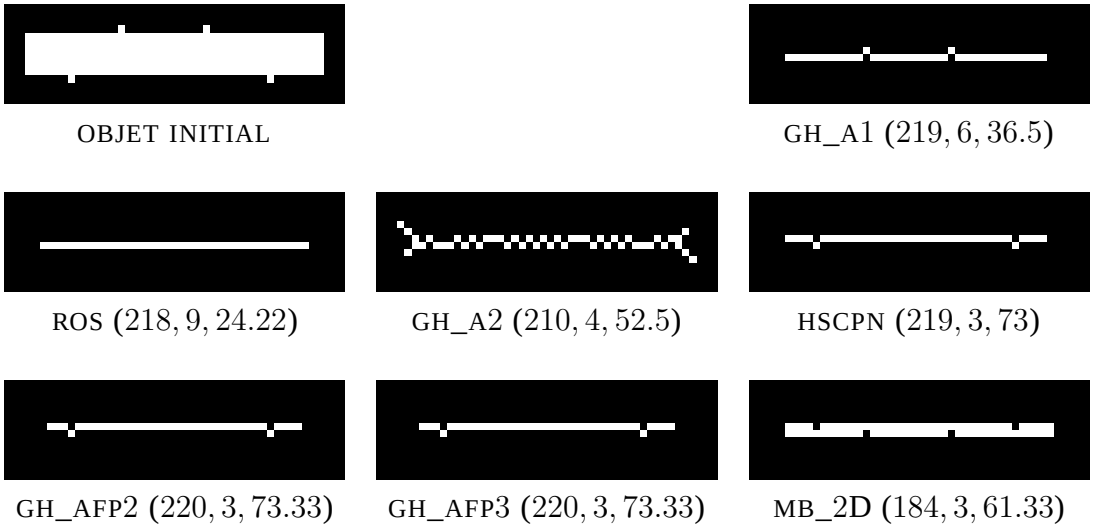


FIG. A.40 – *Squelettisations avec différents algorithmes (ruban épais bruité) (*REPR1).

Annexe B

Algorithmes de squelettisation 3D

Dans ce chapitre sont étudiés des algorithmes pour images 3D binaires ; algorithmes de type sous-itérations, sous-mailles ou fortement parallèles (cf. chapitre 4).

B.1 Représentations et notations utilisées

B.1.1 Représentations

Nous utilisons les représentations suivantes :

- REPR6 : une succession de captures illustrant les résultats obtenus par les (sous-)itérations de suppression. Les voxels retirés lors d'une (sous-)itération sont représentés en vert. Sous l'objet courant, obtenu à la fin de la i-ième (sous-)itération de suppression de points sera indiqué $ite = i$, suivi entre parenthèses du nombre de points supprimés durant cette i-ième (sous-)itération.
- REPR7 : une capture d'écran représentant les points du squelette par des cubes rouges, dans l'objet initial en transparence, à partir duquel le squelette a été obtenu. Sous une telle capture, seront notés entre parenthèses, respectivement le numéro de la dernière (sous-)itération effective de suppression, le nombre de points supprimés et le nombre moyen de points supprimés par (sous-)itération.

B.1.2 Notations

Le système de coordonnées est celui de la figure B.1 (a). Notons que lorsqu'une isométrie est utilisée dans un article de référence, elle est d'abord transposée par rapport à notre système d'axes, puis choisie en conséquence parmi celles proposées dans l'annexe D.

L'orientation utilisée est celle donnée à la figure B.1 (b). À partir de ces 6 directions principales : $Haut$, Bas, $Nord$, Sud, Est, $Ouest$, d'autres seront utilisées "par composition", par exemple $Haut - Sud - Ouest$.

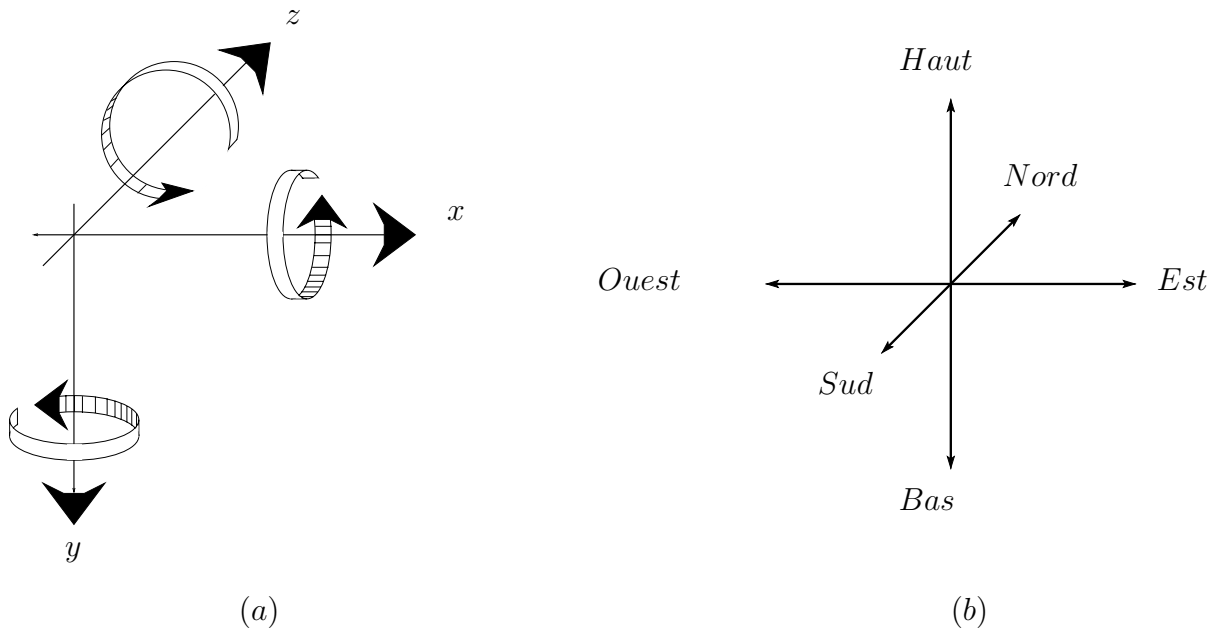


FIG. B.1 – (a) *Système de coordonnées et* (b) *orientation utilisés.*

B.2 Algorithmes de squelettisation de type sous-itérations directionnelles

Nous examinons d'abord deux algorithmes de type 6 sous-itérations, puis deux algorithmes de type 12 sous-itérations.

B.2.1 Algorithme de Palágyi et Kuba - 6 sous-itérations (PAKU6) [PK98a]

B.2.1.1 Description

Cet algorithme de type sous-itérations directionnelles permet l'obtention de squelettes curvilignes. Il est décrit par le schéma suivant : une itération de suppression de points est constituée de 6 sous-itérations ; chacune d'entre elles correspond à une des 6 directions de suppression choisies. Pour chacune des sous-itérations, un point est supprimé si son voisinage vérifie au moins l'un des motifs d'un ensemble de motifs cibles, appelés *templates* par la suite (ensemble déterminé par la direction sous-jacente à la sous-itération).

Nous notons $\mathcal{F}(X, Dir)$ l'opération réalisant la suppression en parallèle des points d'un objet X, et vérifiant au moins l'une des templates de direction Dir. Le schéma de squelettisation est décrit par l'algorithme 30. Selon notre implémentation, l'opération $\mathcal{F}(X, Dir)$ appelle le BDD obtenu pour la direction Dir.

Un BDD est un outil algorithmique (graphe) codant de façon efficace, les configurations vérifiant une propriétée donnée. Un programme recevant ces configurations en entrée génère un code décrivant un tel BDD. Ensuite, si une configuration est donnée en entrée de ce code, alors

nous pouvons savoir très rapidement si elle vérifie la propriété sur laquelle a été construit le BDD. De plus amples détails concernant la création et l'utilisation des BDD sont donnés à l'annexe C.

Y : image initiale
Répéter
$Y = \mathcal{F}(Y, Haut)$;
$Y = \mathcal{F}(Y, Bas)$;
$Y = \mathcal{F}(Y, Nord)$;
$Y = \mathcal{F}(Y, Sud)$;
$Y = \mathcal{F}(Y, Est)$;
$Y = \mathcal{F}(Y, Ouest)$;
Jusqu'à ce qu'il n'y ait plus de suppression durant 6 sous-itérations successives.

Algorithme 30: *Algorithme de Palágyi et Kuba en* 6 *sous-itérations (*PAKU6*).*

Nous devons implémenter les motifs que doit vérifier le voisinage d'un point, afin de savoir si ce point peut être supprimé ou non, et cela pour chacune des directions intrinsèques aux sous-itérations. Nous avons choisi d'utiliser les BDD, pour coder les templates pour chacune des directions mises en œuvre. Les motifs sont donnés pour la seule direction $Haut$ et il faut de plus considérer leurs rotations (90, 180 et 270 degrés) selon l'axe $Haut - Bas$. (Fig. B.2). Ce travail est à réaliser pour les 6 directions.

À la figure B.3 est décrite la chaîne de traitement utilisée pour l'obtention des BDD. Elle se décompose selon les trois étapes suivantes :

- Les configurations vérifiant les 6 motifs pour la direction $Haut$ sont retenues parmi toutes les configurations possibles dans un voisinage $3 \times 3 \times 3$, puis pour chacune d'elles les trois rotations selon l'axe $Haut - Bas$ sont générées (en utilisant les isométries $r_y(iso32)$, $r_y^2(iso23)$, $r_y^3(iso40)$, voir l'annexe D). Nous obtenons ainsi les configurations vérifiant le Masque $Haut$.
- À partir de cet ensemble de configurations (Masque $Haut$) sont déterminées celles vérifiant les 5 autres séries de masque pour les 5 autres directions Sud, Bas, $Nord$, Est et $Ouest$ (par les isométries $r_x(iso8)$, $r_x^2(iso16)$, $r_x^3(iso24)$, $r_z(iso1)$ et $r_z^3(iso2)$).
- Puis les codes des BDD correspondant à ces configurations sont générés (voir l'annexe C).

Nous obtenons ainsi une caractérisation efficace des points pouvant être supprimés pour chacune des directions utilisées par cet algorithme.

B.2.1.2 Commentaires et résultats

Les templates Masque $Haut$ sont des configurations de points 26-simples et de bord $Haut$. La présence d'au moins un cube noir dans les templates M_1 et M_5 évite de supprimer des points 26-simples extrémités. Comme nous l'avons déjà dit auparavant, si l'ordre des directions est changé alors les résultats peuvent être différents.

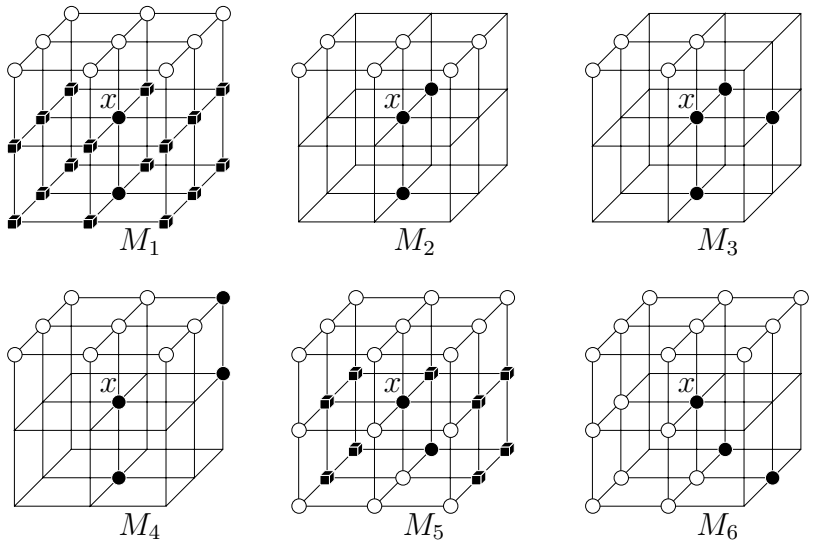


FIG. B.2 – *Masque $Haut$: Templates utilisées par l'algorithme* PAKU6*. Au moins un point représenté par un cube doit appartenir à l'objet dans les masques M_1 et M_5.*

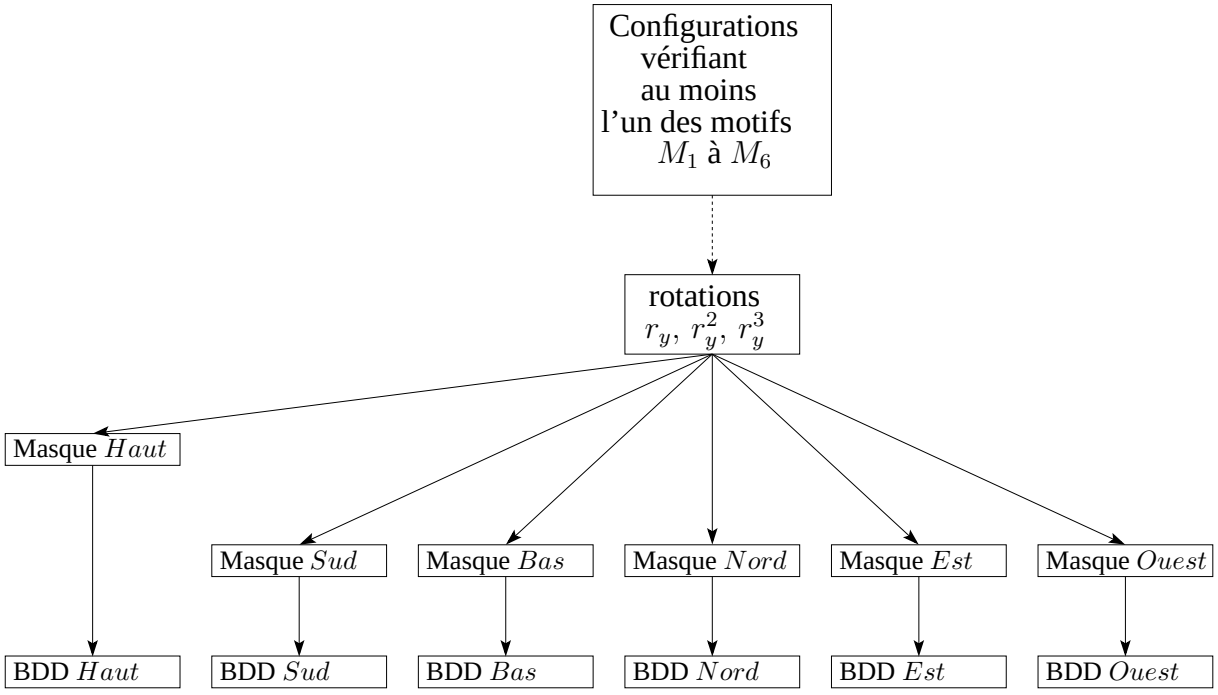


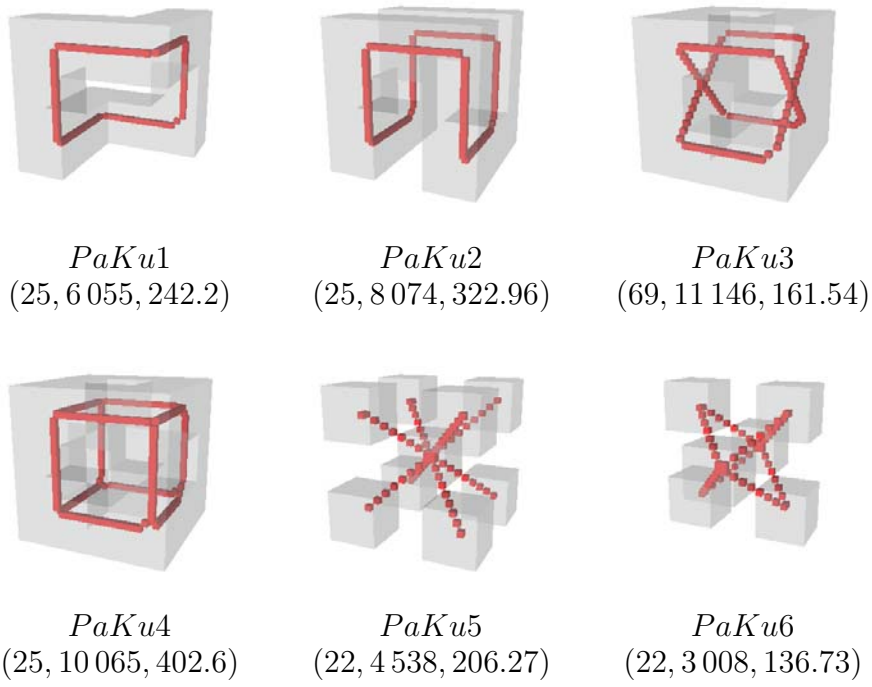
FIG. B.3 – *Chaîne de traitement pour obtenir les masques et leurs codes BDD associés (algorithme* PAKU6*).*

Il y a $705\,214$ configurations vérifiant les templates représentées ; il y en a $2\,124\,283$ avec les rotations (pour une sous-itération).

Des résultats de squelettisation curviligne avec l'algorithme PAKU6 sont présentés à la figure B.4. L'objet constitué de 9 cubes épais (objet $PaKu5$) comporte des branches se terminant par un ou deux points. Certaines branches conservées se terminent par un point 6-adjacent à un autre, tandis que d'autres branches se terminent par un point diamétralement opposé à un autre. Cela exprime le fait que le processus n'est pas isotrope, c'est évidemment le cas car il dépend de directions.

À la figure B.5 sont représentées les 6 premières sous-itérations de suppression, afin d'observer le rôle des directions de suppression.

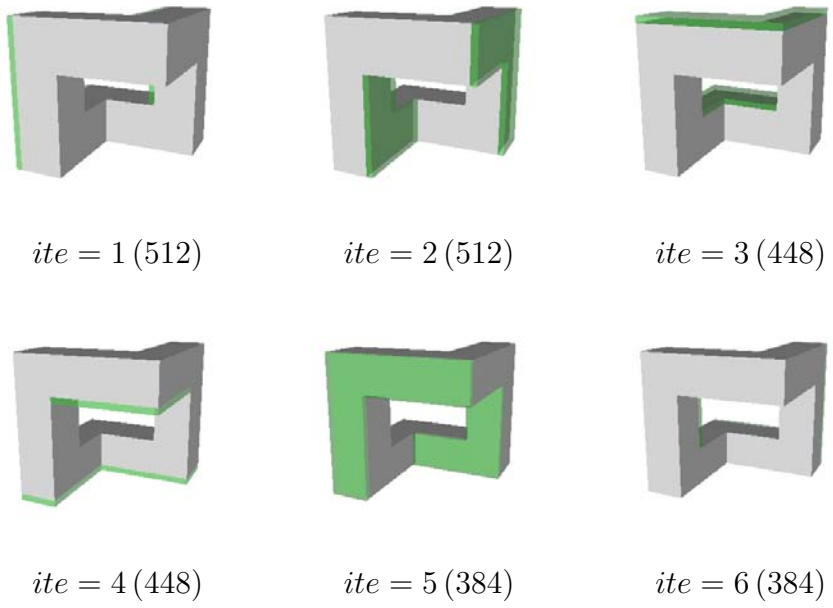
Remarque : Nous avons vérifié avec les points P-simples que cet algorithme préserve la topologie (cf. section 6.5.2.2). Nous avons proposé un algorithme basé sur les points P^x-simples, supprimant au moins les points retirés par cet algorithme tout en préservant les mêmes points extrémités (cf. section 6.5.3) ; nous avons également comparé cet algorithme avec l'algorithme de Gong et Bertrand (GOBE), présenté à la section B.2.2 (cf. section 6.5.4).



$PaKu1$ $(25, 6\,055, 242.2)$ | $PaKu2$ $(25, 8\,074, 322.96)$ | $PaKu3$ $(69, 11\,146, 161.54)$

$PaKu4$ $(25, 10\,065, 402.6)$ | $PaKu5$ $(22, 4\,538, 206.27)$ | $PaKu6$ $(22, 3\,008, 136.73)$

FIG. B.4 – *Squelettisations avec l'algorithme* PAKU6 *(*REPR7*).*



$ite = 1\,(512)$ | $ite = 2\,(512)$ | $ite = 3\,(448)$

$ite = 4\,(448)$ | $ite = 5\,(384)$ | $ite = 6\,(384)$

FIG. B.5 – *Détails des* 6 *premières sous-itérations de l'algorithme* PAKU6 *sur l'objet* $PaKu1$ *(*REPR6*).*

B.2.2 Algorithme de Gong et Bertrand en 6 sous-itérations (GOBE) - [GB90]

Cet algorithme consiste en la répétition jusqu'à stabilité de 6 sous-itérations directionnelles $Haut$, $Nord$, Est, Bas, Sud, $Ouest$ de suppression parallèle de points (selon la connexité $(26,6)$). Il permet l'obtention de squelettes surfaciques.

B.2.2.1 Description

Pour une direction α, parmi les 6 précédemment citées, l'algorithme élimine en parallèle les points qui vérifient les conditions données dans l'algorithme 31 ; voir les notations après l'algorithme et l'illustration des conditions à la figure B.6.

Conditions directionnelles :

$$\begin{cases} (G_1) : \alpha(x) \in \overline{X}, \\ (G_2) : \overline{\alpha}(x) \in X, \\ (G_3) : \forall y \in N^6_\alpha(x), \text{ si } y \in \overline{X} \text{ alors } \alpha(y) \in \overline{X}, \\ (G_4) : \forall y \in N^6_\alpha(x), \forall z \in N^{18}_\alpha(y) \cap N^6_\alpha(x), \\ \qquad \text{soit } t \text{ le seul élément de } N^6_\alpha(z) \cap N^6_\alpha(y) \cap N^{18}_\alpha(x), \\ \qquad \text{si } y \in \overline{X} \text{ et } z \in \overline{X} \text{ et } t \in \overline{X} \text{ alors } \alpha(t) \in \overline{X}. \end{cases}$$

Notons que dans la condition (G_4) : $\alpha(y) \in \overline{X}$ et $\alpha(z) \in \overline{X}$ (d'après (G_3)).

Condition booléenne de ne pas être un point terminal de surface :

$$n \geq 8 \text{ OU } (4 \leq n \leq 7 \text{ ET } (\exists i \in \{1,\ldots,8\}, n_i = 3));$$

avec n le nombre de points dans le 26-voisinage de p, n_i le nombre de points de l'objet 6-adjacents à p dans le i-ième octant du 26-voisinage de p (un octant étant un bloc de $2 \times 2 \times 2$ points, il y en a 8 dans $N_{26}(p)$).

Répéter
  Pour $\alpha = Haut, Nord, Est, Bas, Sud, Ouest$
    Supprimer en parallèle tout point de l'objet, non terminal
    vérifiant les conditions directionnelles selon la direction α
    et qui n'est pas un point terminal de surface
Jusqu'à ce qu'il n'y ait plus de suppression
    durant 6 sous-itérations successives.

Algorithme 31: *Algorithme de Gong et Bertrand en* 6 *sous-itérations (*GOBE*).*

Notations : Considérons une direction α. La direction opposée à α sera notée $\overline{\alpha}$. Nous désignons par $N^6_\alpha(x)$ (resp. $N^{18}_\alpha(x)$), les quatre 6-voisins (resp. les quatre 18-voisins) qui

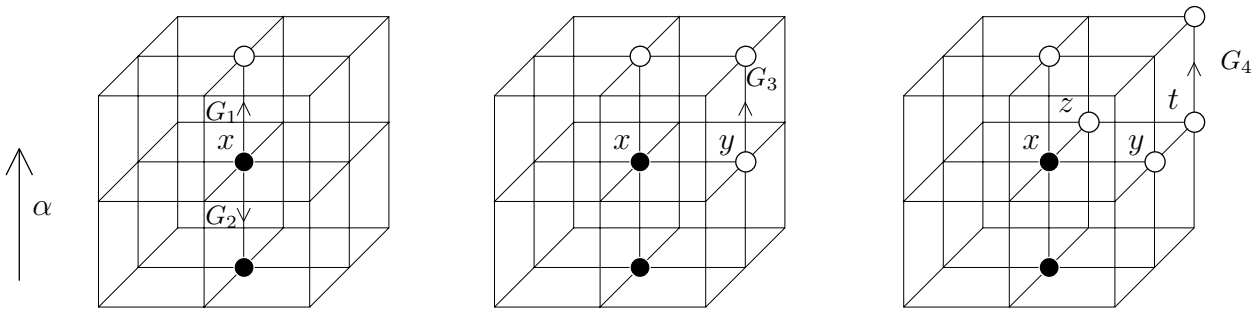


FIG. B.6 – *Conditions imposées dans l'algorithme* GOBE *(*$\alpha = Haut$*).*

appartiennent à la fenêtre 3×3 perpendiculaire à la direction α et de point central x (en fait, $N^6_\alpha(x) = N^*_6(x) \setminus \{\alpha(x), \overline{\alpha}(x)\}$).

B.2.2.2 Commentaires et résultats

C. Pudney [Pud98] a remarqué, à juste titre, que cet algorithme ne permettait pas la préservation des points simples à l'intersection de surfaces ; et avec un processus itératif de suppression de ces points, il peut ne rester qu'un point dans la jonction initiale (ou deux points, selon la parité de la largeur de la jonction) (cf. section 9.5.3). Néanmoins, les squelettes surfaciques obtenus proposent un aspect très satisfaisant.

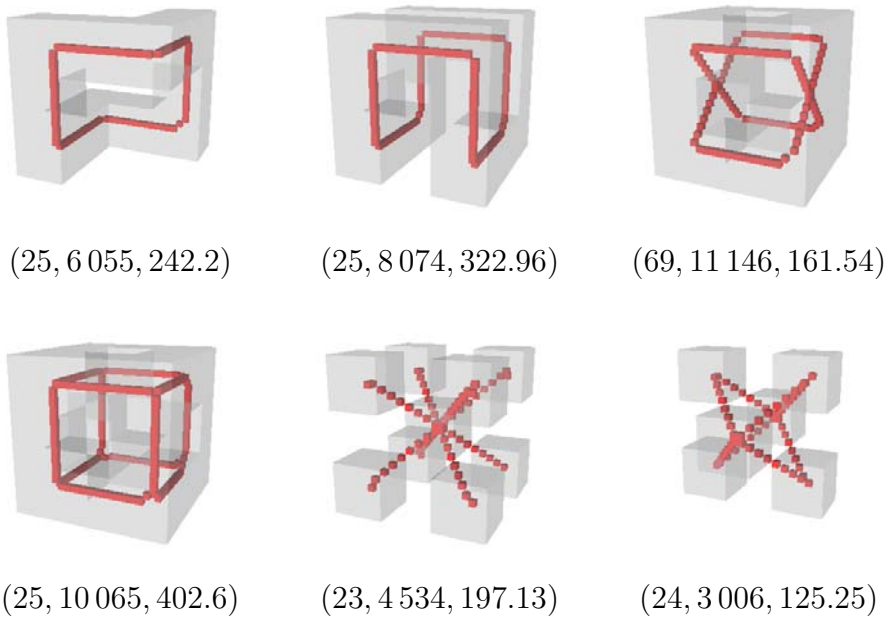
Nous proposons également les résultats obtenus par cet algorithme en remplacant la condition booléenne de point extrémité, afin d'obtenir un squelette curviligne, par la définition classique suivante : un point x est extrémité de courbe s'il n'a qu'un seul voisin dans $N^*_{26}(x) \cap X$, cette variante a été proposée par F. Rolland et al. [RCM92].

Note : nous avons choisi la même séquence de directions pour cet algorithme que celle utilisée par l'algorithme de Palágyi, plutôt que celle suggérée par l'article, afin de rendre plus aisées les comparaisons entre les résultats obtenus par ces deux algorithmes.

Des résultats de squelettisation curviligne ou surfacique sont présentés aux figures B.7, B.8 et B.10. Les 6 premières sous-itérations sur l'objet $PaKu1$ sont détaillées à la figure B.9 (ce sont les mêmes 6 premières sous-itérations qu'avec l'algorithme PAKU6, voir Fig. B.5).

Pour les 4 premiers objets (Fig. B.7), nous obtenons les mêmes résultats (mêmes squelettes, même nombre d'itérations, même nombre de points supprimés) que ceux obtenus avec l'algorithme de Palágyi en 6 sous-itérations (PAKU6) (voir Fig. B.4).

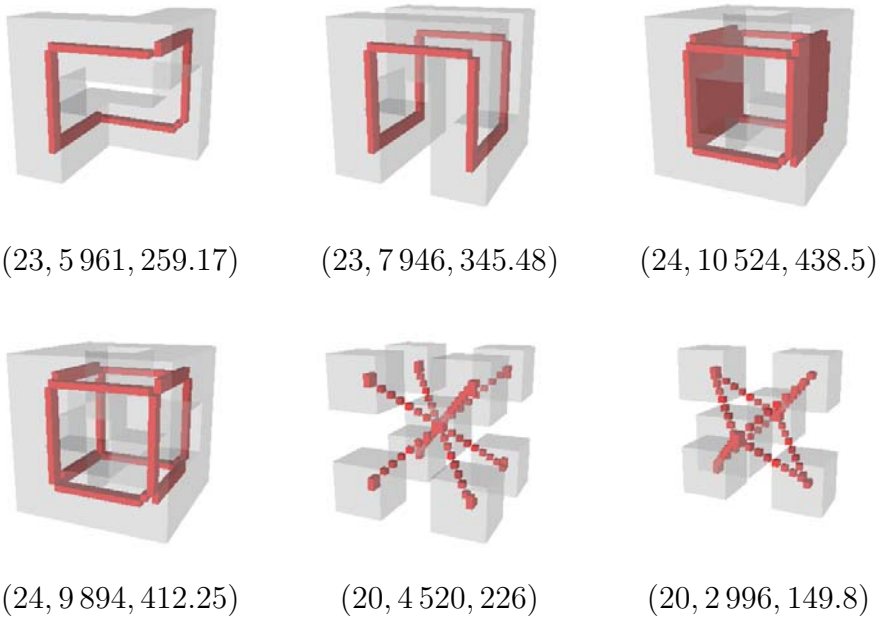
Remarque : Nous avons vérifié avec les points P-simples que cet algorithme préserve la topologie (cf. section 6.5.1.2). Nous avons proposé un algorithme basé sur les points P^x-simples, supprimant au moins les points retirés par cet algorithme tout en préservant les mêmes points extrémités dans sa version curviligne (cf. section 6.5.3) ; nous avons également comparé cet algorithme avec l'algorithme de Palágyi et Kuba en 6 sous-itérations (PAKU6), présenté à la section B.2.1 (cf. section 6.5.4).



$(25, 6\,055, 242.2)$ $(25, 8\,074, 322.96)$ $(69, 11\,146, 161.54)$

$(25, 10\,065, 402.6)$ $(23, 4\,534, 197.13)$ $(24, 3\,006, 125.25)$

FIG. B.7 – *Squelettisations curvilignes avec l'algorithme* GOBE *(*REPR7*).*



$(23, 5\,961, 259.17)$ $(23, 7\,946, 345.48)$ $(24, 10\,524, 438.5)$

$(24, 9\,894, 412.25)$ $(20, 4\,520, 226)$ $(20, 2\,996, 149.8)$

FIG. B.8 – *Squelettisations surfaciques avec l'algorithme* GOBE *(*REPR7*).*

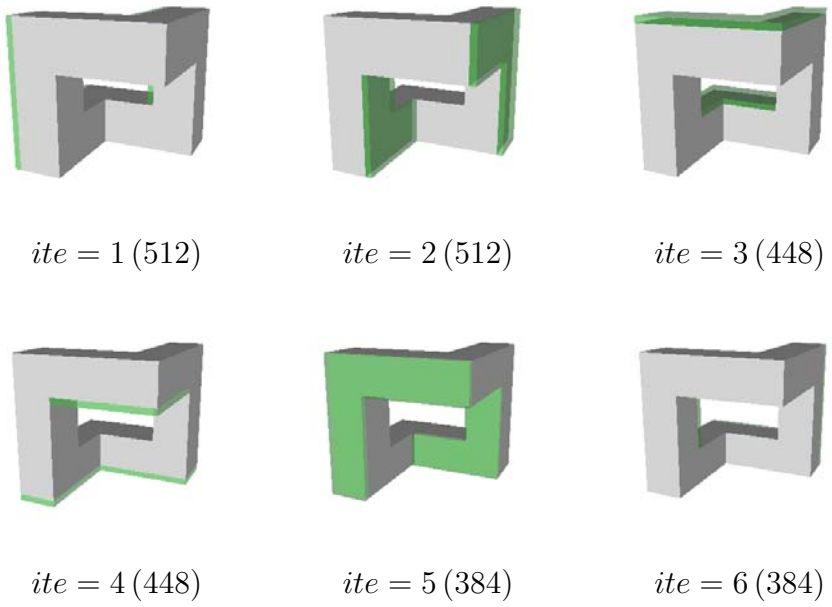


FIG. B.9 – *Détails des* 6 *premières sous-itérations de l'algorithme* GOBE *sur l'objet* $PaKu1$ *(*REPR6*).*

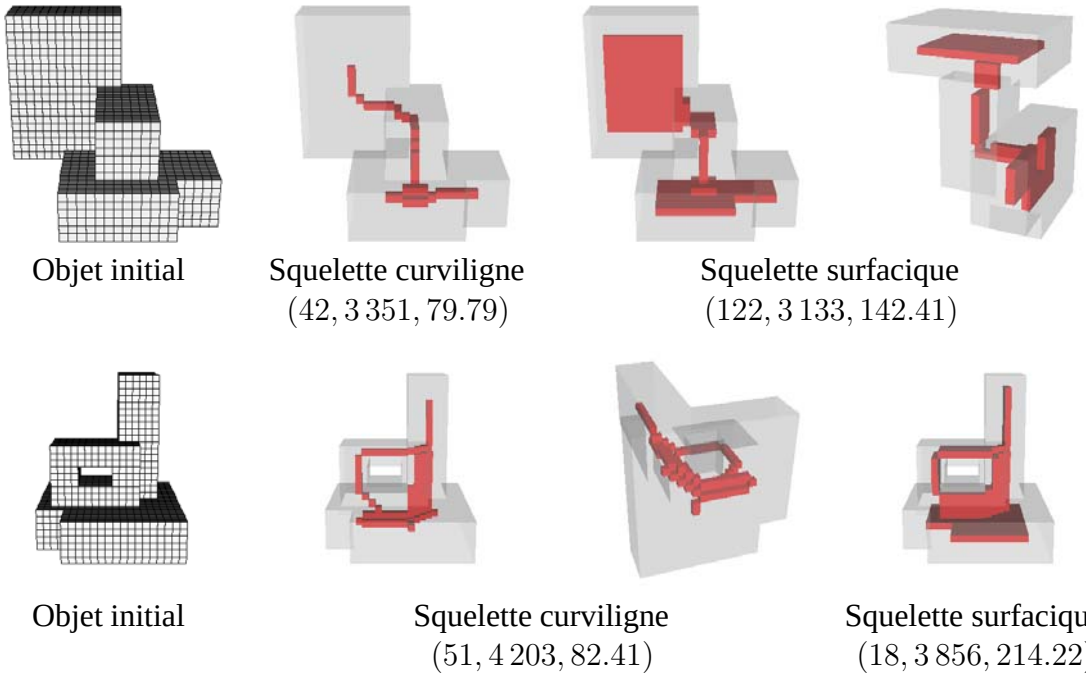


FIG. B.10 – *Autres objets, squelettisations curviligne et surfacique avec l'algorithme* GOBE *(*REPR7*).*

B.2.3 Algorithme de Palágyi et Kuba - 12 sous-itérations (PAKU12_OLD) [PK97a] [PK97b]

B.2.3.1 Description

Cet algorithme, de type sous-itérations directionnelles, permet l'obtention de squelettes curvilignes . Il utilise 12 sous-itérations, correspondant aux 12 directions suivantes : $Haut - Sud$, $Nord - Est$, $Bas - Ouest$, $Haut - Ouest$, $Sud - Est$, $Bas - Nord$, $Haut - Nord$, $Sud - Ouest$, $Bas - Est$, $Haut - Est$, $Nord - Ouest$, $Bas - Sud$. Le principe est le même que celui précédemment décrit à la section B.2.1. Le passage du masque Haut-Sud aux 11 autres masques se réalise par l'utilisation des isométries respectives suivantes : $iso34$, $iso47$, $iso40$, $iso33$, $iso16$, $iso23$, $iso41$, $iso39$, $iso32$, $iso42$, $iso7$ (voir l'annexe D).

Le schéma est donné par l'algorithme 32 et les templates sont décrites à la figure B.11.

Y : image initiale
Répéter
$Y = \mathcal{F}(Y, Haut - Sud)$;
$Y = \mathcal{F}(Y, Nord - Est)$;
$Y = \mathcal{F}(Y, Bas - Ouest)$;
$Y = \mathcal{F}(Y, Haut - Ouest)$;
$Y = \mathcal{F}(Y, Sud - Est)$;
$Y = \mathcal{F}(Y, Bas - Nord)$;
$Y = \mathcal{F}(Y, Haut - Nord)$;
$Y = \mathcal{F}(Y, Sud - Ouest)$;
$Y = \mathcal{F}(Y, Bas - Est)$;
$Y = \mathcal{F}(Y, Haut - Est)$;
$Y = \mathcal{F}(Y, Nord - Ouest)$;
$Y = \mathcal{F}(Y, Bas - Sud)$;
Jusqu'à ce qu'il n'y ait plus de suppression durant 12 sous-itérations successives.

Algorithme 32: *Algorithme de Palágyi et Kuba en 12 sous-itérations (*PAKU12_OLD*).*

B.2.3.2 Commentaires et résultats

Les directions utilisées dans l'algorithme correspondent aux 12 couples de 6-voisins non opposés. Notons que la séquence choisie pour l'algorithme est celle donnée dans [PK97b] et les templates sont celles de [PK97a].

Il y a 1 084 669 configurations qui vérifient les templates de la figure B.11.

Nous constatons que l'aspect géométrique des squelettes obtenus par cet algorithme est différent de celui des squelettes obtenus par l'algorithme PAKU6 (voir Fig. B.4). Les 12 premières sous-itérations de suppression sur l'objet $PaKu5$ sont détaillées à la figure B.13, en notant pour chacune d'elles le nombre de points supprimés.

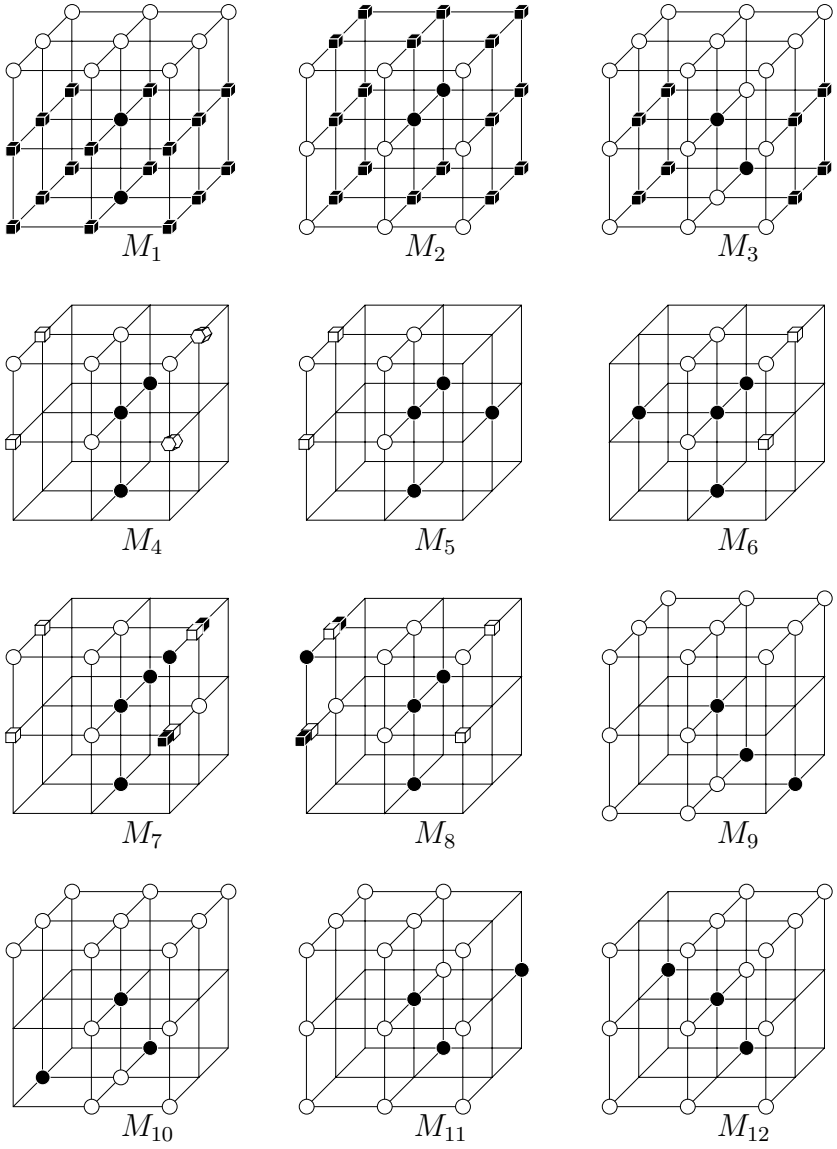


FIG. B.11 – *Masque $Haut - Sud$: templates utilisées pour l'algorithme de squelettisation en 12 sous-itérations de Palágyi et Kuba* PAKU12_OLD. *Au moins un point représenté par un cube noir doit appartenir à l'objet dans les masques M_1 à M_3. Au moins un point représenté par un cube blanc ou un hexagone blanc en perspective doit appartenir au background dans les masques M_4 à M_8. Deux points marqués par les cubes bicolores doivent être différents l'un de l'autre dans les masques M_7 et M_8.*

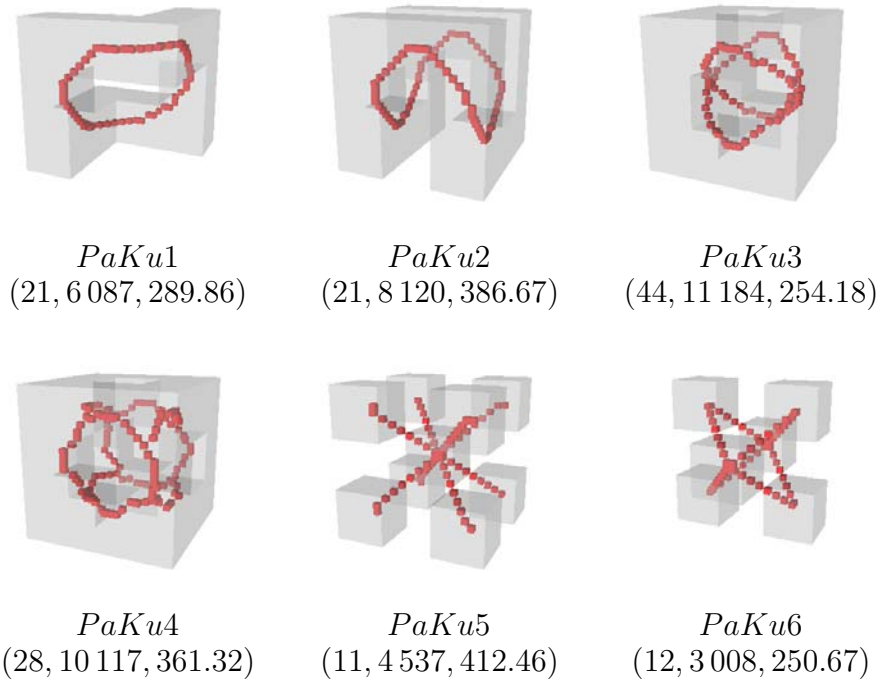


FIG. B.12 – *Squelettisations avec l'algorithme* PAKU12_OLD *(*REPR7*).*

En le comparant sur ces quelques objets synthétiques (comportant de nombreux axes de symétrie), nous observons que l'algorithme PAKU12_OLD nécessite moins de sous-itérations que PAKU6 pour obtenir un squelette sauf pour l'objet $Paku4$, voir la figure B.4. Pour l'objet $PaKu4$, l'algorithme PAKU12_OLD nécessite plus de sous-itérations que l'algorithme PAKU6 ; il supprime également plus de points. Cela n'implique pas que le squelette obtenu est moins épais, nous pouvons simplement constater que le centrage des squelettes n'est pas le même. D'un point de vue pratique, il ne faut pas oublier que cet algorithme nécessite 12 sous-itérations consécutives sans suppression afin d'obtenir la stabilité (contrairement aux 6 sous-itérations avec l'algorithme PAKU6), et dans les indications sous les figures, seule la sernière sous-itération effective de suppression est indiquée. Pour obtenir la dernière sous-itération lors de laquelle se termine l'algorithme, il faut alors ajouter 12 (resp. 6) au numéro de la dernière sous-itération effective de suppression pour l'algorithme PAKU12_OLD (resp. PAKU6), et on observe que PAKU6 s'arrête avant PAKU12_OLD pour les objets $PaKu1$, $PaKu2$ et $PaKu4$. En fait, une petite modification, par rapport aux schémas des algorithmes donnés par les auteurs, a été apportée afin que les algorithmes présentés ici s'arrêtent effectivement lors de l'obtention de la stabilité, *i.e.* lors des 6 ou 12 premières sous-itérations successives sans suppression.

Il s'agit du premier algorithme en 12 sous-itérations, proposé par Palágyi et Kuba. Récemment, ces mêmes auteurs ont proposé un autre algorithme en 12 sous-itérations, permettant l'obtention de squelettes curvilignes ou de squelettes surfaciques. C'est celui que nous détaillons dans la section B.2.4.

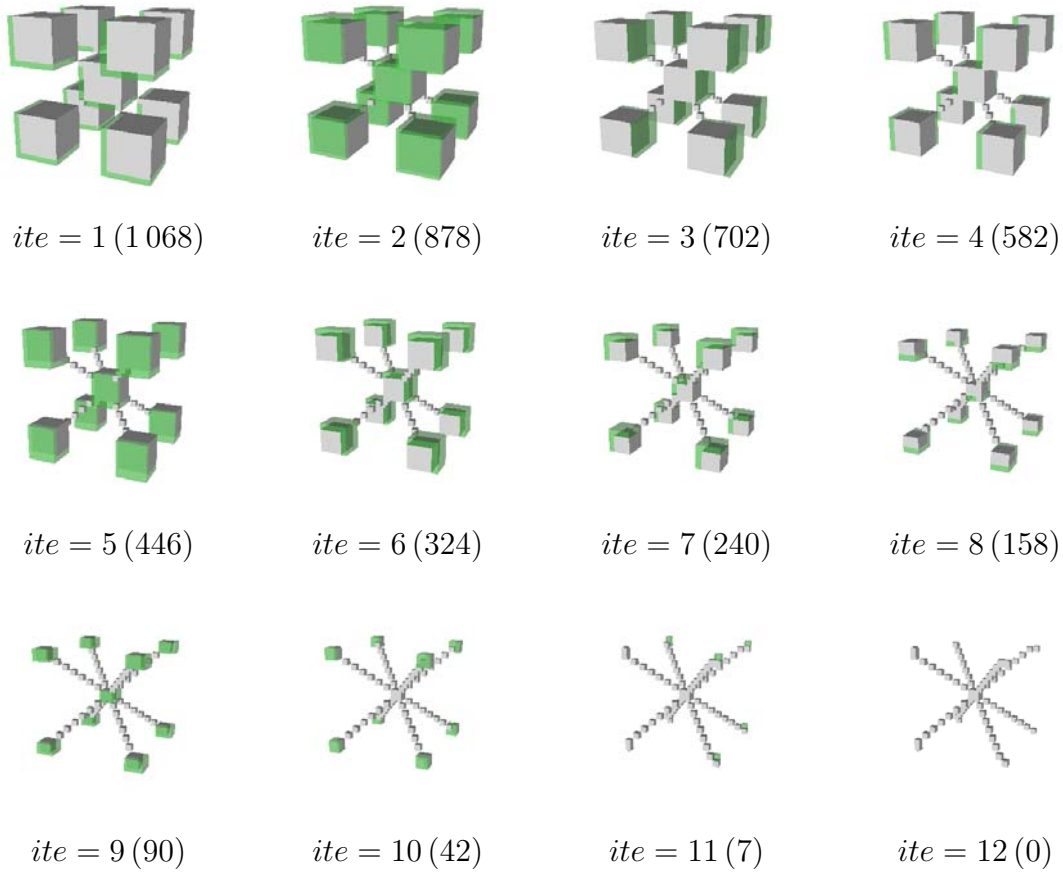


FIG. B.13 – *Détails des* 12 *premières sous-itérations de l'algorithme* PAKU12_OLD *sur l'objet* $PaKu5$ *(*REPR6*).*

B.2.4 Algorithme de Palagyi et Kuba - 12 sous-itérations (PAKU12) [PK99]

B.2.4.1 Description

Cet algorithme permet l'obtention de squelettes curvilignes ou surfaciques. Le schéma est le même que celui décrit par l'algorithme 32, en prenant les templates de la figure B.14 (données selon la direction $Haut\_Sud$). Les 12 directions utilisées sont : $Haut - Sud$, $Nord - Est$, $Bas-Ouest$, $Sud-Est$, $Haut-Ouest$, $Bas-Nord$, $Sud-Ouest$, $Haut-Nord$, $Bas-Est$, $Nord-Ouest$, $Haut-Est$, $Bas-Sud$; les isométries utilisées sont : $iso0$, $iso34$, $iso47$, $iso33$, $iso40$, $iso16$, $iso41$, $iso23$, $iso39$, $iso42$, $iso32$, $iso7$ (voir l'annexe D).

B.2.4.2 Commentaires et résultats

La caractérisation utilisée pour un point terminal de surface est la suivante : un point $p \in X$ est *terminal de surface* si $N_6(p)$ contient au moins une paire de points opposés (par leur direction) et appartenant à $\overline{X}$. Il y a $1\,379\,581$ (resp. $1\,155\,072$) configurations correspondant à des points simples x non terminaux de courbe (resp. de surface) dans $N^*_{26}(x)$. Sur quelques objets de référence, nous donnons les squelettes curvilignes (Fig. B.16) ou surfaciques (Fig. B.17). Les résultats sont du même ordre qu'avec l'algorithme PAKU12_OLD (dans le cadre de la squelettisation curviligne). Le résultat des 12 premières sous-itérations de la squelettisation surfacique sur l'objet $PaKu1$ est représenté figure B.15 : nous pouvons voir apparaître progressivement les bouts de surface qui seront préservés dans le squelette.

Remarque : Nous avons vérifié avec les points P-simples que cet algorithme préserve la topologie (cf. section 6.6.2.1). Nous avons proposé un algorithme basé sur les points P^x-simples, supprimant au moins les points retirés par cet algorithme tout en préservant les mêmes points extrémités pour une version curviligne (cf. section 6.6.3) ou une version surfacique (cf. section 6.6.5).

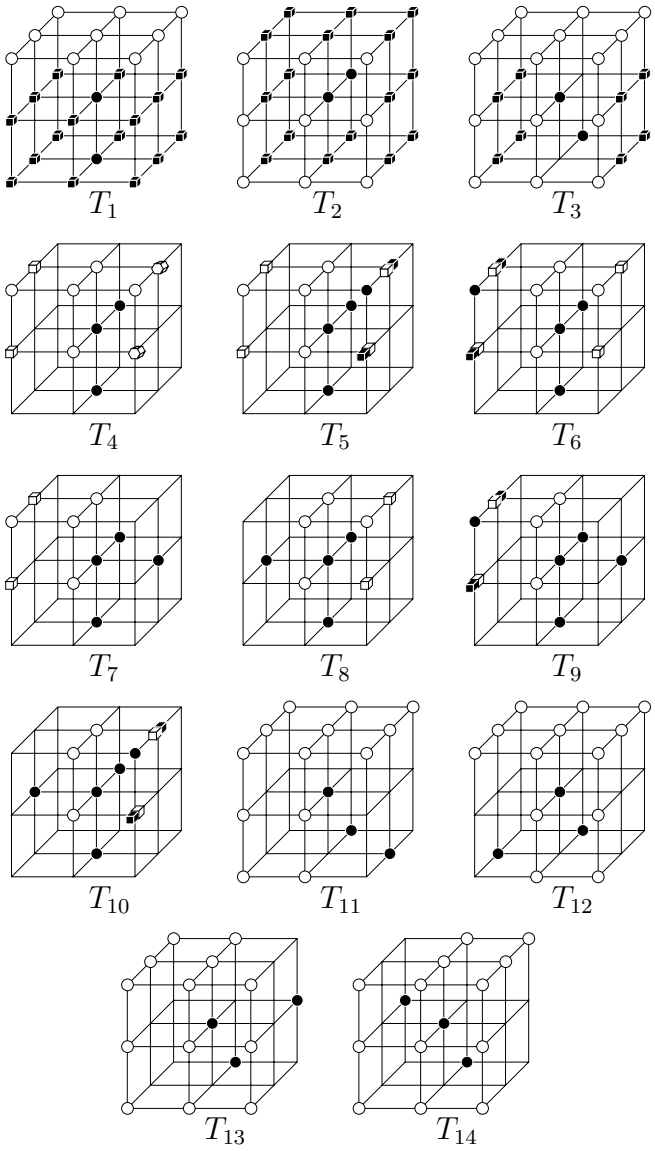


FIG. B.14 – *Masque $Haut - Sud$: templates utilisées pour l'algorithme* PAKU12*. Au moins un point représenté par un cube noir doit appartenir à l'objet dans les masques T_1 à T_3. Au moins un point représenté par un cube blanc et un hexagone blanc en perspective doivent appartenir au background dans les masques T_4 à T_8. Deux points marqués par les cubes bicolores doivent être différents l'un de l'autre dans les masques T_5, T_6, T_9, T_{10}.*

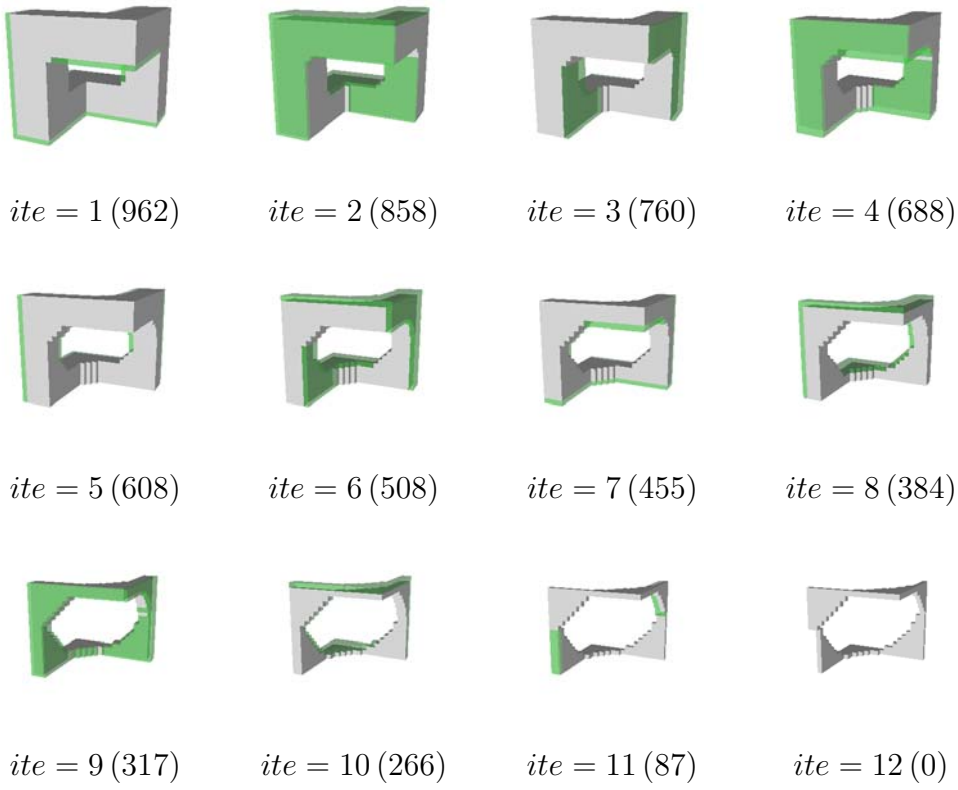
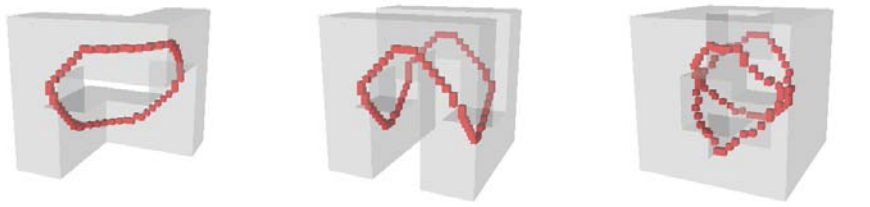
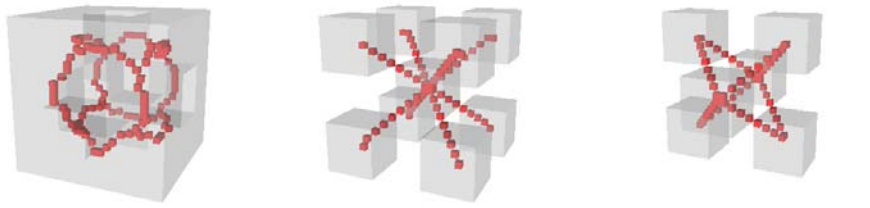


FIG. B.15 – *Détails des* 12 *premières sous-itérations de squelettisation surfacique de l'algorithme* PAKU12 *sur l'objet* $PaKu1$ *(*REPR6*).*

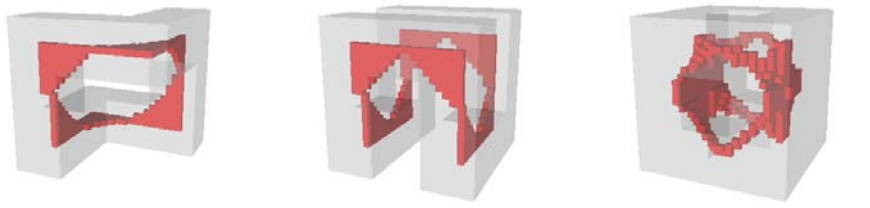


$(21, 6\,088, 289.91)$ $(21, 8\,120, 386.67)$ $(44, 11\,182, 254.14)$

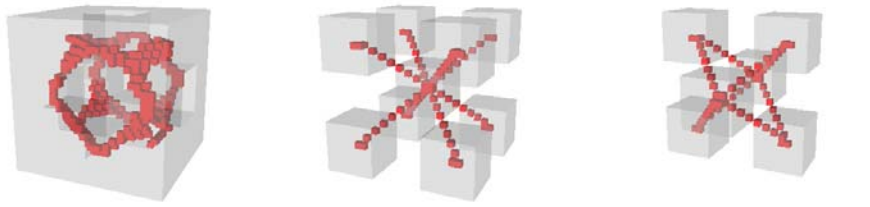


$(29, 10\,117, 348.86)$ $(11, 4\,537, 412.46)$ $(12, 3\,008, 250.67)$

FIG. B.16 – *Squelettisations curvilignes avec l'algorithme* PAKU12 *(*REPR7*).*



$(11, 5\,893, 535.73)$ $(11, 7\,850, 713.64)$ $(23, 10\,792, 469.22)$



$(21, 9\,951, 473.86)$ $(10, 4\,530, 453)$ $(10, 3\,000, 300)$

FIG. B.17 – *Squelettisations surfaciques avec l'algorithme* PAKU12 *(*REPR7*).*

B.3 Algorithmes de squelettisation de type sous-mailles

Nous examinons d'abord un algorithme utilisant 8 sous-mailles, puis un algorithme de squelettisation hybride mélangeant à la fois la stratégie directionnelle et l'approche sous-mailles.

B.3.1 Algorithme de Bertrand et Aktouf (BEAK) [BA94]

C'est un algorithme utilisant la technique des sous-mailles et permettant l'obtention de squelettes surfaciques ou curvilignes.

B.3.1.1 Description

L'espace $\mathcal{Z}^3$ est divisé en 8 sous-mailles (selon la parité de $x+y$, $y+z$ et $x+y+z$, voir Fig. B.18 (a)). Un ordre d'activation des sous-mailles est fixé et chaque point de la sous-maille activée est testé. Si ce point est un point de l'objet non étiqueté, on regarde s'il est point terminal (au sens du squelette désiré). Si c'est le cas, on l'étiquette, sinon, s'il est simple, il sera supprimé en parallèle avec les autres points détectés simples et non terminaux lors de cette sous-itération (la suppression a lieu en fin de cette sous-itération). Le schéma de squelettisation est décrit par l'algorithme 33.

Répéter
  Pour chaque sous-maille S_i **faire**
    En parallèle **Pour** tout $x \in X$ et appartenant à S_i et non déjà étiqueté **faire**
      Si x est terminal alors étiqueter x avec la valeur "TERMINAL",
      Sinon si x est simple alors étiqueter x avec la valeur "À SUPPRIMER"
    En parallèle **Pour** tout x étiqueté "À SUPPRIMER" **faire**
      Supprimer x
Jusqu'à ce qu'il n'y ait plus de suppression durant 8 sous-itérations successives.

Algorithme 33: *Algorithme de Bertrand et Aktouf (*BEAK*).*

Les points terminaux sont ici caractérisés par les nombres topologiques (voir la section 3.7 et le tableau 3.7). Soit (m,n) la connexité de l'image. Considérons une courbe simple X. Les points simples sont les extrémités de courbe ; les autres points x sont tels qu'ils ont deux composantes de l'objet dans leur 26-voisinage, *i.e.* $T_m(x,X)=2$. En marquant les points tels que $T_m(x,X)=2$ et en empêchant de les retirer par la suite (*i.e.* en les marquant comme "TERMINAL") ; la courbe va être partiellement préservée ; seules les extrémités vont être "grignotées". De façon plus générale, afin de tenir compte des intersections de courbe, nous considérons comme terminal pour un squelette curviligne les points x isthmes 1D, *i.e.* tels que $T_n(x,X)\geq 2$.

Considérons un morceau de plan. Les points simples sont les points du contour, les autres points x sont tels qu'ils ont deux composantes du complémentaire dans leur 26-voisinage, soit $T_n(x,\overline{X})=2$.

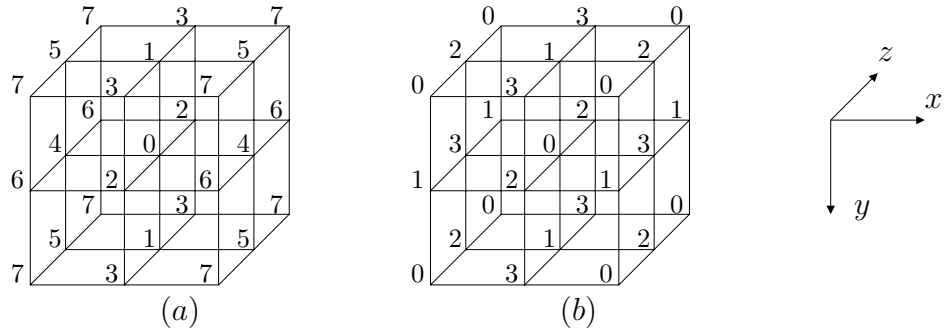


FIG. B.18 – *Mailles pour l'algorithme* BEAK *:* (a) 8 *sous-mailles (utilisée dans l'algorithme),* (b) 4 *sous-mailles (problème, cf. discussion).*

De plus, les courbes doivent être préservées dans un squelette surfacique. De même, afin de tenir compte des intersections (de courbe ou de surface), nous considérons comme terminal pour un squelette surfacique les points x isthmes, *i.e.* tels que $T_m(x, X) \geq 2$ et $T_n(x, \overline{X}) \geq 2$. De la même manière, les contours des morceaux de plan peuvent être grignotés (voir Fig. B.19 et les 10 premières sous-itérations à la figure B.20).

B.3.1.2 Commentaires et résultats

Par exemple, pour l'objet du haut de la figure B.19, cet algorithme exige moins d'itérations pour l'obtention d'un squelette curviligne qu'avec l'algorithme GOBE, et plus d'itérations pour un squelette surfacique qu'avec l'algorithme GOBE (voir l'objet du bas de la figure B.10).

C'est une approche différente de celles précédemment exposées en ce qui concerne la préservation de points extrémités. L'approche habituelle consiste à marquer comme extrémités des points simples vérifiant une certaine condition (généralement empirique). Ici, nous interdisons certains points simples d'être supprimés lors des itérations ultérieures de suppression.

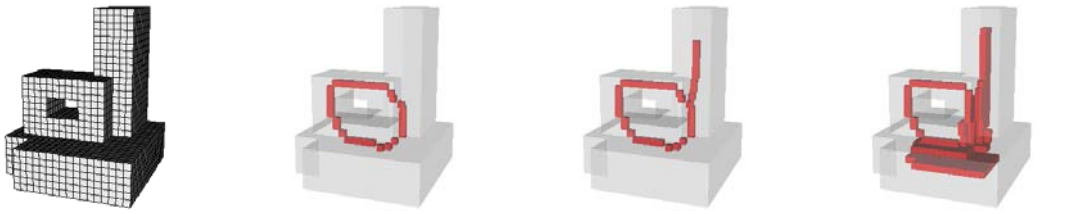
Lors d'une sous-itération de suppression, on ne considère que les points de la sous-maille activée, et c'est à ce moment que l'on étiquette les points terminaux (non encore étiquetés) de la même sous-maille considérée.

Ce problème de "grignotage" de contour des courbes ou des surfaces a pour conséquence la non-idempotence des squelettes, *c.-à-d.* un squelette d'un objet déjà squelettisé peut ne pas donner ce même objet.

Notons qu'une version n'utilisant uniquement que 4 sous-mailles (selon la parité de $x + y$ et $y + z$, voir Fig. B.18 (b)) ne permet pas de préserver des composantes connexes composées de deux points diamétralement opposés, si on adapte la même définition de points terminaux. En effet, ceux-là sont simples et ne sont pas considérés comme points terminaux, ni pour le squelette curviligne, ni pour le squelette surfacique et ils seront supprimés tous les deux lors d'une même sous-itération car ils appartiennent à une même sous-maille ; la topologie n'est alors pas préservée.

En conclusion, l'utilisation des nombres topologiques semble donner de bons résultats pour l'algorithme surfacique, malgré le grignotage de certains contours.

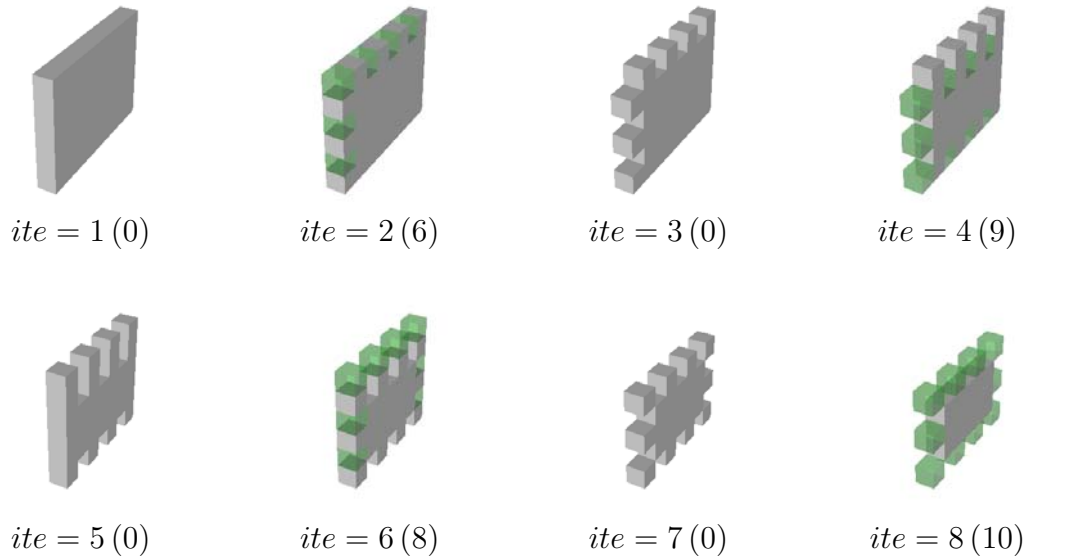
Remarque : Voir [Akt97] pour une discussion d'une part sur le fait de vérifier qu'un point est terminal, seulement à condition qu'il appartienne à la maille active ; et d'autre part sur une variante de l'algorithme selon les connexités utilisées afin d'éviter l'apparition de barbules.



Objet initial | Noyau homotopique $(60, 4\,266, 71.1)$ | Squelette curviligne $(37, 4\,257, 115.05)$ | Squelette surfacique $(23, 4\,046, 175.91)$

Objet initial | Noyau homotopique $(16, 47, 2.94)$ | Squelette curviligne $(16, 47, 2.94)$ | Squelette surfacique $(8, 33, 4.13)$

FIG. B.19 – *Amincissement et squelettisation curvilignes et surfaciques avec l'algorithme* BEAK *(*REPR7*).*



$ite = 1\,(0)$ | $ite = 2\,(6)$ | $ite = 3\,(0)$ | $ite = 4\,(9)$

$ite = 5\,(0)$ | $ite = 6\,(8)$ | $ite = 7\,(0)$ | $ite = 8\,(10)$

FIG. B.20 – *Détails des* 8 *premières sous-itérations de l'algorithme* BEAK *de squelettisation surfacique sur l'objet plan (*REPR6*).*

B.3.2 Algorithme hybride de Palágyi et Kuba (PAKUH) [PK]

B.3.2.1 Description

Palágyi et Kuba ont proposé un algorithme hybride utilisant à la fois une stratégie directionnelle et la technique des sous-mailles. Plus précisément, l'algorithme est composé de huit sous-itérations de suppressions, une pour chacune des huit directions considérées, suivies de deux sous-itérations de suppressions se déroulant alternativement dans l'une des deux sous-mailles choisies.

Notons $\mathcal{F}(X, Dir)$ l'opération réalisant la suppression en parallèle des points d'un objet X et vérifiant au moins l'une des templates de direction Dir. Nous notons $\mathcal{G}(X, Sous-maille\_z)$ l'opération réalisant la suppression en parallèle des points d'un objet X et vérifiant au moins l'une des templates des sous-mailles et telle que ces points appartiennent à la sous-maille $Sous-maille\_z$. Le schéma est donné par l'algorithme 34.

Y : image initiale
Répéter
/*Partie directionnelle*/
$Y = \mathcal{F}(Y, Haut-Sud-Ouest)$;
$Y = \mathcal{F}(Y, Bas-Nord-Est)$;
$Y = \mathcal{F}(Y, Haut-Sud-Est)$;
$Y = \mathcal{F}(Y, Bas-Nord-Ouest)$;
$Y = \mathcal{F}(Y, Haut-Nord-Est)$;
$Y = \mathcal{F}(Y, Bas-Sud-Ouest)$;
$Y = \mathcal{F}(Y, Haut-Nord-Ouest)$;
$Y = \mathcal{F}(Y, Bas-Sud-Est)$;
/*Partie sous-mailles*/
$Y = \mathcal{G}(Y, Sous-maille\_A)$;
$Y = \mathcal{G}(Y, Sous-maille\_B)$;
Jusqu'à ce qu'il n'y ait plus de suppression durant 10 sous-itérations successives.

Algorithme 34: *Algorithme hybride de Palágyi et Kuba en 8 sous-itérations directionnelles et 2 sous-itérations selon 2 sous-mailles (*PAKUH*).*

Les huit directions sont $Haut-Sud-Ouest$, $Bas-Nord-Est$, $Haut-Sud-Est$, $Bas-Nord-Ouest$, $Haut-Nord-Est$, $Bas-Sud-Ouest$, $Haut-Nord-Ouest$, $Bas-Sud-Est$ (Fig. B.21). Les templates pour la direction $Haut-Sud-Ouest$ sont représentées à la figure B.23. Les templates pour les autres directions sont obtenues respectivement par les isométries suivantes : $iso21$, $iso5$, $iso16$, $iso23$, $iso6$, $iso22$, $iso7$ (voir l'annexe D). On note que $6\,157$ configurations vérifient les templates pour un masque.

Les templates pour une sous-maille, à rotations et symétries près, sont représentées à la figure B.24. Les points de la sous-maille $Sous\_maille\_A$ (resp. $Sous\_maille\_B$) sont les points dont la somme des coordonnées $x+y+z$ est impaire (resp. paire) (voir Fig. B.22).

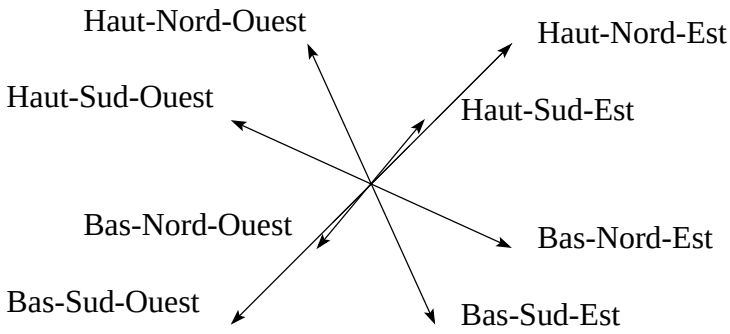


FIG. B.21 – *Orientation utilisée dans l'algorithme* PAKUH.

B.3.2.2 Commentaires et résultats

Il y a $2\,478\,086$ configurations qui vérifient les templates directionnelles avant isométries et $18\,735\,154$ configurations qui vérifient les templates avec isométries.

À la figure B.25 sont représentés les squelettes obtenus par cet algorithme sur 6 objets synthétiques de comparaison des algorithmes PAKU6, PAKU12_OLD, PAKU12 (voir Fig. B.4, B.12 et B.16). Notons la présence de barbules dans les squelettes des objets $PaKu3$ et $PaKu4$, barbules non présentes avec les trois autres algorithmes.

En moyenne, sur ces quelques objets, cet algorithme exige plus de sous-itérations avant d'obtenir un squelette qu'avec les trois algorithmes précédemment cités et retire un nombre similaire de points. Nous pouvons également constaté un aspect dentelé des squelettes, dû à l'utilisation des sous-mailles.

Les 10 premières sous-itérations sur deux petits objets sont représentées aux figures B.26 et B.27. Remarquons sur ces exemples, que la plupart des points sont retirés par les deux itérations se déroulant dans les sous-mailles ($ite = 9$ et $ite = 10$). Notons également que le squelette obtenu diffère d'un objet initial et de son translaté (même problème qu'en 2D, avec les sous-mailles).

Par exemple, la figure B.26 est composée de deux objets dont l'un est le translaté de l'autre, de telle façon que les points similaires de ces deux objets n'appartiennent pas à la même sous-maille. Nous pouvons constater l'aspect rectiligne du squelette de l'objet de gauche et l'aspect crénelé du squelette de l'objet de droite.

Nous pouvons remarquer que les templates $T - D_2$ et $T - D_3$ peuvent être obtenues à partir de la template $T - D_1$ par isométrie. Il en est de même pour les templates $T - D_5$ et $T - D_6$, obtenues par isométrie de $T - D_4$, cela dans le but de préserver des symétries.

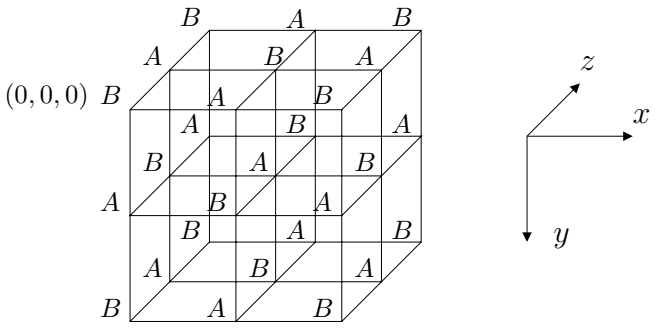


FIG. B.22 – *Maille utilisée pour l'algorithme hybride* PAKUH.

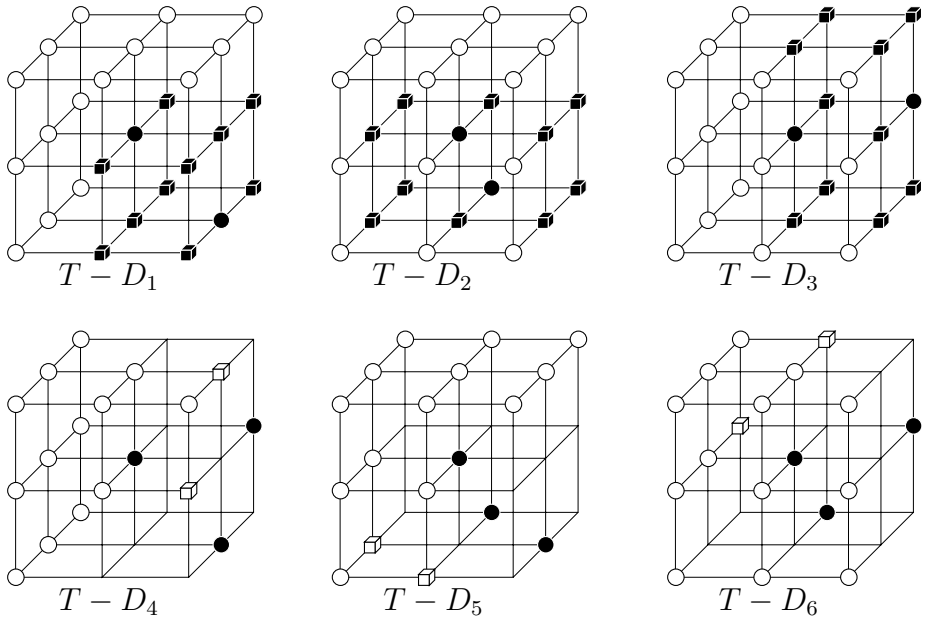


FIG. B.23 – *Masque* $Haut - Sud - Ouest$ *: templates directionnelles utilisées pour l'algorithme* PAKUH. *Au moins un point représenté par un cube noir doit appartenir à l'objet dans les masques* $T - D_1$ *à* $T - D_3$. *Au moins un point représenté par un cube blanc doit appartenir au background dans les masques* $T - D_4$ *à* $T - D_6$.

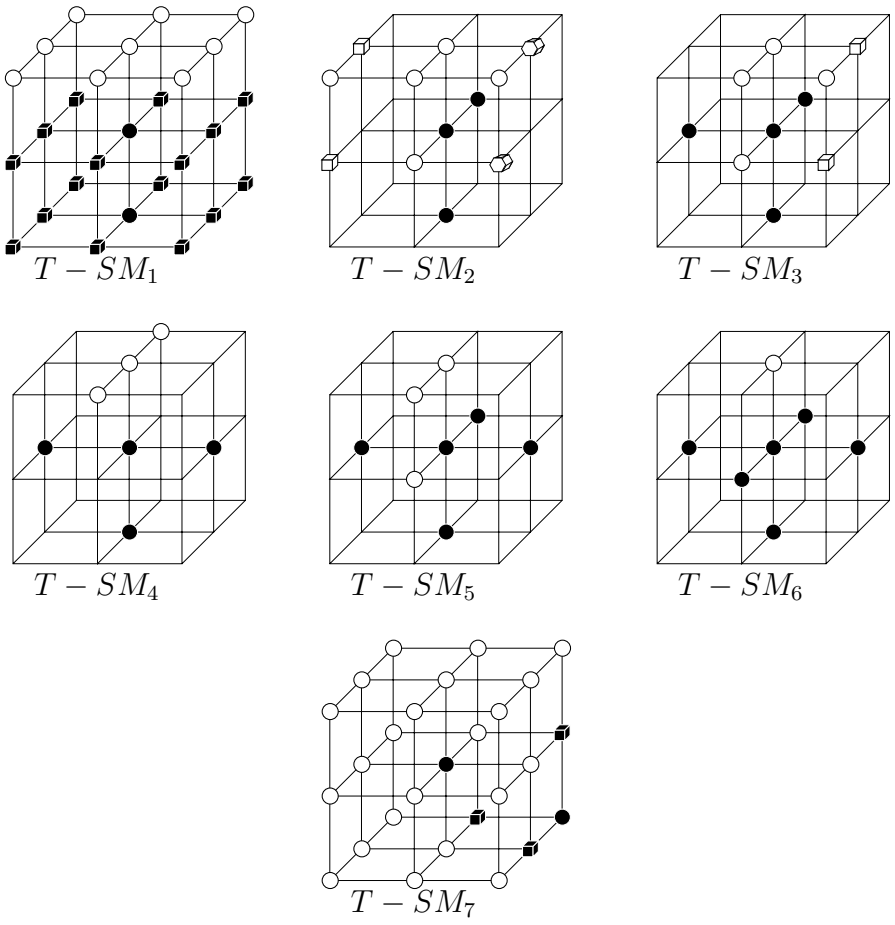
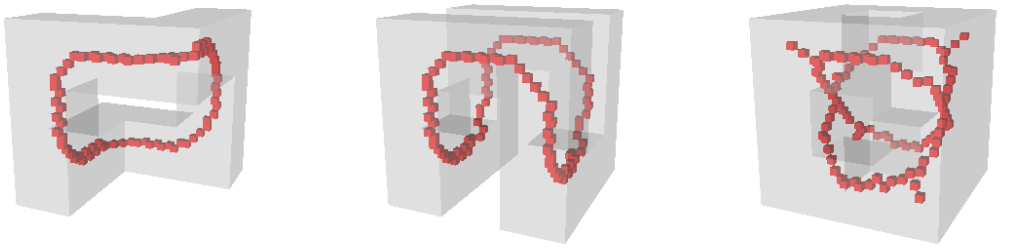
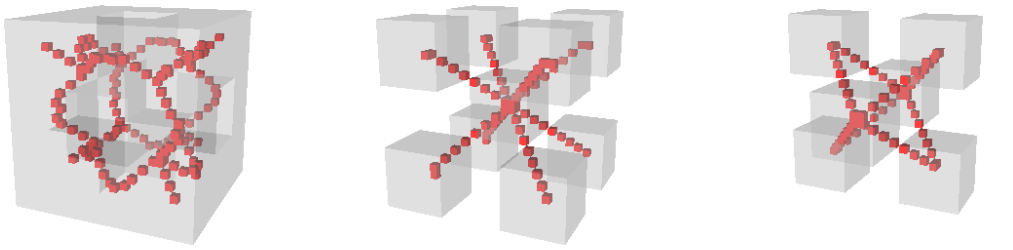


FIG. B.24 – *Templates pour une sous-maille, utilisées pour l'algorithme* PAKUH. *Au moins un point représenté par un cube noir doit appartenir à l'objet dans les masques* $T - SM_1$ *et* $T - SM_7$. *Au moins un point représenté par un cube blanc ou un hexagone blanc en perspective doit appartenir au fond dans les masques* $T - SM_2$ *et* $T - SM_3$.

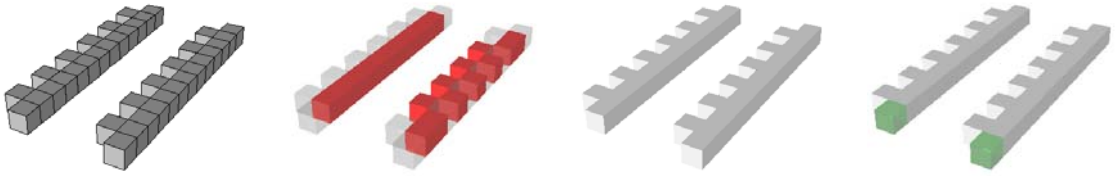


$(39, 6\,071, 155.67)$ $(39, 8\,096, 207.59)$ $(67, 11\,157, 166.52)$

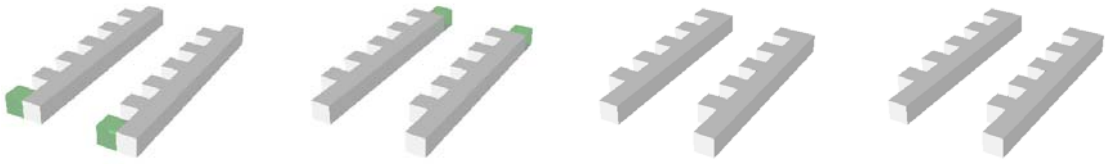


$(56, 10\,104, 180.43)$ $(29, 4\,536, 156.41)$ $(29, 3\,008, 103.72)$

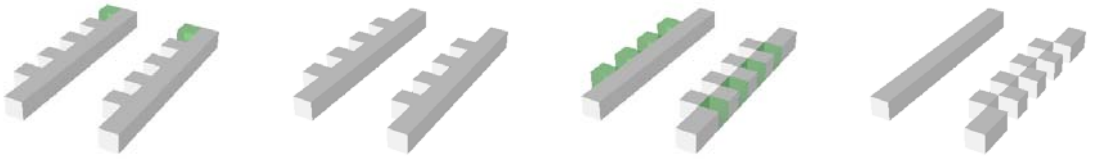
FIG. B.25 – *Squelettisations avec l'algorithme* PAKUH *(*REPR7*).*



Objet initial | Squelette curviligne $(9, 16, 1.78)$ | $ite = 1\,(0)$ | $ite = 2\,(2)$



$ite = 3\,(2)$ | $ite = 4\,(2)$ | $ite = 5\,(0)$ | $ite = 6\,(0)$



$ite = 7\,(2)$ | $ite = 8\,(0)$ | $ite = 9\,(8)$ | $ite = 10\,(0)$

FIG. B.26 – *Détails des* 10 *premières sous-itérations de l'algorithme* PAKUH *sur l'objet* $peigne$ *(*REPR6*).*

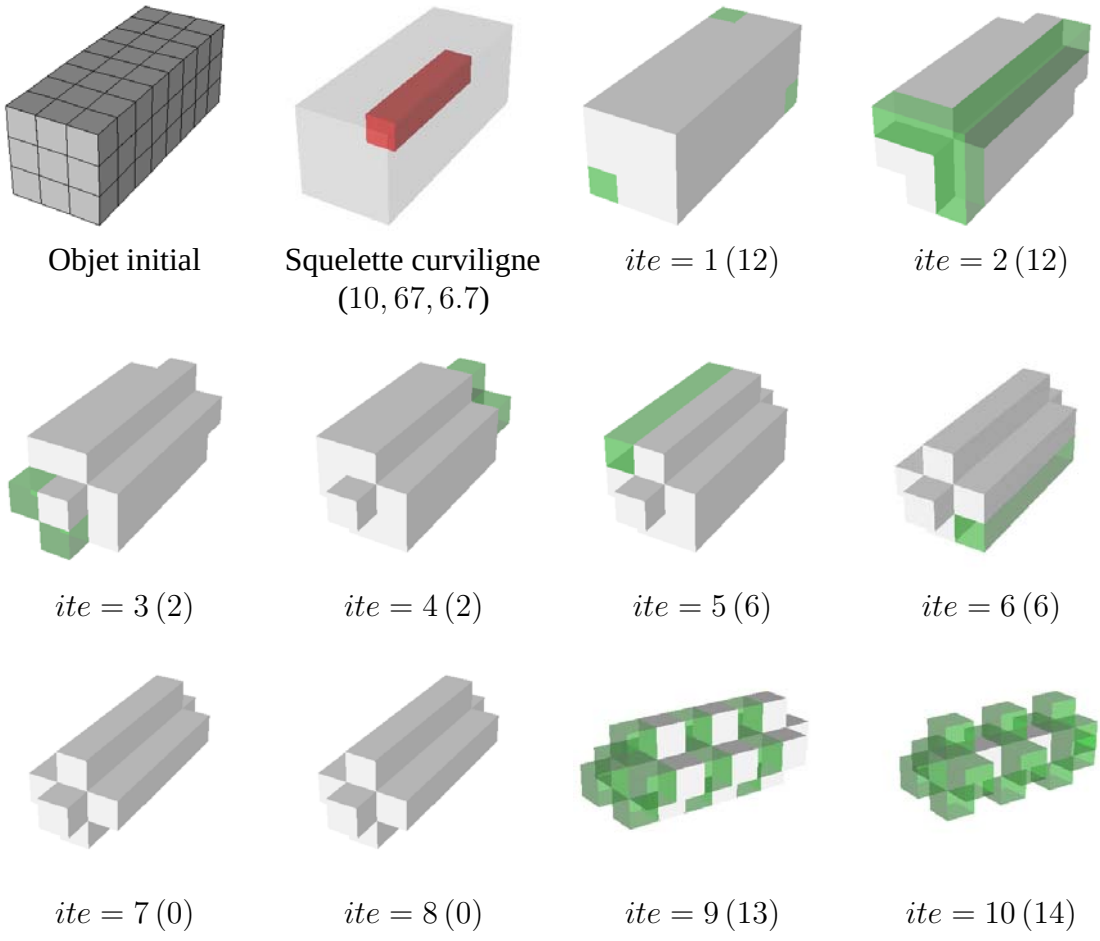


FIG. B.27 – *Détails des* 10 *premières sous-itérations de l'algorithme* PAKUH *sur l'objet* $ruban\,épais$ *(*REPR6*).*

B.4 Algorithmes fortement parallèles

Dans cette section, nous décrivons trois algorithmes fortement parallèles.

B.4.1 Algorithme fortement parallèle de Ma (MA) [Ma95]

B.4.1.1 Description

Il s'agit d'un algorithme fortement parallèle permettant l'obtention de squelettes surfaciques utilisant des templates et un support de 30 points (le point central étant compté).

Le schéma de squelettisation est décrit par l'algorithme 35. Un point sera éliminé s'il vérifie au moins l'une des templates T_1, T_2 ou T_3, ou l'une de ses isométries (Fig. B.28), s'il vérifie les conditions directionnelles et s'il n'est pas extrémité, et s'il ne vérifie pas la règle R_1 (voir l'algorithme 35).

Conditions directionnelles :

– Si p est un point de bord $Nord$, alors $Sud(Sud(p))$ doit être un point de l'objet,
– si p est un point de bord Est, alors $Ouest(Ouest(p))$ doit être un point de l'objet,
– si p est un point de bord $Haut$, alors $Bas(Bas(p))$ doit être un point de l'objet.

Règle R_1 :

Supposons que les 4 coins d'un carré unité sont candidats à la suppression.
Alors le coin dont la somme des coordonnées est la plus petite est préservé
si et seulement si il est non simple après la suppression des trois autres coins.

Définition utilisée pour les points extrémités :

Le point p est un point extrémité si p a exactement un voisin dans son 26-voisinage
ou dans n'importe quel plan $1 \times 3 \times 3$.

Répéter
Supprimer en parallèle les points de l'objet non extrémités
vérifiant au moins l'une des templates T_1, T_2, ou T_3 ou l'une de ses isométries,
les conditions directionnelles, et non préservés par la règle R_1
Jusqu'à ce qu'il n'y ait plus de suppression durant une itération.

Algorithme 35: *Algorithme de Ma (*MA*).*

B.4.1.2 Commentaires et résultats

Considérons un plan (yz) d'épaisseur 1 (voir l'orientation à la figure B.1 (a)). Les points sur le contour de ce plan sont préservés car ils sont soit de type Sud ou $Nord$ et n'ont qu'un voisin dans le plan $1 \times 3 \times 3$ selon (xz), soit de type $Haut$ ou Bas et n'ont qu'un voisin dans le plan

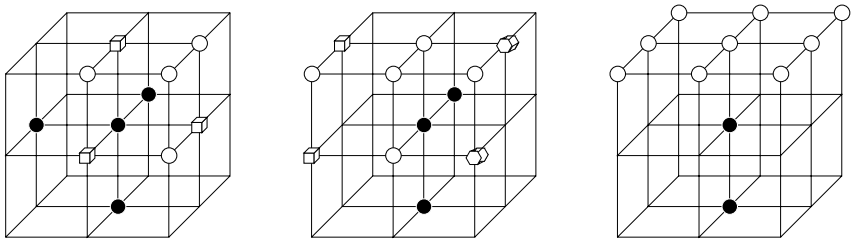


FIG. B.28 – *Les templates utilisées par l'algorithme* MA *(à isométries près). Au moins un point représenté par un cube blanc ou un hexagone blanc en perspective doivent appartenir au complémentaire (dans les templates T_1 ou T_2).*

$1 \times 3 \times 3$ selon (xy). Notons que les 4 coins du plan sont de ces 2 types et n'ont qu'un point dans les plans $1 \times 3 \times 3$ selon (xy) ou (xz). Remarquons que les points éliminés par les templates sont des points 26-simples. Les autres points du plan ne peuvent être éliminés par les templates, car ils ne sont pas simples. Le plan est ainsi totalement préservé. Dans l'algorithme BEAK, par exemple, le contour est grignoté (cf. section B.3.1.2).

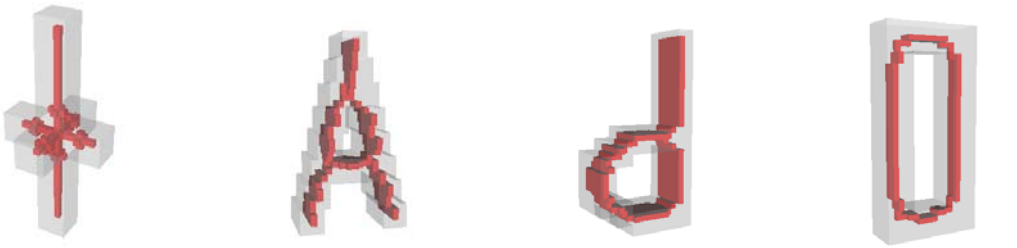
La première partie de la définition d'un point extrémité permet la préservation des courbes dans le squelette, par la préservation des points extrémités de courbe (points avec un seul voisin dans leur 26-voisinage).

Trois templates sont utilisées : le nombre de configurations vérifiant ces templates sont respectivement $458\,752$, $294\,912$, $65\,536$ pour les templates T_1, T_2, et T_3 avant isométries, et $3\,348\,992$, $3\,124\,224$, $387\,200$, après isométries.

Nous pouvons constater sur l'objet qui est la jonction de deux plans $1 \times 5 \times 5$ selon (xz) et (yz), que les deux points extrêmes de la jonction sont détruits, car ils ne sont pas extrémités ; dans le plan $1 \times 3 \times 3$ selon (xy) (resp. (yz), (xz)) se trouvent 4 (resp. 5, 5) points ; ces points vérifient les templates T_1 et T_3 et ils n'appartiennent pas à un carré dont les 4 coins ont à être supprimés (ces points ne sont donc pas concernés par la règle R_1). Rappelons que la jonction de ce même objet était partiellement détruite dans le cas de l'algorithme de Gong et Bertrand [GB90] (cf. la section B.2.2) ou de l'algorithme de Tsao et Fu [TF81] (cf. [Pud98] et la section 9.5.3).

À la figure B.29, sont représentés les squelettes surfaciques obtenus par cet algorithme sur 4 objets synthétiques.

Remarque : À la section 6.7.2, nous avons montré que cet algorithme ne préservait pas la topologie.



$(5, 1468, 293.6)$ $(5, 2215, 443)$ $(3, 1665, 555)$ $(3, 2988, 996)$

FIG. B.29 – *Squelettisation avec l'algorithme* MA *(*REPR7*).*

B.4.2 Algorithme fortement parallèle de Ma et Sonka (MASO) [MS96]

B.4.2.1 Description

Il s'agit d'un algorithme fortement parallèle permettant l'obtention de squelettes curvilignes. Il utilise des templates et un support de 50 points (le point central étant compté).
Les squelettes obtenus peuvent ne pas être minces. Les auteurs proposent alors une phase préalable de modification de l'objet initial afin de le transformer en *objet bien composé*. Ils proposent également d'utiliser une dilatation morphologique pour améliorer l'aspect du squelette.

Le schéma de squelettisation est décrit par l'algorithme 36. Un point sera éliminé s'il vérifie au moins une template des 4 classes A, B, C, ou D (Fig. B.32) et s'il n'est pas extrémité (voir la définition ci-après) ; s'il vérifie une template de la classe D, il faut de plus tester sa simplicité. D'après l'article de référence, il est dit que les templates sont inspirées du travail de Holt et al. (référence [HSCP87], algorithme déjà vu dans la section A.5.1, page 266).

Les définitions suivantes sont adoptées afin de caractériser un point extrémité : p est un *point extrémité de ligne* si p est 26-adjacent à exactement un point de l'objet. Le point p est un *point presque extrémité d'une ligne* si p est 26-adjacent à exactement deux points de l'objet qui sont :

- soit [$Sud(p)$ et $Est(p)$] ou [$Sud(p)$ et $Haut(p)$] mais non les deux (EXT_1),
- soit [$Nord(p)$ et $Ouest(p)$] ou [$Haut(p)$ et $Ouest(p)$] mais non les deux (EXT_2),
- soit [$Nord(p)$ et $Bas(p)$] ou [$Est(p)$ et $Bas(p)$] mais non les deux (EXT_3).

Notons que cette définition est ambiguë : en effet, si p est exactement 26-adjacent à deux points, les deux conditions de chaque clause sont mutuellement exclusives.

Le point p est un *point terminal* s'il est soit un point extrémité de ligne soit un point presque extrémité de ligne.

Répéter
Supprimer en parallèle les points non terminaux de l'objet
vérifiant au moins une des templates des classes A, B, C ou D
(dans le cas de la classe D, il faut tester la simplicité du point) ;
Jusqu'à ce qu'il n'y ait plus de suppression durant une itération.

Algorithme 36: *Algorithme de Ma et Sonka (*MASO*).*

B.4.2.2 Commentaires et résultats

Les squelettes obtenus peuvent ne pas être minces : nous pouvons constater la présence de points redondants dans le squelette de l'objet "O" de la figure B.31. En effet, considérons l'objet (plus simple) de la figure B.30. Il y a quatre points 26-simples : A, B, C et D ; ils vérifient respectivement les templates A_1, A_1, A_3 et A_5. Le point A (resp. D) est extrémité car ses voisins $Est(p)$ et $Bas(p)$ appartiennent à l'objet (EXT_3) (resp. $Haut(p)$ et $Ouest(p)$ (EXT_2)) ; les points B et C ne sont pas des points terminaux. Les deux points 26-simples A et D sont préservés, ce sont alors des points redondants dans le squelette obtenu. Notons également que le squelette

obtenu n'est pas symétrique (un tel résultat était attendu car des directions sont privilégiées dans les templates et dans la définition de point terminal).

Remarque : À la section 6.7.3, nous avons montré que cet algorithme ne préservait pas la topologie.

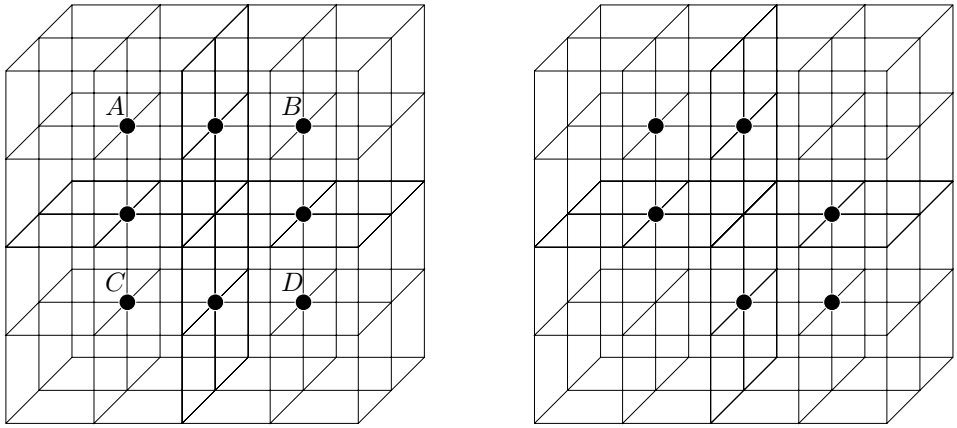


FIG. B.30 – *Présence de points redondants dans un objet - algorithme* MASO *:* (a) *objet initial,* (b) *squelette obtenu.*

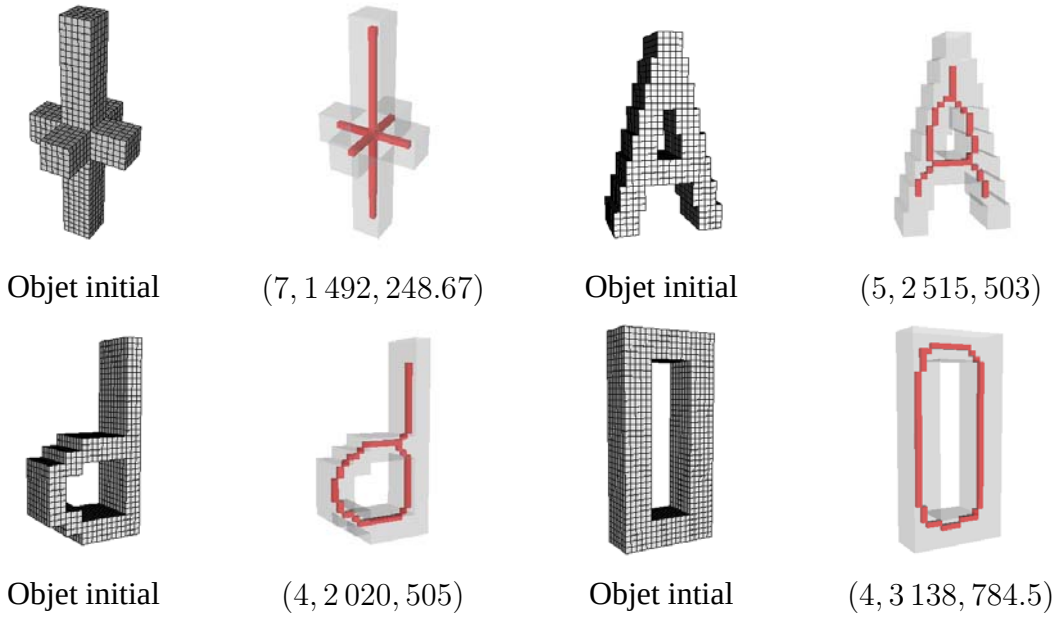


FIG. B.31 – *Squelettisation avec l'algorithme* MASO *(*REPR7*).*

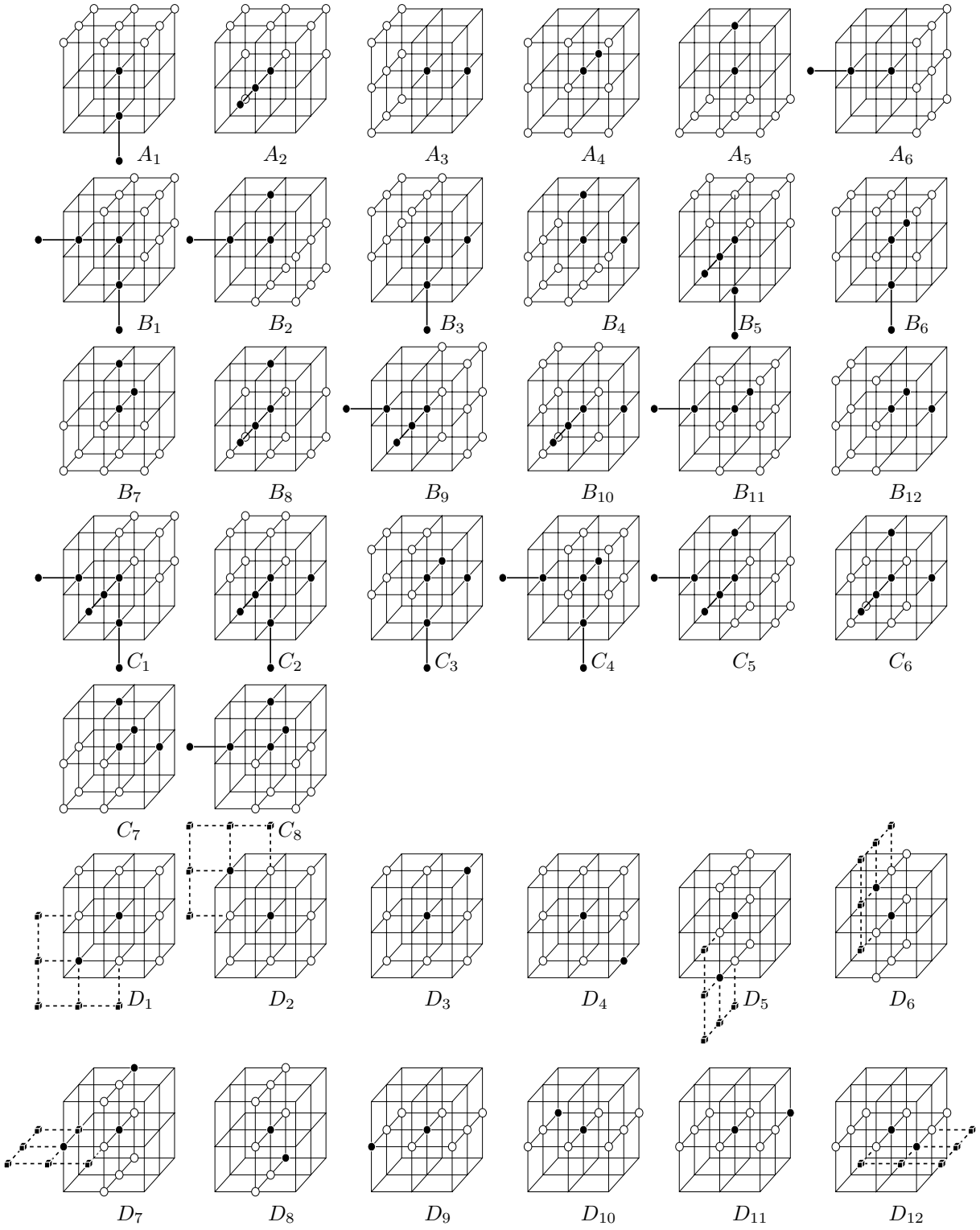


FIG. B.32 – *Les* 4 *séries de templates utilisées par l'algorithme* MASO. *Au moins un point représenté par un cube noir doit appartenir à l'objet (templates* D_1*,* D_2*,* D_5*,* D_6*,* D_7 *ou* D_{12}*).*

B.4.3 Algorithme fortement parallèle de Manzanera et al. (MB_3D) [MBPL99c] [MBPL99a]

B.4.3.1 Description

Il s'agit d'un algorithme fortement parallèle permettant l'obtention de squelettes surfaciques et d'un support de 81 points (le point central étant compté). Il utilise plus de points que les algorithmes fortement parallèles étudiés auparavant (cf. sections B.4.1 et B.4.2).

Le schéma de squelettisation est décrit par l'algorithme 37. Un point x est éliminé s'il vérifie α_1, α_2 ou α_3 (à symétries près), sauf si son 18-voisinage contient le motif β_1, ou si son 26-voisinage contient le motif β_2 (Fig. B.33). Considérons une configuration vérifiant α_1 ou α_2 ou α_3, si de plus son 18-voisinage contient β_1 alors le point x appartient au motif (c'est l'un des points représentés), en revanche si son 26-voisinage contient le motif β_2, le point x n'est pas forcément l'un des deux points noirs de l'objet, représentés dans le motif β_2.

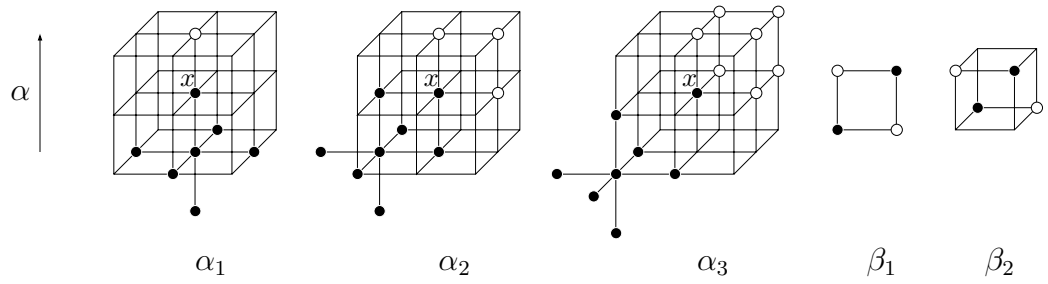


FIG. B.33 – *Un point x est éliminé s'il vérifie α_1, α_2 ou α_3, sauf si son 18-voisinage contient le motif β_1, ou si son 26-voisinage contient le motif β_2.*

Répéter
Supprimer en parallèle les points P de l'objet, vérifiant
le motif α_1 ou le motif α_2 ou le motif α_3,
et tels que les motifs β_1, β_2 ne soient pas inclus resp. dans $N^*_{18}(P)$ ou $N^*_{26}(P)$
(motifs à considérer à rotations de 90 degrés près) selon les axes Ox, O_y et O_z
P étant tel qu'il ne vérifie pas les motifs β_1 et β_2 (à rotations de 90 degrés près)
Jusqu'à ce qu'il n'y ait plus de suppression durant une itération.

Algorithme 37: *Algorithme de Manzanera et Bernard (*MB_3D*).*

B.4.3.2 Commentaires et résultats

Cet algorithme est l'extension directe de la version 2D, étudiée dans la section A.5.6. Parmi les algorithmes que nous avons étudiés dans ce chapitre, c'est le plus facile à implémenter : un

point P est supprimé si c'est un point simple et voisin d'un point intérieur et tel que les motifs β_1 et β_2 ne soiennt pas inclus respectivement dans $N^*_{18}(P)$ ou $N^*_{26}(P)$.

Comme aucune direction n'intervient dans les motifs (contrairement aux algorithmes de Ma (section B.4.1) et de Ma et Sonka (section B.4.2) ; plus précisément des conditions directionnelles sont imposées dans ces deux algorithmes fortement parallèles), alors le processus de squelettisation est homogène (ou isotrope) ; les squelettes obtenus sont alors bien centrés. Néanmoins, et c'est la contrepartie du bon centrage des squelettes, les rubans d'épaisseur 2 ne sont pas amincis. Notons qu'aucune condition implicite n'est imposée au niveau des points extrémités ; le processus préserve les courbes et les bouts de surface d'épaisseur 1 (car leurs points simples ne peuvent être voisins d'un point intérieur).

Remarque : À la section 6.7.4, nous avons montré que cet algorithme préserve la topologie.

B.5 Conclusion

Nous avons passé en revue différents algorithmes des trois classes introduites au chapitre 4, à savoir : les algorithmes de type sous-itérations directionnelles, de type sous-mailles ou fortement parallèles. Nous avons constaté que certains algorithmes fournissent des squelettes curvilignes ou surfaciques ou les deux. Nous avons également remarqué que les problèmes qui apparaissaient en 2D, existent encore maintenant. Le long de cette étude, nous avons mis en évidence différents autres problèmes, tels que le grignotage de contours, une phase préalable d'étiquetage.

Dans la deuxième partie de cette thèse, après avoir rappelé le concept de point P-simple (chap. 5), nous avons proposé de nouveaux algorithmes de squelettisation, basés sur la suppression de tels points (chap. 6). Les problèmes relevés dans ce chapitre nous ont guidés pour l'élaboration de ces nouveaux algorithmes. Nous avons comparé ces nouveaux algorithmes avec ceux de Palágyi et Kuba (en 6 et 12 sous-itérations, resp. PAKU6 et PAKU12), celui de Gong et Bertrand (GOBE) et celui de Manzanera et al. (MB_3D), étudiés dans ce chapitre.

Retenons que les auteurs des algorithmes présentés dans ce chapitre ont apporté des preuves difficiles et énumératives afin de démontrer que leurs algorithmes préservaient bien la topologie. En revanche, l'utilisation des points P-simples permet de concevoir des algorithmes de squelettisation, sans avoir besoin de montrer qu'ils préservent la topologie ; ce qu'ils font par la définition même des points P-simples. Rappelons que nous avons montré, à l'aide des points P-simples, que l'algorithme fortement parallèle de Ma (MA) (cf. section 6.7.2) et que l'algorithme fortement parallèle de Ma et Sonka (MASO) (cf. section 6.7.3) ne préservaient pas la topologie. Nous avons également montré que l'algorithme de Manzanera et al. (MB_3D) préservait la topologie (cf. section 6.7.4.2).

Dans la troisième partie de ce travail, nous avons proposé également des algorithmes de squelettisation pour images 3D binaires, grâce à une approche topologie discrète (chapitre 9). Dans ce cadre, nous avons donné une nouvelle caractérisation de points terminaux de surface (cf. section 9.5.3).

Annexe C

Graphes de décision binaire

Une propriété P qu'un élément x doit satisfaire peut être décrite par une fonction booléenne f : par exemple, la fonction f vaut 1 si l'élément x vérifie la propriété P et 0 dans le cas contraire (P pourrait être la propriété de simplicité d'un point, par exemple). La fonction f peut être stockée sous la forme d'une table de vérité (il faut alors passer en revue toutes les configurations possibles que peut vérifier l'élément x et calculer f pour chacune d'entre elles, dans le cas où f n'est pas donnée explicitement). Nous pouvons également utiliser une table de consultation qui sera indicée par un nombre codant une configuration (plus précisément, à l'ensemble des variables permettant de tester P – ou de calculer f – quelle que soit la configuration), et telle que le contenu associé à cette configuration indique si elle vérifie la propriété P (*i.e.* si $f(x)$ vaut 0 ou 1 pour une configuration x). Prenons le cas d'un algorithme de squelettisation. Afin de déterminer si un point est simple (cela étant à faire pour tous les points de l'image), plutôt que d'examiner son voisinage et de déterminer si ce point est simple ou non (par parcours de composantes connexes dans son voisinage par exemple), nous pouvons accéder directement à la table de consultation, préalablement calculée pour toutes les configurations possibles ; l'indice de la table correspondant au voisinage du point x.

Deux problèmes apparaissent aussitôt : la création de la table et son stockage. Par exemple, si la propriété P consiste à caractériser les points simples en 2D, 2^8 $(= 256)$ configurations (correspondant à toutes les configurations dans $N_8^*(x)$) doivent être générées, et pour chacune d'entre elles, si le point central x d'une telle configuration est simple, alors la valeur 1 est associée à la configuration correspondante (qui représente également l'indice dans la table de consultation). Dans le cas 3D, toujours avec la propriété de simplicité, 2^{26} $(= 67\,108\,864)$ configurations (correpondant à $N_{26}^*(x)$) doivent être générées ; en codant la réponse (0 ou 1) sur 1 bit, la taille de la table nécessite alors 2^{26} bits $(= 8$ Mégaoctets) de mémoire pour être stockée.

Nous présentons ici la notion de graphe de décision binaire (Binary Decision Diagrams - ou BDD). De tels graphes permettent de représenter une fonction booléenne de façon très efficace. Quelques précisions concernant leur création ainsi que quelques propriétés, inconvénients et des exemples d'utilisation seront donnés. Les BDD se sont révélés être des outils puissants et indispensables tout au long de ce travail (nous le verrons par la suite sur des exemples de leur mise en œuvre).

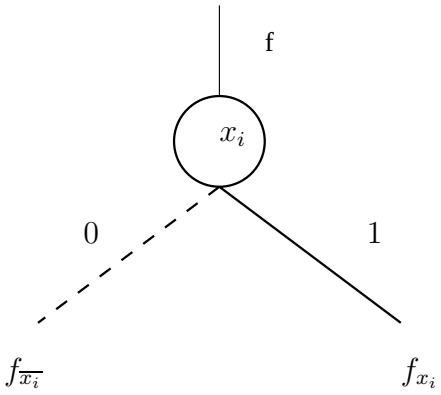


FIG. C.1 – *Décomposition de Shannon représentée par un arbre binaire.*

C.1 Définition

C.1.1 Terminologie générale [Bry86]

Les *graphes de décision binaire* (*Binary Decision Diagrams - BDD*) (introduits par C.Y. Lee [Lee59] et S.B. Akers [Ake78], voir [Bry86]) représentent des fonctions booléennes comme des graphes orientés sans cycle dont les nœuds correspondent aux variables nécessaires à la définition de la fonction (support) et dont les nœuds terminaux (feuilles) sont étiquetés par les valeurs 0 et 1. Un *graphe de décision binaire ordonné* (*Ordered Binary Decision Diagram* - OBDD) est un BDD avec les contraintes que les variables d'entrée soient ordonnées et que chaque chemin source-feuille dans le OBDD visite les variables d'entrée dans un ordre ascendant. Un *graphe de décision binaire ordonné réduit* (*Reduced Ordered Binary Decision Diagram* - ROBDD) est un OBDD dans lequel chaque nœud représente une fonction logique distincte (pour un ordonnancement fixé, deux ROBDD d'une même fonction sont isomorphes).

C.1.2 Définition ([AJCL$^+$00], chap. 2)

Soit $f(x_1, x_2, \ldots, x_n)$, une fonction de $\{0,1\}^n$ dans $\{0,1\}$.
Nous avons $\forall i \in \{1, \ldots, n\}$, $f(x_1, x_2, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n) = \overline{x_i}.f(x_1, x_2, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) + x_i.f(x_1, x_2, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$. Les termes $f(x_1, x_2, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n)$ et $f(x_1, x_2, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$ sont appelés les *cofacteurs négatif et positif* de f par rapport à la variable x_i. Ils sont notés respectivement $f_{\overline{x_i}}$ et f_{x_i}. La *décomposition de Shannon* sur la variable x_i s'écrit $f = \overline{x_i}.f_{\overline{x_i}} + x_i.f_{x_i}$. Cette décomposition est unique.

La décomposition de Shannon peut être représentée par un arbre binaire pour lequel la racine est étiquetée par la variable de décomposition x_i, le fils droit par le cofacteur positif (cas où la variable x_i vaut 1, dans la suite ce choix sera matérialisé par un trait plein) et le fils gauche par le cofacteur négatif (cas où la variable x_i vaut 0, dans la suite ce choix sera matérialisé par un trait en pointillé) (Fig. C.1).

Si cette décomposition de Shannon, avec cette représentation sur les deux cofacteurs, est

réitérée pour toutes les variables de f, alors nous obtenons un arbre binaire, appelé *arbre de Shannon*, pour lequel les feuilles sont les constantes 0 et 1, et les nœuds sont étiquetés par les variables de la fonction. Un tel arbre est une représentation équivalente à la table de vérité de la fonction. Un BDD est un arbre de Shannon dans lequel des contraintes sont imposées (ordre des variables, réduction des sous-graphes isomorphes), décrites ultérieurement (section C.2.1.1).

Introduisons maintenant quelques notations : chaque nœud non terminal v est étiqueté par une variable $var(v)$ et présente des arcs dirigés vers deux fils : $lo(v)$ (en pointillé) correspondant au cas pour lequel la variable vaut 0, et $hi(v)$ (en trait continu) correspondant au cas pour lequel la variable vaut 1. Chaque nœud terminal est étiqueté 0 ou 1.

La valeur de la fonction pour une configuration donnée est déterminée en traçant un chemin de la racine au nœud terminal, en suivant les branches indiquées par les valeurs affectées aux variables ; la valeur de la fonction est alors l'étiquette du nœud terminal.

Une fois qu'un ordre total sur les variables est fixé, puisque la décomposition de Shannon est unique, alors la représentation sous forme d'arbre de Shannon est unique ; l'arbre comporte $2^n - 1$ nœuds ; n étant le nombre de variables. Il existe dans cette représentation des redondances. Nous donnerons les règles de réduction dans la section C.2.1.1. Le graphe sans redondance est appelé *graphe de décision binaire réduit*. Dans le cas où il possède le même ordre de décomposition des variables sur tous ces chemins, on parle de *graphe de décision binaire réduit ordonné (ROBDD)*. Un ROBDD est alors canonique. En raison de leur représentation canonique, ces graphes rendent aisés les tests de propriétés fonctionnelles comme la satisfaisabilité et l'équivalence [Bry86]. La taille effective du ROBDD dépend de l'ordre choisi pour les variables (voir paragraphe C.2.2). Dans la suite, nous ne travaillons qu'avec des ROBDD, que nous renommons dorénavant BDD.

C.2 Obtention d'un BDD

C.2.1 Réduction d'un BDD

C.2.1.1 Règles de réduction [Bry92]

En commençant avec n'importe quel arbre binaire satisfaisant une propriété d'ordonnancement, on peut réduire sa taille en appliquant de façon répétée, les règles de transformation suivantes :

- Règle 1 : Supprimer les nœuds terminaux dupliqués, éliminer tous les nœuds terminaux d'une étiquette donnée, sauf un, et rediriger tous les arcs allant aux nœuds éliminés vers celui restant.
- Règle 2 : Supprimer les nœuds non terminaux dupliqués. Soient deux nœuds non terminaux u et v tels que $var(u) = var(v)$, $lo(u) = lo(v)$, $hi(u) = hi(v)$, alors éliminer l'un de ces deux nœuds et rediriger tous les arcs entrant au nœud à éliminer vers le nœud restant.
- Règle 3 : Supprimer les tests redondants. Soit un nœud non terminal v tel que $lo(v) = hi(v)$, alors éliminer v et rediriger vers $lo(v)$ tous les arcs entrants vers v.

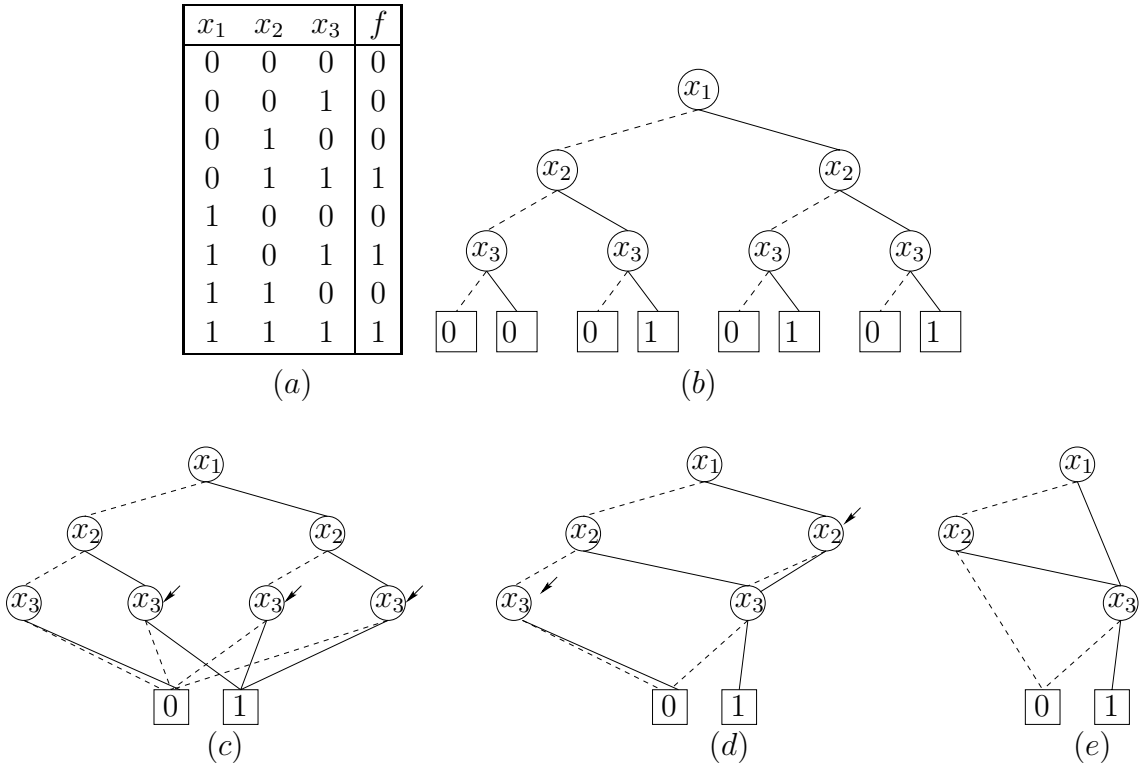


FIG. C.2 – (a) *Table de vérité de* f, (b) *arbre de décision associé. Application des règles de réduction sur l'arbre de Shannon représenté en* (b) *: suppression* (c) *des terminaux dupliqués,* (d) *des non-terminaux dupliqués,* (e) *des tests redondants.*

C.2.1.2 Exemple

La figure C.2 illustre une représentation de la fonction $f(x_1, x_2, x_3)$ définie par la table de vérité donnée à la figure C.2 (a). Pour un BDD, est imposé un ordre total $(<)$ sur l'ensemble des variables ; et pour tout nœud u et quel que soit son fils v non terminal, il est exigé que leurs variables respectives soient telles que $var(u) < var(v)$. Par exemple, sur la figure C.2 (b), l'ordre est $x_1 < x_2 < x_3$. Les figures C.2 (c), (d), (e) représentent successivement l'application des trois règles de réduction. En examinant la figure C.2 (e), nous pouvons alors décrire la fonction f par $(x_1 + \overline{x_1}.x_2).x_3$; ce qui est ici une autre formulation de f peut être vue comme une propriété de f, *i.e.* f est équivalente à la fonction $(x_1 + \overline{x_1}.x_2).x_3$, ce qui n'est pas évident directement par la seule observation de la table de vérité.

C.2.2 Problème de l'ordonnancement des variables

Le problème posé par le calcul d'un ordre qui minimise la taille du graphe est un problème co NP-complet [Bry86] : nous avons une croissance exponentielle de stockage et de l'efficacité des algorithmes manipulant les BDD selon l'ordonnacement choisi ([CLR94] chap. 36).

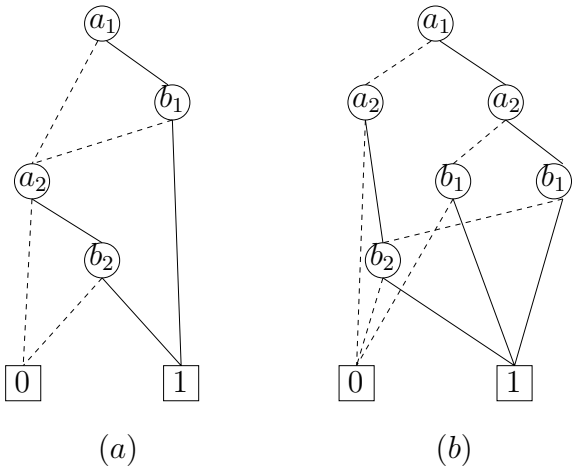


FIG. C.3 – *Représentations de BDD pour la fonction $f = a_1.b_1 + a_2.b_2$ selon deux ordonnancements différents de variables :* (a) $a_1 < b_1 < a_2 < b_2$, (b) $a_1 < a_2 < b_1 < b_2$.

Considérons la figure C.3, où est décrite la formule $f = a_1.b_1 + a_2.b_2$ selon deux ordonnancements différents. Sur la figure C.3 (a), est représenté le BDD selon l'ordonnancement $a_1 < b_1 < a_2 < b_2$. Le graphe comporte 4 nœuds non terminaux. Sur la figure C.3 (b) est représenté le BDD selon l'ordonnancement $a_1 < a_2 < b_1 < b_2$. Le graphe comporte 6 nœuds non terminaux. En généralisant à $2n$ variables, le premier ordonnancement $a_1 < b_1 < a_2 < b_2 < \ldots < a_n < b_n$ conduit à un graphe de $2n$ sommets non terminaux ; le second ordonnancement $a_1 < a_2 < \ldots < a_n < b_1 < b_2 < \ldots < b_n$ mène à un graphe de $2(2^n - 1)$ sommets non terminaux (par récurrence, nous avons un arbre binaire complet sur les n premiers niveaux pour les variables a_i, et nous avons le même nombre de nœuds (non terminaux) pour les variables b_i).

En pratique, l'ordonnancement est choisi manuellement ou par une heuristique (ou dans notre cas, par l'ordonnancement induit par l'expression de la formule booléenne en entrée, par exemple suivant un ordre de balayage vidéo du voisinage d'un point central). On peut noter qu'il existe des packages de création de BDD, qui utilisent un réordonnancement dynamique des variables (voir [CYB97]).

C.2.3 Autres possibilités d'obtention de BDD

Il existe plusieurs façons de concevoir des BDD.

La première consiste à réduire l'arbre de Shannon, comme on l'a vu précédemment. Il est également possible de créer un BDD directement à partir de la formule booléenne [BRB90], plutôt que par réduction de l'arbre de Shannon. Pour cela, est construit récursivement le BDD de façon à éviter de fabriquer un sous-arbre déjà construit. À chaque étape de la récursion, la décomposition de Shannon est réalisée suivant la variable courante et le BDD est construit à partir des cofacteurs positif et négatif de f, de façon à éviter de construire deux fois le même objet (utilisation d'une table à adressage dispersé – hashcode, de règles de simplification afin d'arrêter la récusivité, ...). La version [BRB90] utilise une information supplémentaire sur les arcs

(complement-edge) afin d'inverser le résultat, une table de hachage permettant de maintenir une forme canonique (par unicité – suppression et non-adjonction – de sous-graphes isomorphes), et d'un récupérateur de mémoire (nœuds désalloués). C'est le package que nous avons utilisé pour la création de tous les BDD tout au long de cette thèse. Dans [CYB97], se trouvent des détails concernant les méthodes de construction récursives par largeur (breadth-first approach) ou par profondeur d'abord (depth-first approach) ; la première est telle qu'elle est plus rapide en termes d'accès mémoire et la seconde propose une faible surcharge de mémoire. En fait dans [CYB97], les auteurs proposent une construction hybride tirant parti des deux précédentes méthodes.

C.2.4 Problème de la taille mémoire

Il est difficile d'estimer le nombre de nœuds que comportera le BDD que l'on souhaite obtenir. Tout dépend de la fonction considérée (et de l'ordonnancement). Par exemple, peu de configurations vérifiant une propriété P peuvent donner un BDD de faible taille. Néanmoins, plus de configurations peuvent permettre une meilleure compression. Trop de configurations peuvent empêcher également une trop grande compression. Cela est un problème, car l'obtention d'un BDD associé à une propriété n'est pas garantie. De plus, dans ce cas, la limitation en mémoire peut être éventuellement résolue pour un autre ordonnancement des variables. En fait, tout au long de cette thèse, sauf dans un cas, nous avons toujours pu obtenir les BDD pour une propriété donnée : soit directement, soit indirectement en passant par des BDD intermédiaires (voir explications C.3.4.4). Néanmoins, il est impossible d'obtenir le BDD correspondant aux points P-simples en 3D (voir chapitre 5), quel que soit l'ensemble P, car il est impossible d'obtenir la formule booléenne associée. En effet, elle référence 26 variables du voisinage, qui peuvent prendre deux valeurs "appartient ou non à l'objet" et les mêmes 26 variables (nommées différemment) indiquent leur appartenance à un ensemble P ; 52 variables sont alors exigées et la taille de l'arbre binaire complet comporterait 2^{52} nœuds soit $524\,288$ Gigaoctets, selon le type de stockage précédemment expliqué.

Notons qu'il est possible de construire le BDD de façon dynamique, sans stockage de la formule booléenne ; c'est d'ailleurs ce qui a été fait pendant cette thèse avec le package utilisé. Souvent, nous avons eu recours aux isométries (voir l'annexe D) afin de ne pas passer en revue toutes les configurations possibles, mais un plus petit nombre (principe du crible d'Eratosthène ([Sed91]), afin d'obtenir un ensemble minimal de configurations).

C.3 Utilisations

Dans cette section, nous donnons quelques exemples d'utilisation des BDD, tels que la compression d'images, la synthèse logique, etc.

C.3.1 Compression d'images [SB95]

Soit une image binaire 2D. Les variables du BDD sont les coordonnées des pixels de l'image, et les feuilles (valeurs 0 ou 1) sont des carrés noirs ou blancs. D'après des tests sur quelques

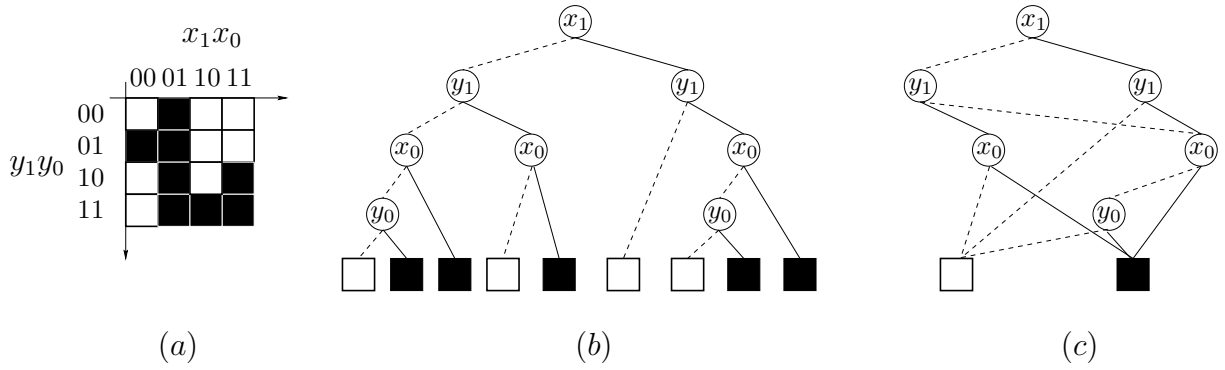


FIG. C.4 – *Utilisation d'un BDD pour la compression d'images.* (a) *Image,* (b) *bintree,* (c) *BDD. Les variables* x_0, x_1, y_0, y_1 *décrivent les coordonnées des points.*

images [SB95], le BDD utilise un quart de nœuds en moins qu'avec un bintree (binary quadtree, structure découpant l'image), et un quart de mémoire en plus (Fig. C.4). Dans ce même article, les BDD sont également utilisés pour stocker une séquence d'image (une racine par image, le graphe est maintenant une arborescence).

C.3.2 Dans les circuits et outils de synthèse logique [Bry86] [CYB97]

C.3.3 Détermination de points simples [RM96]

Il s'agit d'obtenir un BDD dont les nœuds correspondent aux voisins d'un point x nécessaires pour déterminer si x est simple ou non (*i.e.* $N_8^*(x)$ dans le cas 2D, et $N_{26}^*(x)$ dans le cas 3D). Ces BDD ont servi pour certains algorithmes utilisant la détection de points simples [Akt97]. Ils ont également servi lors de l'étude de la segmentation topologique du réseau vasculaire du foie, étude réalisée avec P. Doklàdal [DLPB99a] [DLPB99b] [Dok00].

C.3.4 Dans cette thèse

C.3.4.1 Obtention en pratique du BDD

Nous fournissons une formule booléenne au programme BDD (le package utilisé est celui décrit dans [BRB90]), qui crée un arbre (en Postscript). Avec un traducteur Postscript vers C, nous obtenons un code pouvant être inclus dans un programme, par exemple un algorithme de squelettisation. Ce code est une succession de GOTO (branchement avec étiquette) correspondant aux différents nœuds et directions de l'arbre. Cette technique est celle proposée par [RM96].

C.3.4.2 Utilisation pour les valeurs des nombres topologiques [Loh97]

Nous avons introduit un codage pour décrire les couples de valeurs possibles pour les nombres topologiques. Ce codage a été décrit par des variables complémentaires. Nous avons modifié la

formule booléenne afin qu'elle tienne compte maintenant de ces variables. Nous récupérons en sortie le code du couple de nombres topologiques de la configuration donnée en entrée. En pratique, avec ce seul code, nous pouvons caractériser certains points de l'image (par exemple, les points de surface, voir la classification à la section 3.7). Cela peut être vu comme une variante de l'utilisation d'un BDD pour la compression d'images [SB95] de différents niveaux (le code correspondant au niveau de coloration représentant un couple de valeurs topologiques). En pratique, nous avions prélablement calculé un BDD pour chaque valeur des nombres topologiques. Puis nous avons utilisé ces derniers afin de calculer plus rapidement la formule booléenne servant à créer le BDD final.

C.3.4.3 Obtention des MNS en 2D

Lors de la génération du voisinage 4×3 d'un ensemble $\{p, q\}$ tel que p et q soient simples, les configurations, telles que p devient non simple après que q soit enlevé, sont extraites. Le BDD obtenu à partir de ces configurations peut nous permettre de retrouver les motifs donnés par R.W. Hall, dans le cas des MNS (voir section 2.6.2.2). Pour ce même exemple, nous avons généré toutes les configurations de l'ensemble $\{p, q\}$ à partir du BDD des points simples pour p et q, cela afin d'obtenir les configurations plus rapidement. Ce qui est un gain de temps mineur dans le cas 2D, est important voire nécessaire dans le cas 3D (voir ci-après).

C.3.4.4 Détermination des points α-simples

Dans la troisième partie de ce mémoire, a été donnée la définition récursive de points α_n-simples (cf. section 7.5). Dans le cas 3D, et pour $n = 3$, il est très long de déterminer directement les points α_3-simples parmi toutes les configurations possibles. La façon la plus efficace de déterminer les points α_3-simples (et d'en obtenir un BDD) a alors été de les obtenir à partir du BDD des points α_2-simples (cf. section 9.2).

Nous avons également proposé une notion de point α-simple et terminal de surface (cette notion utilise le concept d'arbre). Nous avons alors utilisé le BDD des points α_3-simples afin de ne retenir que les premiers candidats, à partir desquels nous avons testé la condition d'être terminal de surface. Nous avons créé le BDD associé aux points terminaux de surface. Au final, l'algorithme de squelettisation utilise seulement des BDD afin de détecter les points supprimables.

C.4 Conclusion

En conclusion, un BDD peut être vu comme un outil précieux permettant de stocker une formule booléenne sous une forme très compacte (avantageant la mémoire mise à disposition d'un programme les manipulant) mais également permettant d'extraire certaines propriétés découlant de celle sur laquelle le BDD a été construit. C'est le cas, lorsque le BDD final n'utilise pas une variable d'entrée, cela signifie que la formule est indépendante de cette variable (voir la Règle 3 à la section C.2.1.1). De même, lors de l'examen du résultat, nous pouvons nous apercevoir que

la présence de certaines variables amène tout de suite à une valeur finale, sans tester les autres variables ; cela permet de donner certaines caractérisations efficaces (par exemple, certains motifs dans les MNS n'utilisent pas toutes les variables, cf. section C.3.4.3).

Annexe D

Isométries

Dans cette annexe, sont rappelées quelques généralités concernant les isométries dans $\mathcal{Z}^n$, $n = 1, \ldots, 3$. Les isométries dans $\mathcal{Z}^3$ ont été utilisées lors de l'implémentation des algorithmes utilisant des templates, généralement données à isométries près (voir l'annexe B). Elles permettent également d'obtenir un ensemble minimal de configurations à partir desquelles et par l'utilisation du crible d'Eratosthène, on retrouve toutes les configurations. Ce qui permet de manipuler de plus petits fichiers de données et de tester certaines propriétés plus rapidement. Cela sera expliqué en fin de ce chapitre.

D.1 Isométries dans $\mathcal{Z}$

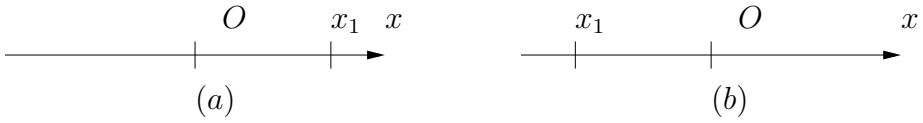


FIG. D.1 – *Isométries dans $\mathcal{Z}$.*

Considérons le repère orthonormé $\mathcal{R}(O, \vec{i})$, $\vec{i}$ vecteur unitaire selon l'axe (Ox). Il y a deux façons de placer un point x_1 à distance 1 de O et sur l'axe Ox (Fig. D.1). Il y a alors 2 isométries dans $\mathcal{Z}$.

Soit $\vec{b}_1 = (1)$. Les images possibles de $\vec{b}_1$ par une isométrie sont $\vec{\beta}_1 = (1)$ (correspondant à l'identité, Fig. D.1 (a)) et $\vec{\beta}_2 = (-1)$ (correspondant à une symétrie centrale par rapport à O, Fig. D.1 (b)).

D.2 Isométries dans $\mathcal{Z}^2$

Considérons le repère orthonormé $\mathcal{R}(O, \vec{i}, \vec{j})$, $\vec{i}$ (resp. $\vec{j}$) vecteur unitaire selon l'axe (Ox) (resp. (Oy)). Il y a quatre façons de placer un point x_1 à distance 1 de O, et pour chacune d'entre elles, il y a 2 façons de placer le point x_2 à distance 1 de O et de manière qu'il ne soit pas sur le même axe que x_1. Il y a alors 8 isométries dans $\mathcal{Z}^2$.

Soient $\vec{b}_1 = \begin{pmatrix}1\\0\end{pmatrix}$ et $\vec{b}_2 = \begin{pmatrix}0\\1\end{pmatrix}$. Les images possibles de $\vec{b}_1$ et $\vec{b}_2$ par une isométrie sont parmi $\vec{\beta}_1 = \begin{pmatrix}1\\0\end{pmatrix}$, $\vec{\beta}_2 = \begin{pmatrix}0\\1\end{pmatrix}$, $\vec{\beta}_3 = \begin{pmatrix}-1\\0\end{pmatrix}$, $\vec{\beta}_4 = \begin{pmatrix}0\\-1\end{pmatrix}$.

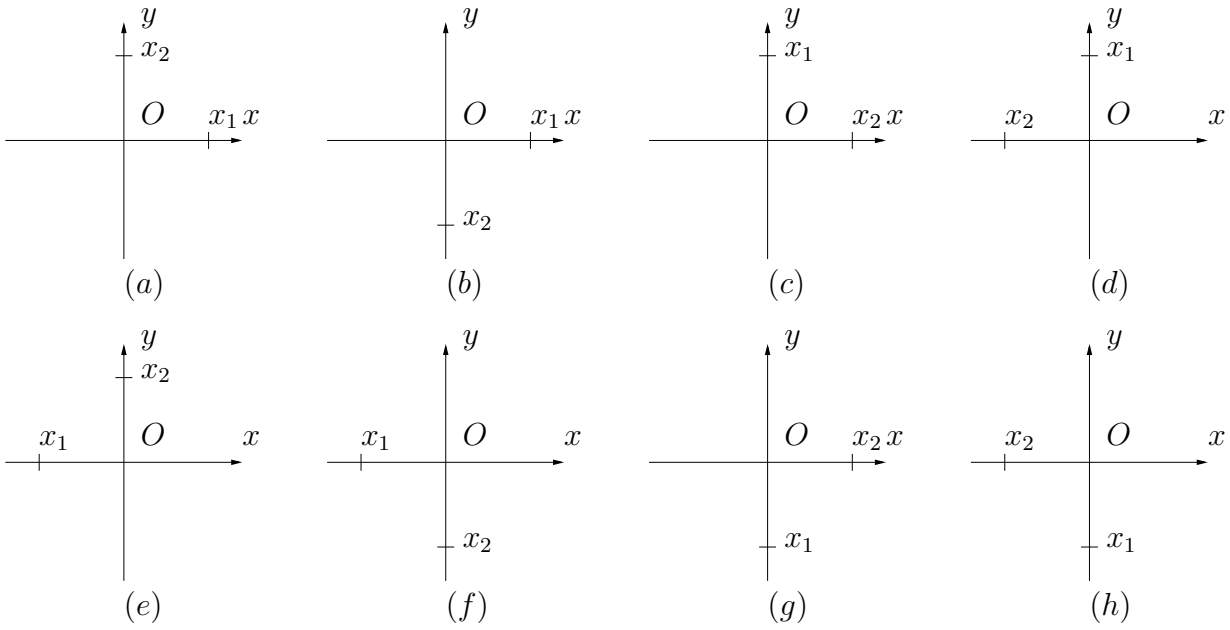


FIG. D.2 – *Isométries dans $\mathcal{Z}^2$.*

Les 8 matrices suivantes décrivent les 8 isométries dessinées à la figure D.2 :

$\begin{pmatrix}1&0\\0&1\end{pmatrix}$	$\begin{pmatrix}1&0\\0&-1\end{pmatrix}$	$\begin{pmatrix}0&1\\1&0\end{pmatrix}$	$\begin{pmatrix}0&-1\\1&0\end{pmatrix}$
identité	s_x	$r_1 \circ s_x$	r_1
(a)	(b)	(c)	(d)
$\begin{pmatrix}-1&0\\0&1\end{pmatrix}$	$\begin{pmatrix}-1&0\\0&-1\end{pmatrix}$	$\begin{pmatrix}0&1\\-1&0\end{pmatrix}$	$\begin{pmatrix}0&-1\\-1&0\end{pmatrix}$
$s_x \circ r_1^2$	r_1^2	r_1^3	$s_x \circ r_1$
(e)	(f)	(g)	(h)

D.3 Nombre d'isométries dans $\mathcal{Z}^n$

- Pour $n = 1$, il y a 1 axe et 2 façons de placer un seul point sur cet axe ; soit 2 isométries,
- Pour $n = 2$, il y a 2 axes et 2 façons de placer le premier point sur chaque axe, et 2 façons de placer le second point sur l'axe restant ; soit 8 isométries,
- Pour $n = 3$, il y a 3 axes et 2 façons de placer le premier point sur chaque axe, 2 façons de placer le deuxième point sur chacun des 2 axes restants, et 2 façons de placer le troisième point sur l'axe restant ; soit 48 isométries.

Plus généralement, pour n axes (dans $\mathcal{Z}^n$, $n \geq 1$), il y a :
- $2 \times n$ façons de placer le premier point sur les n axes,
- $2 \times (n-1)$ façons de placer le deuxième point sur les $n-1$ axes, $\vdots$
- 2×1 façons de placer le dernier point sur l'axe restant.

Soit $2^n(n \times (n-1) \times \ldots 1) = 2^n n!$ isométries, $n!$ désignant la factorielle de n.

D.4 Génération automatique des isométries de $\mathcal{Z}^n$

Soit $\vec{b}_i = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$, avec un "1" à la i-ième ligne, $i = 1, \ldots, n$.

Soient $B_n = \{\vec{b}_i, i = 1, \ldots, n\}$ et $B_{-n} = \{-\vec{b}_i, i = 1, \ldots, n\}$. Les images possibles de $\vec{b}_i$ ($\forall i = 1, \ldots, n$) par une isométrie décrivent $B_n \cup B_{-n} = \{\vec{\beta}_i, i = 1, \ldots, 2n\}$, avec $\vec{\beta}_i = b_i$ si $i = 1, \ldots, n$ et $\vec{\beta}_i = -b_{i-n}$ si $i = n+1, \ldots, 2n$.

Le schéma de programme, décrit à la figure D.3, permet l'affichage des matrices des isométries M_j, avec $M_j = (\vec{b}'_1, \vec{b}'_2, \ldots, \vec{b}'_n)$ et $j = 1, \ldots, 2^n n!$ (ce schéma de programme a été proposé par M. Couprie).

Pour $\vec{b}'_1$ de $\vec{\beta}_1$ à $\vec{\beta}_{2n}$ **faire**
  Pour $\vec{b}'_2$ de $\vec{\beta}_1$ à $\vec{\beta}_{2n}$ tel que $\vec{b}'_2$ ne soit pas aligné avec $\vec{b}'_1$ **faire**
    $\vdots$
      Pour $\vec{b}'_n$ de $\vec{\beta}_1$ à $\vec{\beta}_{2n}$ tel que $\vec{b}'_n$ ne soit pas aligné avec $\vec{b}'_1, \ldots, \vec{b}'_{n-1}$ **faire**
        Afficher $(\vec{b}'_1, \vec{b}'_2, \ldots, \vec{b}'_n)$

FIG. D.3 – *Schéma de programme (ébauche) pour l'obtention des matrices des isométries.*

Soit $\vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{pmatrix}$, avec $x_i \in \{-1, 0, 1\}$, pour $i = 1, \ldots, n$. On définit $|\vec{x}| = \begin{pmatrix} |x_1| \\ \vdots \\ |x_i| \\ \vdots \\ |x_n| \end{pmatrix}$, le vecteur formé des valeurs absolues des coordonnées de $\vec{x}$ et $Max(\vec{x}) = Max(x_1, x_2, \ldots, x_n)$, la plus grande coordonnée de $\vec{x}$.

Avec ces notations, le schéma de programme précédent peut alors être réécrit, voir Fig. D.4.

Pour $\vec{b}'_1$ de $\vec{\beta}_1$ à $\vec{\beta}_{2n}$ **faire**
  Pour $\vec{b}'_2$ de $\vec{\beta}_1$ à $\vec{\beta}_{2n}$ tel que $Max(|\vec{b}'_1| + |\vec{b}'_2|) = 1$ **faire**
    ⋮
      Pour $\vec{b}'_n$ de $\vec{\beta}_1$ à $\vec{\beta}_{2n}$ tel que $Max(|\vec{b}'_1| + \ldots + |\vec{b}'_n|) = 1$ **faire**
        Afficher $(\vec{b}'_1, \vec{b}'_2, \ldots, \vec{b}'_n)$

FIG. D.4 – *Schéma de programme pour l'obtention des matrices des isométries.*

Soit $M_j = (\vec{b}'_1, \vec{b}'_2, \ldots, \vec{b}'_n)$, la matrice de la j-ième isométrie obtenue avec cet algorithme, avec $j = 1, \ldots, 2^n n!$. L'image de $\vec{b}_i$ par M_j est $M_j\vec{b}_i$ et donne pour résultat un vecteur $\vec{\beta}_k$ de $B_n \cup B_{-n}$.

On a $\bigcup_{j=1,\ldots,2^n n!} \{M_j\vec{b}_i, i = 1, \ldots, n\} = \{\vec{\beta}_k, k = 1, \ldots, 2n\}$.

Par exemple, dans $\mathcal{Z}^2$, avec $M_2 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ la matrice de la symétrie par rapport à O_x, et $\vec{b}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, le produit $M_2\vec{b}_2 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ donne $\begin{pmatrix} 0 \\ -1 \end{pmatrix}$ qui est le vecteur $\vec{\beta}_4$.

Remarque : Nous aurions également pu utiliser le fait que toute isométrie de $\mathcal{Z}^n$ s'écrit de façon unique comme un produit d'une matrice du groupe d'ordre 2^n des matrices diagonales à coefficients $+1$ ou -1 et de rang n et d'une matrice du groupe symétrique des permutations de $\{1, \ldots, n\}$ d'ordre $n!$ [BR74] [Cal84].

D.5 Isométries dans $\mathcal{Z}^3$

Parmi les 48 isométries, seules 13 ont été implémentées ; les autres isométries peuvent être retrouvées par une composition d'au plus 2 de ces 13 isométries. Les matrices correspondant à ces 48 isométries sont données au tableau D.1. Les 13 isométries de base retenues sont iso_0, iso_1, iso_3, iso_4, iso_5, iso_7, iso_{12}, iso_{19}, iso_{24}, iso_{26}, iso_{28}, iso_{36}, iso_{43}. Les 48 isométries sont représentées à la figure D.5 : à la première ligne sont représentées successivement iso_0 jusqu'à iso_7, et ainsi de suite.

$\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&1&0\\0&0&-1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&0&1\\0&1&0\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&0&-1\\0&1&0\end{pmatrix}$
$iso_0 = \textit{Identité}$	$iso_1 = s_{xy}$	$iso_2 = r_x \circ s_{xy}$	$iso_3 = r_x$
$\begin{pmatrix}1&0&0\\0&-1&0\\0&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&-1&0\\0&0&-1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&0&1\\0&-1&0\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&0&-1\\0&-1&0\end{pmatrix}$
$iso_4 = s_{xz}$	$iso_5 = r_x^2$	$iso_6 = r_x \circ r_x^2$	$iso_7 = s_{c,-yz}$
$\begin{pmatrix}0&1&0\\1&0&0\\0&0&1\end{pmatrix}$	$\begin{pmatrix}0&1&0\\1&0&0\\0&0&-1\end{pmatrix}$	$\begin{pmatrix}0&0&1\\1&0&0\\0&1&0\end{pmatrix}$	$\begin{pmatrix}0&0&-1\\1&0&0\\0&1&0\end{pmatrix}$
$iso_8 = r_z \circ s_{xz}$	$iso_9 = r_z \circ rx_2$	$iso_{10} = r_z \circ r_x$	$iso_{11} = r_x \circ s_{c,-xz}$
$\begin{pmatrix}0&-1&0\\1&0&0\\0&0&1\end{pmatrix}$	$\begin{pmatrix}0&-1&0\\1&0&0\\0&0&-1\end{pmatrix}$	$\begin{pmatrix}0&0&1\\1&0&0\\0&-1&0\end{pmatrix}$	$\begin{pmatrix}0&0&-1\\1&0&0\\0&-1&0\end{pmatrix}$
$iso_{12} = r_z$	$iso_{13} = r_z \circ s_{xy}$	$iso_{14} = r_z \circ s_{c,-yz}$	$iso_{15} = r_y \circ r_z$
$\begin{pmatrix}0&1&0\\0&0&1\\1&0&0\end{pmatrix}$	$\begin{pmatrix}0&1&0\\0&0&-1\\1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&1\\0&1&0\\1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&-1\\0&1&0\\1&0&0\end{pmatrix}$
$iso_{16} = r_y^2 \circ r_z$	$iso_{17} = r_y \circ s_{c,-yz}$	$iso_{18} = r_y \circ s_{xy}$	$iso_{19} = r_y$
$\begin{pmatrix}0&-1&0\\0&0&1\\1&0&0\end{pmatrix}$	$\begin{pmatrix}0&-1&0\\0&0&-1\\1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&1\\0&-1&0\\1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&-1\\0&-1&0\\1&0&0\end{pmatrix}$
$iso_{20} = s_{c,-xy} \circ r_y$	$iso_{21} = r_x \circ r_z$	$iso_{22} = r_y \circ r_x^2$	$iso_{23} = r_y \circ s_{xz}$
$\begin{pmatrix}-1&0&0\\0&1&0\\0&0&1\end{pmatrix}$	$\begin{pmatrix}-1&0&0\\0&1&0\\0&0&-1\end{pmatrix}$	$\begin{pmatrix}-1&0&0\\0&0&1\\0&1&0\end{pmatrix}$	$\begin{pmatrix}-1&0&0\\0&0&-1\\0&1&0\end{pmatrix}$
$iso_{24} = s_{yz}$	$iso_{25} = r_y \circ r_y$	$iso_{26} = r_y^2$	$iso_{27} = r_x \circ s_{yz}$
$\begin{pmatrix}-1&0&0\\0&-1&0\\0&0&1\end{pmatrix}$	$\begin{pmatrix}-1&0&0\\0&-1&0\\0&0&-1\end{pmatrix}$	$\begin{pmatrix}-1&0&0\\0&0&1\\0&-1&0\end{pmatrix}$	$\begin{pmatrix}-1&0&0\\0&0&-1\\0&-1&0\end{pmatrix}$
$iso_{28} = r_z^2$	$iso_{29} = r_x^2 \circ s_{yz}$	$iso_{30} = r_y^2 \circ s_{xz}$	$iso_{31} = r_x \circ r_z^2$
$\begin{pmatrix}0&1&0\\-1&0&0\\0&0&1\end{pmatrix}$	$\begin{pmatrix}0&1&0\\-1&0&0\\0&0&-1\end{pmatrix}$	$\begin{pmatrix}0&0&1\\-1&0&0\\0&1&0\end{pmatrix}$	$\begin{pmatrix}0&0&-1\\-1&0&0\\0&1&0\end{pmatrix}$
$iso_{32} = r_z \circ r_z^2$	$iso_{33} = s_{c,-xy} \circ r_x^2$	$iso_{34} = r_y^2 \circ s_{c,-xz}$	$iso_{35} = r_x \circ r_y$
$\begin{pmatrix}0&-1&0\\-1&0&0\\0&0&1\end{pmatrix}$	$\begin{pmatrix}0&-1&0\\-1&0&0\\0&0&-1\end{pmatrix}$	$\begin{pmatrix}0&0&1\\-1&0&0\\0&-1&0\end{pmatrix}$	$\begin{pmatrix}0&0&-1\\-1&0&0\\0&-1&0\end{pmatrix}$
$iso_{36} = s_{c,-xy}$	$iso_{37} = r_x^2 \circ r_z$	$iso_{38} = s_{c,-xy} \circ s_{c,-yz}$	$iso_{39} = r_y \circ s_{c,-xy}$
$\begin{pmatrix}0&1&0\\0&0&1\\-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&1&0\\0&0&-1\\-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&1\\0&1&0\\-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&-1\\0&1&0\\-1&0&0\end{pmatrix}$
$iso_{40} = r_y^2 \circ s_{c,-xy}$	$iso_{41} = s_{c,-xz} \circ s_{c,-yz}$	$iso_{42} = s_{yz} \circ s_{c,-xz}$	$iso_{43} = s_{c,-xz}$
$\begin{pmatrix}0&-1&0\\0&0&1\\-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&-1&0\\0&0&-1\\-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&1\\0&-1&0\\-1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&-1\\0&-1&0\\-1&0&0\end{pmatrix}$
$iso_{44} = r_y \circ r_y^2$	$iso_{45} = r_x \circ s_{c,-xy}$	$iso_{46} = r_z^2 \circ s_{c,-xz}$	$iso_{47} = r_y \circ r_z^2$

TAB. D.1 – Les isométries dans $\mathcal{Z}^3$.

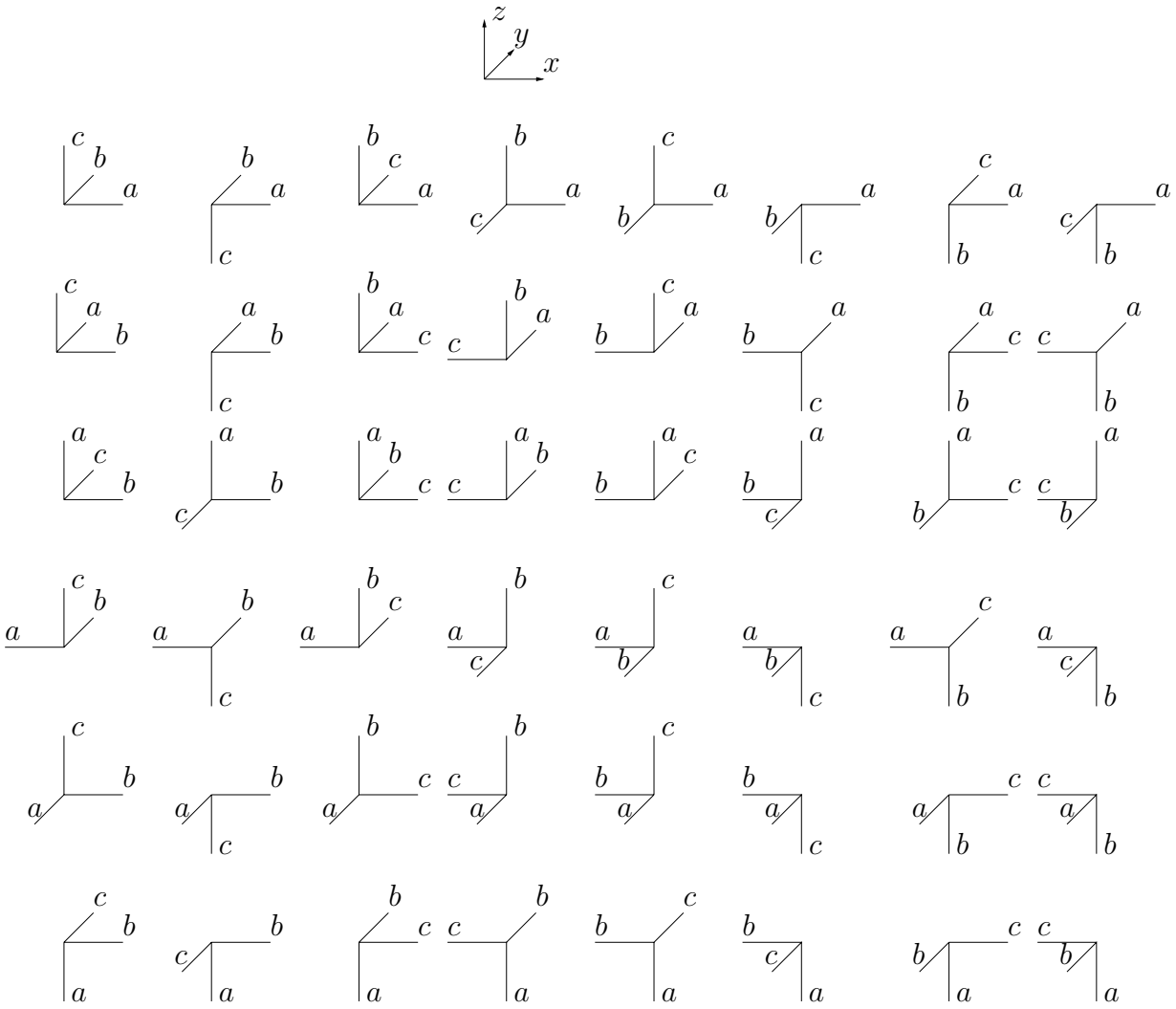


FIG. D.5 – *Isométries dans* $\mathcal{Z}^3$.

D.6 Ensemble minimal de configurations

D.6.1 Obtention d'un ensemble minimal

Nous décrivons dans cette section la façon de procéder afin d'obtenir un ensemble minimal de configurations à partir duquel on retrouve l'ensemble de toutes les configurations possibles par l'application des 48 isométries de $\mathcal{Z}^3$.

Nous créons un tableau T, chacune de ses cases correspond à une configuration et contient un témoin t. Initialement, t est mis à 0 pour chaque configuration. Chaque case du tableau est successivement examinée :

– si le témoin t de cette case vaut 0 alors on insère la configuration correspondante dans un fichier résultat, puis on génère les configurations correspondant aux 48 isométries et on

met le témoin à 1 dans les cases de T correspondant à ces isométries,
- si le témoin vaut 1, cela signifie que la configuration correspondant à cette case a déjà été obtenue par isométrie à partir d'une configuration actuellement présente dans le fichier résultat.

À la sortie de l'algorithme, on obtient un fichier contenant l'ensemble minimal de configurations. Le procédé utilisé est celui du crible d'Eratosthène [Sed91].

Parmi l'ensemble des $67\,108\,864$ configurations possibles décrivant un voisinage $3 \times 3 \times 3$, l'ensemble minimal obtenu comporte $1\,426\,144$ configurations, soit 47.1 fois moins ; valeur inférieure à 48, car une configuration donnée peut donner une même image (*i.e.* une même configuration) avec des isométries différentes.

D.6.2 Utilisation

Par exemple, pour connaître l'ensemble des configurations locales correspondant à un point simple, on peut procéder des deux façons suivantes :
- on retient les configurations correspondant à un point simple (avec les nombres topologiques, par exemple) parmi toutes les configurations possibles,
- (a) on retient les configurations correspondant à un point simple, parmi les configurations provenant de l'ensemble minimal retenu à la section D.6.1,
  (b) puis on génère les 48 isométries des configurations retenues en (a).

La seconde façon est autorisée car la simplicité est une propriété indépendante d'une direction.

De façon plus générale, lorsque l'on souhaite obtenir l'ensemble des configurations vérifiant une propriété donnée, nous pouvons procéder des deux façons suivantes :
- soit la propriété est testée sur toutes les configurations possibles, et celles qui la vérifient sont retenues, formant ainsi un ensemble A,
- soit la propriété est testée sur les configurations de l'ensemble minimal, et celles qui la vérifient sont retenues, formant ainsi un ensemble B ; puis les 48 isométries des configurations de B sont générées, produisant ainsi un ensemble C (qui est égal à A).

La deuxième façon de procéder est autorisée lorsque la propriété est indépendante des directions (par exemple, la simplicité d'un point). Elle est intéressante à utiliser lorsque le test de la propriété sur l'ensemble de toutes les configurations possibles est plus coûteux à la fois que le test de la propriété sur l'ensemble minimal et la génération des isométries des configurations de B.

Bibliographie

[ABP96] Z. Aktouf, G. Bertrand, and L. Perroton. A 3D-hole closing algorithm. In *6th International Workshop, DGCI'96*, pages 36–47, 1996.

[AdB89] C. Arcelli and G. Sanniti di Baja. A one-pass two operation process to detect the skeletal pixels on the 4-distance transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, II(4) :411–414, 1989.

[AJCL+00] P. Amblard, J.-C.Fernandez, F. Lagnier, F. Maraninchi, P. Sicard, and P. Waille. *Architectures logicielles et matérielles. Cours, études de cas et exercices corrigés*, chapter 2. 2000.

[Ake78] S.B. Akers. Binary decision diagrams. *IEEE Transaction on Computer*, C-27(6) :509–516, August 1978. (bib. d'appui).

[Akt97] Z. Aktouf. *Opérateurs topologiques pour images binaires 3D*. PhD thesis, Université Paris-Sud, 1997.

[Ale37] P. Alexandroff. Diskrete Raüme. *Mat. Sbornik 2,*, 44(3) :501–518, 1937. (bib. d'appui).

[AR95] C. Arcelli and G. Ramella. Finding grey-skeletons by iterated pixel removal. *Image and Vision Computing*, 13(3) :159–167, 1995.

[AR96] C. Arcelli and G. Ramella. Sketching a grey-tone pattern from its distance transform. *Pattern Recognition*, 29 :2033–2045, 1996.

[Arc99] C. Arcelli. Topological changes in grey-tone digital pictures. *Pattern Recognition*, 32 :1019–1023, 1999.

[AS98] C. Arcelli and L. Serino. Parallel lowering of digital pictures by topology preserving operations. In *Proceedings Fourteenth International Conference on Pattern Recognition*, pages 1601–1603, Brisbane, Australia, August 1998.

[BA94] G. Bertrand and Z. Aktouf. A three-dimensional thinning algorithm using subfields. In *Proceedings SPIE Conference on Vision Geometry III*, volume 2356, pages 113–124, 1994.

[BC99] G. Bertrand and M. Couprie. A model for digital topology. In *DGCI'99*, volume 1568 of *Lecture notes in Computer Science*, pages 229–241. 8th International Conference on Discrete Geometry for Computer Imagery, Marne-la-Vallée, France, 1999.

[BC00] F. N. Bezerra and M. Couprie. Reducing anisotropy of topological operators for grayscale images. In *Proceedings of Vision Geometry IX*, volume 4117, pages 46–57. SPIE, San Diego, USA, July 2000.

[BEC97] G. Bertrand, J.-C. Everat, and M. Couprie. Image segmentation through operators based upon topology. *Journal of Electronic Imaging*, 6(4) :395–405, 1997.

[Ber84] G. Bertrand. Skeletons in derived grids. In *7th Int. Conf. on Pattern Recognition*, pages 326–329, 1984.

[Ber92] G. Bertrand. Characterization of three-dimensional discrete holes. Technical report, ESIEE, 1992.

[Ber94] G. Bertrand. Simple points, topological numbers and geodesic numbers in cubic grids. *Pattern Recognition Letters*, 15 :1003–1011, 1994.

[Ber95a] G. Bertrand. On P-simple points. *Compte Rendu Académie des Sciences de Paris*, t. 321(Série 1) :1077–1084, 1995.

[Ber95b] G. Bertrand. P-simple points : A solution for parallel thinning. Lecture Notes in Computer Science, pages 233–242. 5th International Conference on Discrete Geometry for Computer Imagery, Sept. 1995.

[Ber95c] G. Bertrand. Sufficient conditions for 3D parallel thinning algorithms. In *Proceedings of Vision Geometry IV*, volume 2573, pages 52–60. Vision Geometry IV, SPIE, USA, 1995.

[Ber99] G. Bertrand. New notions for discrete topology. In *DGCI'99*, volume 1568 of *Lecture Notes in Computer Science*, pages 218–228. 8th International Conference on Discrete Geometry for Computer Imagery, Marne-la-Vallée, France, 1999.

[Blu67] H. Blum. *Models for the Perception of Speech and visual Form*, chapter A transformation for extracting new descriptors of shape, pages 362–380. MIT Press, Cambridge, Massachussets, 1967. (bib. d'appui).

[BM94] G. Bertrand and G. Malandain. A new characterization of three-dimensional simple points. *Pattern Recognition Letters*, 15 :169–175, 1994.

[BM95] G. Bertrand and G. Malandain. A note on "building skeleton models via 3-D medial surface / axis thinning algorithms". *Graphical Models and Image Processing*, 57(6) :537–538, November 1995.

[BM96] G. Bertrand and R. Malgouyres. Some topological properties of discrete surfaces. In Montanvert Miguet and Ubeda, editors, *DGCI'96*, volume 1176 of *Lecture Notes in Computer Science*, pages 325–336. 6th International Conference on Discrete Geometry for Computer Imagery, Lyon, France, Nov. 1996.

[BM99] T. M. Bernard and A. Manzanera. Improved low complexity fully parallel thinning algorithm. In *International Conference on Image Analysis and Processing*, Venice, Italy, Sept 1999.

[BM00] J. Burguet and R. Malgouyres. Strong thinning and polyhedrization of the surface of a voxel object. In G. Borgefors, I. Nyström, and G. Sanniti di Baja, editors,

DGCI'2000, volume 1953 of *Lecture Notes in Computer Science*, pages 222–234. International Conference on Discrete Geometry for Computer Imagery, Uppsala, Sweden, 2000.

[Bor84] G. Borgefors. Distance transformations in arbitrary dimensions. *Computer Vision, Graphics, and Image Processing*, 27 :321–345, 1984.

[Bor86] G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34 :344–371, 1986.

[BR74] A. Bouvier and D. Richard. *Groupes. Observation, théorie, pratique*. Collection Formation des enseignants – Actualités scientifiques et industrielles, 1371, 1974. (bib. d'appui).

[BRB90] K.S. Brace, R.L. Rudell, and R.E. Bryant. Efficient implementation of a bdd package. In *Proc. of the 27th IEEE Design Automation Conference*, pages 40–45, Orlando, Fl., June 1990.

[Bry86] R.E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computer*, Vol. C-35(8) :677–691, Aug 1986.

[Bry92] R. E. Bryant. Symbolic boolean manipulation with ordered binary decision diagrams. *ACM Computing Surveys*, 24(3) :293–318, September 1992.

[Cal84] J. Calais. *Éléments de théorie des groupes*. Presses Universitaires de France – PUF Mathématiques, 1984. (bib. d'appui).

[CB00] M. Couprie and G. Bertrand. Tesselations by connection in orders. In *DGCI'2000*. 9th International Conference on Discrete Geometry for Computer Imagery, Uppsala, Sweden, 2000.

[CBB99] M. Couprie, F. N. Bezerra, and G. Bertrand. Grayscale image processing using topological operators. In *Proceedings of Vision Geometry VIII*, volume 3811, pages 261–272. SPIE, Denver, USA, July 1999.

[CBK02] M. Couprie, G. Bertrand, and Y. Kenmochi. Discretization in 2D and 3D orders. In *DGCI 2002*. 10th International Conference on Discrete Geometry for Computer Imagery, Bordeaux, France, 2002. to appear.

[CCS95] S.S.O. Choy, C. Sze-Tsan Choy, and W.-C. Siu. Note "new single pass algorithm for parallel thinning". *Computer Vision and Image Understanding*, 62(1) :69–77, July 1995.

[CLR94] T. Cormen, C. Leiserson, and R. Rivest. *Introduction à l'algorithmique*, chapter 2. The MIT Press, Cambridge, Massachusets, "Introduction to Algorithms", 1994.

[CM91] J.-M. Chassery and A. Montanvert. *Géométrie discrète en analyse d'images*. Traité des Nouvelles Technologies, série Images. Hermes Eds., 1991.

[CWSI87] R.T. Chin, H.-K. Wan, D.L. Stover, and R.D. Iverson. A one-pass thinning algorithm and its parallel implementation. *Computer Vision, Graphics and Image Processing*, (40) :30–40, 1987.

[CYB97] Y.-A. Chen, B. Yang, and R.E. Bryant. Breadth-first with depth-first bdd construction : A hybrid approach. Technical report, CMU-CS-97-120, March 1997.

[DC02] X. Daragon and M. Couprie. Segmentation du néo-cortex cérébral depuis des données IRM dans le cadre de la topologie des ordres. In *RFIA2002*, volume 3, pages 809–818. Congrès francophone AFRIF-AFIA de Reconnaissances des Formes et Intelligence Artificielle (RFIA), Angers, France, 2002.

[DLPB99a] P. Dokládal, C. Lohou, L. Perroton, and G. Bertrand. Liver blood vessels extraction by a 3D topological approach. In *Second International Conference on Medical Image Computing and Computer-Assisted Intervention*. MICCAI99, Cambridge, England, September 1999.

[DLPB99b] P. Dokládal, C. Lohou, L. Perroton, and G. Bertrand. A new thinning algorithm and its application to extraction of blood vessels. In *1'st Conference Modelling and Simulation in Biology, Medicine and biomedical Engineering*, pages 32–37. BioMedSim'99, Noisy-le-Grand, France, April 1999.

[Dok00] P. Dokládal. *Grey-Scale Image Segmentation : A topological Approach*. PhD thesis, Université de Marne-la-Vallée, 2000.

[DR79] C.R. Dyer and A. Rosenfeld. Thinning algorithms for gray-scale pictures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(1) :88–89, 1979.

[EB97] J.-C. Everat and G. Bertrand. New topological operators for segmentation. In *Proceedings of ICIP'96, IEEE Signal Processing Society*, volume III/III, pages 45–48, Lausanne, Switzerland, 1997.

[EBC97] J.-C. Everat, G. Bertrand, and M. Couprie. Reconstruction operators for image segmentation. In *Proceedings of 3rd International Workshop on Vision Form., IAPR*, Capri, Italy, 1997.

[Eve97] J.-C. Everat. *Topologie des coupes et segmentation d'images par extraction de minima*. PhD thesis, Université Paris 7, 1997.

[FKVL99] A. Farina, Z.M. Kovács-Vajna, and A. Leone. Fingerprint minutiae extraction from skeletonized binary images. *Pattern Recognition*, pages 877–889, 1999. (bib. d'appui).

[GB90] W. Gong and G. Bertrand. A simple parallel 3D thinning algorithm. pages 188–190, Atlantic City, NJ, 1990. International Conference on Pattern Recognition.

[GBC96] J.-C. Everat G. Bertrand and M. Couprie. Topological approach to image segmentation. In *Proceedings of Vision Geometry V*, volume 2826, pages 65–76. SPIE, Denver, USA, 1996.

[GH89] Z. Guo and R.W. Hall. Parallel thinning with two-subiteration algorithms. *Comm. ACM*, 32(3) :359–373, March 1989.

[GH92] Z. Guo and R.W. Hall. Fast fully parallel thinning algorithms. *CVGIP : Image Understanding*, 55(3) :317–328, May 1992.

[Hal89] R.W. Hall. Fast parallel thinning algorithms : Parallel speed and connectivity preservation. *Comm. ACM*, 32(1) :124–131, January 1989.

[Hal92] R.W. Hall. Connectivity preserving parallel operators in 2D and 3D images. In *Proceedings of Vision Geometry*, volume 1832, pages 172–183. SPIE, USA, 1992.

[Hal96] R.W. Hall. *Topological Algorithms for Digital Image Processing*, volume 19 of *Machine Intelligence and Pattern Recognition*, chapter Parallel Connectivity-Preserving Thinning Algorithms, pages 145–179. Elsevier Science B.V., 1996.

[Hil69] C.J. Hilditch. *Linear skeletons from square cupboards. Machine Intelligence*, volume 4, pages 403–420. Edinburgh Univ. Press, 1969. (bib. d'appui).

[HSCP87] C.M. Holt, A. Stewart, M. Clint, and R.H. Perrott. An improved parallel thinning algorithm. *Comm. ACM*, 30(2) :156–160, February 1987.

[JC90] B.-K. Jang and R.T. Chin. Analysis of thinning algorithms using mathematical morphology. *IEEE Trans. of PAMI*, 12(6), 1990.

[Jon00] P.P. Jonker. Morphological operations on 3D and 4D images : From shape primitive detection to skeletonization. In *9th DGCI*, volume 1953 of *LNCS*, pages 371–391, 2000.

[KCM91] M.K. Kundu, B.B. Chaudhuri, and D. Dutta Majumder. A parallel graytone thinning algorithm. *Pattern Recognition Letters*, 12 :491–496, 1991.

[Kha60] E. Khalimsky. On topologies on generalized segments. *Soviet Math. Dokl.*, pages 1508–1511, 1960. (bib. d'appui).

[KKM90] E. Khalimsky, R. Kopperman, and P.R. Meyer. Computer graphics and connected topologies on finite ordered sets. *Topology and its Applications*, 36 :1–17, 1990.

[KKM91] T.Y. Kong, R. Kopperman, and P.R. Meyer. A topological approach to digital topology. *Am. Math. Monthly*, 98 :901–917, 1991.

[KMW91] R. Kopperman, P.R. Meyer, and R.G. Wilson. A Jordan surface theorem for three-dimensional digital spaces. *Discrete Computational Geometry*, 6 :155–161, 1991.

[Kon93] T.Y. Kong. On the problem of determining whether a parallel reduction operator for n-dimensional binary images always preserves topology. In *SPIE Vision Geometry II*, volume 2060, pages 69–77, 1993.

[Kon95] T.Y. Kong. On topology preservation in 2-D and 3-D thinning. *International Journal of Pattern Recognition and Artificial Intelligence*, 9(5) :813–844, 1995.

[Kon97] T.Y. Kong. Topology-preserving deletion of 1's from 2-, 3- and 4-dimensional binary images. In *DGCI'97*, volume 1347 of *Lecture Notes in Computer Science*, pages 3–18. 7th International Conference on Discrete Geometry for Computer Imagery, Montpellier, France, 1997.

[Kop94] R. Kopperman. The Khalimsky line as a foundation for digital topology. *Shape in Picture, NATO ASI Series, Series F : Computer and Systems Sciences,*, 126 :3–20, 1994.

[Kov89] V.A. Kovalevsky. Finite topology as applied to image analysis. *Computer Vision, Graphics and Image Processing*, 46 :141–161, 1989.

[Kov94] V. A. Kovalevsky. Topological foundations of shape analysis. *Shape in Picture, NATO ASI Series, Series F : Computer and Systems Sciences*, 126 :21–36, 1994.

[KR85] T.Y. Kong and A.W. Roscoe. Continuous analogs of axiomatized digital surfaces. *Computer Vision Graphics Image Processing*, 29 :60–86, 1985. (bib. d'appui).

[KR89] T.Y. Kong and A. Rosenfeld. Digital topology : introduction and survey. *Computer Vision, Graphics and Image Processing*, 48 :357–393, 1989.

[LB] C. Lohou and G. Bertrand. A 3D 12-subiteration thinning algorithm based on P-simple points. submitted to *Pattern Recognition Letters*.

[LB99] C. Lohou and G. Bertrand. Poset approach to 3D parallel thinning. In *Proceedings of Vision Geometry VIII*, volume 3811, pages 45–56. SPIE, Denver, USA, July 1999.

[LB00a] C. Lohou and G. Bertrand. New parallel thinning algorithms for 2D grayscale images. In *Proceedings of Vision Geometry IX*, volume 4117, pages 58–69. SPIE, San Diego, USA, July 2000.

[LB00b] C. Lohou and G. Bertrand. Nouvel algorithme de squelettisation parallèle d'images binaires 2D. In *RFIA2000*, volume 1, pages 493–504. 12ème congrès francophone AFRIF-AFIA de Reconnaissances des Formes et Intelligence Artificielle (RFIA), Paris, France, 2000.

[LB01] C. Lohou and G. Bertrand. A new 3D 12-subiteration thinning algorithm based on P-simple points. In Sébastien Fourey, Gabor T. Herman, and T. Yung Kong, editors, *IWCIA2001, 8th Int. Workshop on Combinatorial Image Analysis, Aug. 23-24*, pages 39–58, 2001.

[LB02a] C. Lohou and G. Bertrand. Détection de la non-validité de deux algorithmes de squelettisation pour images 3d binaires, par l'utilisation des points p-simples. Congrès à l'occasion du soixantième anniversaire de Denis Richard, submitted, 2002.

[LB02b] C. Lohou and G. Bertrand. A new 3D 6-subiteration thinning algorithm based on P-simple points. In *DGCI 2002*. 10th International Conference on Discrete Geometry for Computer Imagery, Bordeaux, France, 2002. to appear.

[LB02c] C. Lohou and G. Bertrand. New fully parallel thinning algorithms for 2d and 3d binary images based on p-simple points. In *Proceedings of Vision Geometry XI*. SPIE, Seattle, USA, July 2002. to appear.

[Lee59] C.Y. Lee. Representation of switching circuits by binary decision programs. *Bell System Technical Journal 38*, pages 985–999, 1959. (bib. d'appui).

[LKC94] T.-C. Lee, R.L. Kashyap, and C.-N. Chu. Building skeleton models via 3-D medial surface / axis thinning algorithms. *CVGIP : Graphical Models and Image Processing*, 56(6) :462–478, Nov. 1994.

[Loh97] C. Lohou. Analyse d'images 3D. Rapport de DEA, Université de Marne-la-Vallée, 1997.

[LS95] L. Lam and C.Y. Suen. An evaluation of parallel thinning algorithms for character recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(9) :914–919, September 1995.

[LVG80] S. Lobregt, W. Verbeek, and F.C.A. Groen. Three-dimensional skeletonization : Principle and algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2 :75–77, 1980. (bib. d'appui).

[LW86] H.E. Lü and P.S.P. Wang. A comment on "a fast parallel algorithm for digital patterns". *Comm. ACM*, 29(3) :239–242, March 1986.

[Ma93] C.M. Ma. Topology preservation in 3D images. *SPIE Conference on Vision Geometry II*, pages 201–207, 1993.

[Ma94] C.M. Ma. On topology preservation in 3D thinning. *Computer Vision, Graphics, and Image Processing : Image Understanding*, 59(3) :328–339, May 1994.

[Ma95] C.M. Ma. A 3D fully parallel thinning algorithm for generating medial faces. *Pattern Recognition Letters*, 16 :83–87, 1995.

[Mal97] R. Malgouyres. A definition of surfaces of $\mathcal{Z}^3$, a new 3D discrete Jordan theorem. *Theoretical Computer Science*, 186 :1–41, 1997. (bib. d'appui).

[Mar87] A. Marion. *Introduction aux techniques de traitement d'images*. Eyrolles, 1987.

[MBA93] G. Malandain, G. Bertrand, and N. Ayache. Toplogical segmentation of discrete surfaces. *International Journal of Computer Vision*, 10(2) :183–197, 1993.

[MBPL99a] A. Manzanera, T. M. Bernard, F. Prêteux, and B. Longuet. Medial faces from a concise 3D thinning algorithm. In *Proceedings ICCV*, 1999.

[MBPL99b] A. Manzanera, T. M. Bernard, F. Prêteux, and B. Longuet. Ultra-fast skeleton based on an isotropic fully parallel algorithm. In G. Bertrand, M. Couprie, and L. Perroton, editors, *DGCI'99*, number 1568 in Lecture Notes in Computer Science, pages 313–324. 8th International Conference DGCI'99, Marne-la-Vallée, March 1999.

[MBPL99c] A. Manzanera, T. M. Bernard, F. Prêteux, and B. Longuet. A unified mathematical framework for a compact and fully parallel n-D skeletonization procedure. In *Proceedings of Vision Geometry VIII*, volume 3811, pages 57–68. SPIE, Denver, USA, July 1999.

[MF98] R. Malgouyres and S. Fourey. Strong surfaces, surface skeletons and image superimposition. In *Vision Geometry VII*, volume 3454 of *SPIE*, pages 16–27, 1998.

[ML00] R. Malgouyres and A. Lenoir. Topology preservation within digital surfaces. *Graphical Models*, 62 :71–84, 2000.

[Mor80] D.G. Morgenthaler. Three-dimensional digital topology : The genus. Technical Report TR-980, Computer Vision Laboratory, University of Maryland, November 1980.

[Mor81] D.G. Morgenthaler. Three-dimensional simple points : Serial erosion, parallel thinning, and skeletonization. Technical Report TR-1009, Computer Vision Laboratory, University of Maryland, February 1981.

[MS96] C. Min Ma and M. Sonka. A fully parallel 3D thinning algorithm and its applications. *Computer Vision and Image Understanding*, 64(3) :420–433, 1996.

[NA85] A. Nakamura and K. Aizawa. On the recognition of properties of three-dimensional pictures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(6) :708–713, Nov. 1985.

[Pal89] S. K. Pal. Fuzzy skeletonization of an image. *Pattern Recognition Letters*, 10 :17–23, 1989.

[PK] K. Palágyi and A. Kuba. A hybrid thinning algorithm for 3D medical images. *Journal of Computing and Information Technology, CIT 6, 1998, 2*, pages 149–164.

[PK97a] K. Palágyi and A. Kuba. A parallel 12-subiteration 3D thinning algorithm to extract medial lines. In *Proc. 7th International Conference, CAIP'97, Lecture Notes in Computer Science*, volume 1296, pages 400–407, 1997.

[PK97b] K. Palágyi and A. Kuba. A thinning algorithm to extract medial lines from 3D medical images. In *Information Processing in Medical Imaging, Proc. 15th International Conference, IPMI'97, Lecture Notes in Computer Science*, volume 1230, pages 411–416, Berlin, 1997.

[PK98a] K. Palágyi and A. Kuba. A 3D 6-subiteration thinning algorithm for extracting medial lines. *Pattern recognition Letters*, 19 :613–627, 1998.

[PK98b] K. Palágyi and A. Kuba. A hybrid thinning algorithm for 3D medical images. *Journal of Computing and Information Technology*, CIT 6(2) :149–164, 1998.

[PK99] K. Palágyi and A. Kuba. A parallel 3D 12-subiteration thinning algorithm. *Graphical Models and Image Processing*, 61 :199–221, 1999.

[PR71] C.M. Park and A. Rosenfeld. Connectivity and genus in three dimensions. Technical Report TR-156, Computer Science Center, University of Maryland, May 1971.

[Pud98] C. Pudney. Note : Distance-ordered homotopic thinning : a skeletonization algorithm for 3D digital images. *Computer Vision and Image Understanding*, 72(3) :404–413, December 1998.

[RCM92] F. Rolland, J.-M. Chassery, and A. Montanvert. 3D medial surfaces and 3D skeletons. *Visual Form*, pages 443–450, 1992. in C.Arcelli et al. editor, Plenum Press, New York.

[RK82] A. Rosenfeld and A.C. Kak. *Digital Picture Processing*, volume 2. Academic Press, 1982.

[RM96] L. Robert and G. Malandain. Fast binary image processing using binary decision diagrams. Technical Report n 3001, pp 1-24, INRIA, Oct 1996.

[Ron86] C. Ronse. A topological characterization of thinning. *Theoretical Computer Science*, 43 :31–41, 1986.

[Ron88] C. Ronse. Minimal test patterns for connectivity preservation in parallel thinning algorithms for binary digital images. *Discrete Applied Mathematics*, 21 :67–79, 1988.

[Ron98] C. Ronse. Set-theoretical algebraic approaches to connectivity in continuous or digital spaces. *Journal of Mathematical Imaging and Vision*, 8 :41–58, 1998.

[Ros75] A. Rosenfeld. A characterization of parallel thinning algorithms. *Information Control*, 29 :286–291, 1975. (bib. d'appui).

[Ros84] A. Rosenfeld. The fuzzy geometry of image subsets. *Pattern Recognition Letters*, 2 :311–317, 1984.

[Rut66] D. Rutovitz. Pattern recognition. *J. Roy. Statist. Soc. Ser. A*129, pages 504–530, 1966. (bib. d'appui).

[SB95] M. Starkey and R.E. Bryant. Using ordered binary-decision diagram for compressing images and image sequence. Technical report, CMU-CS-95-105, January 1995.

[SCM97] P.K. Saha, B.B. Chaudhuri, and D. Dutta Majumder. A new shape preserving parallel thinning algorithm for 3D digital images. *Pattern Recognition*, 30(12) :1939–1955, 1997.

[Sed91] R. Sedgewick. *Algorithmes en langage C*. InterEditions, Addison-Wesley Europe, 1991.

[Ser82] J. Serra. *Image Analysis and mathematical morphology*. Academic Press, 1982.

[SNK94] G. Székely, M. Näf, and Olaf Kübler. Skeletal description of complex 3D shapes with application to brain anatomy. In *DGCI'94*, pages 169–182. 4th International Conference on Discrete Geometry for Computer Imagery, Grenoble, France, 1994. (bib. d'appui).

[Ste94] A. Stewart. A one-pass thinning algorithm with interference guards. *Pattern Recognition Letters*, 15 :825–832, August 1994.

[TF81] Y.F. Tsao and K.S. Fu. A parallel thinning for 3D pictures. *Computer Graphics Image Processing*, 17 :315–331, 1981.

[TF82] Y.F. Tsao and K.S. Fu. A general scheme for constructing skeleton models. *Inform. Sci.*, pages 53–87, 1982. (bib. d'appui).

[WB00] Q.J. Wu and J.D. Bourland. Three-dimensional skeletonization for computer-assisted treatment planning in radiosurgery. *Computerized Medical Imaging and Graphics*, 24 :243–251, 2000. (bib. d'appui).

[WT92] R.-Y. Wu and W.-H. Tsai. A new one-pass parallel thinning algorithm for binary images. *Pattern Recognition Letters*, 13 :715–723, October 1992.

[Xia89] Y. Xia. Skeletonization via the realization of the fire front's propagation and extinction in digital binary shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, II(10) :1076–1086, 1989.

[ZS84] T.Y. Zhang and C.Y. Suen. A fast parallel algorithm for digital patterns. *Comm. ACM*, 27(3) :236–239, March 1984.

Dans certaines références, apparaît la note : bib. d'appui ; cela signifie qu'il s'agit d'une référence non possédée ou lue partiellement.

Index

Résumé

Cette thèse propose de nouveaux algorithmes de squelettisation d'images 2D et 3D selon deux approches : l'approche topologie digitale et l'approche topologie discrète.

Dans la première partie, nous rappelons les notions fondamentales de topologie digitale et quelques algorithmes de squelettisation parmi les plus connus, basés sur la suppression de certains points simples. Puis, nous proposons une méthodologie permettant la production d'algorithmes de squelettisation fondés sur la suppression en parallèle de points P-simples. De tels algorithmes sont élaborés de façon à ce qu'ils suppriment au moins les points retirés par un autre algorithme donné. Nous appliquons cette méthodologie et produisons deux nouveaux algorithmes. Bien que les résultats semblent satisfaisants, la recherche et la mise en œuvre de tels algorithmes restent difficiles.

Dans la deuxième partie, nous utilisons le cadre mathématique des ordres. De façon plus directe qu'auparavant, nous proposons un algorithme de squelettisation consistant en la répétition de la suppression parallèle de points α_n-simples, puis en la suppression parallèle de points β_n-simples. Nous avons également proposé des définitions originales de points terminaux, nous permettant l'obtention de squelettes curvilignes ou surfaciques. Le schéma général de squelettisation est utilisé sur des images 2D, 3D binaires et 2D en niveaux de gris. Enfin, une étude de filtrage parallèle de squelettes est développée.
Mots-clés : algorithmes de squelettisation, simplicité, point simple, point P-simple, point α_n-simple, squelette curviligne, squelette surfacique, topologie digitale, ordre, topologie discrète.

Title: Contribution to the topological analysis of images: study of thinning algorithms for 2D or 3D images, according to either a digital topology approach or a discrete topology approach.

Abstract

This thesis proposes new thinning algorithms for 2D or 3D images according to two approaches using either the digital topology or the discrete topology.

In the first part, we recall some fundamental notions of digital topology and several thinning algorithms among the well-known ones, which delete simple points. Then, we propose a methodology to produce new thinning algorithms based on the parallel deletion of P-simple points. Such algorithms are conceived in order to they delete at least the points removed by another one given existent thinning algorithm. By applying this methodology, we produce two new algorithms. Although results seem to be satisfying, the proposal and encoding of new proposed algorithms are not easy.

In the second part, we use the concept of partially order set (or poset). We propose more straightforwardly than before, a thinning algorithm consisting in the repetition of parallel deletion of α_n-simple points, followed by the parallel deletion of β_n-simple points. We also have proposed new definitions of end points which permit us to obtain either curve skeletons or surface skeletons. The thinning scheme is used on 2D, 3D binary images, or on 2D grayscale images. At last, a study of a parallel filtering of skeletons is developped.
Keywords: thinning algorithms, simplicity, simple point, P-simple point, α_n-simple point, curve skeleton, surface skeleton, digital topology, order, poset, discrete topology.

DISCIPLINE : Informatique Fondamentale et Applications
Université de Marne-la-Vallée - Laboratoire A^2SI de l'ESIEE.