Structuration de nuages de points issus d'acquisitions LiDAR multiples à l'aide de graphes locaux connectés

Arnaud Bletterer, Frédéric Payan, Marc Antonini

▶ **To cite this version:**

Arnaud Bletterer, Frédéric Payan, Marc Antonini. Structuration de nuages de points issus d'acquisitions LiDAR multiples à l'aide de graphes locaux connectés. Groupe de Recherche et d'Etudes du Traitement du Signal et des Images (GRETSI), Aug 2019, Lille, France. hal-02147755v2

HAL Id: hal-02147755
https://hal.science/hal-02147755v2

Submitted on 12 Jun 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Structuration de nuages de points issus d'acquisitions LiDAR multiples à l'aide de graphes locaux connectés

Arnaud BLETTERER, Frédéric PAYAN, Marc ANTONINI

Université Côte d'Azur, CNRS, I3S
2000, Route des lucioles 06903 Sophia Antipolis
{arnaud.bletterer, frederic.payan, marc.antonini}@univ-cotedazur.fr

Résumé – Nous montrons comment une représentation basée sur des graphes locaux permet de structurer des nuages de points générés par agrégation d'acquisitions LiDAR. Ces graphes sont définis sur les cartes de profondeur générées par les capteurs, puis connectés pour obtenir une représentation unique et globale du site numérisé. Les résultats expérimentaux montrent que cette structure permet de manipuler et de traiter des nuages de points gigantesques même s'ils dépassent largement les capacités mémoire de la machine sur laquelle ces calculs sont effectués, et ce quel que soit le nombre d'acquisitions ou de points capturés par acquisition.

Abstract – We show how a local graph-based representation can be used for structuring 3D point clouds provided by multiple LiDAR acquisitions. Those graphs are defined from the depth maps generated by the sensors, and connected together to get a single and global representation of the scanned site. Experimental results show that this structure is particularly suitable for processing gigantic points clouds, even if they largely exceed the memory capacity of the machine on which those processings are performed, whatever the number of points and of acquisitions.

1 Introduction

Certains scanners sont capables aujourd'hui d'acquérir des centaines de millions de points en une seule acquisition. Lors de certaines campagnes sur des sites étendus, des centaines d'acquisitions sont nécessaires, et leur agrégation produit des nuages contenant des milliards de points. Ces données sont particulièrement difficiles à traiter pour plusieurs raisons : elles sont volumineuses (des centaines de gigaoctets), et dépassent donc la capacité mémoire de la plupart des machines actuelles. Elles sont non-structurées, ce qui signifie qu'elles rendent difficile l'inférence de la topologie de la surface ayant été numérisée. Ce problème complexifie de nombreux traitements car la notion de voisinage (fournie par la connectivité des sommets dans un maillage surfacique par exemple) est absente. Enfin, elles peuvent présenter de nombreuses *zones de recouvrement*, des zones capturées par plusieurs acquisitions. Pour certaines applications, ces zones nécessitent d'être "nettoyées" afin d'éliminer la redondance d'information. Ces particularités limitent drastiquement l'usage direct de tels nuages de points : ceux-ci nécessitent d'être préalablement structurés.

La méthode la plus répandue pour structurer un nuage de points est d'utiliser un arbre de partitionnement volumique. Efficace pour compresser ou visualiser les données [4, 6], cette structure peut s'avérer inadaptée pour certains traitements car elle fournit une structure de l'espace ambiant et non pas de la surface sous-jacente au nuage de points. Récemment, une structure basée graphe [3] a été proposée comme alternative pour décrire un nuage de points. Cette structure a l'avantage d'ajouter une connectivité à la scène numérisée et facilite donc les traitements faits en aval sur la surface.

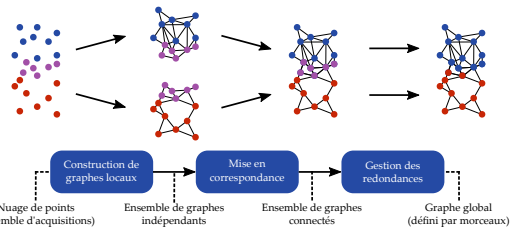
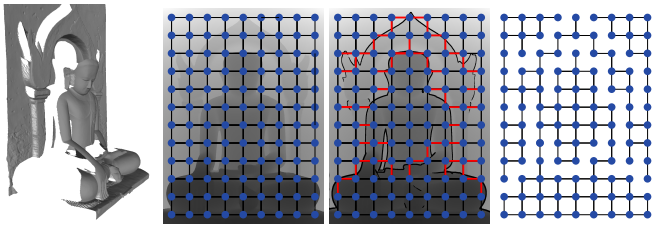


FIGURE 1 – Méthode proposée pour structurer un nuage de points à l'aide de graphes locaux.

Aujourd'hui, nous proposons une alternative à [3] pour structurer à l'aide de graphes les nuages de points issus de l'agrégation d'acquisitions LiDAR. La figure 1 présente notre méthode dans ses grandes lignes. Au lieu de traiter le nuage de points 3D dans son ensemble, nous construisons un graphe par acquisition, à partir de la carte de profondeur associée à celle-ci. Chaque graphe fournit une représentation locale structurée d'une partie de la surface numérisée. Ensuite, pour traiter les données de manière globalement cohérente, mais aussi pour gérer les redondances, nous connectons ces graphes en mettant en correspondance les points présents dans les zones de recouvrement (en violet sur la figure 1). Comme nous le verrons dans la section expérimentale, cette structure permet de traiter les nuages de points par morceaux, et va nous permettre de manipuler de manière efficace les surfaces représentées par des nuages de points de manière *out-of-core*, même sur des machines ayant une capacité mémoire limitée.

Ce travail a été financé par une bourse doctorale de la *Région Sud*.



(a) Nuage 3D. (b) Carte et graphe initial. (c) Segmentation. (d) Graphe final.

FIGURE 2 – Construction d'un graphe à partir d'une carte de profondeur (dans cet exemple, tous les pixels ont une intensité non nulle, ils sont donc tous présents dans le graphe initial).

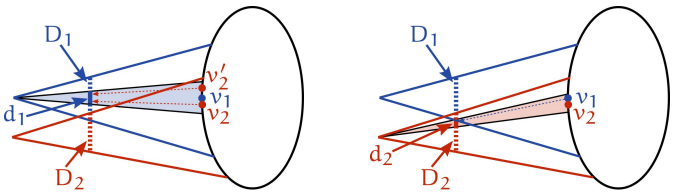
2 Méthodologie

On considère comme données d'entrée un ensemble de N cartes de profondeur, toutes recalées dans un même repère global. Cet ensemble peut, par exemple, être issu d'une campagne d'acquisitions LiDAR effectuées à différents endroits sur un site, ou de manière synthétique à partir de modèles 3D [7].

Construction des graphes locaux La figure 2 présente les principales étapes de notre méthode pour construire un graphe à partir d'une carte de profondeur. Lors d'une acquisition, un LiDAR génère une carte de profondeur D qui peut être vue comme un domaine de paramétrisation d'un morceau de la scène numérisée S (fig. 2(a)), fournissant ainsi une connectivité aux points 3D. Le passage de la carte à l'ensemble des points 3D se fait à l'aide d'une fonction de projection. Un premier graphe $G = (V, E)$ peut être construit : les sommets V sont associés aux pixels non-nuls de la carte (les pixels d'intensité nulle sont des points fuyants), et les arêtes E relient les pixels voisins (fig. 2(b), avec un 4-voisinage dans cet exemple). Pour modéliser précisément la surface capturée, les sommets voisins qui ne sont pas associés à des points 3D géodésiquement proches sur la surface doivent être dissociés. En effet, une fois "projetés" sur la carte, des éléments peuvent sembler voisins, bien qu'éloignés dans la réalité. Pour détecter ces configurations, nous avons utilisé la méthode des gradients morphologiques [8] pour seuiller les fortes variations d'intensité dans la carte qui correspondent à de fortes distances sur la surface (fig. 2(c)). La taille de l'élément structurant permet de directement contrôler la finesse de la détection. Les arêtes associées à ces forts gradients sont ensuite supprimées (fig. 2(d)). En faisant cela sur chaque carte, nous générons un ensemble de N graphes $\{G_1, G_2, ...G_N\}$, locaux et indépendants.

Mise en correspondance des graphes Afin de permettre par la suite des traitements globalement cohérents sur l'ensemble du nuage, nous avons choisi de *connecter* les parties des graphes décrivant les zones de recouvrement. Plutôt que de fusionner les différents graphes locaux sous la forme d'un unique graphe global, nous connectons ces graphes de manière explicite. Cela permet de garder le contrôle sur la taille des données en mémoire, quel que soit le nombre de points ainsi que le nombre d'acquisitions.

Ainsi, pour un graphe donné, chaque sommet possède une



(a) $D2 \rightarrow D1$. (b) $D1 \rightarrow D2$.

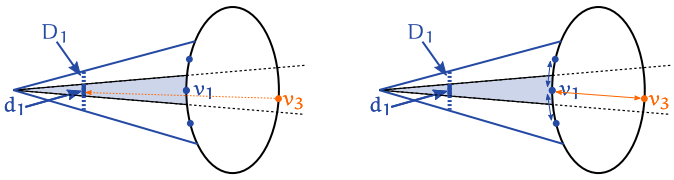
FIGURE 3 – Mise en correspondance de points entre graphes.

liste de sommets correspondants contenant l'ensemble des sommets des autres graphes qui représentent le même point dans l'espace 3D (plus précisément le point 3D le plus proche, en tenant compte de la discrétisation des domaines de paramétrisation initiaux). Ces listes sont stockées comme information additionnelle aux sommets des graphes. Elles permettront par la suite de transférer des résultats d'opérations ou des informations d'un graphe à l'autre, afin d'éviter des traitements redondants sur plusieurs graphes décrivant la même zone. Dans les zones de non recouvrement, ces listes resteront vides.

Le principe est illustré par un cas d'école (fig. 3), où l'on met en correspondance les sommets associés aux cartes de profondeur D_1 et D_2. On considère tout d'abord le sommet du graphe G_1 associé au pixel d_1 (fig. 4(a)). Ce pixel, par sa fonction de projection, est associé au point v_1 dans l'espace 3D. Si on considère maintenant les points capturés par D_2, on observe que les points v_2 et v'_2 seraient eux aussi capturés par ce même pixel d_1 dans D_1. On en déduit qu'ils représentent le même échantillon sur la surface : les sommets associés à v_2 et v'_2 sont donc des sommets correspondants au sommet associé au point v_1. Inversement, et selon le même principe, v_1 est un sommet correspondant de v_2 dans le graphe G_2 (fig. 4(b)).

Cette approche n'est pas suffisante pour gérer les occultations, comme l'illustre la figure 4. Dans cet exemple très simple, le sommet de G_3 associé au point v_3 capturé par une carte D_3 (non représentée sur la figure) sera en effet considéré comme un sommet correspondant au sommet associé à $v1$, puisqu'en utilisant les fonctions de projection des cartes D_1 et D_3, il va se projeter lui aussi sur d_1 bien que ne représentant pas la même zone numérisée. Pour éviter cela, nous ajoutons une simple fonction qui calcule l'espacement moyen entre v_1 et ses voisins directs issus de la carte de profondeur D_1 : si cet espacement n'est pas du même ordre de grandeur que la distance entre v_1 et v_3, on considère que v_3 n'appartient pas au même morceau de surface et ne doit donc pas être mis en correspondance avec v_1. Cette technique, bien que perfectible, nous permet d'éviter les fausses mises en correspondance de la plupart des données.

Gestion des redondances Les zones de recouvrement nécessitent une attention particulière, notamment pour éviter des calculs redondants sur plusieurs graphes. De plus, la précision de certaines fonctions dépend fortement de la densité d'échantillonnage des graphes sur lesquels les calculs sont appliqués. Nous avons donc décidé de distinguer pour chaque graphe 2 sous-ensembles de sommets : V^+, l'ensemble des sommets de ce graphe ayant une plus grande densité que leurs sommets cor-



(a) $D3 \rightarrow D1$. (b) Solution retenue.

FIGURE 4 – Gestion des occultations : bien que projeté sur d_1, v_3 ne doit pas être associé à v_1.

respondants dans les autres graphes, et $V^- = V \setminus V^+$, les autres sommets de ce graphe. Pour cela, nous avons utilisé la densité d'échantillonnage locale de chaque sommet $w(v)$, tel que : $\frac{1}{w(v)} = \frac{1}{|\mathcal{N}_k(v)|} \sum_{v' \in \mathcal{N}_k(v)} ||v' - v||^2$, $\mathcal{N}_k(v)$ étant le k voisinage du sommet v dans le graphe dont il est issu. Ainsi, dans les zones de recouvrement, pour un graphe donné, nous effectuerons un maximum d'opérations sur l'ensemble V^+, et les autres sommets de ce graphe (V^-) récupèreront le résultat des fonctions auprès de leurs sommets correspondants.

Gestion par morceaux d'une acquisition massive Une seule acquisition peut capturer plusieurs centaines de millions de points. Dans ce cas, un unique graphe peut déja dépasser les capacités mémoire de nombreuses machines. Pour être capable de traiter des ensembles d'acquisitions quel que soit le nombre de points capturés lors de chaque acquisition, notre méthode intègre la possibilité de découper *a priori* une carte de profondeur en tuiles avec chevauchements. Un tel ensemble de tuiles peut alors être vu comme un ensemble d'acquisitions : un graphe est construit sur chaque tuile, et la mise en correspondance se fait implicitement dans les zones de chevauchement (qui sont, par construction, parfaitement connues), de la même manière que dans les zones de recouvrement entre cartes. En jouant sur la taille des tuiles, on peut donc structurer n'importe quel nuage de points sur n'importe quelle machine, la limite de consommation mémoire étant fixée par la taille de la plus grande tuile.

3 Expérimentations

Calcul de géodésiques Pour valider notre structure, nous vérifions tout d'abord si celle-ci permet d'effectuer des calculs précis de géodésiques sur la surface sous-jacente de nuages de points. Nous avons adapté l'algorithme du plus court chemin de Dijkstra, pour que le calcul puisse se faire à travers les différents graphes connectés de notre structure. Nous avons comparé ces résultats à ceux obtenus avec [5], qui permet de calculer de manière exacte des géodésiques sur la surface de maillages polygonaux. Nous avons donc, pour cela, généré des cartes de profondeur synthétiques à partir de modèles 3D, comme expliqué au début de la section 2. La figure 5 propose une comparaison visuelle des deux méthodes. Les géodésiques calculées sur notre structure sont similaires aux géodésiques calculées sur la surface. La différence, mineure, vient de l'utilisation de l'algorithme de Dijkstra, qui construit des géodésiques le long des arêtes du graphe, et ne peut donc être aussi précis que [5]. Néanmoins, pour de nombreux traitements, cette erreur n'aura que peu d'influence. Dans un contexte de rééchantillonnage ou de reconstruction par exemple (voir plus bas), les distances calculées entre points voisins seront généralement courtes par rapport à la taille des nuages, le biais induit est donc négligeable.

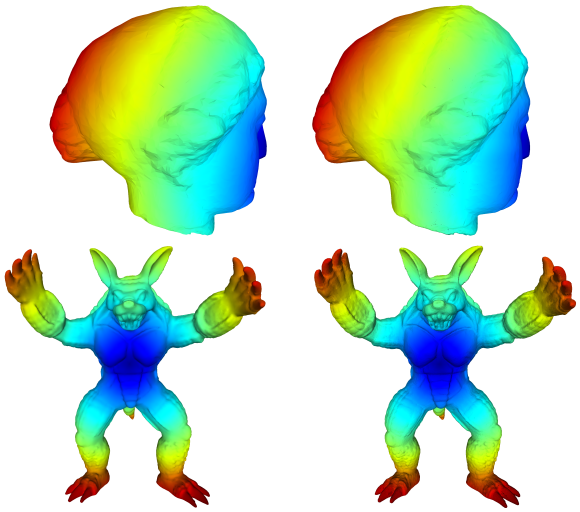


FIGURE 5 – Géodésiques calculées à partir d'un même et unique point vers tous les autres sur la surface d'un maillage (à gauche) et sur notre structure de graphes locaux (à droite). Bleu/rouge = faible/forte distance.

Pour attester que le découpage en tuiles de certaines cartes n'introduit aucun biais lors d'un calcul de géodésiques, la figure 6 présente le résultat d'un calcul sur une carte avec ou sans découpage. La mise en correspondance des sommets entre les tuiles étant parfaite, les géodésiques calculées sont exactement les mêmes. Grâce à cette gestion par tuiles, nous avons un contrôle direct sur les pics de consommation mémoire.

Consommation mémoire Nous analysons maintenant la consommation mémoire de notre algorithme. Les données testées sont issues d'acquisitions réelles fournies par *CyArk/Google Open Heritage Program* et *Art Graphique et Patrimoine*. Nos résultats ont été générés avec un Intel Core i7 @ 3.00GHz, avec 32Go RAM et 1To HDD. La table 1 présente le pic mémoire

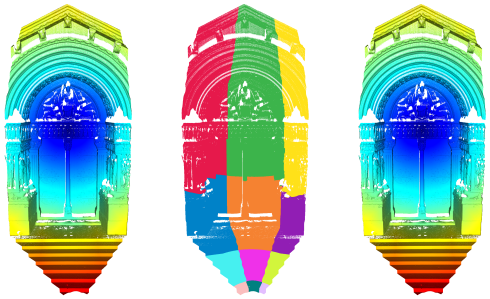


FIGURE 6 – Géodésique calculée sur un graphe local (à gauche) et sur un ensemble de graphes (à droite). Au milieu, les sommets V^+ des graphes associés aux tuiles sont identifiés. Donnée *St Trophime*.

Nom Points (M) / Acquisitions	Pic mémoire (Gio)	Facteur de Réduction
Meeting House 1 493 / 50	5,7	24,6
Ananda Oak Kyaung 1 703 / 126	1,9	83,5
Wat Phra Si Sanphet 5 313 / 177	5,7	112,6

TABLE 1 – Consommation mémoire de notre méthode de structuration pour différentes campagnes d'acquisition.

atteint lors de la construction de la structure, et le facteur de réduction par rapport au coût binaire global des données. Cela donne une idée du gain par rapport à une méthode basée sur une structure globale, qui nécessiterait de tout stocker en mémoire, avant de traiter le nuage. Nous remarquons la considérable réduction du coût mémoire, d'autant plus marquée que le nombre d'acquisitions est important. Cela nous permet de traiter des nuages *a priori* trop volumineux sur des machines de capacité standard. Par exemple, la figure 7 montre un nuage du *Palais de la Découverte* simplifié avec [1] qui est basée sur la structure proposée. La donnée initiale, composée de 977 millions de points, soit un coût de stockage de 72,15 Gio, a pu être traitée sur notre machine, le pic mémoire ne dépassant pas les 12 Gio. Et malgré cela, la surface reconstruite avec [2] à partir de ce nuage simplifié est très satisfaisante (figs. 7(b) et 7(c)).

(a) Nuage de points simplifié (~977 → ~25 millions de points).

(b) Morceau de la surface reconstruite à partir de ce nuage simplifié.

(c) Vue rapprochée de la triangulation générée sur une autre partie.

FIGURE 7 – *Palais de la Découverte* simplifié à l'aide d'un échantillonnage en disques de Poisson basé sur notre structure [1], puis reconstruit avec [2].

4 Conclusion

Nous avons présenté une nouvelle structure pour manipuler la surface sous-jacente à un nuage de points. Des graphes locaux sont construits à partir de la connectivité des cartes de profondeur issues du processus d'acquisition. Ces cartes sont ensuite connectées afin de naviguer entre les domaines de paramétrisation, afin d'éviter les calculs redondants dans les zones capturées par plusieurs acquisitions. La segmentation naturelle fournie par l'ensemble des acquisitions, combinée à un éventuel découpage en tuiles, permet de limiter la mémoire requise pour générer la structure, mais aussi pour divers traitements faits en aval. Ainsi, il est possible de traiter des nuages contenant des milliards de points sur des ordinateurs ayant des capacités mémoire bien inférieures au volume nécessaire pour représenter ces nuages.

Références

[1] Arnaud Bletterer, Frédéric Payan, Marc Antonini, and Anis Meftah. Towards the reconstruction of wide historical sites : A local graph-based representation to resample gigantic acquisitions. In *16th EUROGRAPHICS Workshop on Graphics and Cultural Heritage (EG GCH)*, pages 1 – 9, Vienna, Austria, November 2018.

[2] Dobrina Boltcheva and Bruno Levy. Surface reconstruction by computing restricted voronoi cells in parallel. *Computer-Aided Design*, 90 :123–134, 2017.

[3] Siheng Chen, Dong Tian, Chen Feng, Anthony Vetro, and Jelena Kovačević. Fast resampling of three-dimensional point clouds via graphs. *IEEE Transactions on Signal Processing*, 66(3) :666–681, 2018.

[4] Jan Elseberg, Dorit Borrmann, and Andreas Nüchter. One billion points in the cloud–an octree for efficient processing of 3d laser scans. *ISPRS Journal of Photogrammetry and Remote Sensing*, 76 :76–88, 2013.

[5] Joseph SB Mitchell, David M Mount, and Christos H Papadimitriou. The discrete geodesic problem. *SIAM Journal on Computing*, 16(4) :647–668, 1987.

[6] Jingliang Peng and C-C Jay Kuo. Geometry-guided progressive lossless 3d mesh coding with octree (ot) decomposition. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 609–616. ACM, 2005.

[7] Nico Pietroni, Marco Tarini, Olga Sorkine, and Denis Zorin. Global parametrization of range image sets. In *ACM Transactions on Graphics (TOG)*, volume 30, page 149. ACM, 2011.

[8] Jean-Francois Rivest, Pierre Soille, and Serge Beucher. Morphological gradients. *J. Electronic Imaging*, 2(4) :326–336, 1993.