Semaine d'Etude Mathématiques et Entreprises 6 : Représentation des fonctions de réponse radiométrique

Basad Al-Sarray, Beniamin Bogosel, Eric Dalissier, Jean-Matthieu Etancelin, Janne-Kathrin Gunther, Rosalie Plantefève

Semaine d'Etude Mathématiques et Entreprises 6

Grenoble — 10 au 14 juin 2013

Représentation des fonctions de réponse radiométrique

Basad Al-Sarray[1], Beniamin Bogosel[2], Eric Dalissier[3],
Jean-Matthieu Etancelin[4], Janne-Kathrin Günther[5], Rosalie Plantefève[6]

Sujet proposé par l'entreprise Kolor

Correspondant : Meler Antoine[7]

Interlocuteur universitaire : Patrick Chenin[8]

[1]LMB, Université de Franche-Comté, basad.al-sarray@univ-fcomte.fr
[2]LAMA, Université de Savoie, Beniamin.Bogosel@univ-savoie.fr
[3]LJK, Université de Grenoble-Alpes, Eric.Dalissier@imag.fr
[4]LJK, Université de Grenoble-Alpes, Jean-Matthieu.Etancelin@imag.fr
[5]MRU, Université du Luxembourg; IECL, Université de Lorraine, janne.guenther@uni.lu
[6]ALTRAN, INRIA Université de Lille 1, rosalie@plantefeve.fr
[7]Kolor, antoine.meler@kolor.com
[8]LJK, Université de Grenoble-Alpes, Patrick.Chenin@imag.fr

Résumé

Ce rapport rassemble les différentes pistes de réponses apportées au problème posé par l'entreprise Kolor lors de la 6ème Semaine d'Étude Maths-Entreprises de juin 2013.

Le problème porte sur la représentation de fonctions de réponse radiométrique utilisées dans de nombreux logiciels de manipulation de photographies. Une grande partie du travail effectué a consisté à comprendre le problème et ses enjeux afin de le formaliser de façon mathématique. Après une description formelle des outils envisagés, nous les évaluons par rapport aux contraintes du problème afin de déterminer leurs avantages et inconvénients. D'un point de vue pratique, les outils sont présentés dans l'objectif d'être éventuellement développés et intégrés à un logiciel existant. Nous avons donc tenté, dans la mesure du possible, de prendre en compte la simplicité de ces outils que ce soit du côté développement ou du côté utilisation.

Ce rapport s'articule en six parties. Après une description pratique du problème, nous en détaillons les principales caractéristiques. Dans une troisième partie, nous décrivons les trois outils envisagés. Les deux parties suivantes constituent l'étude des outils par rapport aux contraintes du problème. Finalement nous donnons une conclusion de cette étude.

Table des matières

1 Introduction

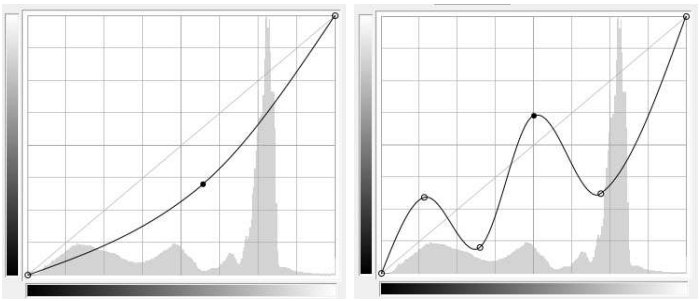
Lors de l'acquisition d'une image, divers phénomènes physiques font que l'intensité lumineuse reçue par un capteur n'est pas toujours proportionnelle à l'intensité du pixel correspondant. Ce phénomène est modélisé par une fonction de réponse radiométrique :

$$F : [0, 1] \to [0, 1],$$

qui donne les intensités d'un canal de couleur d'un pixel en fonction de l'intensité reçue par le capteur. Pour améliorer la qualité de l'image, il est donc souhaitable de pouvoir modifier les intensités afin de retrouver la luminosité réelle. Dans un cadre plus artistique, il est également possible de vouloir modifier les intensités des couleurs dans le but d'obtenir un effet spécifique. En pratique, cette opération de modification de l'intensité des pixels est effectuée grâce à l'outil courbe d'un logiciel de retouche d'image. Cet outil permet de définir une fonction :

$$f : [0, 1] \to [0, 1],$$

qui associe à chaque valeur d'intensité une nouvelle valeur. Mathématiquement, cette fonction n'est pas différente de la fonction radiométrique, mais elle ne représente pas un phénomène physique. Les Figures 1 donnent un exemple d'utilisation de l'outil courbe du logiciel GIMP [1].



(a) fonction croissante (b) fonction oscillante

FIGURE 1: Deux exemples de fonctions f.

Ces fonctions peuvent être exprimées dans différents espaces colorimétriques. Dans le système RGB, où chaque couleur est représentée par une combinaison linéaire de l'intensité de chacun des canaux rouge (R), vert (G) et bleu (B), trois fonctions f_R, f_G et f_B sont utilisées pour rendre compte de la modification des intensités de chacun des canaux. Un autre système colorimétrique fréquemment utilisé est le système HSV. Une couleur y est représentée comme combinaison de trois paramètres, la teinte (H), la saturation (S) et la valeur (V). Les fonctions utilisées pour la modification des intensités sont alors appelées f_H, f_S, et f_V.

Les fonctions f peuvent être définies par l'utilisateur ou calculées par un algorithme. Par exemple, lors de la création d'un panorama à partir de photographies possédant des valeurs

d'exposition différentes, un algorithme peut être utilisé pour trouver les fonctions f qui permettent de faire correspondre les couleurs des zones identiques sur les différentes images. Pour que l'algorithme ne donne pas de résultats aberrants, comme sur la Figure 2b, il est nécessaire de contraindre le domaine de recherche de ces fonctions. Les fonctions f doivent donc avoir certaines propriétés :

— Bijectivité de $[0, 1]$ dans $[0, 1]$,
— Continuité,
— $f(0) = 0$ et $f(1) = 1$.

(a) Image modifiée en utilisant la fonction f définie en Figure 1a

(b) Image modifiée en utilisant la fonction f définie en Figure 1b

FIGURE 2: Exemple de résultats d'une transformation f sur chacun des canaux RGB.

De plus, différentes opérations sont fréquemment effectuées sur les fonctions f comme la composition, l'inversion, la correction gamma – composition avec une fonction de puissance non entière – et le changement d'espace colorimétrique.

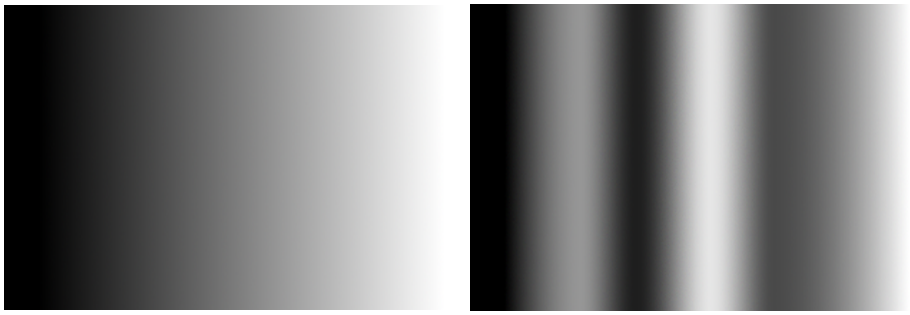
L'objectif est alors de définir une base ou un système de paramétrisation des fonctions f, idéalement identique dans tous les espaces colorimétriques, qui satisfasse à toutes les propriétés demandées et qui permette d'effectuer les calculs précédemment cités de manière simple.

2 Exposition détaillée du problème

2.1 Effet des propriétés des fonctions f

Les fonctions f sont fréquemment utilisées en traitement d'image. Elles permettent, entre autre, de corriger les problèmes liés à l'exposition de la photographie, ou de modifier les couleurs d'une image en augmentant, par exemple, les intensités du canal rouge et en diminuant celles du canal bleu. La plupart des logiciels de traitement d'image ne limitent pas l'utilisateur quant au choix de la fonction f. Cependant, certains choix peuvent conduire à des phénomènes non désirés.

Si une fonction f non bijective est appliquée sur les trois canaux RGB de l'image d'un dégradé noir et blanc, des bandes claires et foncées apparaissent (Figure 3). En effet, plusieurs valeurs d'intensité de départ sont transformées vers une même valeur d'intensité d'arrivée. Les



(a) image originale (b) image transformée

FIGURE 3: Transformation d'une image test par une fonction f non bijective identique sur chacun des canaux RGB.

zones du dégradé correspondant aux intensités de départ ne sont pas contiguës mais possèdent la même intensité après transfomation et des bandes de même intensité apparaissent donc sur l'image.

Des problèmes de dynamique sur la gamme des couleurs se posent également. Ces problèmes surviennent en raison de la précision limitée des valeurs des intensités stockées pour une image. Une fonction f comme celle de la Figure 4c augmente très fortement la dynamique de la zone claire et provoque l'apparition de bruit dans les zones correspondant à ces intensités (Figure 4b).

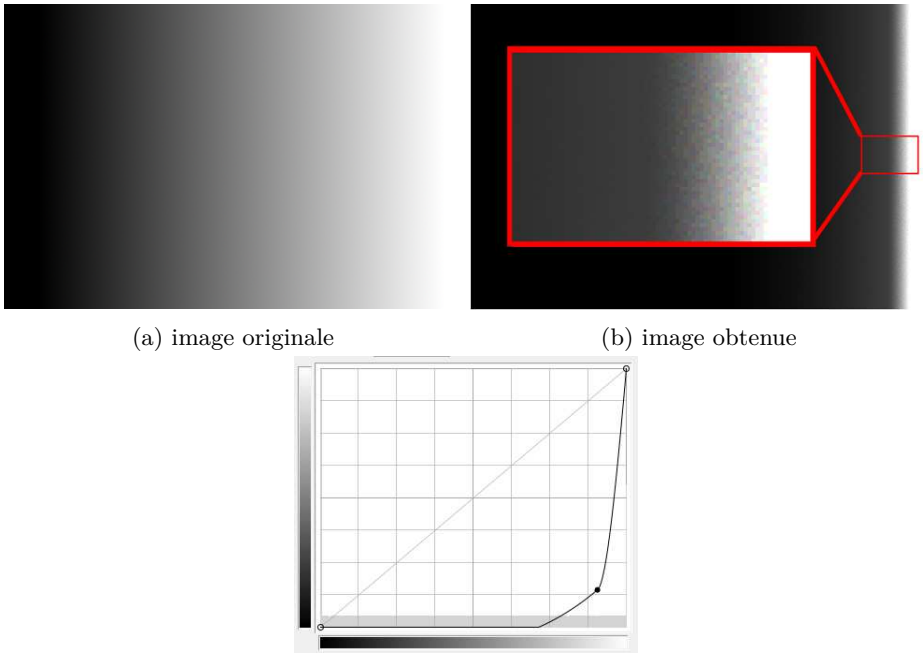
L'utilisation d'une fonction f continue mais non dérivable, donc C^0 seulement, provoque aussi des artefacts sur l'image, qui sont plus ou moins génants en fonction du contenu de l'image. Si une telle fonction f est appliquée au dégradé noir et blanc (Figure 5), une ligne plus sombre apparaît. Cette ligne n'est pas réellement plus sombre que les pixels situés à sa droite. Mais, le contraste, plus important à droite qu'à gauche, provoque l'effet visuel décrit plus haut.

À cause des effets décrits ci-dessus, il est parfois souhaitable d'interdire l'utilisation de fonctions non bijectives, à dérivées discontinues, ou de pentes trop importantes. La Section 3 présente trois systèmes de représentation pour les fonctions f qui permettent de garantir la continuité de la dérivée et d'imposer, ou de vérifier très simplement, la bijectivité.

2.2 Opérations fréquemment effectuées sur les fonctions f

Lorsqu'un utilisateur construit une fonction f, il peut décider de procéder par étapes. Par exemple, s'il connaît la fonction radiométrique F du capteur ayant enregistré l'image sur laquelle il travaille, il peut commencer par appliquer à l'image l'inverse de cette fonction pour travailler ensuite sur une échelle linéaire. En général, la fonction F est une fonction gamma, et son inverse l'est donc également. Il est ainsi important que le système de représentation choisi puisse fidèlement représenter les fonctions gamma.

Ensuite, l'utilisateur peut vouloir appliquer une transformation sur l'échelle linéaire pour, par exemple, renforcer ou atténuer le contraste des teintes claires. Ensuite, il peut appliquer de nouveau une fonction f pour obtenir un autre effet et ainsi appliquer successivement plusieurs transformations. Si l'utilisateur souhaite appliquer une nouvelle fois sa séquence de fonctions f à son image ou à une autre image, il est utile de connaître la composée des fonctions utilisées



(a) image originale (b) image obtenue

(c) fonction de transformation f

FIGURE 4: Transformation d'une image test par une fonction provoquant un problème de dynamique.

pour effectuer toutes les transformations en une seule étape.

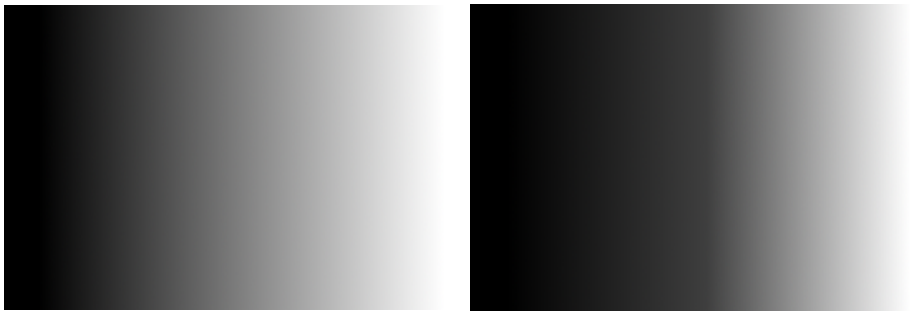
Il est également possible que l'utilisateur, ayant enregistré la fonction f qu'il avait utilisée pour modifier une image, souhaite connaître l'inverse de cette fonction pour retrouver l'échelle linéaire. Il est donc nécessaire que le système de représentation choisi permette de calculer facilement la représentation de l'inverse d'une fonction.

Enfin, après avoir effectué toutes les transformations voulues sur une image, il peut être utile d'appliquer une fonction gamma pour un meilleur affichage sur l'écran. Il est donc intéressant d'étudier comment se comporte le système de représentation vis à vis de la composition avec une fonction de puissance non entière.

2.3 Résumé du problème

Le but de cette étude est de trouver un système de représentation des fonctions f tel que :

- Les fonctions f puissent être définies de manière simple tout en respectant les propriétés suivantes :
  - Bijectivité de $[0,1]$ dans $[0,1]$,
  - Appartenance à la classe C^1,
  - $f(0) = 0$ et $f(1) = 1$.
- Les opérations suivantes puissent être effectuées avec une faible erreur tout en restant dans le même système de représentation :



(a) image originale (b) image obtenue

(c) fonction de transformation f

FIGURE 5: Transformation par une fonction non dérivable en un point.

  - Inversion.
  - Composition.
- Les fonctions gamma ($g_\gamma : x \mapsto x^\gamma$) puissent être représentées avec une faible erreur.

3 Trois systèmes de représentation des fonctions f

Trois systèmes de représentation des fonctions f seront étudiés. Le premier système est une interpolation des fonctions f par des splines cubiques monotones. Les deux autres systèmes ne sont pas des interpolations, mais des outils de conception de courbes. Il s'agit des courbes de Bézier, et des B-splines. Ce second système utilise d'ailleurs, tout comme le système de représentation par interpolation choisi, des fonctions splines. Dans tous les cas, l'utilisateur, ou l'algorithme, devra définir les fonctions f en plaçant des points de contrôle. Ces points seront sur la courbe de la fonction f dans le cas de l'interpolation, mais ce ne sera en général pas le cas pour les courbes de Bézier ni les B-splines.

3.1 Interpolation par fonctions splines

3.1.1 Définition

Les splines sont des fonctions polynomiales par morceaux régulières aux noeuds. Les interpolations par splines ont l'avantage, par rapport aux interpolations polynomiales, d'obtenir une erreur plus faible avec des polynômes de degrés identiques. Les interpolations par splines

sont utilisées dans les applications graphiques car leur construction est assez simple et qu'elles approximent des objets complexes avec une faible erreur.
Soit une suite de m noeuds t_i non décroissante :

$$a = t_0 < t_1 < \cdots < t_{m-2} < t_{m-1} = b.$$

Une spline $C : [a, b] \to \mathbb{R}$ est definie sur l'intervalle composé des $m-1$ sous-intervalles disjoints $[t_{i-1}, t_i]$. La restriction de C sur chaque intervalle est un polynôme :

$$\pi_i : [t_{i-1}, t_i] \to \mathbb{R},$$

i.e.

$$C(t) = \pi_i(t) \quad \forall t \in [t_{i-1}, t_i].$$

Le plus grand degré des polynômes π_i est l'ordre de la spline C.

Les splines cubiques présentent l'avantage d'être simple tout en permettant une grande liberté dans le choix de la forme de leur courbe représentative. Pour la partie concernant l'interpolation, ce rapport se limitera donc à l'étude de la représentation des fonctions f par interpolation par splines cubiques.

Étant donné un ensemble de points $\boldsymbol{P}_i = (t_i, y_i)$ l'interpolation par splines cubiques se construit en calculant pour $i = 1, \ldots, k$ l'unique polynôme π_i de degré trois satisfaisant pour $i = 2, \ldots, m-2$ les quatres contraintes suivantes :

$$\begin{aligned} \pi_i(t_{i-1}) &= y_{i-1}, \\ \pi_i'(t_{i-1}) &= d_{i-1}, \\ \pi_i(t_i) &= y_i, \\ \pi_i'(t_i) &= d_i, \end{aligned}$$

où d_{i-1} et d_i sont les nombres dérivés des tangentes aux points $\boldsymbol{P}_{i-1}$ et $\boldsymbol{P}_i$ respectivement, et pour $i = 1$ et $i = m-1$ ces contraintes sont remplacées par :

$$\begin{aligned} \pi_i(t_0) &= y_0, \\ \pi_i''(t_0) &= 0, \\ \pi_i(t_m) &= y_m, \\ \pi_i''(t_m) &= 0. \end{aligned}$$

Il est possible de choisir la valeur des nombres dérivés d_i tels que la spline cubique soit C^2, mais il existe également d'autres méthodes pour les évaluer qui ne garantissent qu'une régularité C^1.

3.1.2 Propriétés

L'interpolation par splines cubiques possède quelques propriétés intéressantes. Tout d'abord, s'agissant d'une interpolation, la courbe de la fonction f tracée avec cet outil passe par les points de contrôle. Donc, si l'utilisateur souhaite être sûr qu'une certaine intensité sera bien transformée en une autre de son choix, il lui suffit de placer un point de contrôle à cet endroit là.

De plus, la modification d'un point de contrôle produit simplement une perturbation locale de la fonction interpolante. En fait, si le point de contrôle $\boldsymbol{P}_i = (x_i, y_i)$ est déplacé, la fonction

interpolante n'est modifiée que sur l'intervalle $[x_{i-1}, x_{i+1}]$.

Par définition, les splines cubiques sont des fonctions régulières. Et, lorsqu'elles sont C^2, elles permettent d'approximer n'importe quelle fonction régulière avec une erreur bornée par $R.h^2 \sup |f^{(iv)}|$, où R est une constante, h est la distance maximale entre les abscisses des points de contrôle et $f^{(iv)}$ est la quatrième dérivée de la courbe. Elles permettent donc une grande liberté dans la définition de la courbe des fonctions f. L'ordre peut être amélioré en choisissant des conditions aux limites appropriées.

Par contre, les splines cubiques ne sont pas invariantes par transformations affines. En effet, elles dépendent de la position des axes de coordonnées et l'image d'une spline cubique par une transformation affine n'est en général pas la spline cubique definie par les images de ses points de contrôle.

3.2 Courbes de Bézier

3.2.1 Définition

Les courbes de Bézier ont été inventées dans les années 1960, indépendamment par Pierre Bézier et Paul de Casteljau, respectivement ingénieurs des compagnies Renault et Citroën, pour le design de pièces d'automobiles. Elles ont été developpées comme une amélioration des splines, les courbes utilisées jusqu'à cette date, qui présentaient le défaut de ne pas être invariantes par changement de repère. Aujourd'hui les courbes de Bézier sont un outil important dans la Conception Assistée par Ordinateur (CAO).

Une courbe de Bézier $\mathcal{B}$ de dégré n est la courbe paramétrique

$$\mathcal{B} : [0;1] \to \mathbb{R}^d$$

définie comme suit :

$$\mathcal{B}(t) \;=\; \sum_{i=0}^{n} B_i^n(t)\boldsymbol{P}_i \quad \forall t \in [0,1]$$

où $\boldsymbol{P}_i \in \mathbb{R}^d$ est le $i^{\text{ème}}$ point de contrôle de la courbe et B_i^n est le $i^{\text{ème}}$ polynôme de Bernstein de dégré n donné par :

$$B_i^n(t) \;=\; C_n^i t^i (1-t)^{n-i}.$$

Grâce à l'existence d'une formule récursive pour les polynômes de Bernstein, il est possible de calculer les coordonnées de $\mathcal{B}(t^0)$ pour $t^0 \in]0,1[$ en utilisant l'algorithme de De Casteljau. Il faut construire la suite de points $\boldsymbol{P}_i^{[k]}(t^0)$ à l'aide de la formule suivante :

$$\begin{array}{rcll} \boldsymbol{P}_i^{[0]}(t^0) & := & \boldsymbol{P}_i \quad \forall i \in \{0,\dots,n\} & \text{et} \\ \boldsymbol{P}_i^{[k]}(t^0) & := & (1-t^0)\boldsymbol{P}_i^{[k-1]}(t^0) \;+\; t^0 P_{i+1}^{[k-1]}(t^0) & \begin{array}{c}\forall k \in \{1,\dots,n\} \\ \forall i \in \{0,\dots,n-k\}\end{array} \end{array}$$

et finalement :

$$\mathcal{B}(t^0) \;=\; \boldsymbol{P}_0^{[n]}(t^0).$$

3.2.2 Propriétés

Les courbes de Bézier possèdent quelques propriétés intéressantes. Une courbe de Bézier se trouve toujours à l'intérieur de l'enveloppe convexe de ses points de contrôle. De plus, elle commence au point P_0 et se termine au point P_n, mais ne passe en général pas par les autres points de contrôle. Les courbes de Bézier sont lisses (C^∞) et – comme mentionné précédemment et contrairement aux splines – invariantes par transformation affine. En effet, l'image d'une courbe de Bézier par une telle transformation est la courbe de Bézier definie par les images de ses points de contrôle. Les avantages de ces propriétés par rapport au problème de départ seront détaillées dans la Section 5.

Cependant, une courbe de Bézier est définie de manière globale. C'est à dire que la modification d'un seul point de contrôle entraine la modification de toute la courbe, et il faut donc recalculer la courbe complète à chaque modification. Cette propriété est illustrée en Figure 6 où la modification d'un noeud de la courbe rouge donne la courbe verte qui est totalement différente. De plus, il n'est pas possible de donner des poids différents aux différents points de contrôle. Cela signifie qu'il est impossible de faire en sorte que la courbe s'approche davantage de certains points. L'espace des courbes représentables par une courbe de Bézier est donc assez limité.

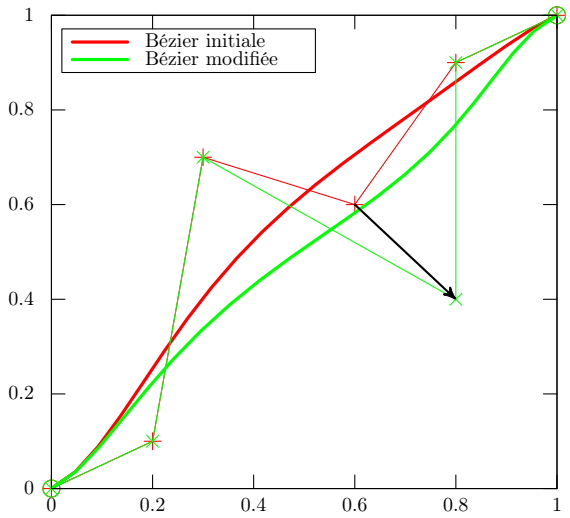


FIGURE 6: Exemple de modification d'un noeud d'une courbe de Bézier.

Néanmoins, pour ce deuxième point, il existe une généralisation des courbes de Bézier qui permet une plus grande liberté dans la définition de la courbe. Il s'agit des courbes de Bézier rationnelles. Elles se definissent comme quotient d'une courbe de Bézier, dont les points de contrôle sont multipliés par des poids, divisé par la somme de ces poids multipliés par les polynômes de Bernstein. En utilisant cette représentation, il est possible de rapprocher ou d'éloigner la courbe d'un point de contrôle donné. Les courbes de Bézier rationnelles ne seront cependant pas traitées dans ce rapport qui se concentre sur d'autres systèmes de représentation.

3.3 B-splines

3.3.1 Définition

Les B-splines sont une généralisation des courbes de Bézier. Ce sont des fonctions paramétriques polynomiales à support compact d'un degré et d'une régularité fixés. Comme les courbes de Bézier, ces courbes sont très largement utilisées en informatique graphique et en CAO. Elles ont été introduites vers la fin des années 70 par de Boor [4].

Étant donné une suite de m noeuds t_i non décroissante :

$$t_0 \leqslant t_1 \leqslant \cdots \leqslant t_{m-1},$$

une B-spline S de degré n est la courbe $S : [t_n; t_{m-n-1}] \to \mathbb{R}$ définie comme suit :

$$\boldsymbol{S}(t) = \sum_{i=0}^{m-n-2} \boldsymbol{P}_i b_{i,n}(t), \tag{1}$$

où les points $\boldsymbol{P}_i \in \mathbb{R}^d$ sont les $m-n-1$ points de contrôle de la courbe. Les fonctions de base d'une B-spline de degré n, $\{b_{i,n}\}_{i\in\{0,\ldots,m-n-2\}}$, sont définies à l'aide de la formule récursive de Cox-de Boor :

$$b_{i,0}(t) = \begin{cases} 1 & \text{si } t_i \leqslant t \leqslant t_{i+1} \\ 0 & \text{sinon} \end{cases}, \quad i = 0, \ldots, m-2$$

$$b_{i,n}(t) = \frac{t - t_i}{t_{i+n} - t_i} b_{i,n-1}(t) + \frac{t_{i+n+1} - t}{t_{i+n+1} - t_{i+1}} b_{i+1,n-1}(t), \quad \begin{matrix} i = 0, \ldots, m-n-2 \\ n = 1, \ldots, m-2. \end{matrix}$$

Il est intéressant de remarquer que ces fonctions ont la propriété d'être à support compact.

De manière analogue à l'algorithme de De Casteljau pour les courbes de Bézier, l'algorithme de de Boor permet d'évaluer une B-spline en un point t^0 à partir des $n+1$ points de contrôle associés aux fonctions de base dont le support contient t^0. L'algorithme est basé sur la construction d'une suite de points $\boldsymbol{P}_i^{[k]}$ telle que $\boldsymbol{P}_i^{[0]} = \boldsymbol{P}_i$. Si $t^0 \in [t_l; t_{l+1}[$, alors $S(t^0) = \boldsymbol{P}_l^{[n]}$ et

$$\begin{cases} \boldsymbol{P}_i^{[k]} = (1 - \alpha_{k,i})\, \boldsymbol{P}_{i-1}^{[k-1]} + \alpha_{k,i} \boldsymbol{P}_i^{[k-1]} \\ \alpha_{k,i} = \dfrac{t^0 - t_i}{t_{i+n+1-k} - t_i} \end{cases}, \quad k = 1, \ldots, n \text{ et } i = l-n+k, \ldots, l.$$

Il est possible d'obtenir la dérivée de manière simple à partir des propriétés des fonctions de base des B-splines. La littérature fournit également des algorithmes d'insertion de noeuds, algorithmes de Boehm [2] ou d'Oslo [3].

3.3.2 Propriétés

De la même manière qu'une courbe de Bézier, une B-spline est contenue dans l'enveloppe convexe de ses points de contrôle, et est invariante par transformation affine.

Mais, contrairement aux courbes de Bézier, les B-splines présentent également l'avantage d'être locales car, comme dit précédemment, les fonctions de base ont un support compact. La Figure 7 montre que le déplacement d'un point de contrôle entraîne une modification seulement locale de la courbe en $n-1$ noeuds. Dans l'exemple, la B-spline est de degré trois, la modification concerne deux noeuds, soit trois intervalles. Ainsi, l'utilisateur peut aisément modifier localement ses courbes.

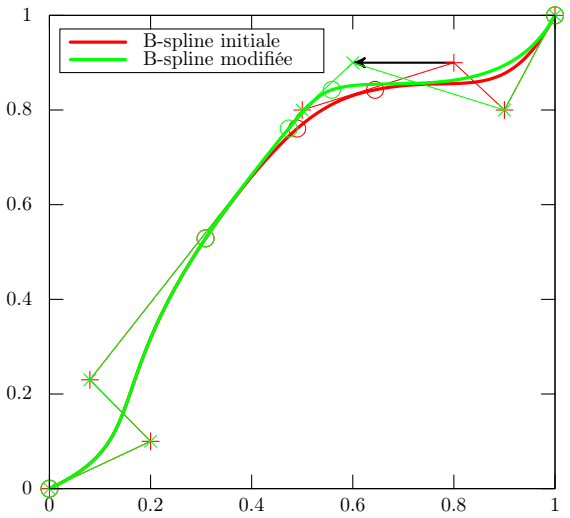


FIGURE 7: Exemple de modification d'un noeud d'une courbe B-spline.

La Figure 8 donne un exemple des fonctions de base pour des B-splines cubiques (de degré trois) pour dix noeuds uniformément répartis, $t_i = i/10$ ainsi qu'une courbe B-spline construite sur cette base.

Comme pour les courbes de Bézier rationnelles, il est possible de renforcer le poids de certains points de contrôle d'une B-spline en jouant sur la multiplicité des noeuds. En particulier, si un noeud i possède une multiplicité égale au degré de la courbe B-spline, alors la courbe passe par le point de contrôle $\boldsymbol{P}_i$. La fonction de base associée atteint la valeur 1.0 en t_i et les autres sont nulles. Cependant, la régularité de la courbe est seulement C^0 – C^{n-k} pour un noeud de multiplicité k. La Figure 9 permet de visualiser ce phénomène où les noeuds sont pris uniformes, $t_i = i/10$, sauf pour $t_5 = t_6 = 4/10$ $(= t_4)$ ainsi que la courbe associée.

La propriété de multiplicité des noeuds est intéressante en particulier lorsque les premiers et derniers noeuds sont de multiplicité égale au degré de la B-spline, alors la courbe ainsi définie atteint les points de contrôle aux extrémités.

De plus, si les noeuds t_j avec $0 \leqslant j < m = 2n + 2$ d'une B-spline de degré n et les points de contrôle $\boldsymbol{P}_i$ avec $0 \leqslant i \leqslant n$ sont choisis comme suit :

$$(t_0, \ldots, t_n, t_{n+1}, \ldots, t_{2n+1}) = (0, \ldots, 0, 1, \ldots, 1),$$

la courbe obtenue est en fait la courbe de Bézier de degré n définie par les points de contrôle $\boldsymbol{P}_i$. En effet, dans ce cas, les fonctions de base des B-splines $\{b_{i,n}\}_{i\in\{0,\ldots,n\}}$ coïncident avec les polynômes de Bernstein $\{B_i^n\}_{i\in\{0,\ldots,n\}}$.
Toute courbe de Bézier peut ainsi être construite comme une B-spline et les courbes de Bézier sont donc des B-splines particulières.

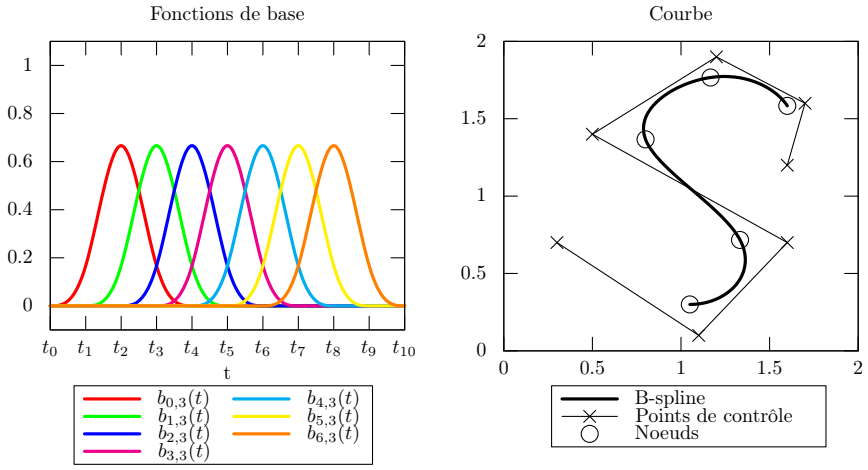


FIGURE 8: Exemple de B-spline cubique.

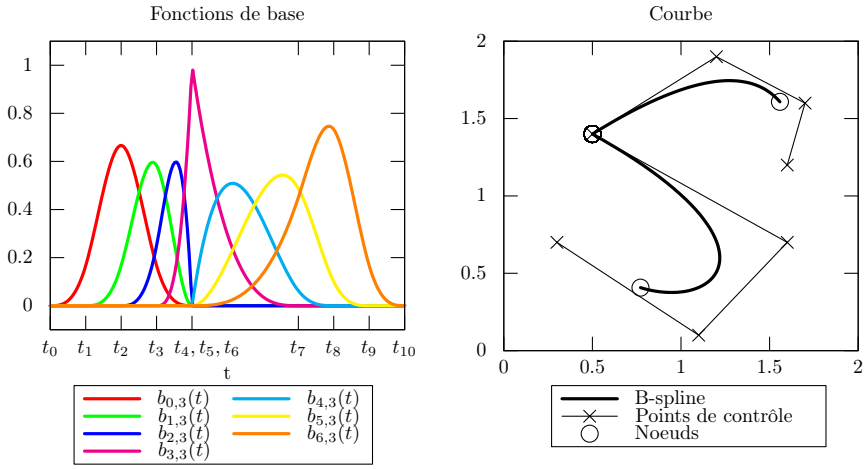


FIGURE 9: Exemple de B-spline cubique (t_4 de multiplicité trois).

4 Obtention de la monotonie

Pour obtenir la bijectivité des fonctions f conçues à l'aide des systèmes de représentation choisis, il suffit d'imposer que $f(0) = 0$, $f(1) = 1$ et que la courbe représentative de la fonction soit strictement monotone.

4.1 Pour l'interpolation par splines cubiques

Même en choisissant une suite monotone pour les points de contrôle (on considère la relation d'ordre $\boldsymbol{P}_i = (x_i, y_i) \leq \boldsymbol{P}_j = (x_j, y_j)$ si $x_i \leq x_j$ et $y_i \leq y_j$), la fonction interpolante peut ne pas être monotone à cause de la condition de régularité C^2. Mais comme les fonctions f doivent seulement être C^1, il est possible d'abandonner la régularité C^2 et de modifier le calcul de l'interpolation par splines cubiques pour obtenir la monotonie. Par commodité de langage, nous continuerons d'appeler ces objets « spline » bien que cela ne corresponde pas à la terminologie habituelle qui impose un lien entre le degré des polynômes par morceaux et la régularité de la fonction.

Fritsch et Carlson [8] donnent les conditions nécessaires et suffisantes pour obtenir une spline cubique monotone. Ils proposent un algorithme qui change les dérivées d_i et d_{i+1} en chaque point, afin de rendre la courbe monotone. Ils ont fait une comparaison entre leur méthode et les méthodes existantes telles que celle d'Akima ou les taut splines de Boor (Figure 10). Sur ces graphiques, seule la spline cubique de Fritsch et Carlson est monotone dans ce cas test. L'approche d'Akima donne de bons résultats dans beaucoup de cas mais est mise en défaut dans ce cas-là.

Voici le détail de l'approche de Fritsch et Carlson [8]. Les notations suivantes seront utilisées :
— d_i est la dérivée de la fonction interpolante en x_i ;
— $\Delta_i = (y_{i+1} - y_i)/h_i$ avec $h_i = x_{i+1} - x_i$.

Si $\Delta_i = 0$, alors la fonction interpolante ne peut être monotone sur $[x_i, x_{i+1}]$ que si $d_i = d_{i+1} = 0$. Ce cas sera exclu car la suite des points de contrôle sera choisie de sorte à être strictement monotone, donc $\Delta_i \neq 0$. Soient $\alpha = d_i/\Delta_i$ et $\beta = d_{i+1}/\Delta_i$. Alors le polynôme cubique interpolant et ses dérivées première et seconde sont données par :

$$
\begin{aligned}
c_i(x) &= \frac{\Delta_i}{h_i^2}[(\alpha+\beta-2)(x-x_i)^3 - (2\alpha+\beta-3)h_i(x-x_i)^2 + \alpha h_i^2(x-x_i)] + y_i,\\
c_i'(x) &= \frac{\Delta_i}{h_i^2}[3(\alpha+\beta-2)(x-x_i)^2 - 2(2\alpha+\beta-3)h_i(x-x_i) + \alpha h_i^2],\\
c_i''(x) &= \frac{2\Delta_i}{h_i^2}[3(\alpha+\beta-2)(x-x_i) - (2\alpha+\beta-3)h_i].
\end{aligned}
$$

La monotonie de $c_i(x)$ est directement reliée à la position de l'extrémum de c_i' :

$$
x^* = x_i + \frac{h_i}{3}\left(\frac{2\alpha+\beta-3}{\alpha+\beta-2}\right),
$$

et à sa valeur :

$$
c_i'(x^*) = \phi(\alpha,\beta)\Delta_i,\ \phi(\alpha,\beta) = \alpha - \frac{1}{3}\frac{(2\alpha+\beta-3)^2}{\alpha+\beta-2}.
$$

L'ensemble de α, β tel que c_i est monotone est la region bornée par les axes des coordonneés

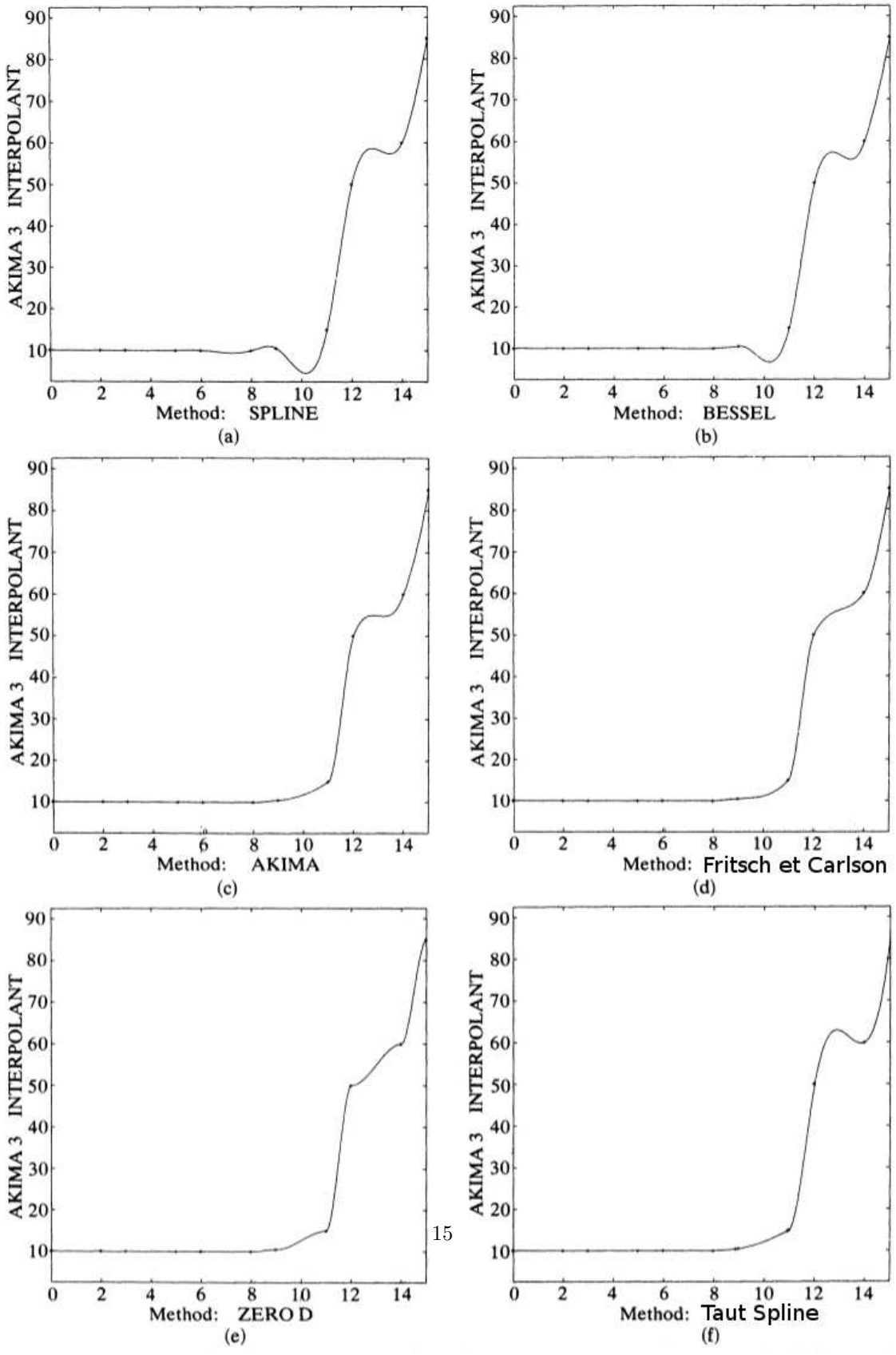


FIGURE 10: Comparaison des différentes méthodes d'interpolation par splines cubiques.

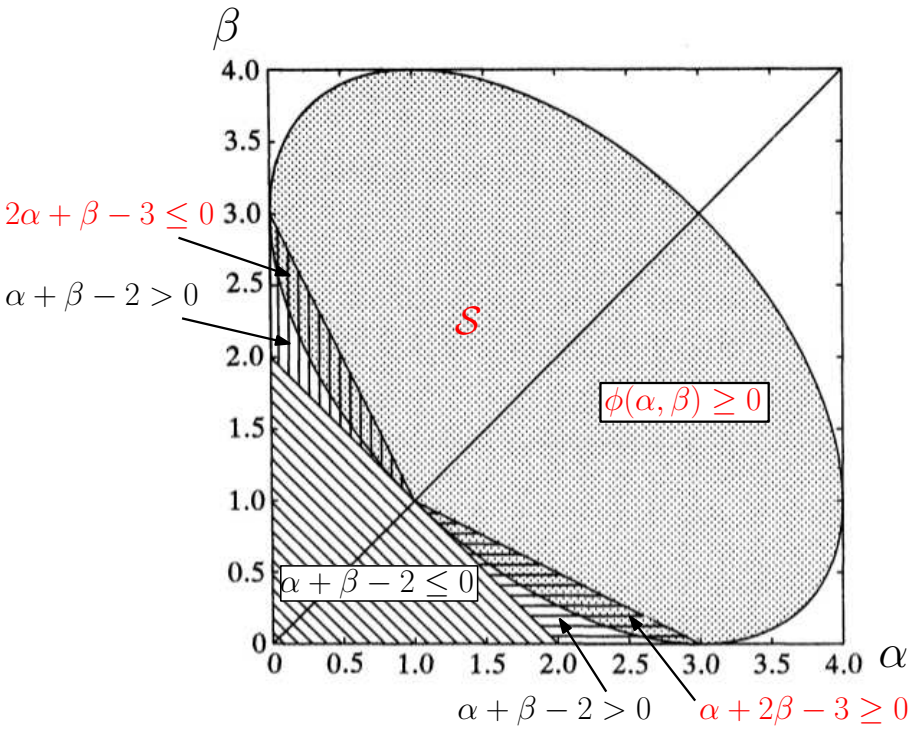


FIGURE 11: Représentation de l'espace pour lequel la courbe est monotone[8].

et $\phi(\alpha, \beta) = 0$. On dénote cette région par $\mathcal{S}$ (Figure 11).

Voici l'algorithme utilisé pour construire une interpolation par splines cubiques monotones d'une famille monotone de points :

Étape 1 : Initialisation de d_i par combinaison convexe de Δ_{i-1} et Δ_i.

Étape 2 : Calcul de α, β.

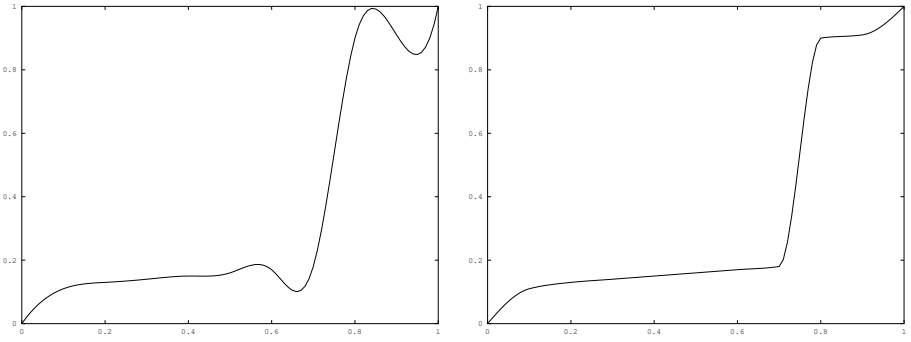
Étape 3a : Si $\alpha, \beta \geq 0$ et $\alpha, \beta \in \mathcal{S}$, passage à l'intervalle suivant.

Étape 3b : Si $\alpha, \beta \geq 0$ et $\alpha, \beta \notin \mathcal{S}$, calcul du plus grand $\tau \in [0, 1]$ tel que $(\tau\alpha, \tau\beta) \in \mathcal{S}$ et modification de d_i et d_{i+1} : $d_i = \tau d_i,\ d_{i+1} = \tau d_i$.

Exemple 1. Soient les points suivants :

$$\{\boldsymbol{P}_i\} = \{(0,0); (0.1, 0.11); (0.2, 0.13); (0.3, 0.14); (0.4, 0.15); (0.5, 0.16); (0.6, 0.17); (0.7, 0.18); (0.8, 0.9); (0.9, 0.91); (1,1)\}.$$

L'interpolation classique par splines cubiques C^2 donne une fonction qui n'est pas monotone, même si les points d'interpolation forment une suite monotone. Par contre, la seconde fonction, construite à l'aide de l'algorithme précédent, est bien monotone (voir Figure 12).



(a) Spline classique (b) Spline monotone

FIGURE 12: Les mêmes points de contrôle sont utilisés pour les deux courbes.

On sait que pour l'interpolation par splines cubiques avec une condition de régularité C^2, on a une erreur bornée par $Ch^2 \sup |f^{(iv)}|$. Lorque l'on utilise l'interpolation monotone, on perd la régularité C^2 et la preuve qui est utilisée pour le cas C^2 ne fonctionne pas ici.

Pour obtenir une estimation d'erreur dans le cas de l'interpolation monotone, on peut procéder dans la manière suivante. On travaille sur l'intervalle $I = [x_i, x_{i+1}]$. On considère S_i, le polynôme cubique pour le cas de régularité globale C^2 sur $[x_i, x_{i+1}]$, et M_i, le polynôme cubique réalisant l'interpolation monotone de f sur cet intervalle.

On peut écrire l'estimation suivante :

$$\sup_I |f(x) - M_i(x)| \leq \sup_I |f(x) - S_i(x)| + \sup_I |S_i(x) - M_i(x)|.$$

Parce qu'on a une erreur d'ordre $C(f)h^4$ pour la première expression, il est suffisant d'obtenir une estimation pour la dernière expression.

S_i et M_i sont deux polynômes cubiques et $S_i(x_l) = M_i(x_l)$ pour tout $l \in \{i, i+1\}$. Alors, il existe c, d tels que

$$S_i(x) - M_i(x) = (x - x_i)(x_{i+1} - x)(cx + d)$$

et on peut atteindre que

$$|S_i(x) - M_i(x)| \leq M_i h^2.$$

On obtient ainsi une erreur d'ordre h^2 pour l'interpolation monotone.

4.2 Pour les courbes de Bézier et les B-splines

Une condition suffisante pour obtenir la monotonie des B-splines, dont les courbes de Bézier sont un cas particulier, est d'imposer que la suite des points de contrôle soit monotone. Cependant cette condition n'est pas nécessaire, et restreint grandement l'ensemble des courbes monotones qu'il est possible de définir à l'aide d'une B-spline.

4.2.1 Tester la monotonie

Une courbe B-spline n'est pas, par définition, une fonction. Ainsi il est nécessaire d'effectuer un test de monotonie de la courbe. Notons séparément les composantes de la courbe :

$$\boldsymbol{S}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}.$$

La monotonie de la courbe est assurée par la monotonie des fonctions $x(t)$ et $y(t)$. Un moyen simple de vérification consiste à calculer la dérivée de la courbe et à vérifier son signe. L'évaluation de la dérivée est obtenue par calcul d'un nouveau jeu de $m - n - 2$ points de contrôle $\boldsymbol{Q}_i$, la dérivée d'une courbe B-spline de degré n reste une courbe B-spline mais de degré $n - 1$.

$$\boldsymbol{S}'(t) = \sum_{i=0}^{m-n-3} \boldsymbol{Q}_i b_{i+1,n-1}(t),$$

$$\boldsymbol{Q}_i = \frac{n}{t_{i+n+1} - t_{i+1}} (\boldsymbol{P}_{i+1} - \boldsymbol{P}_i).$$

Les dérivées des courbes B-splines présentées en Figures 8 et 9 sont données respectivement en Figures 13a et 13b.

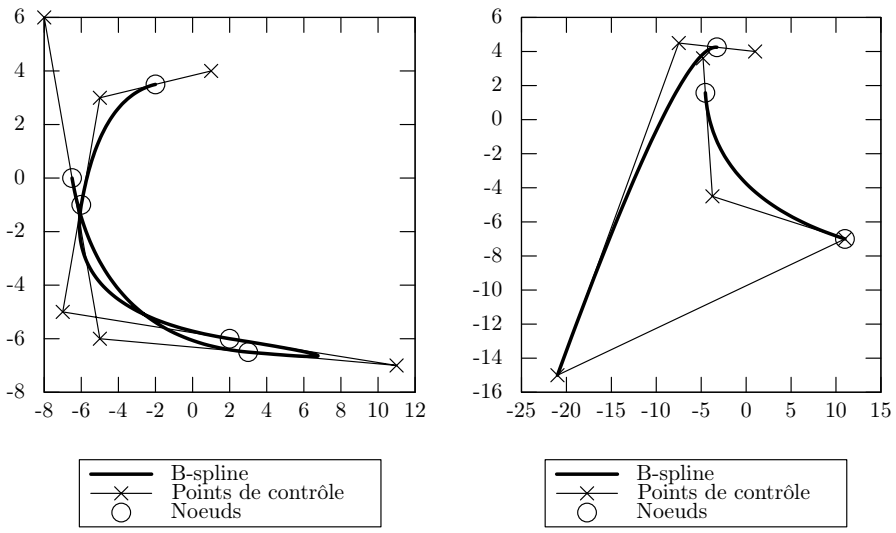
4.2.2 Imposer la monotonie

Les B-splines étant utilisées dans le monde du design, les recherches ont plus porté sur la monotonie de la courbure de la courbe que sur la monotonie du tracé en lui-même, comme dans le cas des splines cubiques.

Pour imposer la monotonie de la courbe, plusieurs méthodes existent. Toutes ces méthodes sont basées sur la définition de la courbure.

Définition :
Soit un point $X = X(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}$ d'une courbe et X_l et X_r deux points voisins, avec pour condition que ces trois points ne soient pas collinéaires ; il existe alors un cercle unique passant par ces trois points. Ce cercle a une courbure k (Figure 14) :

$$k(t) = \frac{|\dot{X}(t) \times \ddot{X}(t)|}{|\dot{X}(t)|^3}$$



(a) Noeuds uniformes (b) Noeuds de multiplicité trois

FIGURE 13: Exemples de dérivées.

avec $\dot{X} = \frac{dX}{dt}$, $\ddot{X} = \frac{d^2X}{dt^2}$.
Cette courbure peut aussi s'exprimer dans le cas d'un plan :

$$k = \frac{\ddot{x}\dot{y} - \ddot{y}\dot{x}}{[\dot{x}^2 + \dot{y}^2]^{\frac{3}{2}}}.$$

Cette autre formulation est utilisée dans certaines méthodes, notamment lors du déplacement des points afin d'augmenter la régularité de la fonction.

La dérivée de cette courbe k s'exprime sur une portion d'arc de cercle :

$$\dot{k} = \frac{|\dot{X} \times \ddot{X}||\dot{X}.\dot{X}| - 3|\dot{X} \times \ddot{X}||\dot{X}.\ddot{X}|}{|\dot{X}|^6}.$$

Une des premières méthodes de modification de courbe en vue d'imposer la monotonie avait aussi pour but d'augmenter sa régularité. Farin et Sapadis [7] proposent en effet un algorithme permettant de déplacer un point de contrôle $\boldsymbol{P}_i$ rendant la courbe monotone et de classe C^3. Cet algorithme ne s'applique que dans le cas où la courbure n'est pas deux fois différentiable. Le nouveau point de contrôle (Figure 15) est alors :

$$\hat{\boldsymbol{P}}_i = \frac{(t_{i+2} - t_i)l_i + (t_i - t_{i-2})r_i}{t_{i+2} - t_{i-2}}$$

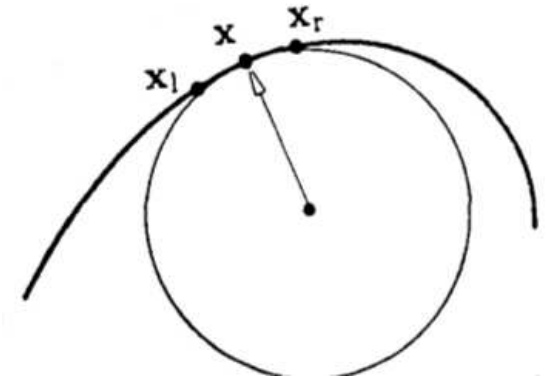


FIGURE 14: Représentation de la courbure [7].

avec

$$l_i = \frac{(t_{i+1} - t_{i-3})\boldsymbol{P}_{i-1} - (t_{i+1} - t_i)\boldsymbol{P}_{i-2}}{t_i - t_{i-3}} \quad \text{et}$$
$$r_i = \frac{(t_{i+3} - t_{i-1})\boldsymbol{P}_{i+1} - (t_i - t_{i-1})\boldsymbol{P}_{i+2}}{t_{i+3} - t_i}.$$

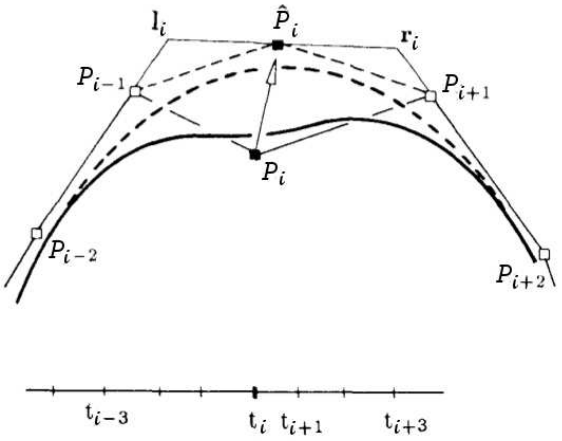


FIGURE 15: Changement d'un point de contrôle afin de rendre la courbe plus régulière (C^3) [7].

Le déplacement des points de contrôle est une méthode facilement inversible, ce qui n'est pas le cas des méthodes qui suppriment ou ajoutent des points de contrôle.

Li *et al.* donnent un exemple de cet algorithme sur une courbe donnée (Figure 16). Le signe de la courbure peut ainsi être corrigé et ne pas changer. La régularité de la courbure est aussi augmentée pour une variation minime de la courbe.

Après cette approche de modification des courbes qui entraine la monotonie, des conditions mathématiques ont été énoncées qui ont permis le développement d'autres méthodes d'obten-

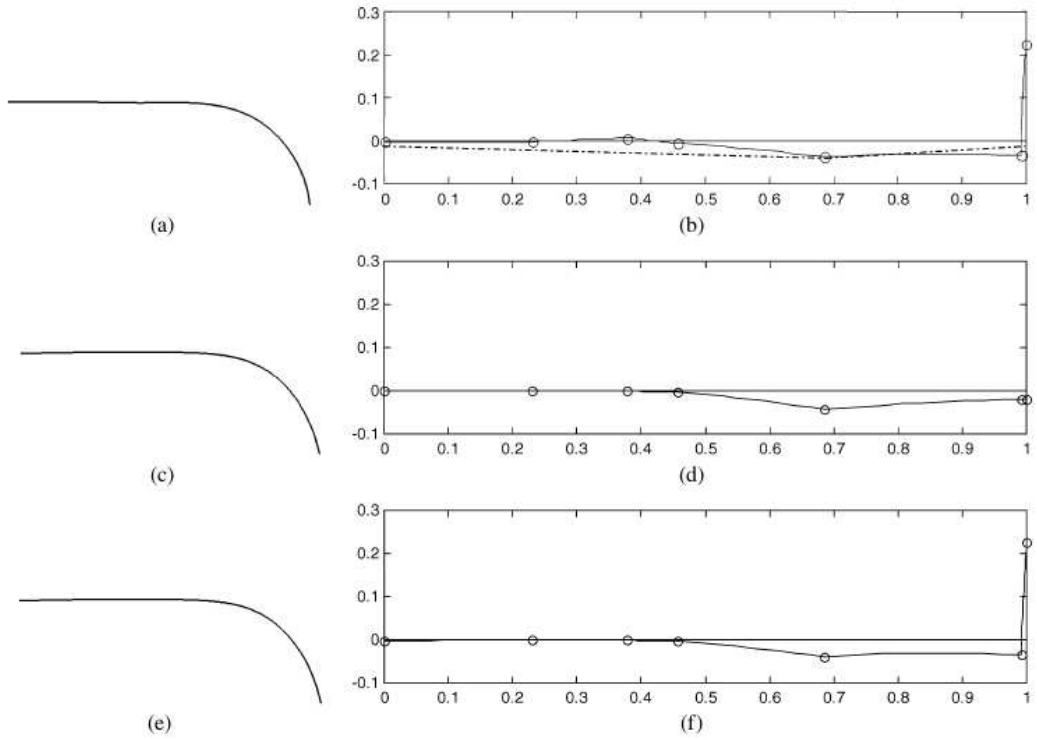


Fig. 11. The given curve (a) and its curvature plot (b), the curve faired by our algorithm (c) and its curvature plot (d), the curve faired by *Sapidis–Farin* algorithm (e) and its curvature plot (f).

FIGURE 16: Modification des points de contrôle d'une courbe donnée [9].

tion d'une courbe monotone.
Sapidis et Frey [11] ont énoncé une condition nécessaire et suffisante pour les courbes de Bézier quadratiques. Cette condition est basée sur le positionnement et la symétrie des points de contrôle.
Mineur *et al.* [10] proposent, à partir des travaux de Sapidis et Frey, des conditions sur les angles d'une courbe de Bézier au premier et au dernier point afin d'obtenir la monotonie de la courbure (Figure 17).
Ensuite Wang *et al.* [12] ont proposé une condition suffisante pour les courbes de Bézier ainsi que les B-splines.

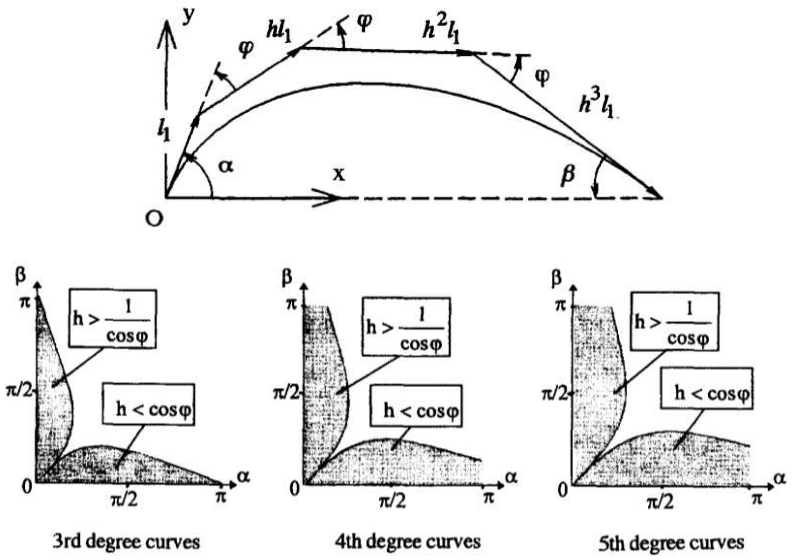


Fig. 2. Regions for a monotonic curvature variation.

FIGURE 17: Conditions de monotonie [10].

Les B-splines de degré n sur chaque séquence $t = (t_0, \ldots, t_{m-1})$ sont de la forme :

$$S(t) = \sum_{i=0}^{m-n-2} \boldsymbol{P}_i b_{i,n}(t).$$

Par l'identité de de Boor-Fix, les fonctions de base des B-splines $b_{i,n}(t)$ se réécrivent pour $t \in [t_j, t_{j+1}]$:

$$b_{i,n}(t) = \sum_{l=0}^{n} \beta_{l,j}^{i} B_n^l(t),$$

où $B_n^l(t)$ sont les polynômes de Bernstein de degré n.
En dérivant cette forme, la fonction de courbure plane $\lambda_j(t)$ se réécrit :

$$\lambda_j(t) = \sum_{0}^{4n-6} \varepsilon_{s,j} B_{4n-6}^{s}(t).$$

Le théorème concernant la condition suffisante de la monotonie de la courbure d'une B-Spline est alors :

Théorème
Le polynôme de courbure (4.2.2) *est monotone s'il satisfait l'un des critères suivants :*
— $\varepsilon_{i,j} \geq 0$ *pour i=0, . . . , 4n-6*
— $\varepsilon_{i,j} \leq 0$ *pour i=0, . . . , 4n-6*
avec $\exists\ \varepsilon_{i,j} \neq 0$ *pour i=0,. . . ,4n-6.*

Wang *et al.* ont alors proposé un algorithme permettant de modifier une B-Spline afin de la rendre de courbure monotone (implémenté dans le cas DONPL2 de Spellucci).
Pour cela, les dérivées des fonctions par morceaux sont calculées, puis les coefficients ε_l où $l = 0, \ldots, 4n-6$ sont dérivés :

$$\frac{\partial \varepsilon_l}{\partial x_i} = n^4(n-1)\Big((n-2)\chi_{1,x_i} - 3(n-1)\chi_{2,x_i}\Big)$$

où χ_{1,x_i} et χ_{2,x_i} sont des fonctions dépendantes des points de contrôle.

Enfin, les gradients sont contraints. La courbe ainsi générée est de courbure monotone. Les corrections se font sur les points de contrôle.

Cet algorithme est rapide et donne des résultats satisfaisant. Il est aussi utilisé dans de nombreux domaines, tels que la biologie.

5 La qualité des systèmes de représentation

Les trois systèmes de représentation présentés en Section 3 permettent de construire des fonctions régulières. De plus, la Section 4 montre qu'il est possible d'obtenir la monotonie des fonctions créées. Ce dernier point, combiné avec le fait que les trois méthodes de conception permettent d'imposer $f(0) = 0$ et $f(1) = 1$, assure la bijectivité des fonctions f.

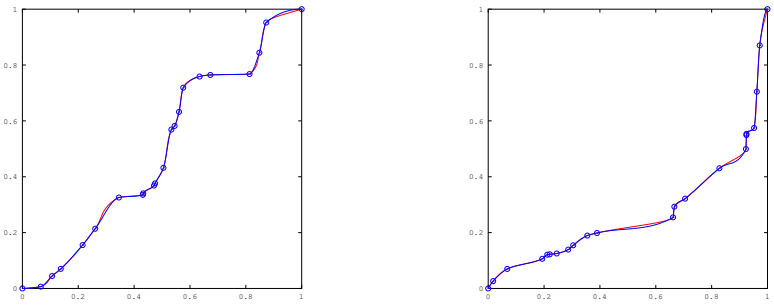
Cependant, s'assurer que les fonctions f possèdent les propriétés voulues en laissant à l'utilisateur, ou à l'algorithme, une grande liberté de choix pour ces fonctions n'est pas suffisant. Il faut en plus que les opérations fréquemment effectuées sur les fonctions f puissent être effectuées simplement avec peu ou pas d'erreur tout en restant dans le même système de représentation. Il est donc nécessaire d'étudier le comportement de chaque système de représentation par rapport aux opérations d'inversion et de composition. De plus, la fonction x^γ étant très fréquemment utilisée en traitement d'image, l'erreur commise par son approximation dans chacun des systèmes sera analysée.

5.1 Inverse

5.1.1 Interpolation par splines cubiques

Une première approche, pour l'inversion d'une interpolation par splines cubiques, serait d'inverser géométriquement, par symétrie axiale par rapport à la première bissectrice, les points de contrôle puis de réaliser une interpolation par splines cubiques à partir de ces nouveaux points de contrôle. Deux exemples d'inverse calculés selon cette méthode sont présentés à la Figure 18.

Visuellement, l'inverse exacte de la courbe interpolante, en bleu, et l'interpolation des points inversés, en rouge, sont très proches.



(a) (b)

FIGURE 18: Interpolation de l'inverse des points d'interpolation.

Une autre solution, qui ne sera pas détaillée ici, serait de calculer l'inverse exacte de la spline cubique et de placer des points de contrôle sur celle-ci en fonction de sa pente, puis d'effectuer une nouvelle interpolation à partir de ces points.

5.1.2 Courbes de Bézier et B-splines

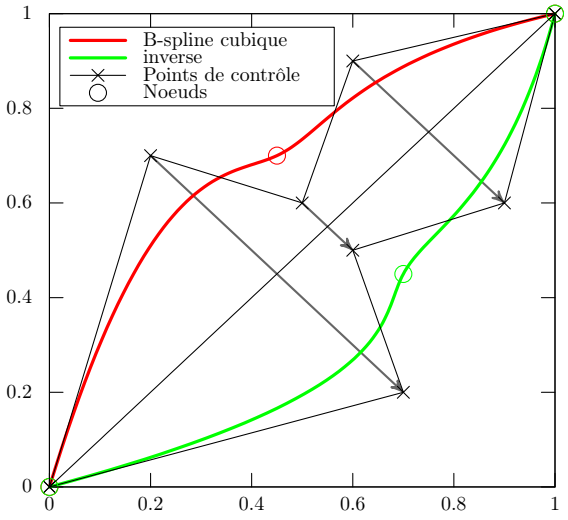


FIGURE 19: Exemple d'obtention de l'inverse.

La courbe B-spline inverse s'obtient simplement comme une courbe B-spline définie sur l'inverse des points de contôle comme illustré sur la Figure 19. Sur cette figure, la courbe verte est une courbe B-spline définie sur l'inverse des points de contrôle de la courbe rouge. L'inverse

est obtenue géométriquement par symétrie par rapport à la première bissectrice.

$$\boldsymbol{P}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}, \quad \boldsymbol{P}_i^{-1} = \begin{bmatrix} y_i \\ x_i \end{bmatrix}.$$

Comme les B-splines sont une généralisation des courbes de Bézier, l'inverse d'une courbe de Bézier s'obtient de la même manière.

5.2 Composition

5.2.1 Interpolation par splines cubiques

Le comportement de l'interpolation par splines cubiques vis à vis de l'opération de composition ne sera pas traitée en détail dans ce rapport. Mais, une première idée pour calculer $f_1 \circ f_2$ serait de calculer exactement la nouvelle position $\boldsymbol{P}_i^{2'}$ des points de contrôle $\boldsymbol{P}_i^2$ de la courbe représentative de f_2 par la fonction f_1 :

$$\boldsymbol{P}_i^{2'} = f_1(\boldsymbol{P}_i^2),$$

à partir de l'expression analytique de f_1, puis de réaliser une interpolation par splines cubiques sur ces nouveaux points de contrôle.

Une seconde idée serait de calculer la composée de manière exacte, puis de choisir sur cette courbe un certain nombre de points de contrôle et de réaliser une interpolation par splines cubiques à partir de ces points.

5.2.2 Courbes de Bézier et B-splines

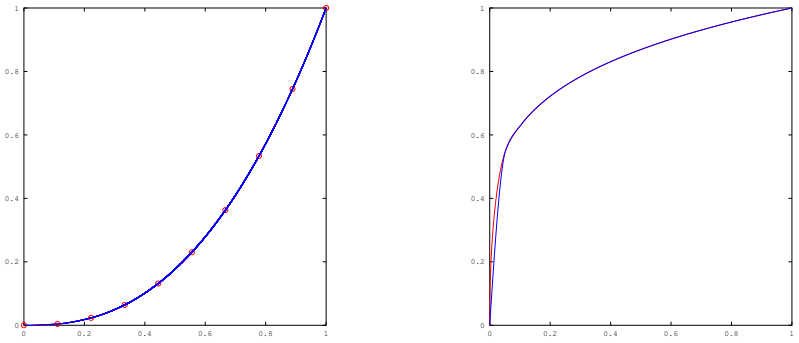
La composition de B-splines semble réalisable, [6], comme extension de la composition de courbes de Bézier, [5].

5.3 Représentation de la fonction gamma

5.3.1 Interpolation par splines cubiques

Comme les interpolations par splines cubiques passent par leur points de contrôle il est assez simple de représenter la fonction gamma. Il suffit de choisir des points de contrôle sur la courbe représentative de la fonction gamma voulue. Plus il y aura de points de contrôle, meilleure sera l'approximation de la fonction gamma. La Figure 20b montre la différence entre deux fonctions gamma, en bleu, et leur interpolation par splines cubiques, en rouge. Pour chacune des courbes dix points de contrôle répartis uniformément entre 0 et 1 ont été utilisés. L'approximation est meilleure pour $\gamma = 2.5$ car la pente de la courbe n'est jamais trop importante. En revanche, pour les faibles intensités, l'erreur entre les deux courbes est visible pour $\gamma = 0.45$.

Cependant, en changeant la répartition des points de contrôle, il est possible de réduire cette erreur. La Figure 21 montre qu'une répartition en x^2 diminue grandement l'erreur d'approximation.



(a) $\gamma = 2.5$ (b) $\gamma = 0.45$

FIGURE 20: Comparaison de l'interpolation par splines cubiques de deux fonctions gamma. Les deux interpolations utilisent dix points de contrôle uniformément répartis entre 0 et 1.

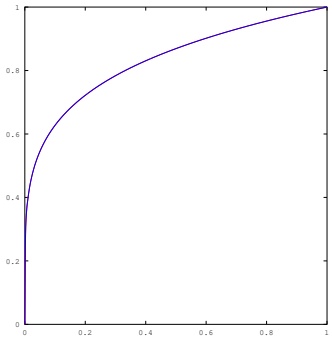


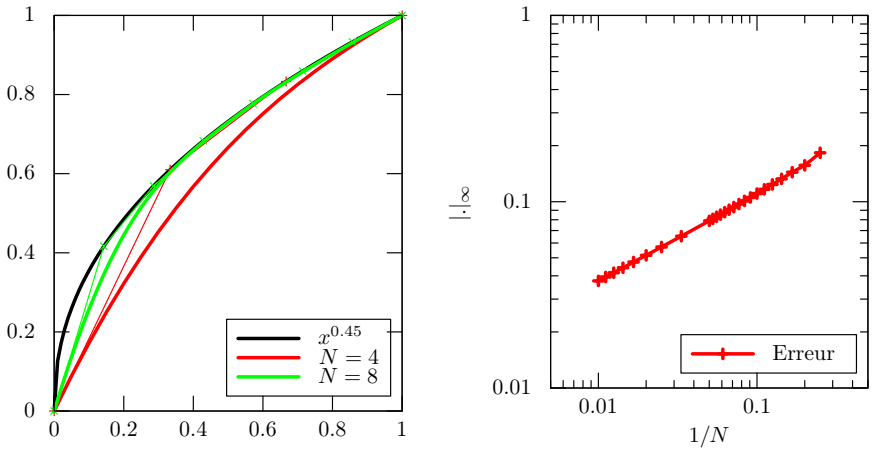
FIGURE 21: Interpolation par splines cubiques de la fonction gamma avec $\gamma = 0.45$ utilisant dix points de contrôle répartis suivant x^2.

5.3.2 Courbes de Bézier et B-splines

Les courbes de Bézier et les B-splines sont regroupées dans cette section aussi car la représentation de la fonction x^γ est équivalente dans les deux cas. Une telle fonction ne peut pas être directement obtenue. Une première idée est d'approcher cette fonction en définissant une suite de points de contrôle (x_n, y_n) telle que $y_n = x_n^\gamma$. La courbe générée ne peut pas être égale à la fonction x^γ car elle ne passe pas par les points de contrôle, néanmoins, l'augmentation du nombre de points permet de s'en approcher. Soit U_N la suite de $N+1$ points uniformément répartis sur $[0;1]$ définis par :

$$U_n = \left(\frac{n}{N+1}, \left(\frac{n}{N+1}\right)^\gamma\right).$$

La Figure 22a donne deux exemples d'approximation de la fonction $x^{0.45}$ utilisant les suites U_4 et U_8. Comme le montre la Figure 22b, la méthode converge. Cependant la convergence est assez lente. Il faudrait donc un très grand nombre de points pour définir une courbe B-spline comme bonne approximation de x^γ. Il est possible qu'un meilleur choix pour la suite des points



(a) Exemples de courbes (b) Erreur d'approximation

FIGURE 22: Approximation de la fonction x^γ avec des B-splines cubiques.

de contrôle permette d'obtenir une meilleure convergence.

Une seconde idée, qui ne sera pas traitée en détail, serait d'utiliser un algorithme d'optimisation minimisant l'erreur entre la courbe et la fonction x^γ en jouant sur le nombre et la position des points de contrôle. A titre indicatif, la Figure 23 montre une approximation réalisée en positionnant à la main les points de contrôle. L'erreur obtenue est déjà inférieure à la moitié de l'erreur obtenue avec la suite U_{100} (0.016 au lieu de 0.039). Donc, lorsque la courbe est consue par un utilisateur et non un algorithme, il est envisageable d'afficher la courbe représentative de la fonction x^γ afin de guider celui-ci.

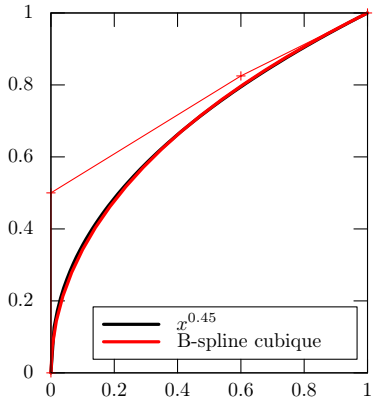


FIGURE 23: Exemple d'approximation à la main.

6 Conclusion

Au terme de cette étude, nous avons pu dégager différentes pistes pour la représentation des fonctions de réponse radiométrique. Ces pistes ont été explorées selon les aspects mathématiques issus du problème initial. La compréhension et la formalisation du problème de manière mathématique étaient déjà un problème en soi et pourrait être affinée par rapport à certains points qui nous auraient échappé. L'étude a permis de dégager trois outils potentiels pouvant répondre au problème. Même si tous ces outils semblent envisageables, nous attirons l'attention du lecteur sur la simplicité d'utilisation des courbes B-splines. En effet, la manipulation de ces courbes est assez intuitive et la prise en main de l'outil se réalise en quelques essais.

Pour tous les outils envisagés, la composition par une fonction puissance non entière du fait de sa nature non polynomiale reste un problème car les outils se basent sur des polynômes. Cependant, il est possible d'obtenir une bonne approximation de cette fonction.

Références

[1] GNU Image Manipulation Program.

[2] Wolfgang Boehm. Inserting new knots into b-spline curves. *Computer-Aided Design*, 12(4) :199–201, 1980.

[3] Elaine Cohen, Tom Lyche, and Richard Riesenfeld. Discrete B-splines and subdivision techniques in computer-aided geometric design and computer graphics. *Computer graphics and image processing*, 14(2) :87–111, 1980.

[4] Carl De Boor. *A practical guide to splines*. Springer-Verlag New York, 1978.

[5] Tony D DeRose. Composing bézier simplexes. *ACM Transactions on Graphics (TOG)*, 7(3) :198–221, 1988.

[6] Tony D DeRose, Ronald N Goldman, Hans Hagen, and Stephen Mann. Functional composition algorithms via blossoming. *ACM Transactions on Graphics (TOG)*, 12(2) :113–135, 1993.

[7] G. Farin and N. Sapidis. Curvature and the fairness of curves and surfaces. *Computer Graphics and Applications, IEEE*, 9(2) :52–57, 1989.

[8] F. Fritsch and R. Carlson. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2) :238–246, 1980.

[9] Weishi Li, Shuhong Xu, Jianmin Zheng, and Gang Zhao. Target curvature driven fairing algorithm for planar cubic b-spline curves. *Computer Aided Geometric Design*, 21(5) :499 – 513, 2004.

[10] Yves Mineur, Tony Lichah, Jean Marie Castelain, and Henri Giaume. A shape controled fitting method for bézier curves. *Computer Aided Geometric Design*, 15(9) :879 – 891, 1998.

[11] Nickolas S. Sapidis and William H. Frey. Controlling the curvature of a quadratic bézier curve. *Computer Aided Geometric Design*, 9(2) :85 – 91, 1992.

[12] Yulin Wang, Bingyan Zhao, Luzou Zhang, Jiachuan Xu, Kanchang Wang, and Shuchun Wang. Designing fair curves using monotone curvature pieces. *Computer Aided Geometric Design*, 21(5) :515 – 527, 2004.