REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE SAAD DAHLAB DE BLIDA
FACULTE DES SCIENCES
DEPARTEMENT D'INFORMATIQUE

MEMOIRE DE FIN D'ETUDES

Pour l'obtention

d'un Diplôme de Master en Informatique

Option : Ingénierie des Logiciels

THÈME :

Fusion Multidirectionnelle des Triangulations de Delaunay en 3D

Réalisé par :

M^elle^. DJEBBAR Meriem

M^elle^. KOUDJI Radhia

Soutenu devant :

Mr. BEY Mohamed	CDTA,	Encadreur
Mr. BENDIFALLAH Hassène	CDTA,	Encadreur
Mme. TCHANTCHANE Zahida	CDTA,	Encadreur
Mr. KAMECHE Abdallah Hicham	USDB,	Promoteur
Mme. Toubaline Nesrine	USDB,	Président
Mme. Hadj Henni Malika	USDB,	Examinateur

2018/2019

Résumé :

La reconstruction des modèles numériques 3D avec une bonne précision est une tâche difficile, coûteuse en temps et en espace mémoire. Par conséquent, la solution à ce problème n'est pas triviale. De nombreuses méthodes ont été développées pour tenter d'y apporter une solution. Les nuages de points sont très denses et très volumineux ce qui augmente les temps de traitement et par conséquent les coûts. Pour cela, le nuage de points est subdivisé en plusieurs sous nuages et chaque sous nuage est approximé par des tétraèdres en utilisant la triangulation de Delaunay et le paradigme « Diviser pour Régner ». Par la suite, une stratégie de fusion a été appliquée afin de fusionner les différentes subdivisions de l'objet 3D.

Mots-clés : Fusion, triangulation de Delaunay, Productique, Robotique, fabrication assistée par ordinateur, Conception assistée par Ordinateur, CFAO

الملخص:

تعد إعادة بناء النماذج الرقمية ثلاثية الابعاد بدقة جيدة مهمة صعبة ومكلفة في الوقت ومساحة الذاكرة. لذلك، فان حل هذه المشكلة ليس سهلا، فقد تم تطوير العديد من الطرق لمحاولة إيجاد حل.

السحابة النقطية عبارة عن مجموعة نقاط كثيفة وضخمة جدا مما يزيد من وقت المعالجة وبالتالي التكاليف. لذلك، وجب تقريب شكل السحابة النقطية بواسطة رباعي الأسطح باستخدام تثليث Delaunay ونموذج الانقسام للتملك. تم اقتراح منهجية واستراتيجية دمج لدمج التقسيمات الفرعية المختلفة للكائن ثلاثي الابعاد.

Abstract :

The reconstruction of 3D digital models with good accuracy is a difficult task, costly in time and memory space. Therefore, the solution to this problem is not trivial. Many methods have been developed to try to find a solution.

The point clouds are very dense and very huge which increases the processing time and consequently the costs. For this, the point cloud is approximated by tetrahedra using Delaunay's triangulation and the "Divide and Conquer" paradigm. A methodology and a merge strategy have been proposed to merge the different subdivisions of the 3D object.

Remerciements

Merci à Allah le miséricordieux de sa grâce, source de notre force et courage tout au long de nos études universitaires.

La réalisation de ce mémoire a été possible grâce au concours de plusieurs personnes à qui nous voudrons témoigner toute notre gratitude.

Nous voudrons tout d'abord adresser toute notre reconnaissance et notre profonde gratitude à Mr. BEY, Mr. BENDIFALLAH et Mme. TCHANTCHANE de nous avoir encadrés dans notre mémoire de fin d'études ainsi que pour leurs patiences, leurs disponibilités et surtout leurs judicieux conseils.

Nous remercions également Mr. Kamèche pour ses encouragements.

Nous désirons aussi remercier les professeurs de l'université de Blida, qui nous ont fourni les outils nécessaires à la réussite de nos études universitaires.

Nous tenons à remercier aussi toute l'équipe du CDTA.

Nous remercions les membres du jury pour avoir accepté d'évaluer notre travail.

Nous voudrions exprimer notre sincère reconnaissance envers nos familles pour leur soutien aussi bien moral que financier et pour leurs sacrifices.

Un grand merci aux amis et collègues qui nous ont apporté leurs soutiens moral et intellectuel tout au long de notre démarche.

Nous ne pourrons terminer sans remercier tous ceux qui ont participé de près ou de loin dans l'élaboration de ce projet de fin d'études.

Merci !!

Dédicaces

Je dédie ce travail qui n'aura jamais pu voir le jour sans les soutiens indéfectibles et sans limite de mes chers parents qui ne cessent de me donner avec amour le nécessaire pour que je puisse arriver à ce que je suis aujourd'hui.

Je dédie également ce modeste travail :

~ A mes très chères sœurs pour m'avoir soutenu et aidé tout au long de ce projet.

~ A toute ma famille : mes grands-parents, mes tantes et mes cousines, en particulier Selma, Khadidja et Hadjer.

~ A la personne qui m'a soutenu toute l'année, mon binôme et sœur de cœur « Radia », merci.

~ A tous les membres de l'équipe CFAO.

~ A tous mes camarades et équipe du CDTA en particulier : Ahmed, Amir, Asma, Fadwa, Hafsa, Kamilia, Latifa et Rym.

~ A mes amis :Assia, Khalil, Meroua, Nesrine, Rania, Samira, Seif Eddine, et Sidali.

~ Enfin, a tous les étudiants de la promotion 2018 / 2019 IL.

Meriem

Dédicaces

Je dédie ce travail qui n'aura jamais pu voir le jour sans les soutiens indéfectibles et sans limite de mes chers parents qui ne cessent de me donner avec amour le nécessaire pour que je puisse arriver à ce que je suis aujourd'hui.

Je dédie également ce modeste travail :

~ A mes très chères frères et ma sœur pour m'avoir soutenu et aidé tout au long de ce projet.

~ A la personne qui m'a soutenu toute l'année, mon binôme et sœur de cœur « Meriem », merci.

~ A tous les membres de l'équipe CFAO.

~ A mes amis : Othmane, Dalila, Houria, Asma, Kamilia, Ahmed, Amir, Younes.

~ Enfin, a tous les étudiants de la promotion 2018 / 2019 IL.

Radia

Table des matières

Table des Figures

Chapitre 1 : Généralités et Etat de l'Art

Chapitre 2 : Conception de l'Application et Démarches de Résolution

Chapitre 3 : Tests et validation

Introduction générale

Le développement rapide de l'informatique nous offre de nombreuses nouvelles technologies utiles les unes des autres. Parmi ces technologies, la Conception et la Fabrication Assistées par Ordinateur (CFAO).

La reconstruction des objets 3D est un outil très répondus dans ce domaine et par conséquent, dans le monde industriel. En effet, plusieurs techniques de reconstruction d'objets 3D ne cessent de s'améliorer car ces dernières permettent de modéliser des phénomènes et d'en étudier les caractéristiques en minimisant le recours à l'expérience. A ce titre, de nombreux efforts considérables ont été engagés dans le développement de méthodes numériques efficaces et fiables capables de réaliser des simulations à grande échelle.

Toutefois, la majorité des méthodes de la reconstruction des objets 3D dans l'industrie conduisent à des solutions très coûteuses en termes de temps de calcul et d'espace mémoire nécessaires à la réalisation d'un calcul. En effet, les calculs peuvent durer entre quelques heures et plusieurs jours, voire des semaines.

Problématique

La reconstruction 3D a pour but de générer un modèle numérique de la surface externe d'un objet physique à partir d'un nuage de points dont les coordonnées 3D son acquises par un dispositif de digitalisation. Une telle problématique se retrouve dans de nombreux domaines d'applications tels que le design industriel, aéronautique, automobile, imagerie médicale, systèmes d'information géographiques, etc.

La reconstruction des modèles numériques 3D avec une bonne précision est une tâche ardue, coûteuse en temps. Par conséquent, la solution à ce problème n'est pas évidente. Plusieurs méthodes ont été développées pour tenter d'y apporter une solution. Les nuages de points sont très denses et très volumineux ce qui augmentent les temps de traitement et par conséquent les coûts. Pour cela, le nuage de points est approximé par des tétraèdres en utilisant la triangulation de Delaunay et le paradigme « Diviser pour Régner ».

Objectif du travail

Ce travail s'insère dans le cadre de développement de modules logiciels pour la production des surfaces de formes complexes initiés par l'équipe Conception et Fabrication

Assistées par Ordinateur (CFAO) de la Division Productique et Robotique (DPR) du Centre de Développement des Technologies Avancées (CDTA).

Dans ce projet, nous nous intéressons à la fusion multidirectionnelle des triangulations de Delaunay en 3D. Le travail consiste à proposer une méthodologie permettant la concaténation de la triangulation de Delaunay deux par deux en utilisant le paradigme « Diviser pour Régner ».

Le but de ce travail est la conception, le développement et l'intégration à la plateforme logicielle de production des surfaces complexes développée par l'équipe CFAO du CDTA, d'un module logiciel graphique et interactif sous Windows automatisant le processus de fusion des triangulations de Delaunay en 3D pour générer la triangulation finale de l'objet.

Structuration du mémoire

Le présent mémoire est composé de trois chapitres :

- Le premier chapitre est consacré à l'étude bibliographique concernant la modélisation et la reconstruction d'objet en particulier la triangulation de Delaunay.
- Dans le deuxième chapitre, une modélisation explicite de l'approche adoptée et le développement de tous les algorithmes utilisés est détaillé.
- Le dernier chapitre présente l'implémentation informatique de l'application et les tests de validation des résultats du travail effectué.

Nous terminons ce mémoire par une conclusion générale et des perspectives.

CHAPITRE I

Généralités et Etat de l'Art

Introduction

Depuis des années, les exigences et les différents besoins de la fabrication des pièces mécaniques n'ont cessé d'évoluer. C'est d'ailleurs la cause d'augmentation des solutions liées à la conception, à la fabrication et à la production des composants mécaniques.

Dans le marché mondial intensément concurrentiel d'aujourd'hui, les entreprises productrices sont à la recherche de nouveaux moyens pour réduire les délais de développement de nouveaux produits qui répondent à toutes les attentes des clients. En général, l'entreprise investie dans la Conception et la Fabrication Assistées par Ordinateur, dans le domaine de prototypage rapide et dans une gamme de nouvelles technologies fournissant des avantages commerciaux.

1. Format d'échange de données

Dans un système homogène où la Conception Assistée par Ordinateur et la Fabrication Assistées par Ordinateur sont intégrées dans un seul environnement, l'information passe directement. Mais dans le cas où la CAO et la FAO ne font pas partie du même environnement ou tournent sous deux environnements différents, il est nécessaire d'utiliser des translateurs nommés « Formats d'échange de données » pour assurer la transmission de l'information entre ces deux plateformes et garantir la compréhensibilité [1]. Le format le plus courant dans le domaine de production est le format STL (Figure 1).

1.1. STL (Standard Tessellation Language)

STL (STereoLithography) appelé également « Standard Tessellation Language » est un format de fichier natif du logiciel de CAO créé par 3D Systems. Ce format de fichiers est pris en charge par de nombreux logiciels. Il est largement utilisé pour le prototypage rapide et la fabrication assistée par ordinateur. Les fichiers STL décrivent uniquement la géométrie des surfaces d'un objet tridimensionnel sans aucune représentation de couleur, de texture ou autres attributs communs du modèle CAO [2].

Figure 1. Passage de la CAO à la FAO.

1.2. Caractéristiques du ficher STL

- Décrit la géométrie de la surface d'un objet 3D mais absence d'informations concernant la couleur, la texture ou autres attributs du modèle CAO communs.
- Le fichier STL décrit un objet par sa surface externe qui doit être fermée.
- La surface est composée d'une série de triangles (facettes) (Figure 2).
- Chaque triangle est défini par les coordonnées de ses trois (03) sommets.
- Chaque triangle doit partager deux (02) sommets avec les triangles le juxtaposant.
- Le sommet d'un triangle ne doit pas être sur l'arête d'un autre triangle.
- La densité des facettes triangulaires change selon la géométrie [3].

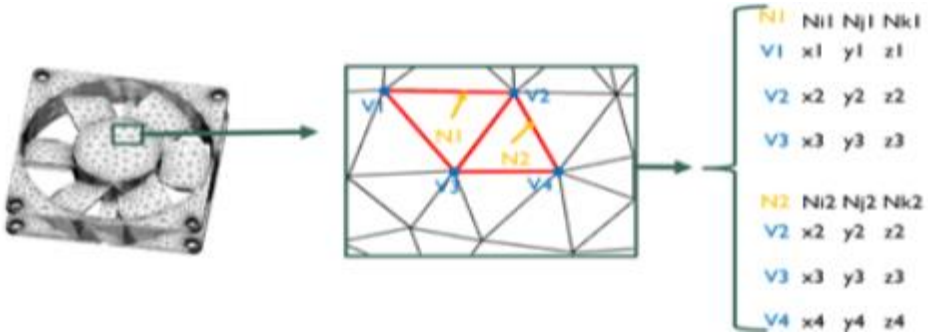


Figure 2. Exemple d'un modèle STL.

1.3. Structure d'un fichier STL

Le format STL spécifie à la fois les représentations ASCII et binaires. Les fichiers binaires sont plus communs car ils sont plus compacts [2].

- **Format ASCII :** un fichier STL ASCII commence par la ligne « solid name » où name est une chaine de caractère. Chaque triangle est défini par les coordonnées de ses sommets et par les composantes de son vecteur normal unitaire. Le fichier se termine par l'indication « endsolid name » (Figure 3).
- **Format binaire :** un fichier STL binaire est architecturé de la façon suivante :
  - Les 80 premiers octets sont un commentaire.
  - Les 4 octets suivants forment un entier codé sur 32 bits représentant le nombre de triangles présents dans le fichier.
  - Ensuite, chaque triangle est codé sur 50 octets, selon la décomposition suivante :

- 3 fois 4 octets, chaque paquet de 4 octets représente un nombre à virgule flottante correspondant respectivement aux composantes x, y et z de la normale du triangle.
- 3 paquets de 3 fois 4 octets, chaque groupe de 4 octets représente un nombre à virgule flottante correspondant respectivement aux coordonnées x, y et z de chacun des sommets du triangle.
- Deux octets représentant un octet de contrôle [3].

Solide projet
Facet normal n_i,n_j, n_k
Outer loop
Vertex $v1_x$ $v1_y$ $v1_z$
Vertex $v2_x$ $v2_y$ $v2_z$
Vertex $v3_x$ $v3_y$ $v3_z$
endloop
endfacet
endprojet

Figure 3. Format STL ASCII.

UINT8[80] – Header
UINT32 – Number of triangles
foreach triangle
REAL32[3] – Normal vector
REAL32[3] – Vertex 1
REAL32[3] – Vertex 2
REAL32[3] – Vertex 3
UINT16 – Attribute byte count
end

Figure 4. Format STL binaire.

2. Reverse Engineering

Reverse Engineering ou Ingénierie Inverse est considérée comme l'une des technologies qui fournissent des avantages commerciaux en raccourcissant le cycle de développement du produit. La Figure 5 montre comment l'ingénierie inverse permet les possibilités de fermeture de la boucle entre ce qui est « tel que conçu » et ce qui est « réellement fabriqué ». De gauche vers la droite, on retrouve les outils de Conception Assistée par Ordinateur (CAO), Fabrication Assistée par Ordinateur (FAO) et de prototypage rapide. Le Reverse Engineering permet alors un retour du produit fini vers le modèle CAO (double flèche).

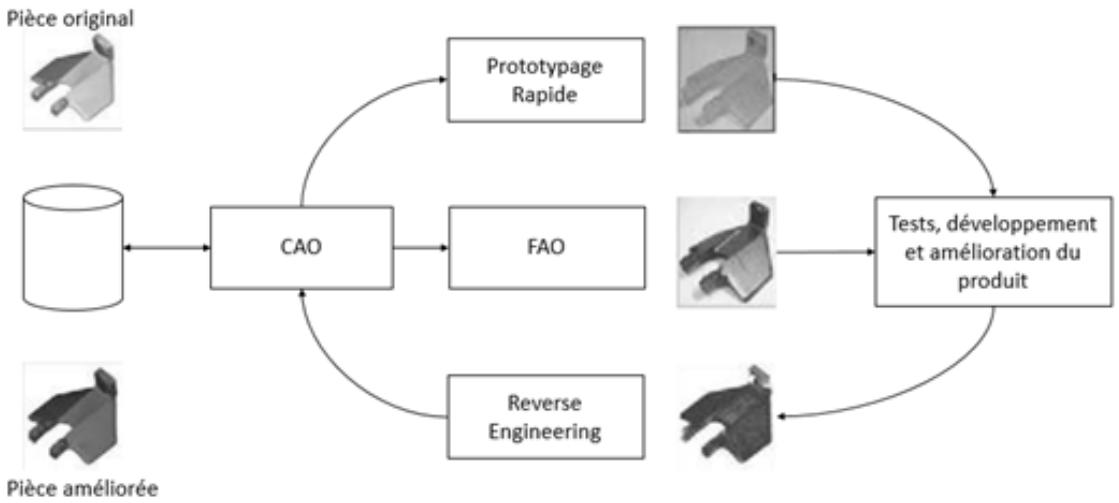


Figure 5. Processus de prototypage rapide et de Reverse Engineering

2.1. Définition de l'Ingénierie Inverse

L'ingénierie, de manière générale, est le processus de modélisation, de fabrication, d'assemblage et de maintenance des produits et des systèmes.

Ils existent deux types d'ingénierie :

- **Forward Engineering (Ingénierie Directe) :** c'est le processus traditionnel de passer d'une abstraction de haut niveau et de conception logique à une implémentation physique d'un système. Dans certaines situations, il peut y avoir une partie ou un produit physique sans aucun détail technique, tels que des dessins, des nomenclatures ou sans données d'ingénierie.
- **Reverse Engineering (Ingénierie Inverse) :** est le processus de duplication d'une pièce existante, d'un sous-assemblage ou d'un produit sans dessins, sans documentation ou sans modèle informatique.

L'ingénierie inverse est également définie comme le processus d'obtention d'un modèle CAO géométrique à partir de points 3D acquis par numérisation des pièces existantes [4].

Selon le domaine d'application, ils existent différentes traductions pour le terme « Reverse Engineering » tels que « Rétro-Conception », « Duplication » ou encore « Rétro-Ingénierie », et plusieurs définitions du Reverse Engineering existent comme :

- Le processus d'extraction des données de conception et de fabrication d'une pièce existante.

- Le concept permettant de fabriquer une pièce à partir de son modèle physique sans utiliser aucun dessin de définition.
- Le processus permettant de récupérer la nouvelle géométrie d'une pièce fabriquée en la numérisant et en modifiant son modèle CAO existant.
- L'analyse d'un produit existant (qui peut être logiciel, une pièce mécanique, etc.) afin de produire une copie ou une version améliorée de celui-ci.

La Figure 6 propose une corrélation entre les outils de conception (CAO) et de rétro-conception (CARE) [4].

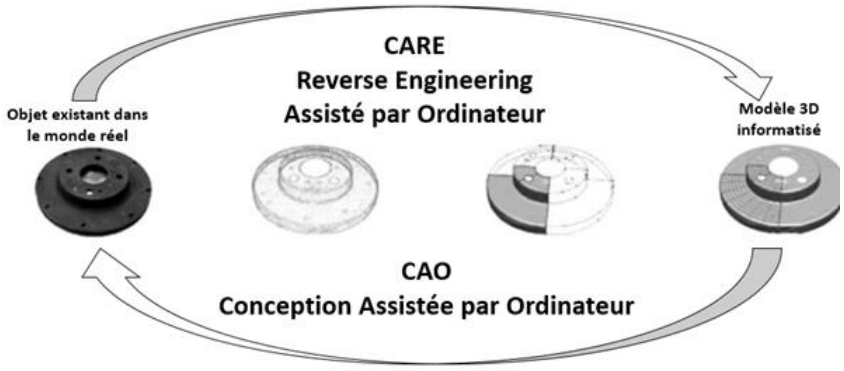


Figure 6. Corrélation entre les outils de conception et de rétro-conception.

2.2. Processus de l'Ingénierie Inverse

Dans la majorité des études, le processus de RE se décompose en trois étapes (Figure 7). Les trois phases sont :

- **Acquisition des données (Scanning) :** est réalisée par l'utilisation de matériels standards pour les données en 2D comme les appareils photo. Cependant notre intérêt se focalise sur les données 3. Ce type de données apporte un niveau d'informations plus haut (géométrie et topologie). Ils existent deux technologies d'acquisition pour les données 3D : avec contact et sans contact.
- **Traitement de données (Traitement de points) :** permet d'extraire des informations pertinentes comme les informations géométriques ou topologiques. Ces informations permettent de reconstruire la maquette numérique. Ensuite, l'obtention des données dans le cas d'assemblage de plusieurs pièces, fournit un ensemble de points ou de pixels pour lesquels, il est impossible de déterminer les limites de chaque composant de manière spontanée. Dans la plupart des cas, on applique alors une segmentation de cette donnée en fonction de son type (2D ou 3D).

- **Reconstruction du modèle CAO (Applications) :** il s'agit d'utiliser les informations extraites de l'étape précédente, afin de reconstruire la géométrie et la topologie de chaque composant. Cette phase étudie la reconstruction de la maquette numérique au-delà du modèle CAO [4].

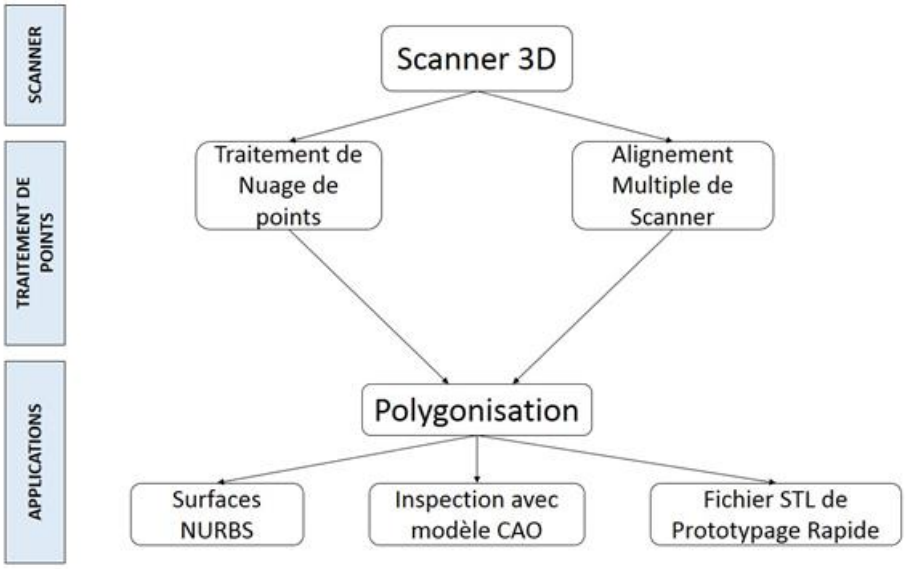


Figure 7. Processus générique du Reverse Engineering.

La stratégie d'Ingénierie Inverse doit prendre en compte les éléments suivants :

- Raison de l'ingénierie inverse d'une pièce.
- Nombre de pièces à numériser (une ou plusieurs).
- Taille de la pièce (grande ou petite).
- Complexité partielle (simple ou complexe).
- Matériau partiel (dur ou mou).
- Finition partielle (brillante ou mate).
- Géométrie partielle (organique ou prismatique et interne ou externe).
- Précision requise (linéaire ou volumétrique) [4].

3. Méthodes de reconstruction des modèles d'objets à partir d'un nuage de points quelconque

La reconstruction est l'approximation d'une courbe ou d'une surface par une représentation analytique ou géométrique. Elle utilise en entrée, un échantillonnage de la courbe ou de la surface sous forme d'ensemble de points et en sortie, on obtient une courbe ou

une surface définie par une représentation analytique ou géométrique. La surface reconstruite peut être approximante ou interpolante.

3.1. Représentation paramétrique

NURBS, comme modèle mathématique, est un des principes de la représentation et de la reconstruction de modèles d'objets, couramment utilisé pour générer et représenter des courbes et des surfaces. Ces derniers offrent une grande flexibilité et une grande précision pour la gestion de formes analytiques et de formes libres. Les surfaces de forme libre n'ont pas de dimensions radiales rigides, contrairement aux surfaces régulières telles que les plans, les cylindres et les surfaces coniques. Ils sont utilisés pour décrire des formes telles que les aubes de turbines, les carrosseries de voitures et les coques de bateaux.

Les formes de surfaces de formes libres et courbes, ne sont pas stockées ni définies dans les logiciels de CAO en terme d'équations polynomiales, mais par leurs pôles, leurs degrés et leurs nombre de patchs (segments avec courbes Splines).

Une surface paramétrique est définie par un ensemble de trois fonctions réelles X(), Y() et Z(). Ces fonctions dépendent de deux paramètres u et v appartenant à l'intervalle [0[0,1],1]. Cette forme est donnée par la formule suivante :

$$F(u,v) = \big(X(u,v), Y(u,v), Z(u,v)\big) \ldots\ldots (1)$$

Chaque point F(u.v) de la surface paramétrique dépend des cordonnées paramétriques u et v (Figure 8). En général, les surfaces paramétriques ne sont pas utilisées séparément, mais plusieurs morceaux de surfaces paramétriques sont assemblés avec des contraintes de continuité pour former une surface plus complexe [5].

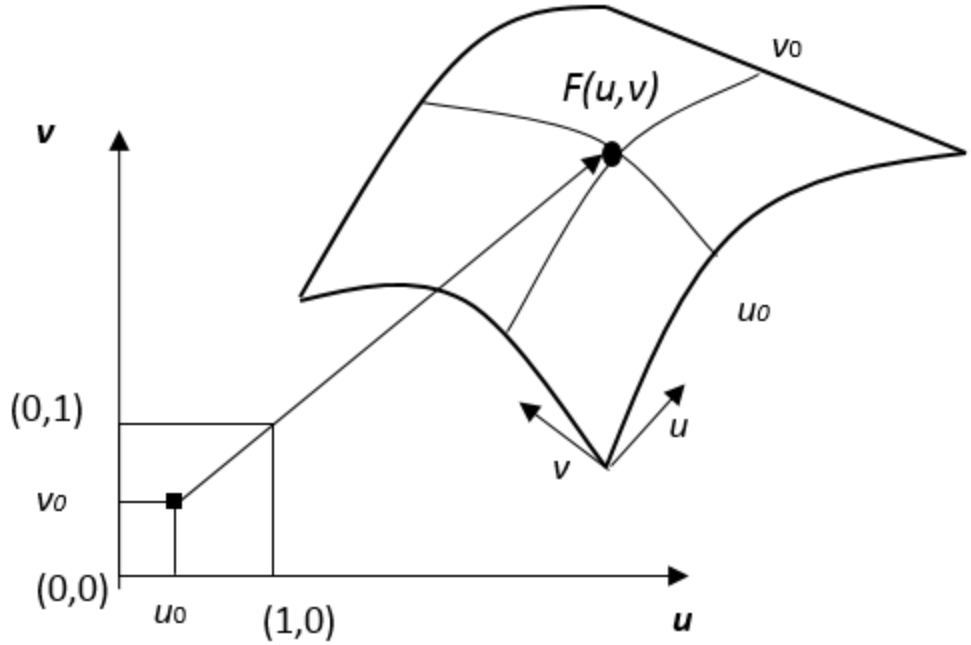


Figure 8. Localisation d'un point sur une surface paramétrique

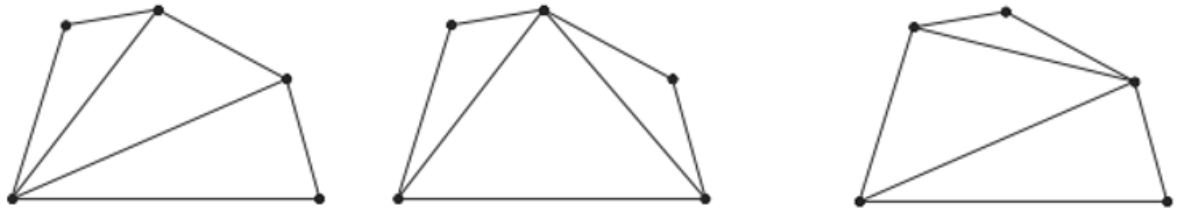
3.2. Représentation géométrique

La représentation géométrique concerne principalement les différentes triangulations. Ces dernières, sont des objets géométriques très fréquemment utilisées. En effet, manipuler uniquement des points n'est pas satisfaisant. Les triangulations, en reliant les points entre eux, créent une partition de l'espace ou d'un domaine.

4. Triangulation de Delaunay

4.1. Différentes triangulations

Les triangules sont des objets géométriques fréquemment utilisés. Les liaisons géométriques et topologiques qu'ils définissent, permettent de localiser rapidement un objet, de décomposer l'espace de travail d'un robot, de reconstruire des objets tridimensionnels à partir de coupes, de visualiser une surface avec un rendu d'image, d'effectuer des opérations booléennes sur les surfaces et des calculs de volume. Ils existent cependant deux grandes classes de triangulation. Les triangulations géométriques, qui ne dépendent que des coordonnées des points, et les triangulations dépendantes des données où on peut associer un coût aux sommets ou aux arêtes. Dans ce mémoire, seules les triangulations géométriques sont considérées.



a. Triangulation de Delaunay. b. Triangulation de poids minimal. c. Triangulation gloutonne.

Figure 9. Différentes triangulations :

Il existe un très grand nombre de façons pour trianguler un ensemble de points. En fonction du domaine d'application, la triangulation doit respecter certains critères, ce qui limitera le nombre de possibilités. [6]

Pour des problèmes de communication, on cherchera à minimiser la longueur totale des côtés des triangles ou, plus généralement, la somme des poids associés aux trois côtés des triangles : c'est la triangulation du poids minimal. Elle a l'avantage d'être unique, systématique, pour des points en position générale, mais reste difficile à calculer du point de vue algorithmique. La complexité de ce problème n'est pas connue.

La triangulation gloutonne est une triangulation obtenue en ajoutant une à une les arêtes et en choisissant toujours la plus courte qui ne croise aucune de celles retenues auparavant, celle-ci n'est pas toujours égale à la triangulation de poids minimum (Figure 9). Elle a aussi l'avantage d'être unique pour des points en position générale, mais par contre peut se calculer en $\theta(nlogn)$. [7]

Pour des problèmes d'approximation de surfaces (avec interpolation) ou de calcul par éléments finis, on préfèrera avoir des triangles les plus équilatéraux possibles, ce qui justifie le choix de la triangulation de Delaunay pour ces applications. Cette dernière maximise l'angle minimal parmi tous les triangles. Elle a aussi l'énorme avantage d'être unique (pour des points en position générale) et peut se calculer efficacement en $\theta(nlogn)$ dans le plan. [8]

4.2. Définition de la triangulation de Delaunay

La triangulation de Delaunay d'un ensemble E de points est la donnée d'un ensemble T de triangles tel que :

- $\forall P \in E \exists t \sim (P1, P2, P3) \in TtelqueP \in \{P1, P2, P3\}$(i.e. tout point P appartient à au moins un triangle de sommet P1, P2, P3.).

- $\forall t \sim (P1, P2, P3) \in T \forall P \in Ptelquep\{P1, P2, P3\} \Rightarrow PCt$ (i.e. pour tout triangle, un point qui n'est pas un sommet n'appartient pas au cercle circonscrit à ce triangle qui est alors dit vide).

4.3. Propriétés des triangulations de Delaunay

4.3.1. Dans le plan

a) Propriétés fondamentales

- Le cercle circonscrit à n'importe quel triangle de Delaunay ne contient aucun site en son intérieur strict (Figure 10). C'est la propriété du cercle vide ou du cercle circonscrit. Réciproquement, si une triangulation vérifie ce critère, elle est de Delaunay.

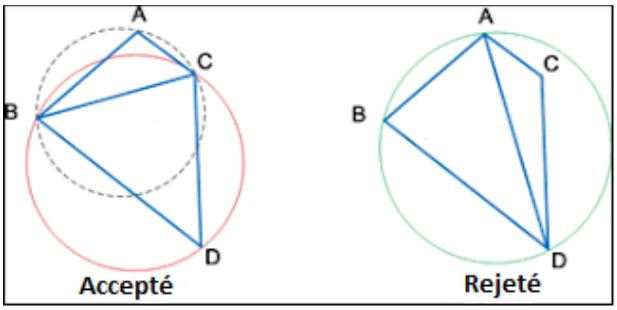


Figure 10. Triangulation de Delaunay acceptée et triangulation de Delaunay refusée.

- On appelle finesse d'une triangulation $T(S)$ le vecteur$F\big(T(S)\big) = (a_1, a_2, \dots a_{3t})$où les a_i sont les angles internes des t triangles, classés par ordre croissant. On définit sur ces vecteurs la relation d'ordre lexicographique $_l$ par :

$$F\big(T(S)\big) = (a_1, a_2, \dots a_{3t}) \quad {}_lF\big(T'(S)\big) = (b_1, b_2, \dots b_{3t})$$

$$\Leftrightarrow \exists_j\{\forall \quad {}_i < j\{a_i = b_i eta_j < b_j\}\dots.. (2)$$

La triangulation qui maximise la finesse selon l'ordre lexicographique est une triangulation de Delaunay. En particulier, la triangulation de Delaunay maximise l'angle minimal parmi tous les triangles.

- On déduit de la propriété précédente la règle de régularisation locale ou critère d'équiangularité locale ou critère de l'angle Min-Max : si la diagonale de n'importe quel quadrilatère strictement convexe formé par l'union de deux triangles de Delaunay est remplacée par la diagonale opposée, la valeur de l'angle minimal parmi les six angles internes n'augmente pas. Réciproquement, si ce

critère d'équiangularité locale est partout vérifié, alors la triangulation est de Delaunay.

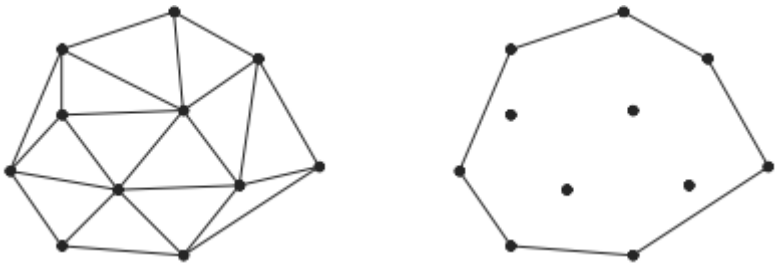
b) Autres propriétés

- Les arêtes de l'enveloppe convexe sont des arêtes de Delaunay (Figure 11).
- Un point et son plus proche voisin définissent toujours une arête de Delaunay.
- Comme dans n'importe quelle triangulation de n sites, le nombre de triangles est$t = 2n - n_{ec}$, et le nombre d'arêtes $e = 3n - 3 - n_{ec}$, n_{ec} étant le nombre de points (ou d'arêtes) de l'enveloppe convexe. La relation d'Euler $t - e + n = 1$ est vérifiée.
- On a vu que la triangulation de Delaunay maximise le plus petit angle, mais elle optimise d'autres paramètres tels que :

  - Elle minimise le plus grand cercle circonscrit.
  - Elle minimise le plus grand cercle englobant (on dit aussi qu'elle a la granularité la plus fine). On appelle cercle englobant d'un triangle le plus petit cercle contenant ce triangle. Ce cercle est égal au cercle circonscrit si et seulement si le triangle ne contient pas d'angle obtus.

4.3.2. En dimension quelconque

- Certaines propriétés existantes dans le plan, se généralisent à une dimension quelconque.
- La boule circonscrite à n'importe quel simplexe de Delaunay ne contient aucun site en son intérieur strict. C'est la propriété de la boule vide ou de la sphère circonscrite. Réciproquement, si une triangulation vérifie ce critère, elle est de Delaunay.



a. Triangulation de Delaunay. b. Enveloppe convexe.

Figure 11. Triangulation de Delaunay et l'enveloppe convexe.

- On associe à chaque simplexe X la plus petite sphère qui le contienne. Soit r_x son rayon. Le grain d'une triangulation est $G(T(S)) = \max_{x \in T(S)} r_x$. Les triangulations de Delaunay de S sont celles qui ont le grain le plus fin.
- Les faces de l'enveloppe convexe sont des faces de la triangulation de Delaunay.
- Un point et son plus proche voisin définissent toujours une arête de Delaunay.
- La taille (c'est-à-dire le nombre de faces) d'une triangulation de Delaunay en dimension k est bornée par $\theta\left(n^{\left\lceil \frac{k}{2} \right\rceil}\right)$

5. Paradigme « Divide and Conquer »

Le schéma de division et conquête (Divide & Conquer) de la triangulation de Delaunay est un schéma $\theta(nlogn)$. L'avantage de ce schéma est qu'il fournit un moyen efficace de fusionner deux triangulations. C'est la raison pour laquelle nous avons choisi cet algorithme pour la parallélisation. [9]

Pendant l'opération de fusion, il doit accéder à la triangulation du voisin dans la zone de fusion. Cela signifie que, pour la fusion entre processeurs, un échange de données successif entre les processeurs est nécessaire. La meilleure solution consiste à estimer les zones affectées de deux triangulations, puis à échanger la zone affectée et à la fusionner. L'algorithme et les propriétés théoriques de la zone affectée seront présentés dans le prochain chapitre. La Figure 12 illustre la fusion parallèle de triangulations de blocs en décomposant le domaine en quatre sous-domaines. L'autre phase nécessaire pour paralléliser, consiste à partitionner l'ensemble de points. Une méthode de partitionnement parallèle de l'algorithme de recherche de Hoare [10] est conçue pour partitionner le jeu de points en blocs 2D. Cette partition peut réduire la longueur des limites devant fusionner entre tous les processeurs. Comme dans la Figure 12.a, les blocs d'une même colonne fusionnent une partie de la coupe dans la direction x. Seuls les processeurs qui fusionnent entre les colonnes sont nécessaires pour fusionner la coupe dans la direction y complète (Figure 12.b). Par conséquent, la longueur de la limite est plus courte que celle des coupes à une dimension. Comme dans la figure, l'ensemble de points sera partitionné dans la direction x et distribué au premier (1er) bloc de chaque colonne, puis les points de chaque colonne seront partitionnés dans la direction y et distribués aux autres blocs. Chaque bloc trie ses points, triangule le jeu de points et crée la zone de triangulation affectée, puis fusionne la zone affectée. La séquence de fusion

étant dans l'ordre inverse de l'ensemble de points de partitionnement parallèle, le nombre de phases requises pour la fusion est donc $O(logP)$.

a. Triangulation de chaque bloc. b. Fusion de B1 et B3 avec B0 et B2. c. Fusion de B0 avec B2.

Figure 12. Algorithme Divide and Conquer.

Conclusion

L'étude effectuée dans cette partie, portait dans un premier temps sur la présentation des formats d'échange de données et principalement sur le format STL. Par la suite, une présentation sur le Reverse Engineering est faite, celle-ci a traité le processus de l'ingénierie inverse. Dans un second temps, différentes méthodes de reconstruction ont été présentées, notamment les méthodes paramétriques et géométriques. Ensuite, La triangulation de Delaunay, qui est une étape cruciale dans la reconstruction 3D, a été expliquée, tout en mettant l'accent sur ses propriétés dans le plan et dans une dimension quelconque. Enfin, le principe Divide and Conquer a été présenté et les étapes importantes de la fusion multidirectionnelle des triangulations de Delaunay en 3D ont été vaguement présentées. Les détails seront apportés dans le chapitre suivant, la conception de l'application est également abordée.

CHAPITRE II

Conception de l'Application et Démarches de Résolution

Introduction

Pour discrétiser la surface complexe à des fins d'analyse et de calcul, le maillage est couramment utilisé dans de nombreux domaines, tels que l'informatique graphique, la mécanique des solides computationnelle, la dynamique des fluides numérique et l'électromagnétisme numérique. Avec l'utilisation d'un ordinateur parallèle, il n'est pas inhabituel de résoudre le problème des dizaines de millions de points maillés. Une telle quantité d'exigences de calcul fait de la génération de maillage un problème de temps et de stockage lors d'utilisation d'un seul processeur. Par conséquent, nous avons besoin de l'aide d'un ordinateur parallèle pour générer le maillage d'un très grand ensemble de données. Dans la génération automatique de maillage, ils existent deux classes principales de méthodes : la triangulation de Delaunay et la méthode de front avancé. Le critère d'angle max-min de la triangulation de Delaunay fait de la triangulation un maillage bien formé. Ils existent de nombreux algorithmes séquentiels pour la triangulation de Delaunay tels que l'algorithme de Bowyer-Watson, la coque rapide, la ligne de balayage, la construction incrémentielle, Divide and Conquer, etc. Parmi ces algorithmes, le schéma de division et de conquête est le plus efficace et puissant pour générer la triangulation de Delaunay sur des ensembles de points avec n'importe quelle distribution. Nous utiliserons le schéma de Divide and Conquer comme approche pour la fusion multidirectionnelle des triangulations de Delaunay en 3D.

1. Problématique

La reconstruction des modèles numériques 3D à partir d'un nuage de point très dense et très volumineux, avec une bonne précision est une tâche ardue, coûteuse en temps. Par conséquent, la solution à ce problème n'est pas triviale. Plusieurs méthodes ont été développées pour tenter d'y apporter une solution. La densité et le volume très important du nuage de points augmentent les temps de traitement et par conséquent les coûts. Pour apporter une solution à ce problème, le nuage de points subdivisé en plusieurs blocs selon des critères bien définis, par la suite, chaque bloc est approximé par des tétraèdres en utilisant la triangulation de Delaunay et le paradigme « Diviser pour Régner » pour fusionner les différents blocs.

2. Architecture globale de l'application

L'application développée consiste à réaliser la fusion des sous nuages de points qui sont traités par la triangulation de Delaunay lors de l'étape de la subdivision, en un temps optimal avec un coût pertinent, en utilisant de bonnes stratégies.

Ce travail, prend comme base de départ, certains résultats obtenus préalablement dans le cadre des activités de l'équipe CFAO, ces résultats concernent, la lecture du nuage de points, la subdivision de ce dernier et la triangulation de Delaunay pour chaque octree sans fils. Nous continuerons ainsi sur les résultats précédents.

Cette partie est consacrée à la fusion des différentes triangulations traitées, en utilisant les zones affectées, afin d'obtenir le modèle STL désiré. La Figure 1 donne l'architecture globale de l'application à développer.

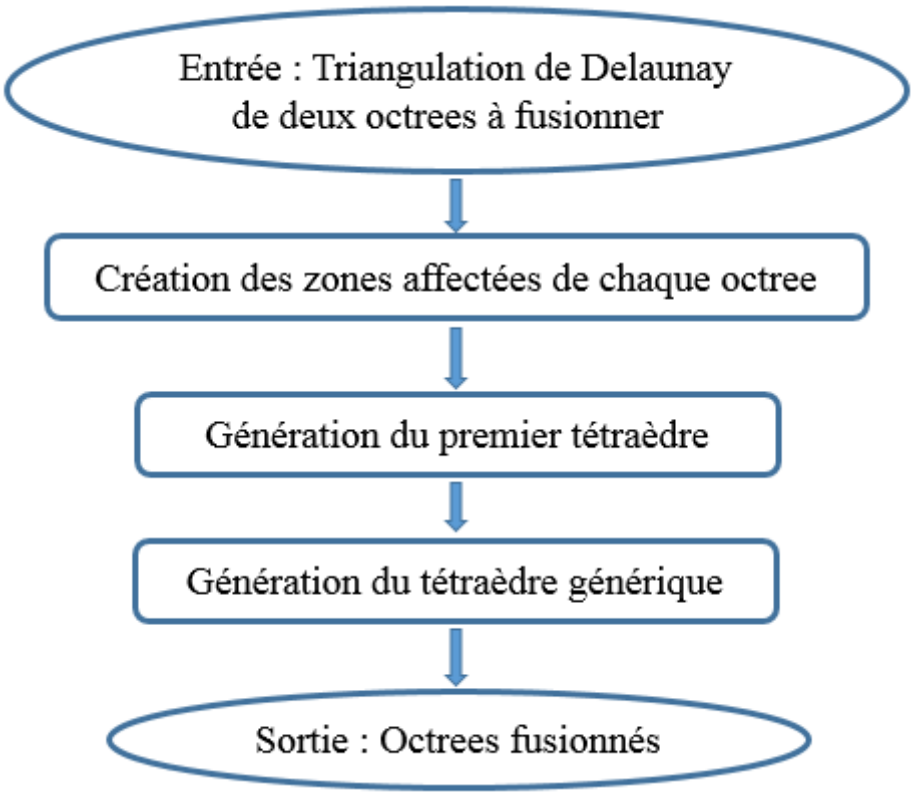


Figure 1. Architecture globale de l'application.

2.1. Description de l'organigramme de l'architecture globale de l'application

L'organigramme de l'architecture globale de l'application est divisé en quatre (04) étapes, où chaque étape est subdivisée en plusieurs sous étapes.

2.1.1. Création des zones affectées de chaque cellule

Les zones affectées par définition, sont les triangulations qui sont susceptible d'être modifiées au cours de la fusion de deux sous-triangulations de Delaunay. Le processus de création de ces zones est décrit par la Figure 2.

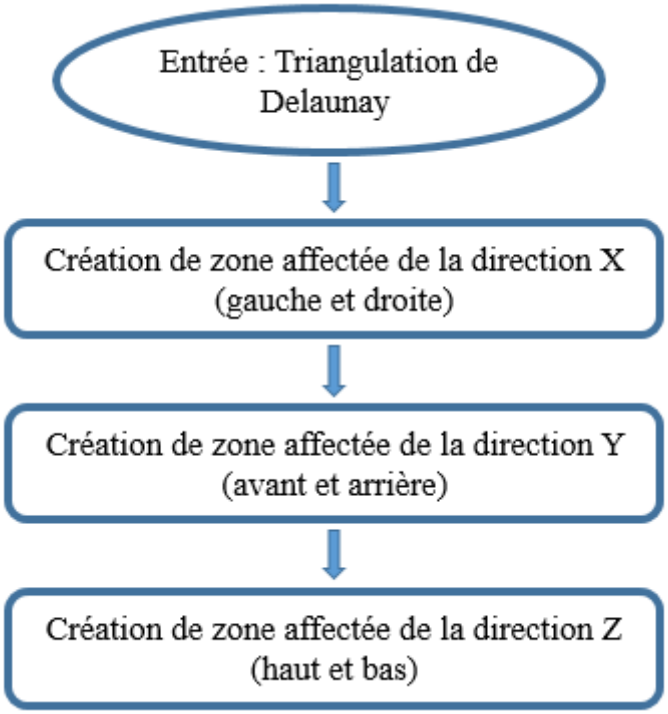


Figure 2. Organigramme de la création des zones affectées de chaque cellule

Ils existent trois directions de zones affectées correspondant aux trois axes de l'espace tridimensionnel X, Y et Z.

- **Zones Affectées selon X : droites et gauches**

Afin de déterminer la zone affectée dans cette direction, on suit le processus suivant :

- Vérification de l'intersection entre la sphère circonscrite d'un tétraèdre de l'Octree droit (gauche) et le plan Xmax (Xmin) (Figure 3).

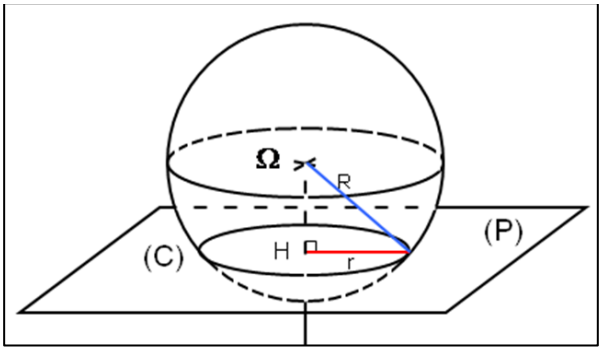
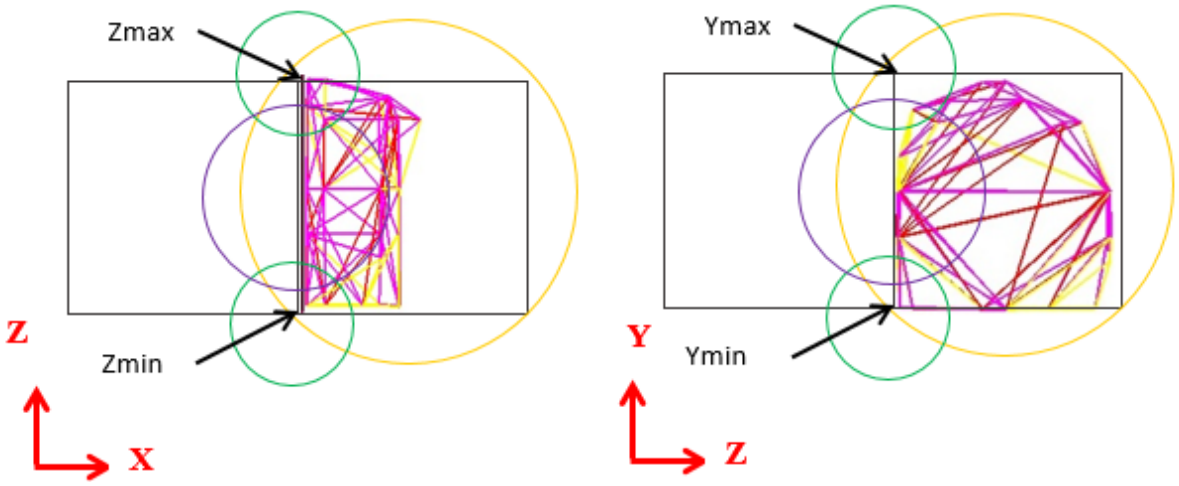


Figure 3. Intersection d'une sphère avec un plan.

- Si vérifié, alors calcul du rayon du cercle d'intersection entre la sphère et le plan.
- Vérification du rayon s'il est compris entre Ymin et Ymax.
- Vérification du rayon s'il est compris entre Zmin et Zmax (Figure 4).



a. Vérification selon Z. b. Vérification selon Y.

Figure 4. Génération de la zone affectée selon X (gauche-droite).

- **Zones Affectées selon Y : avant et arrière**

Afin de déterminer la zone affectée dans cette direction, on suit le processus suivant :

- Vérification de l'intersection entre la sphère circonscrite d'un tétraèdre de l'Octree avant (arrière) et le plan Ymax (Ymin).
- Si vérifié, alors calcul du rayon du cercle d'intersection entre la sphère et le plan.
- Vérification du rayon s'il est compris entre Xmin et Xmax.
- Vérification du rayon s'il est compris entre Zmin et Zmax.

- **Zones Affectées selon Z : haut et bas**

Afin de déterminer la zone affectée dans cette direction, on suit le processus suivant :

- Vérification de l'intersection entre la sphère circonscrite d'un tétraèdre de l'Octree haut (bas) et le plan Zmax (Zmin).
- Si vérifié, alors calcul du rayon du cercle d'intersection entre la sphère et le plan.
- Vérification du rayon s'il est compris entre Xmin et Xmax.
- Vérification du rayon s'il est compris entre Ymin et Ymax

3. Fusion des Octrees

La fusion est la base du processus de reconstruction du nuage de points subdivisé. Elle consiste à concaténer les Octrees deux à deux en utilisant les zones affectées des boxes adjacents jusqu'à atteindre l'Octree père.

Chaque Octree a trois zones affectées selon sa position dans son Octree père. Pour la simplification de la procédure de fusion, la stratégie adoptée dans ce travail est décrite par l'organigramme ci-dessous (Figure 5).

3.1. Stratégie de fusion adoptée

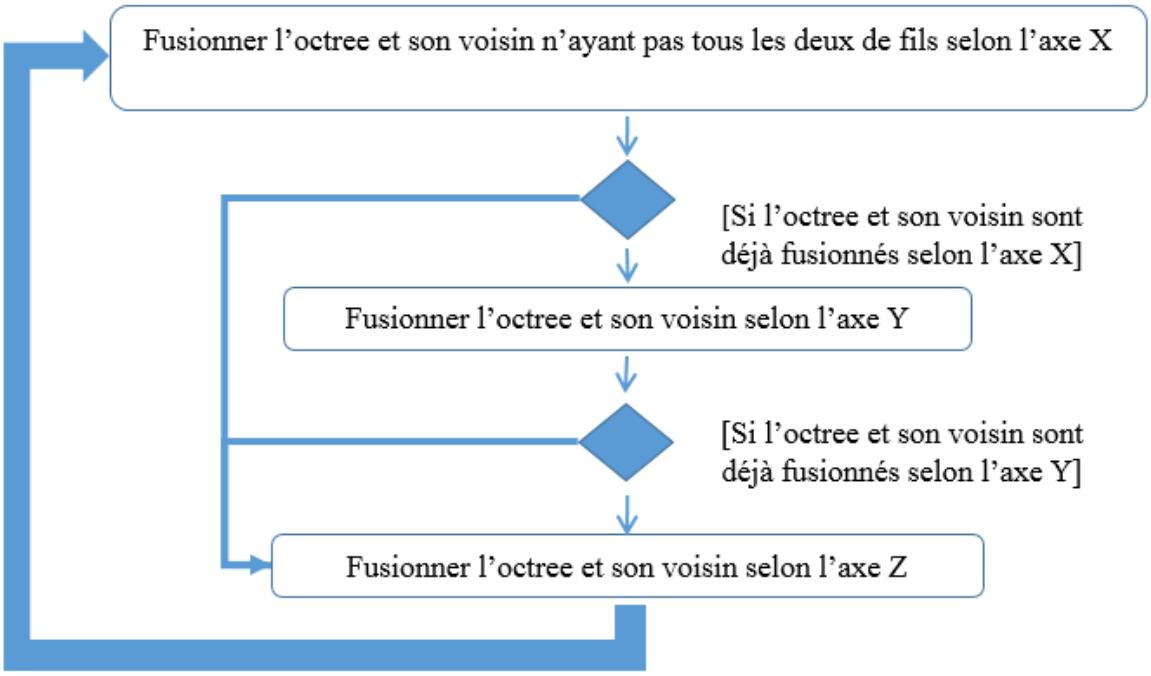


Figure 5. Organigramme de la stratégie de fusion proposé.

- Sélectionner et fusionner les Octrees selon l'axe X sur tous les niveaux de l'arborescence.

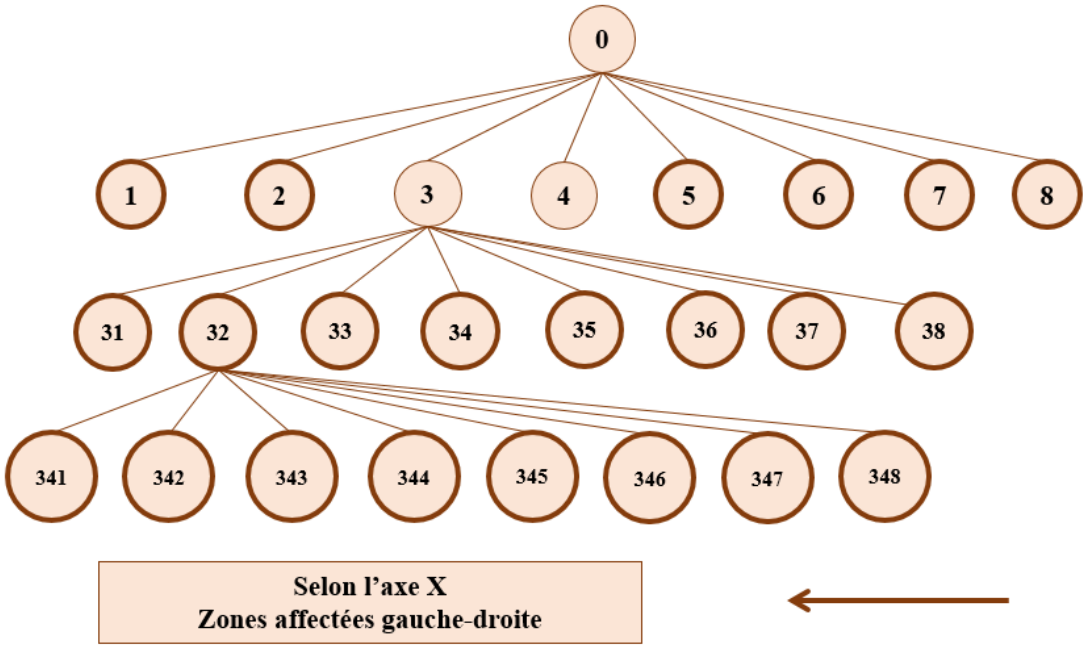


Figure 6. Sélection des Octrees à fusionner selon l'axe X.

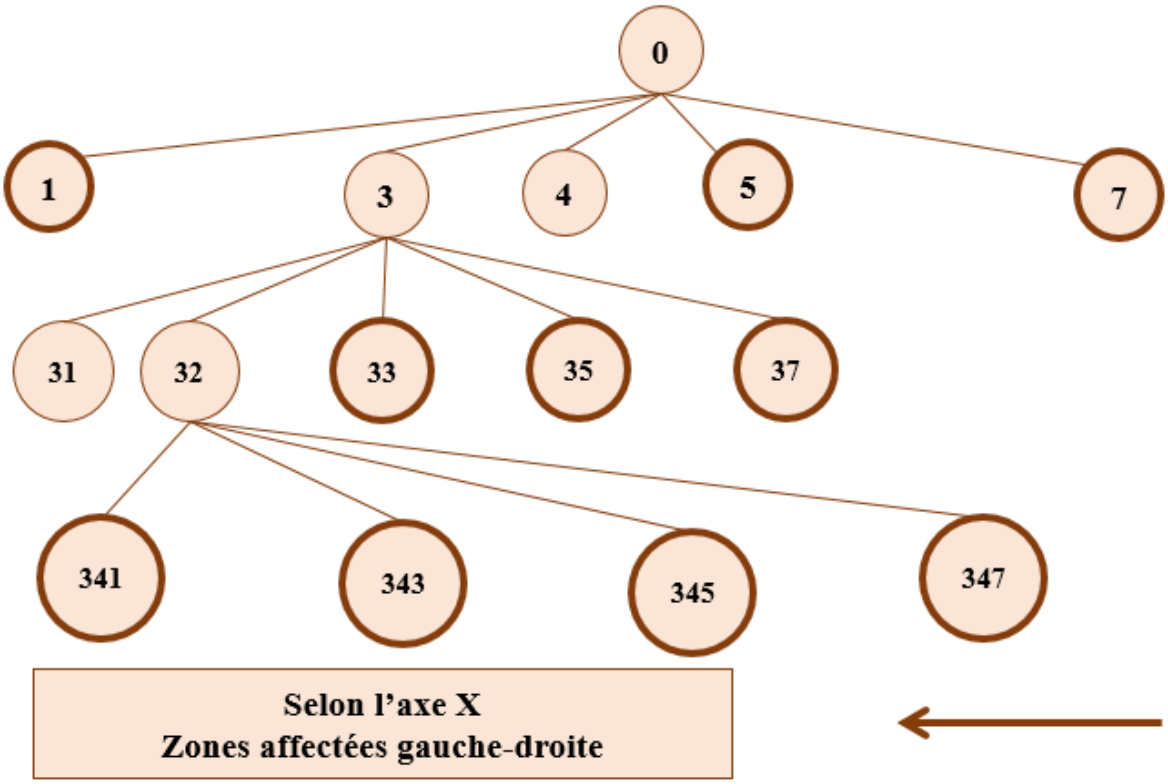


Figure 7. Fusion des Octrees sélectionnés selon l'axe X.

- Sélectionner et fusionner les Octrees selon l'axe Y sur tous les niveaux :
  Deux Octrees peuvent être fusionnés selon l'axe Y s'ils sont déjà fusionnés selon l'axe X.

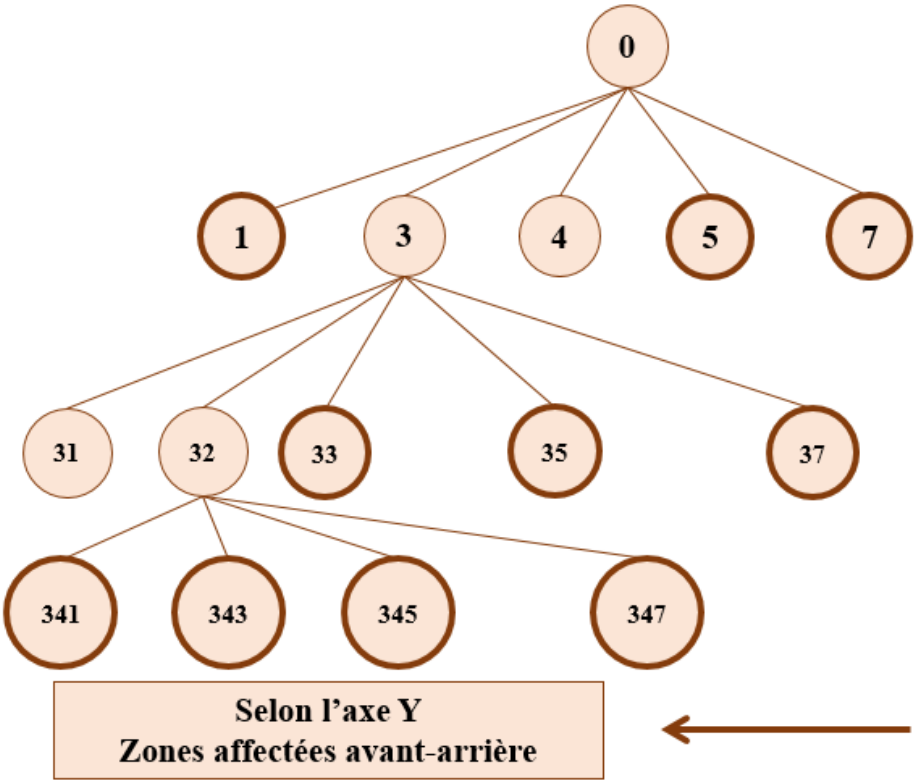


Figure 8. Sélection des Octrees à fusionner selon l'axe Y.

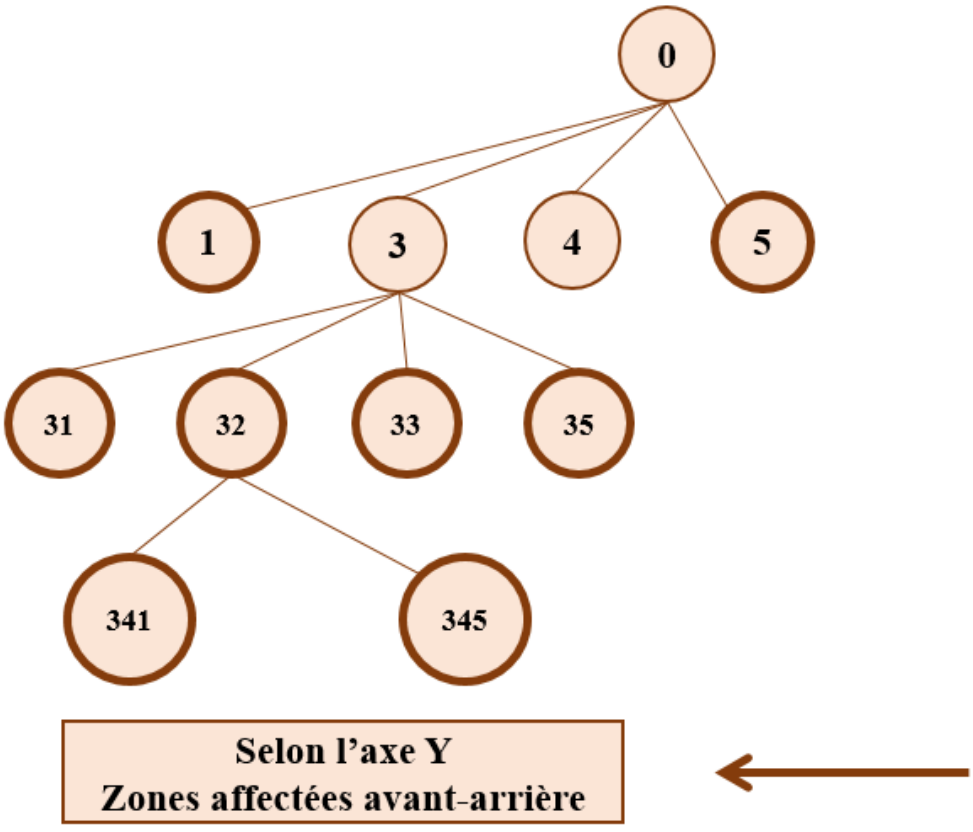


Figure 9. Fusion des Octrees sélectionnés selon l'axe Y.

- Sélectionner et fusionner les Octrees selon l'axe Z sur tous les niveaux :
  Deux Octrees peuvent être fusionnés selon l'axe Z s'ils sont déjà fusionnés selon l'axe Y. Le résultat de la fusion est stocké dans l'Octree père.

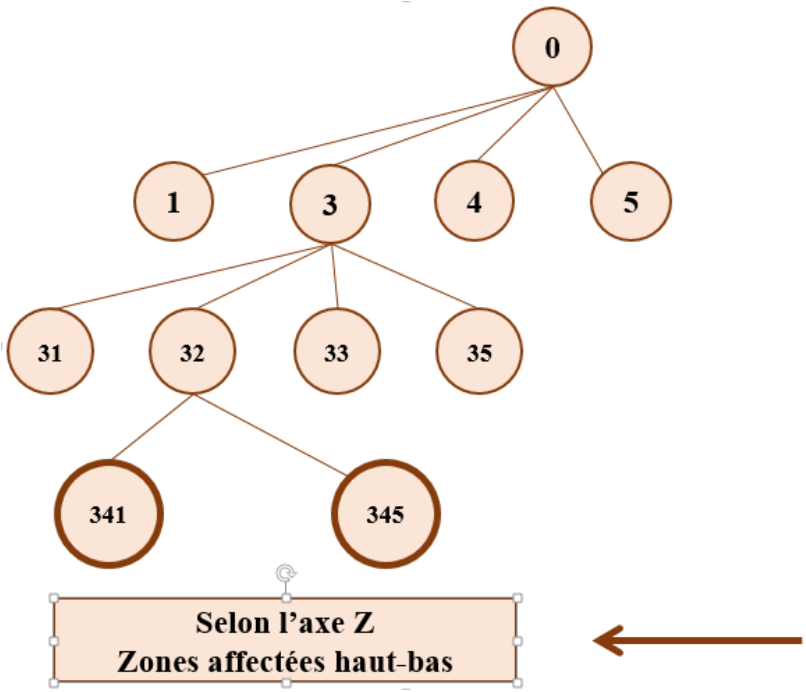


Figure 10. Sélection des Octrees à fusionner selon l'axe Z.

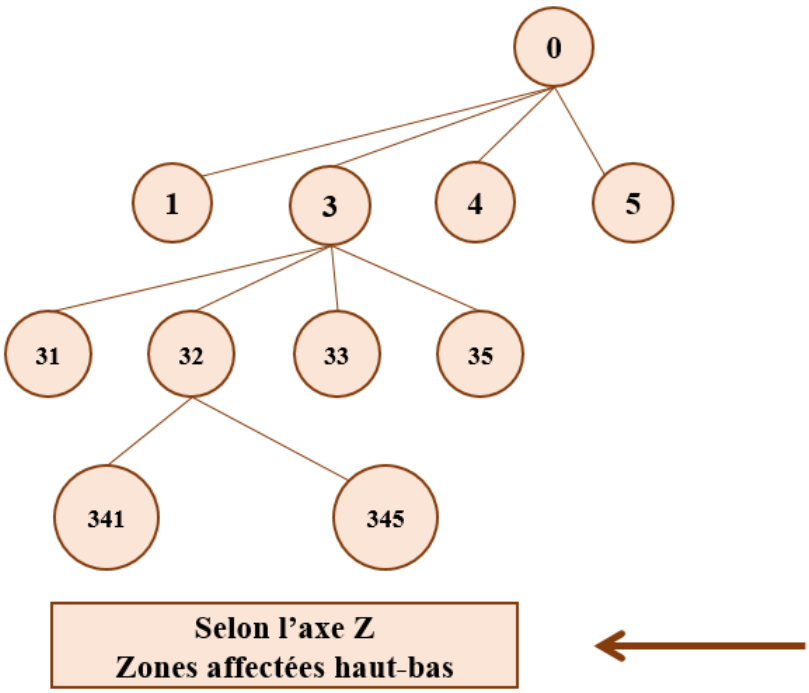


Figure 11. Fusion des Octrees sélectionnés selon l'axe Z.

La procédure est réitérée jusqu'à la fusion de tous les Octrees.

3.2. Optimisation de la procédure de la fusion :

a. Affecter une zone droite à chaque Octree impair et gauche à chaque Octree pair :

- Affecter la zone droite au premier Octree et gauche au deuxième Octree.
- Affecter la zone droite au troisième Octree et gauche au quatrième Octree.
- Affecter la zone droite au cinquième Octree et gauche au sixième Octree.
- Affecter la zone droite au septième et gauche au huitième Octree.

b. Affecter une zone arrière aux Octrees impairs et avant aux Octrees pairs :

- Affecter la zone arrière au premier Octree et l'avant au troisième Octree.
- Affecter la zone arrière au cinquième Octree et l'avant au septième Octree.

c. Affecter une zone haute et basse aux Octrees impairs :

- Affecter la zone haute au premier Octree et bas au cinquième Octree.

La fusion des cellules commence par la concaténation selon l'axe X, puis l'axe Y, et enfin l'axe Z, les étapes sont décrites ci-dessous :

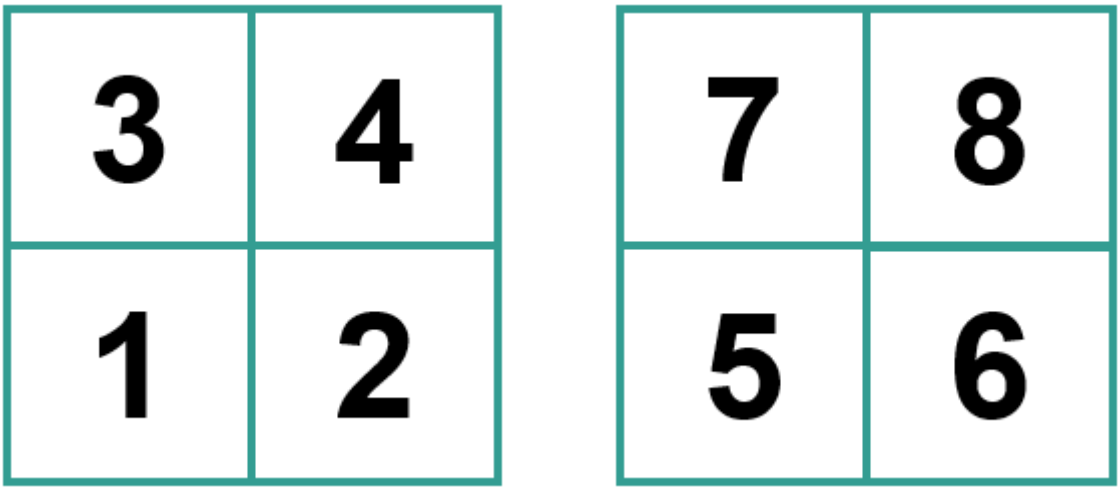


Figure 12. Répartition des Octrees.

4. Approche de fusion

a. Fusion selon l'axe X en utilisant les zones affectées gauches et droites :

- Fusion du premier et du deuxième Octree et le stockage est au niveau du premier Octree.

- Fusion du troisième et du quatrième Octree et le stockage est au niveau du troisième Octree.
- Fusion du cinquième et du sixième Octree et le stockage est au niveau du cinquième Octree.
- Fusion du septième et du huitième Octree et le stockage est au niveau du septième Octree.

b. Fusion en utilisant les zones affectées avant et arrière :

- Fusion du premier et du troisième Octree et le stockage est au niveau du premier Octree.
- Fusion du cinquième et du septième Octree et le stockage est au niveau du cinquième Octree.

c. Fusion en utilisant les zones affectées hautes et basses :

- Fusion du premier et du cinquième Octree et le stockage est au niveau du premier Octree.

4.1. Premier tétraèdre

Après la génération des zones affectées, l'étape suivante est la génération du premier tétraèdre, support des autres tétraèdres. L'organigramme suivant décrit les étapes essentielles de cette procédure (Figure 13).

La génération du premier tétraèdre commence par l'identification de deux zones affectées, puis l'identification des deux points les plus proches dans chaque zone (sommets), vient ensuite, la génération des faces qui partagent le sommet dans chaque zone, et pour chaque face, la création du tétraèdre associé.

Le test de la sphère circonscrite consiste à tester l'appartenance des points de la zone affectée droite dans le cas où le tétraèdre se trouve dans la zone affectée gauche et vice versa. Dans le cas où la condition de la sphère vide n'est pas vérifiée, le tétraèdre doit être supprimé tout en mettant à jour la visibilité de ces voisins, sinon il faut tester la sphère vide de chaque tétraèdre à fusionner avec les points de chaque zone affectée. Dans le cas où les deux tétraèdres sont valides, celui dont le rayon de la sphère est le plus petit est le premier tétraèdre choisi. [13]

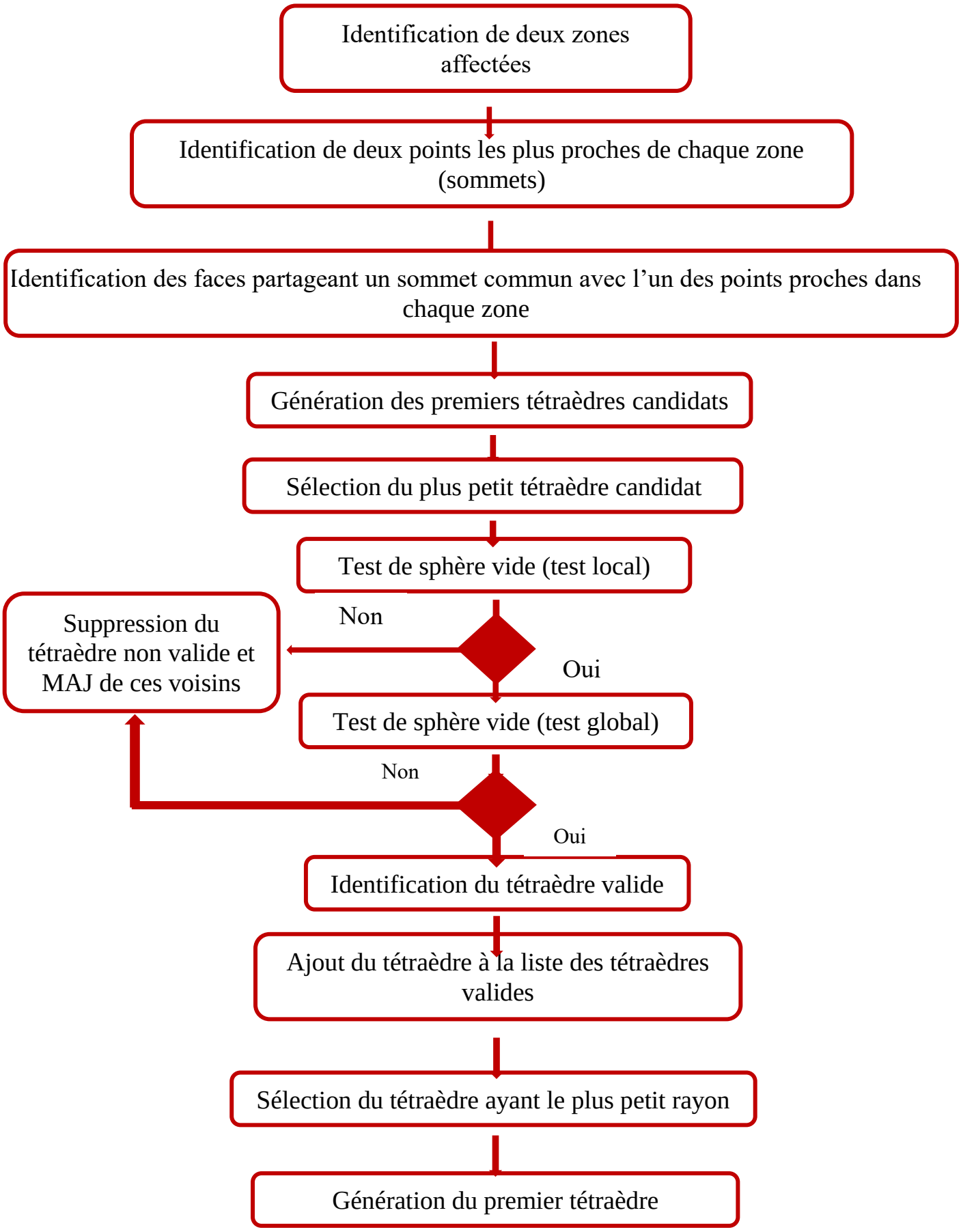


Figure 13. Organigramme de la génération du premier tétraèdre.

4.2. Tétraèdre générique

La procédure de la génération du tétraèdre générique suit le même principe que la génération du premier tétraèdre (Figure 14). Deux tests sont à vérifier :

1. Test local de la sphère circonscrite : tester l'appartenance des points de la zone affectée droite dans le cas où le tétraèdre se trouve dans la zone affectée gauche et vice versa.
2. Test global de la sphère circonscrite : tester les tétraèdres à fusionner avec les points de chaque zone. [14]

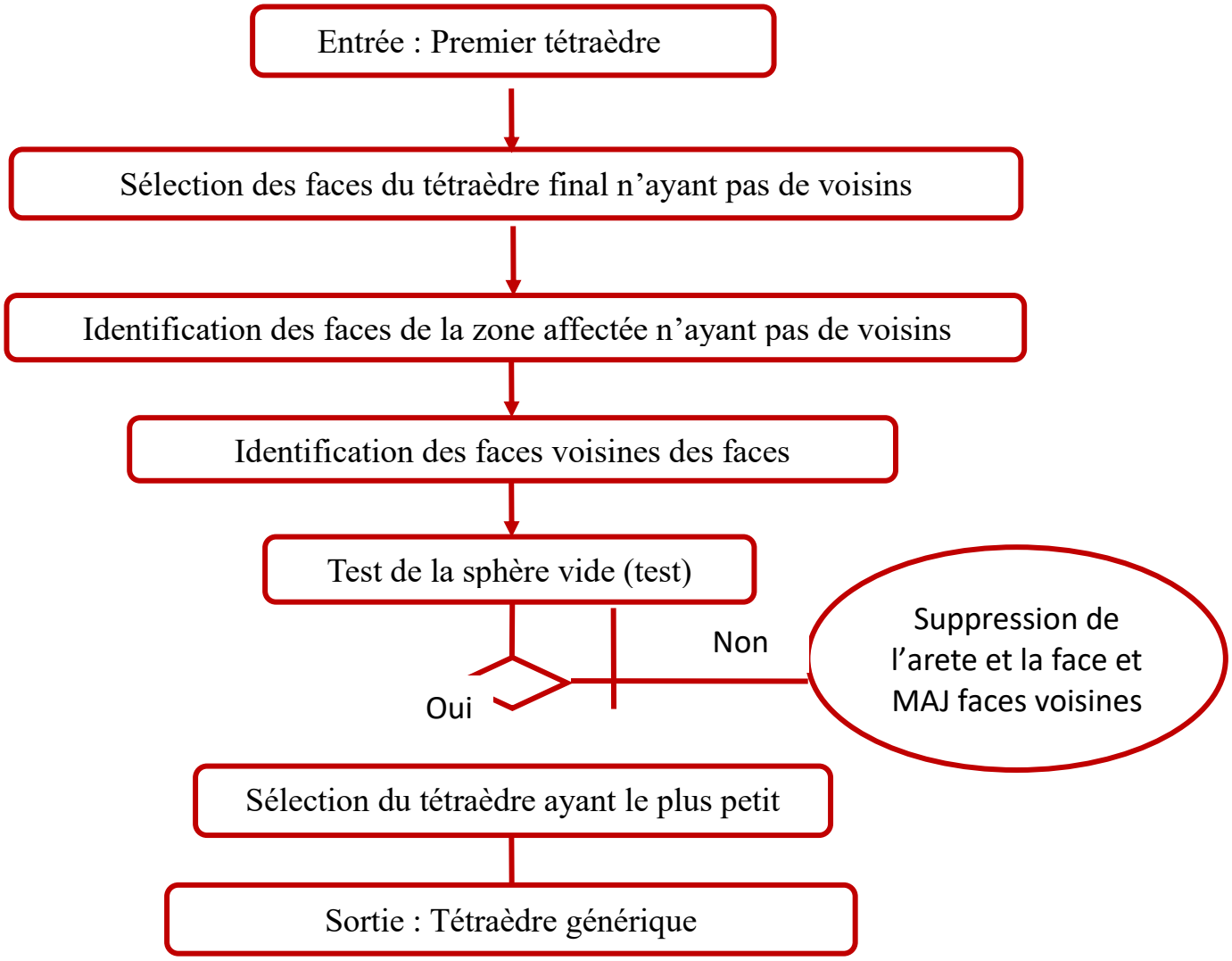


Figure 14. Organigramme de la génération du tétraèdre générique.

La partie sensible lors de la génération du tétraèdre générique est la mise-à-jour des faces et des arêtes après une suppression d'une face ou d'une arête qui ne vérifie pas la condition de

la sphère vide. Au niveau de cette procédure, il se génère deux types de tétraèdres candidats, et un seul qui va représenter le tétraèdre générique.

- Premier type du tétraèdre générique candidat « RBT » (Figure 15) :

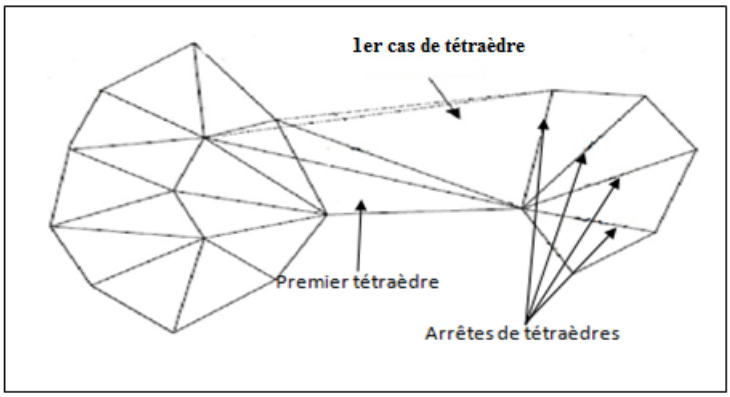


Figure 15. Premier type du tétraèdre candidat générique.

Dans ce cas, il faut sélectionner une des faces du premier tétraèdre qui n'a pas de voisins, et créer un tétraèdre à partir de cette face et une arrête de l'autre zone affectée (si le premier tétraèdre appartient à la zone gauche donc l'arrête choisie doit être de la zone affectée droite et vice versa).

- Le deuxième type du tétraèdre générique candidat « LBT » (Figure 16) :

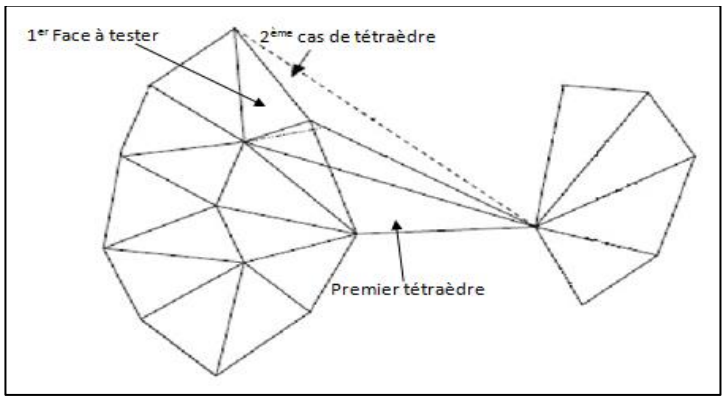


Figure 16. Deuxième type du tétraèdre candidat générique.

Dans le deuxième cas, à partir de la même face qu'on a créé le tétraèdre générique du premier type on crée un tétraèdre avec la face du premier tétraèdre qui appartient à la zone affectée (la face de base).

Dans le cas d'existence des deux types de tétraèdres génériques qui vérifient toutes les conditions présentées précédemment, le tétraèdre ayant le plus petit rayon de la sphère

circonscrite va être le tétraèdre générique et devient la base pour la création d'autres tétraèdres génériques.

5. Modélisation de l'application avec UML

Cette partie concerne la modélisation de l'application réalisée. La conception en fait c'est une étape très importante pour identifier les grandes fonctionnalités d'un système donné, les exigences de la réalisation de ce système, et notamment les contraintes qui doivent être considérées.

Le langage UML « Unified Modeling Language» ou le langage de la modélisation unifiée, a été pensé pour être un langage de modélisation visuelle commun, et riche sémantiquement et syntaxiquement, il est destiné à l'architecture, la conception et la mise en œuvre de systèmes logiciels complexes par leur structure aussi bien que leur comportement. L'UML a des applications qui vont au-delà du développement logiciel, notamment pour les flux de processus dans l'industrie, il ressemble aux plans utilisés dans d'autres domaines et se compose de différents types de diagrammes. Dans l'ensemble, les diagrammes UML décrivent la limite, la structure et le comportement du système et les objets qui s'y trouvent [12]. Les diagrammes UML utilisés :

- Diagramme de cas d'utilisation : utilisé pour l'expression des besoins.
- Diagramme de classe : pour la conception de l'application.

5.1. Diagramme des cas d'utilisation

Le diagramme des cas d'utilisation « Use Case Diagram » permet de :

- Identifier et modéliser les besoins des utilisateurs.
- Préciser les grandes fonctionnalités, les contraintes, les limites et les ressources utilisées pour réaliser l'application.
- Représenter les interactions entre le système et les utilisateurs.

La Figure 17 donne le diagramme cas d'utilisation général de l'application logicielle à développer.

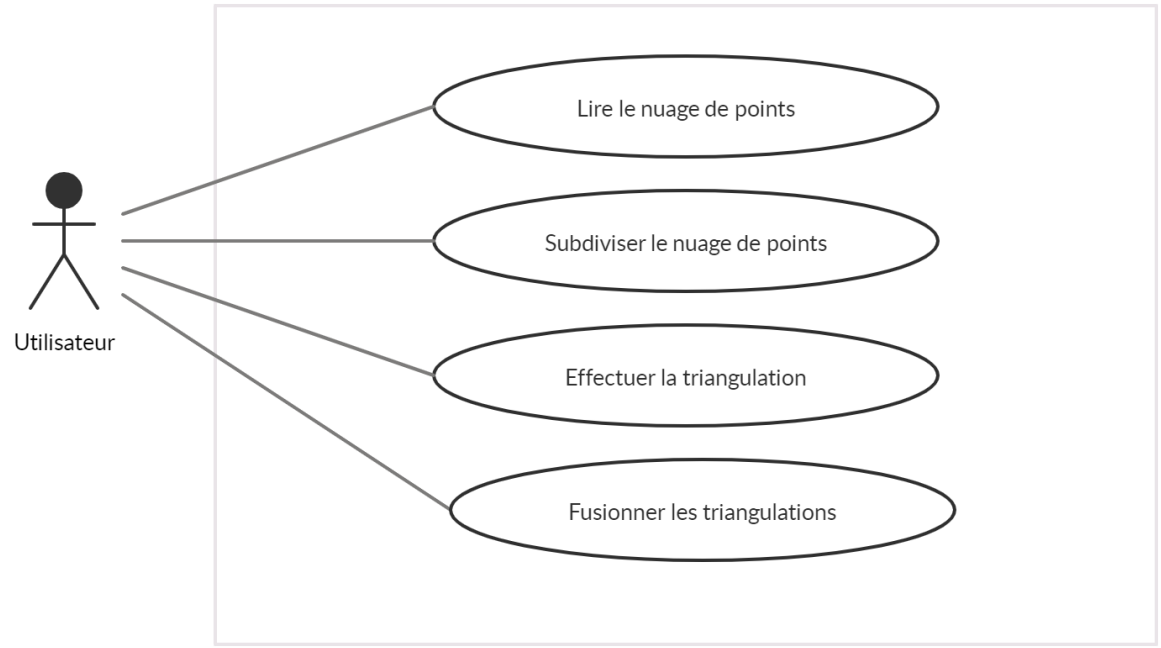


Figure 17. Diagramme cas d'utilisation général.

La lecture du nuage de points, la subdivision ainsi que la triangulation sont effectuées dans les projets antérieurs.

Le scénario relatif au premier cas est le suivant :

Nom : Lire le nuage de points.

Acteurs : utilisateur.

Description : l'utilisateur doit être capable de lire un nuage de points quelconque.

Le scénario nominal :

1. Le système affiche un onglet ayant un bouton pour récupérer le fichier du nuage de points.
2. L'utilisateur choisi le fichier.
3. Le système calcule et affiche le nombre de points, les dimensions du brut, les coordonnées des limites du brut et offre la possibilité de visualiser les résultats.

Le scénario relatif au deuxième cas d'utilisation :

Nom : Subdiviser le nuage de points.

Acteurs : Utilisateur.

Description : l'utilisateur a la possibilité d'effectuer la subdivision du nuage de points selon les trois directions X, Y, Z.

Le scénario nominal :

1. L'utilisateur spécifie la puissance de la division.
2. L'utilisateur clique sur le bouton « Création des boxes », il peut choisir entre une création séquentielle ou parallèle.
3. Le système affecte les points aux boxes.
4. L'utilisateur spécifie le nombre de points par Octree.
5. L'utilisateur clique sur le bouton « Création des Octrees »
6. Le système affecte les boxes et les points aux Octrees.
7. Le système offre les différents paramètres de visualisation.

Le scénario relatif au troisième cas d'utilisation est le suivant :

Nom : Effectuer la triangulation.

Acteurs : Utilisateur.

Description : la triangulation se fait au niveau de chaque Octree n'ayant pas de fils.

Le scénario nominal :

1. L'utilisateur choisi le mode de triangulation (séquentiel ou parallèle) et clique sur le bouton « Triangulation des nœuds sans fils ».
2. Le système effectue la triangulation de Delaunay.
3. Le système offre les différents paramètres de visualisation.

La fusion est la dernière étape dans l'enchainement des fonctionnalités de cette application. Cette étape consiste à fusionner les Octrees triangulés, en commençant par la création du premier tétraèdre après la génération des zones affectées et se terminent par la détermination des tétraèdres génériques.

Les diagrammes des cas d'utilisations qui suivent montrent le détail du cas d'utilisation « Fusionner les triangulations ».

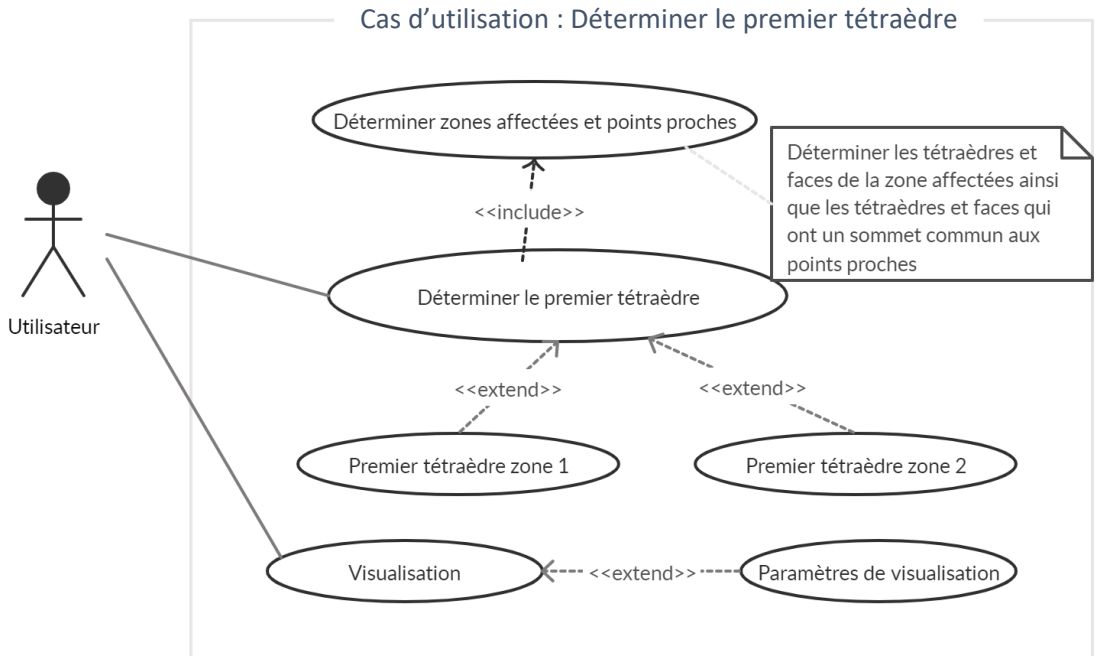


Figure 18. Diagramme du cas d'utilisation de la génération du premier tétraèdre.

La détermination du tétraèdre générique est l'étape qui suit la génération du premier tétraèdre, cette phase est modélisée par le diagramme de cas d'utilisation suivant :

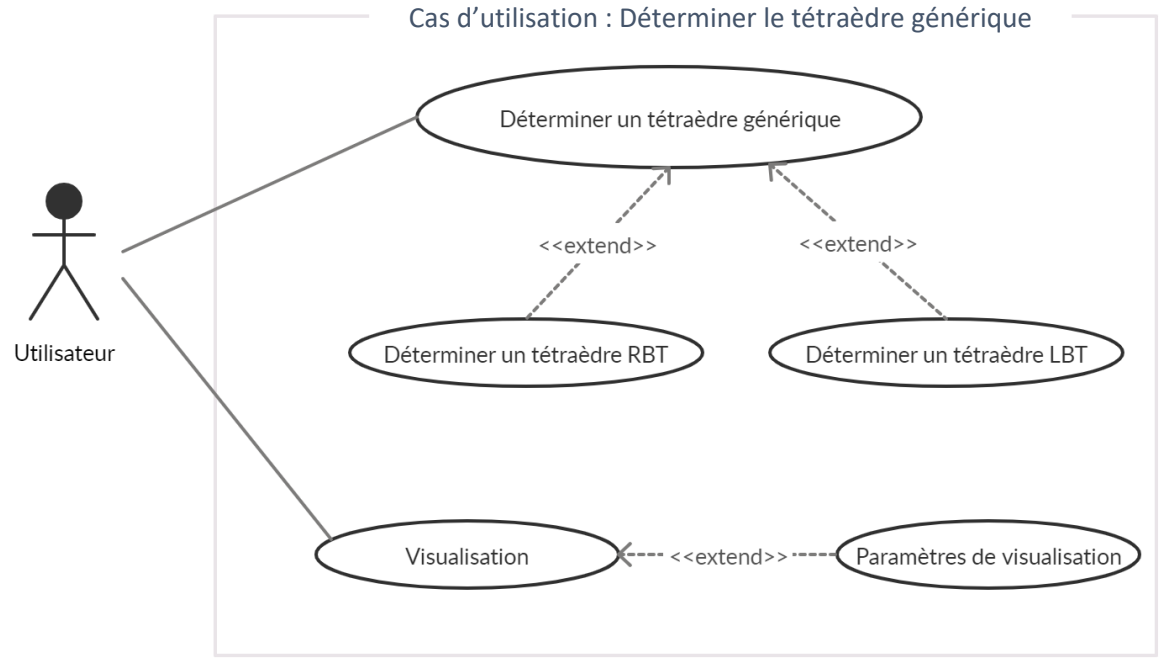


Figure 19. Diagramme du cas d'utilisation de la génération du tétraèdre générique.

5.2. Diagramme de classes

Les diagrammes de classes sont l'un des types de diagrammes UML les plus utiles, car ils décrivent clairement la structure d'un système particulier en modélisant ses classes, ses attributs, ses opérations et les relations entre ses objets. Ils représentent un type de diagramme de structure, car ils décrivent ce qui doit être présent dans le système modélisé.

Les diagrammes de classes présentent de nombreux avantages pour n'importe quel type d'organisation entre autres :

- Illustrer des modèles de données pour des systèmes d'information, quel que soit leur degré de complexité.
- Comprendre au mieux l'aperçu général des schémas d'une application.
- Exprimer visuellement les besoins d'un système et diffuser cette information dans toute l'entreprise.
- Créer des schémas détaillés qui mettent l'accent sur le code spécifique qui doit être programmé et mis en œuvre dans la structure décrite.
- Fournir une description indépendante de l'implémentation des types utilisés dans un système, qui sont ensuite transmis entre ses composants [11].

Comme il a été mentionné, ce travail se veut une suite de certains travaux effectués précédemment par l'équipe « CFAO » dans le domaine de la reconstruction d'objets à partir de nuages de points non structurés. De ce fait, les classes qui sont déjà créées et lister ci-dessous, sont utilisées directement avec les nouvelles classes implémentées.

- Classe Couleur_multi
- Classe Point_multi
- Classe Brute
- Classe Nuage_point_multi
- Classe Box_multi
- Classe octree
- Classe Face
- Classe Tetraedre_k

Les nouvelles classes implémentées dans ce travail sont :

- Classe Face_fusion
- Classe Octree_a_fusionner
- Classe Arete
- Classe Normale

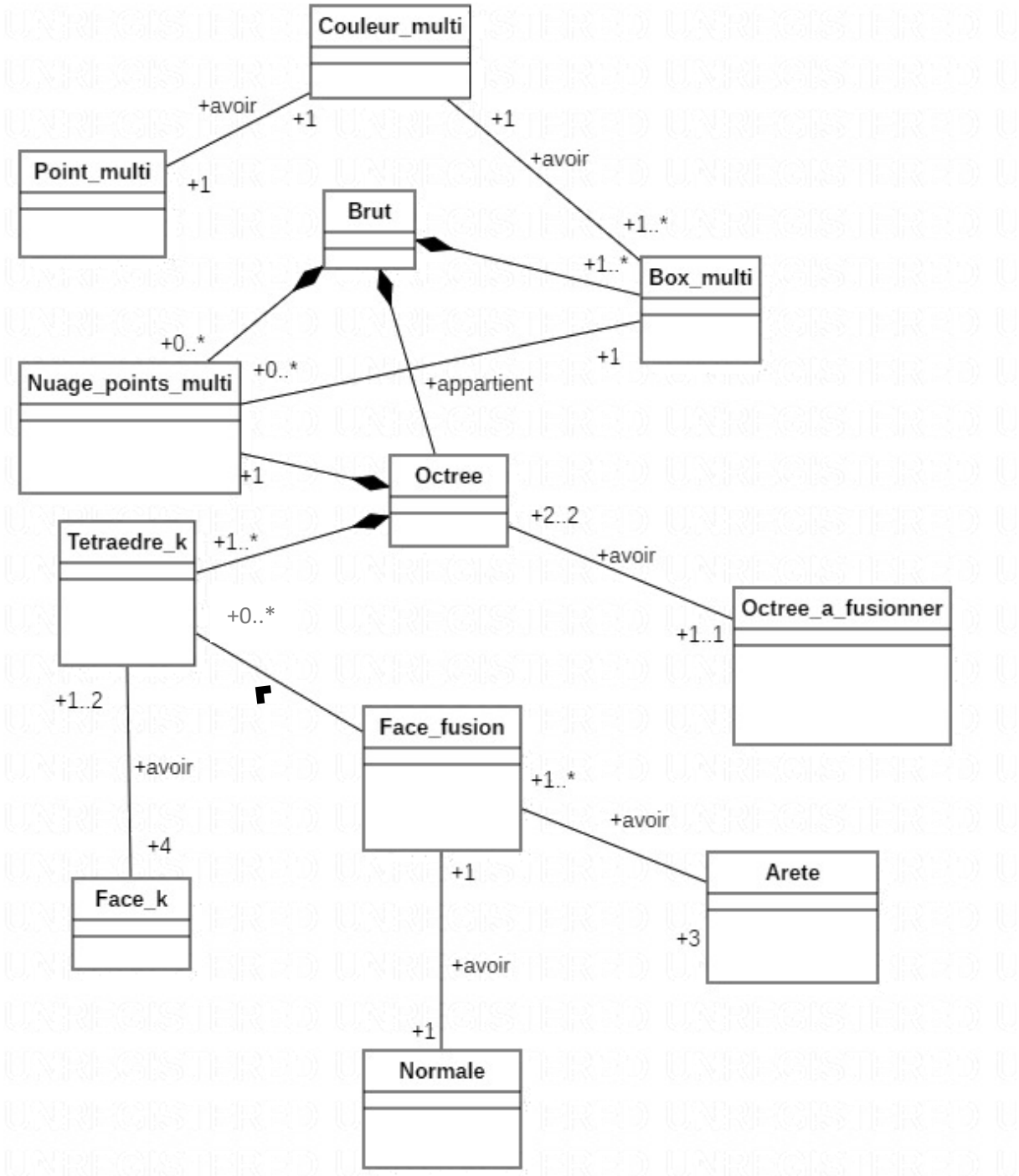


Figure 20. Diagramme de classes général

La représentation des attributs et méthodes des différentes classes implémentées est décrite ci-dessous :

- **Classe Octree** : c'est une classe de la deuxième subdivision du nuage de points. Elle regroupe un ensemble de points spécifié par l'utilisateur d'où chaque Octree contient un ensemble de boxes. Parmi les fonctions de la classe **Octree,** nous avons :
  - calcule_nb_point() : retourner le nombre de points de chaque Octree,

- calcule_voisin (inti, int j) : retourne les voisins de chaque Octree,
- zone_affectee_gauche () : retourne la zone affectée gauche d'un Octree,
- points_proches () : retourne le point le plus proche de chaque zones affectées.

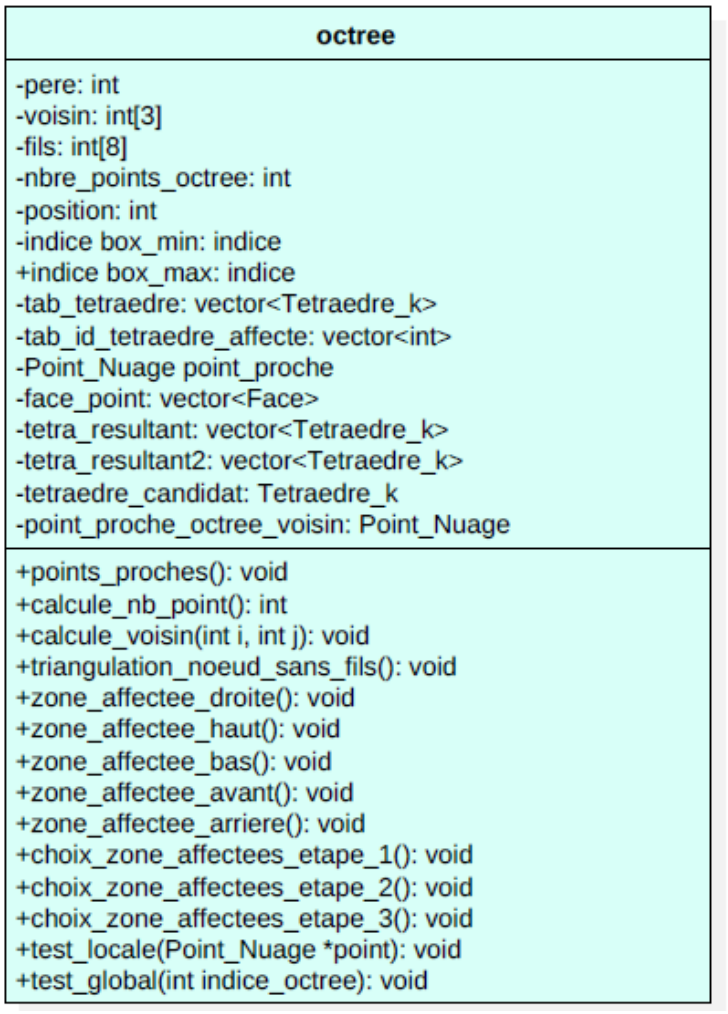
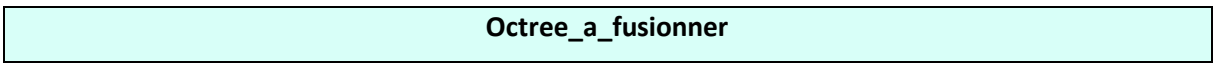


Figure 21. Classe Octree.

❖ **Classe Octree_a_fusionner :** cette classe est décrite par plusieurs méthodes en plus des méthodes de la classe **Octree**, parmi ses méthodes :

- dessiner_tetraedre_RBT(int i): qui dessine un tétraèdre RBT (premier type du tétraèdre générique).
- tetraedre_generique_R(): permet de générer un tétraèdre RBT.
- tetraedre_generique_L(): permet de générer un tétraèdre LBT.



-p1_zone_proche: point_multi -p2_zone_proche: point_multi -xmax: double -ymax: double -zmax: double -xmin: double -ymin: double -zmin: double -xmoy :double -ymoy :double -zmoy :double -face_non_sommets_communs: Face_fusion -tab_tetraedre_arete: vector<Tetraedre_k> -tab_tetraedres_fusion: vector<Tetraedre_k> -tab_faces_premier_tetra: vector<Face_fusion> -tab_toutes_aretes_communes: vector<arete> -tab_toutes_aretes_zone_1: vector<arete> -tab_toutes_aretes_zone_2: vector<arete>
+maj_visibilite_voisins(Face_fusion Face, tab_tetraedres_sommet_commun): void +dessiner_aretes_LBT(point_multi p_zone_proche, tab_toutes_aretes_zone): void +mise_a_jour_generique_LBT(Face_fusion face, tab_toutes_aretes, tab_faces_communes): void +tetraedre_LBT(Face_fusion face, tab_toutes_aretes_zone tab_tetraedre_indice_octree) :void +suppression_tetraedre(arete a, tab_faces_zone_affectee, tab_faces_communes): void +dessiner_tetraedre_RBT(int i): void +tetraedre_generique_R(): void +tetraedre_generique_L(): void +construire_face(double A, double B, double C, Point_Nuage p1, Point_Nuage p2): void +mise_a_jour_faces(Face_fusion FACE, tab_faces_communes, int num): void +point_sommet(in Point_Nuage *pp, in Point_Nuage* pp0): bool +f_tetraedre_final(): void +determiner_zone_affectee_X(): void +determiner_zone_affectee_Y(): void +determiner_zone_affectee_Z(): void +test_point_tetraedre(in Tetraedre_k tetraedre, in tab_points_zone): bool

Figure 22. Classe Octree_a_fusionner.

- **Classe Tetraedre_k** : cette classe regroupe la définition d'un tétraèdre. Parmi les fonctions de la classe Tétraèdre_k :
  - creer_tetraedre (Point_Nuage *p1, Point_Nuage *p2, Point_Nuage *p3, Point_Nuage *p4) : cette fonction retourne un tétraèdre.

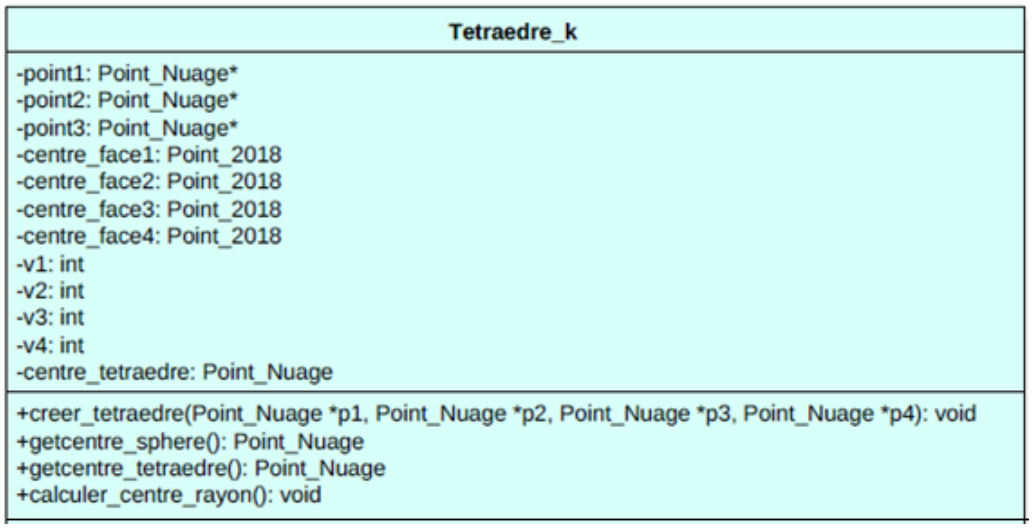


Figure 23. Classe Tetraedre_k.

- **Classe Face_fusion :** elle regroupe un ensemble de points, des arêtes, normale, etc. Ses différents attributs permettent de caractériser une face d'un tétraèdre.

Face_fusion
- Point1 :Point_Nuage - Point2 :Point_Nuage - Point3 :Point_Nuage - Point4_hors_face : Point_Nuage - voisin : int - premier_tetraedre_associe :Tetraedre_k - LBT_associee : Tetraedre_k -tetraedre_pere : Tetraedre_k - a1 : arete - a2 : arete - a3 : arete -moy_x : double -moy_y : double - moy_z : double - indice_pere : int - visible : bool - normal : Normale
+ modifier_parametres_faces_voisines (bool a, int mere) : void + modifier_parametres_face (bool a, int mere) : void + comparer_faces (Face_fusion face) : bool + calcul_centre_face () : void + gettetraedre_pere () : Tetraedre_k +dessiner_faces_rendu () : void

Figure 24. Classe Face_fusion.

- **Classe Normale :** cette classe donne la normale d'une face.

Normale
-nx : double -ny : double -nz : double
+ Normale () : void + dessiner_Normale () : void + getnx () : double + setnx (double xx) : void

Figure 25. Classe Normale.

- **Classe Arete :** cette classe permet de décrire les arêtes d'une face donnée.

Arete
-Point1 :Point_Nuage -Point2 :Point_Nuage -face_mere : int -arete_visible :bool -moy_x : double -moy_y : double -moy_z : double
+comparer_arete (arete a) : bool + calculer_milieu_arete () : void +dessiner_arete () : void

Figure 26. Classe Arete.

Conclusion

Dans ce chapitre, la conception de l'application développée a été présentée. Cette présentation a été entamée par la définition de la problématique du projet, ainsi que les différents objectifs visés à atteindre. Ensuite, la présentation des solutions adoptées avec les différents organigrammes et diagrammes de conception ont été détaillés en focalisant sur le concept de la modélisation orienté objet et le langage de modélisation UML.

Le prochain chapitre sera réservé à l'implémentation et à la validation de l'application.

CHAPITRE III

Tests et Validation

Introduction

Dans le cadre d'optimiser les différentes procédures de reconstruction d'objets en 3D à partir d'un nuage de points non structuré, une application logicielle graphique et interactive a été intégrée à l'environnement de production d'objets de formes complexes développé par l'équipe CFAO du CDTA. Le travail réalisé peut être visualisé par la présentation de l'application conçue, de l'environnement de développement et des différents outils utilisés, ainsi qu'une évaluation du système à travers un exemple de test.

Ce dernier chapitre est consacré à la présentation des résultats obtenus après l'implémentation de notre stratégie dédiée à l'optimisation de la phase de la fusion des différentes triangulations d'un nuage de point donné.

1. Implémentation

Par le biais d'Embarcadero C++ Builder Seattle et la plateforme Windows 8.1 la réalisation de la stratégie proposée a été développée. En utilisant un ensemble de fenêtres Windows on peut traiter l'interaction entre l'utilisateur et le système conçu et visualiser tous les objets manipulés par la bibliothèque OpenGL.

1.1. Langages de programmation utilisés

Le travail réalisé a été implémenté en langage C++.

Langage C++ : apparu au début des années 90, le langage C++ est actuellement l'un des plus utilisés dans le monde, aussi bien pour les applications scientifiques que pour le développement des logiciels. En tant qu'héritier du langage C, le C++ est d'une grande efficacité du fait qu'il a en plus, des fonctionnalités puissantes, comme par exemple la notion de classe, qui permet d'appliquer les techniques de la programmation-objet [1].

C++Builder : c'est un environnement de développement rapide d'applications qui permet aux développeurs de logiciels de développer du code avec une vitesse et une productivité supérieures. L'environnement est fourni avec un certain nombre de composants qui rendent le codage logiciel plus simple et plus rapide. C ++ Builder est destiné à la plupart des plateformes et systèmes d'exploitation modernes tels qu'Android, iOS, Windows et OS X [2].

Embarcadero C ++ Builder 10 Seattle : est le moyen le plus rapide de créer et de mettre à jour des applications riches en données, hyper connectées et visuellement

engageantes pour Windows 10, Mac, Mobile, IoT et plus encore en utilisant le standard C ++. Mettez à jour rapidement et facilement les applications VCL et FMX vers Windows [3].

OpenGL (Open Graphics Library) : est l'interface de programme d'application (API) standard de l'industrie informatique qui permet de définir des images graphiques 2D et 3D. Avant OpenGL, toute entreprise développant une application graphique devait généralement réécrire la partie graphique de celle-ci pour chaque plateforme de système d'exploitation et devait également connaître le matériel graphique. Avec OpenGL, une application peut créer les mêmes effets dans n'importe quel système d'exploitation [15].

1.2. Matériel utilisé

La réalisation de notre travail a été accomplie en utilisant le matériel suivant :

Un PC sous Windows 8.1, Intel® Core™ i5-3210M CPU @ 2.50GHz 2.50 GHz 4Go de RAM. La configuration minimale pour l'exécution de l'application développée est la suivante :

- RAM : 02 Go
- Carte graphique : NVIDIA
- Processeur multi-cœurs.

2. Présentation des fenêtres

Notre application est composée de plusieurs onglets, chacun présente une tâche précise. La fenêtre principale de notre projet est composée de six (06) onglets (Figure 1 et Figure 2).

- Onglet 1 : Lecture du nuage de points.
- Onglet 2 : Création des boxes.
- Onglet 3 : Création des octrees.
- Onglet 4 : Triangulation de Delaunay.
- Onglet 5 : Ordre de fusion.
- Onglet 6 : Zones affectées.

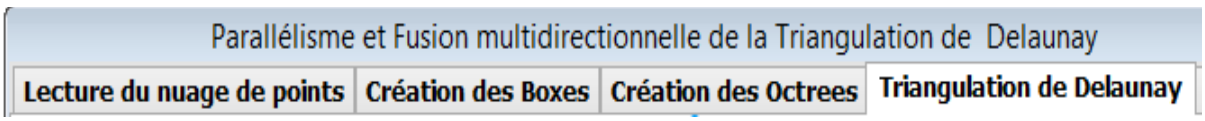


Figure 1. Fenêtre principale de l'application (1).

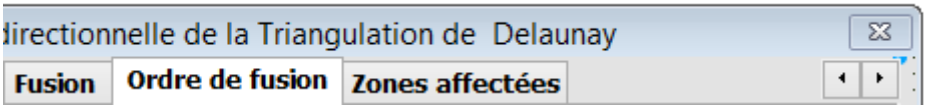


Figure 2. Fenêtre principale de l'application (2).

2.1. Lecture du nuage de points

Cet onglet permet de visualiser le nuage de points choisi avec son brut en trois étapes :

- Lecture du fichier contenant le nuage de points par un clic sur le bouton « ouvrir le fichier de points ».
- Calcul des coordonnées du brut (X_min, X_max, Y_min, Y_max, Z_min, Z_max).
- Calcul des dimensions du brut (Hauteur, Largeur et Longueur) et même le nombre de points du nuage est calculé et affiché (Figure 3).

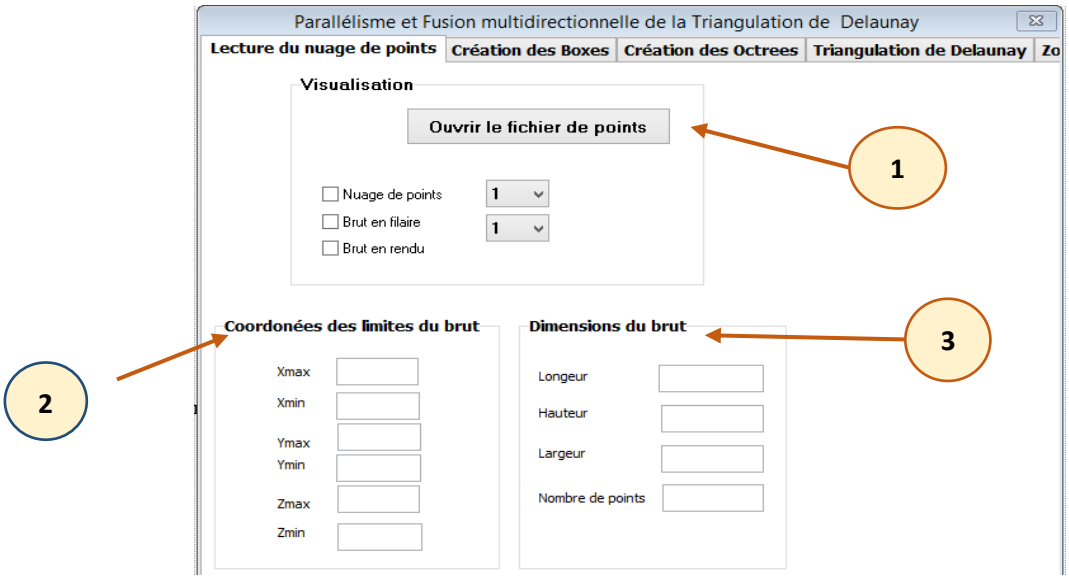


Figure 3. Onglet « Lecture du nuage de points ».

2.2. Création des boxes

Dans cet onglet, l'utilisateur a la possibilité de créer les boxes suivant deux modes :

- Création des boxes en mode séquentiel.
- Création des boxes en mode parallèle.

L'utilisateur a le choix entre la visualisation de tous les boxes ou d'un seul box.

Après la phase de création des boxes, les points du nuage de points sont affectés aux boxes correspondants. L'affection peut se faire en deux modes :

- Affectation séquentielle des points.

- Affectation parallèle des points.

Une fois l'opération est terminée, le temps du traitement est affiché (Figure 4).

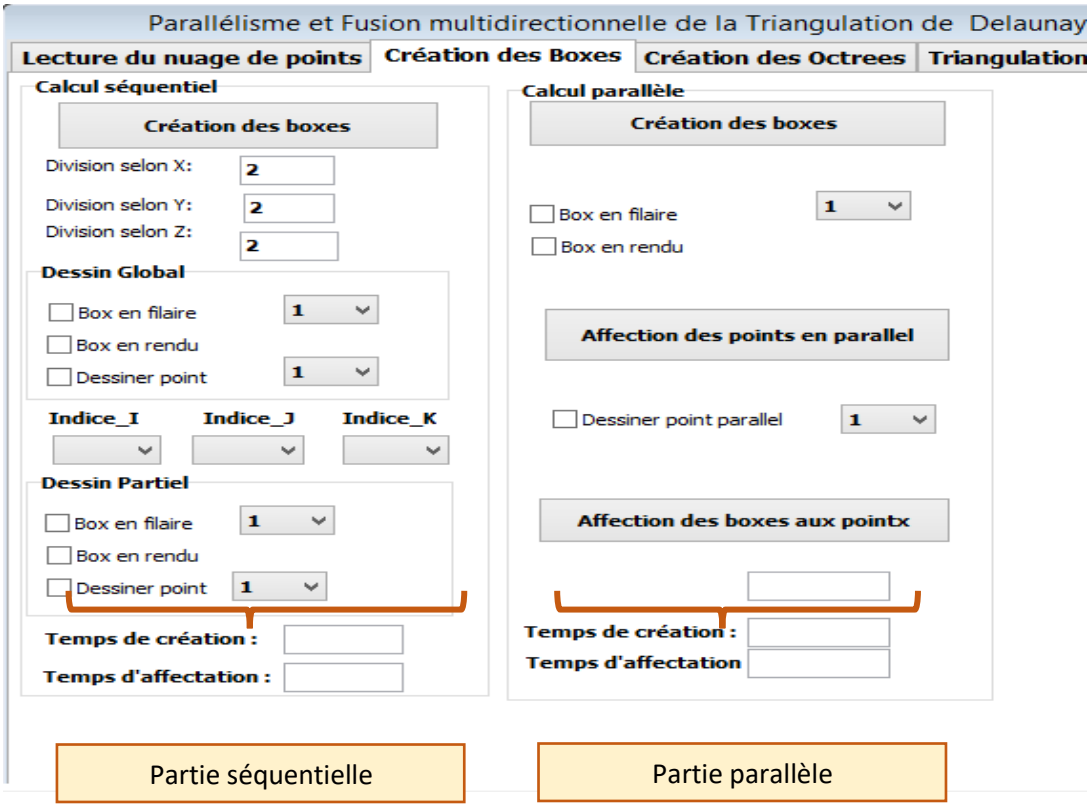


Figure 4. Onglet « Création des boxes ».

2.3. Création des Octrees

Cet onglet est dédié à la création des octrees avec la spécification de nombre de points de chaque Octree de la part de l'utilisateur par un simple clic sur le bouton « Création des Octrees ». L'utilisateur a la possibilité de choisir entre plusieurs modes de visualisation :

- Visualisation de tous les Octrees.
- Visualisation d'un seul Octree.
- Visualisation de chaque Octree avec son Octree père, fils et voisin

Après cette phase, les points du nuage de points et les boxes sont affectés aux Octrees.

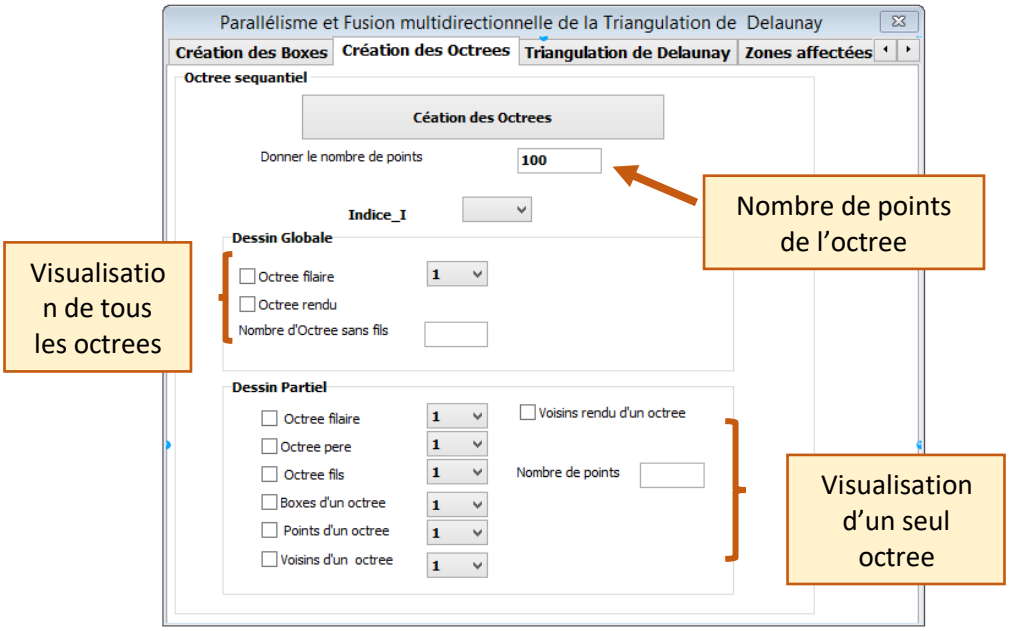


Figure 5. Onglet « Création des Octrees ».

2.4. Triangulation de Delaunay

Cet onglet est divisé en deux grandes parties de telle manière que l'utilisateur a le choix entre deux modes de triangulation :

- Triangulation séquentielle de Delaunay.
- Triangulation parallèle de Delaunay.

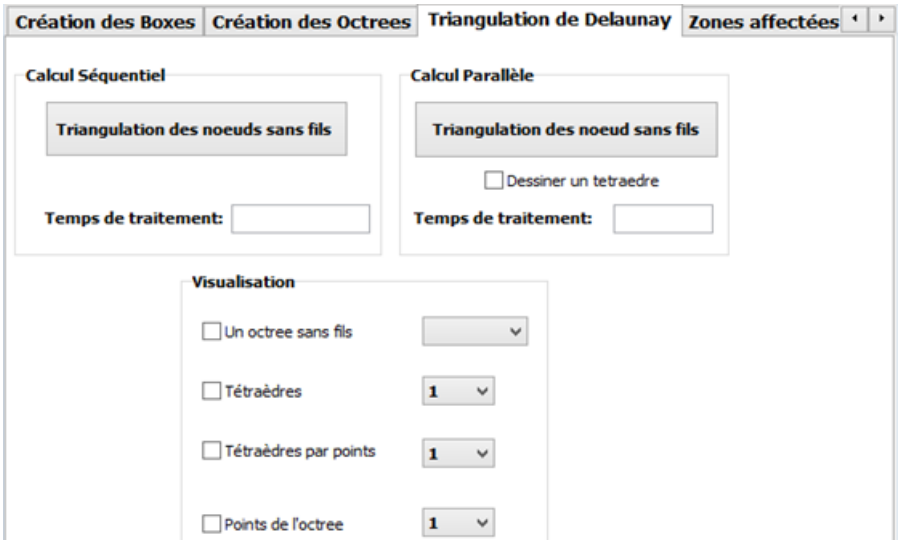


Figure 6. Onglet « Triangulation de Delaunay des Octrees ».

Une fois la triangulation est obtenue, le temps de traitement est calculé. Dans les deux cas, l'utilisateur peut visualiser partiellement la triangulation d'un seul Octree donné.

2.5. Ordre de fusion

Cet onglet a pour but de montrer la manière de fusion des Octrees après l'étape de la triangulation de Delaunay. La fusion est réalisée selon l'ordre des axes X, Y puis Z partiellement et globalement. L'utilisateur peut visualiser suivant les trois directions :

- L'indice du premier Octree à fusionner.
- L'indice du deuxième Octree à fusionner.
- L'indice de l'Octree père.
- Le niveau de la fusion.
- La direction de la fusion.

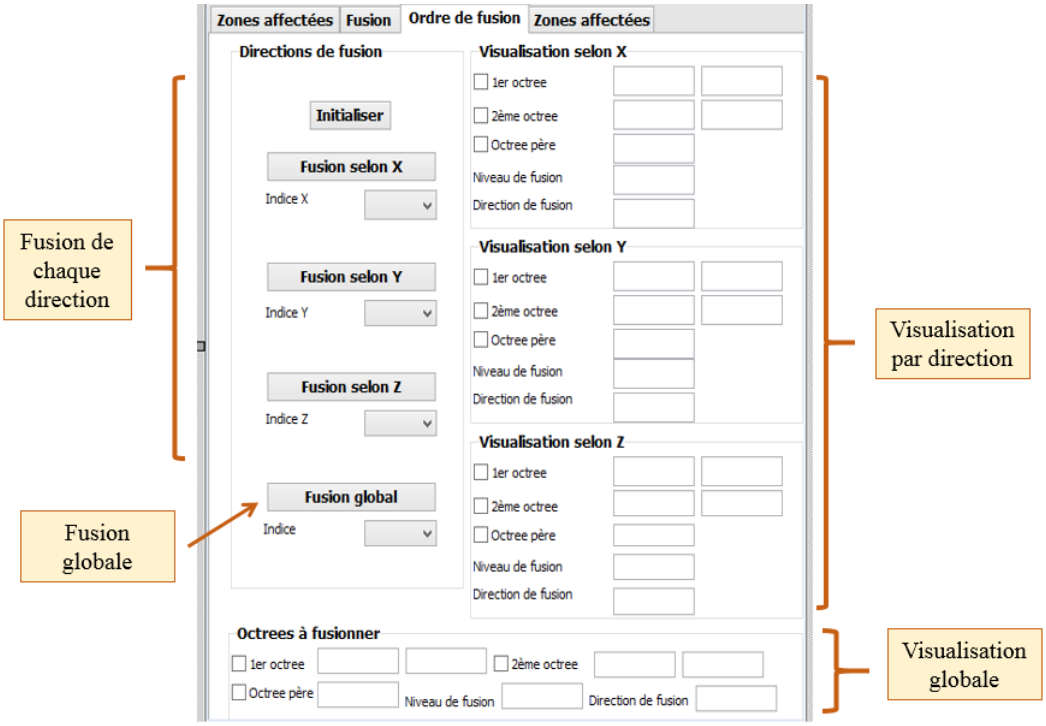


Figure 7. Onglet « Ordre de fusion ».

2.6. Zones affectées

Cet onglet contient les étapes nécessaires à la fusion. Il est divisé en quatre parties (04) et chaque paire d'Octrees suit le processus de fusion manuellement :

2.6.1. Direction de fusion

Cette partie a pour but de spécifier la direction de fusion choisie X, Y, ou Z.

L'utilisateur peut visualiser l'ensemble des tétraèdres de chaque zone affectée de deux Octrees candidats pour la fusion, ainsi que les deux points proches dans chaque zone affectée.

2.6.2. Premier tétraèdre :

Dans cette partie, les deux tétraèdres candidats pour la procédure de la fusion sont visibles, ainsi que le choix du premier tétraèdre, sa sphère et ses faces.

2.6.3. Visualisation des Octrees et des zones affectées

Cette partie est consacrée à la vision des :

- Tétraèdres de deux zones affectées.
- Faces de deux zones affectées.
- Points de deux zones affectées.
- Arêtes et faces communes aux points proches.

2.6.4. Tétraèdre générique

Le but de cette partie est de traiter la procédure de la fusion. Elle permet de :

- Visualiser le tétraèdre « RBT » et « LBT ».
- Visualiser le tétraèdre générique.

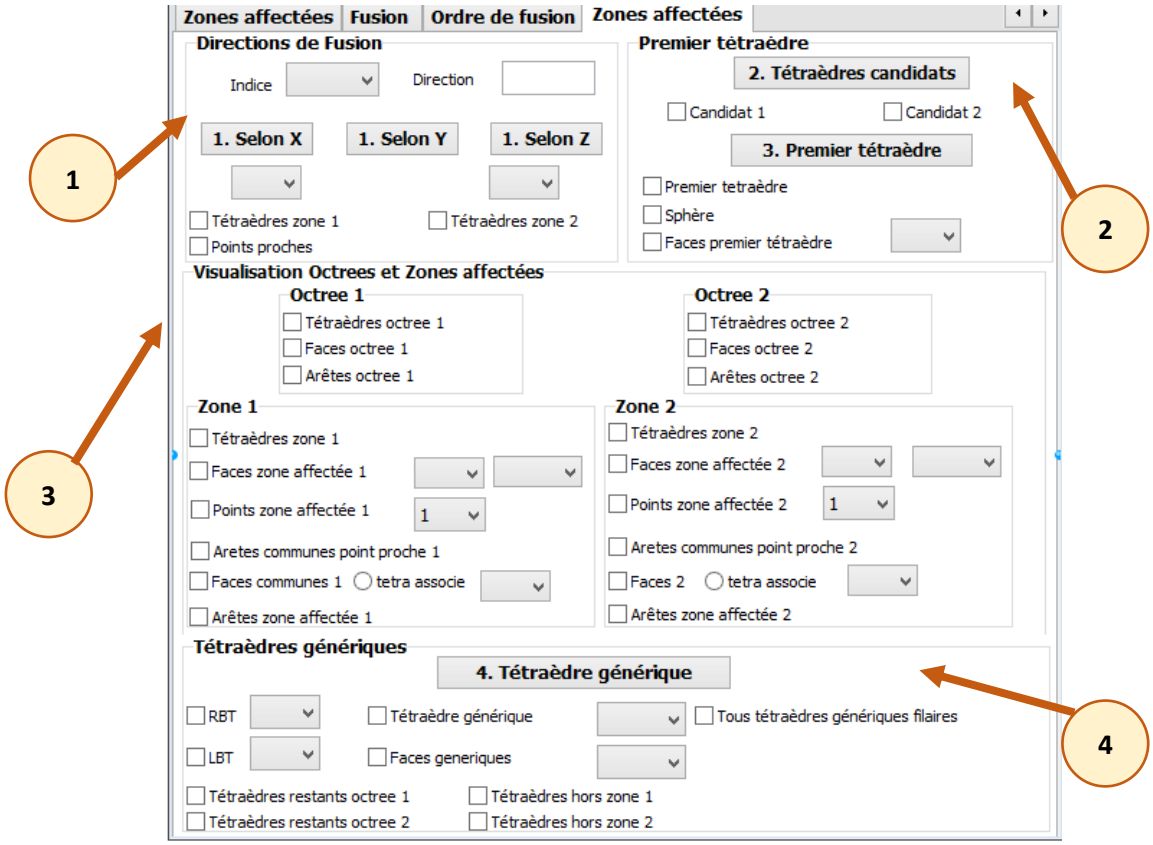


Figure 8. Onglet « Zones affectées ».

La figure qui suit montre la partie automatique de l'exécution de la génération du premier tétraèdre et des zones affectées.

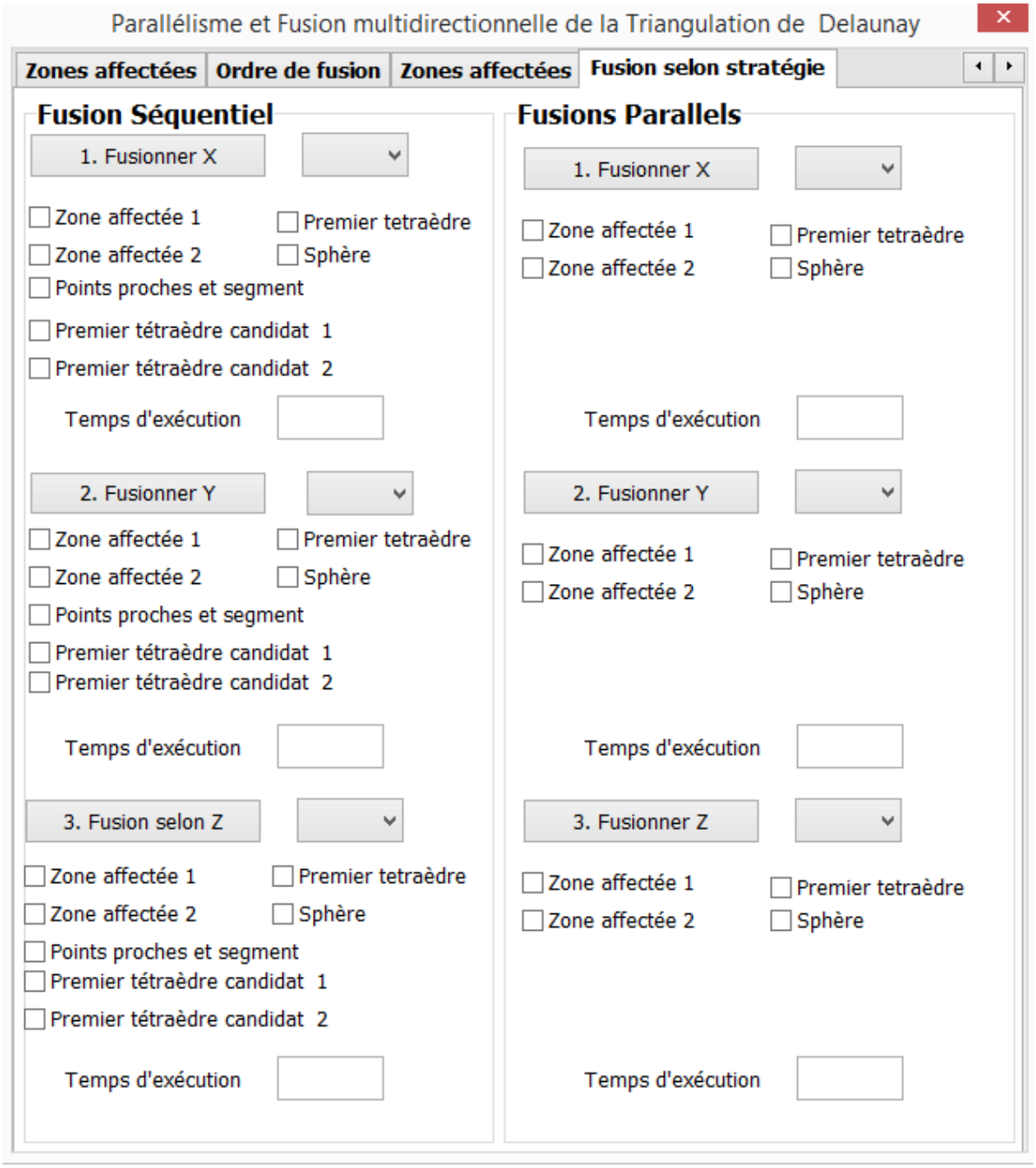


Figure 9. Génération du premier tétraèdre et des zones affectées

3. Validation

La validation de ce travail a été menée sur l'exemple du nuage de points présenté dans la Figure 10. Le traitement est effectué sur ce nuage de points récupéré de la digitalisation d'un objet depuis la lecture du fichier jusqu'à l'obtention d'un modèle STL.

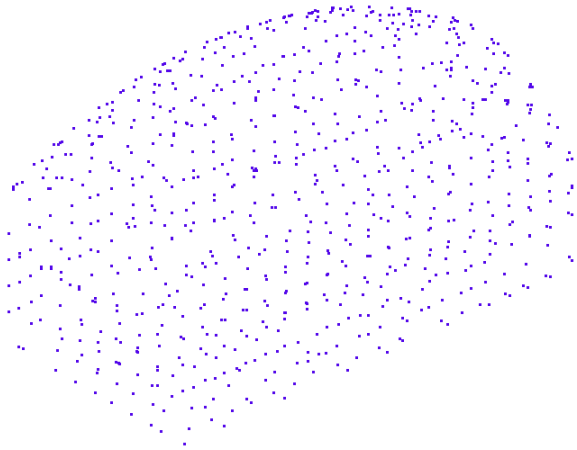


Figure 10. Nuage de points de l'objet.

3.1. Lecture du nuage de points

La première étape consiste à lire le nuage de points présenté, à calculer, afficher et visualiser ses limites et ses dimensions. Ce fichier contient un nombre de points égal à 809.

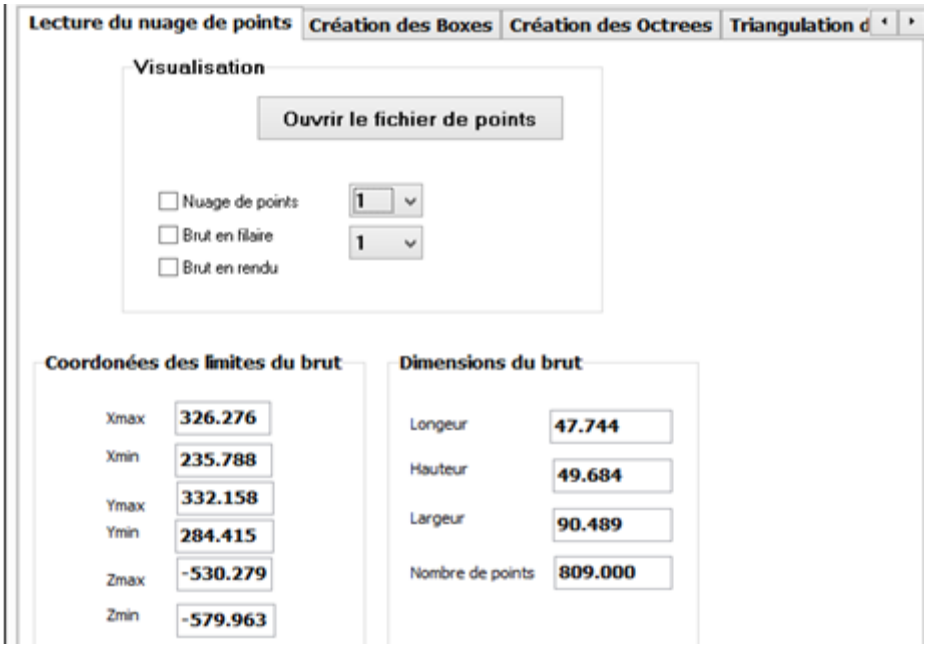
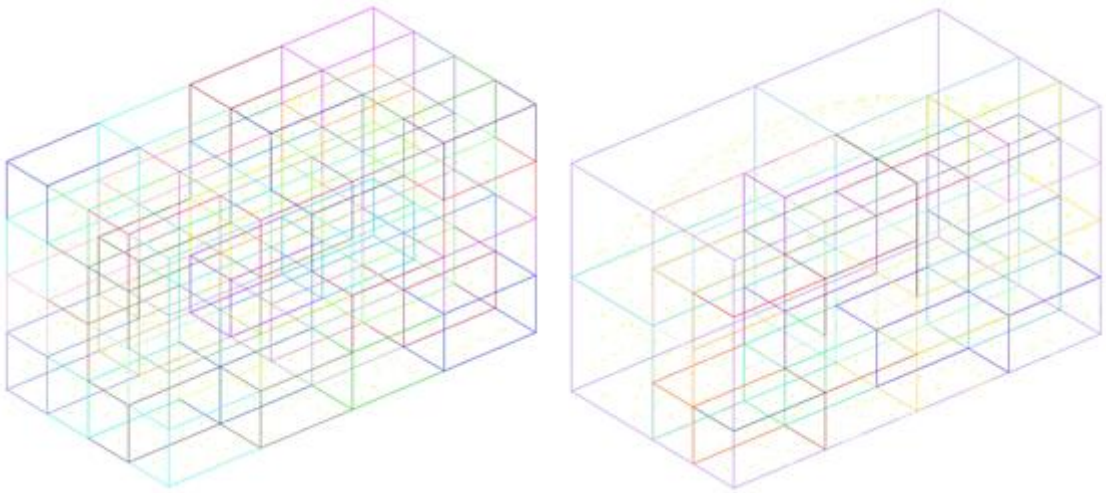


Figure 11. Résultat de lecture du nuage de points

3.2. Subdivision du nuage de point

La subdivision du nuage de point selon le paradigme « Divide and Conquer » a été effectuée en créant 2^n boxes dans les trois directions X, Y et Z avec n=2. Le seuil fixé pour la création des octrees est égale à 100 points par Octree. Le résultat obtenu est représenté par la Figure 12.



a. Subdivision en boxes. b. Subdivision en Octrees.

Figure 12. Subdivision du nuage de points.

3.3. Triangulation

Le paradigme de « Divide and Conquer » permet de traiter la triangulation dans chaque Octree dans le mode séquentiel ou dans le mode parallèle.

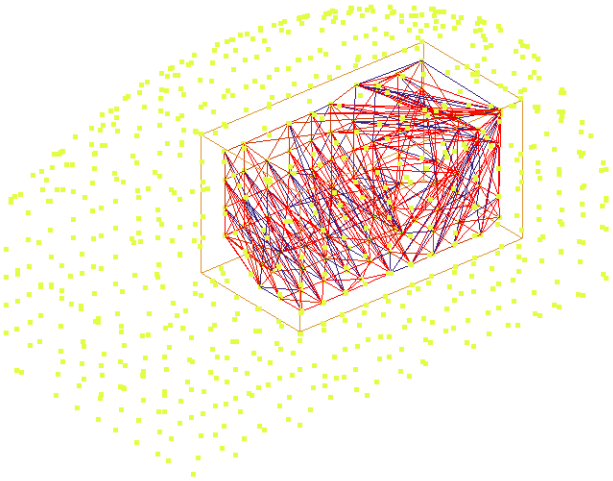


Figure 13. Visualisation de la triangulation d'un Octree sans fils.

3.4. Fusion multidirectionnelle des triangulations

Dans le chapitre précédent, la stratégie de fusion adoptée a été présentée. La mise en œuvre de cette stratégie avec les paramètres utilisés (2^2 boxes, 100 points par octree) est illustrée dans l'exemple suivant. Pour cet exemple et ces paramètres l'arbre de subdivision est donné par les figures suivantes.

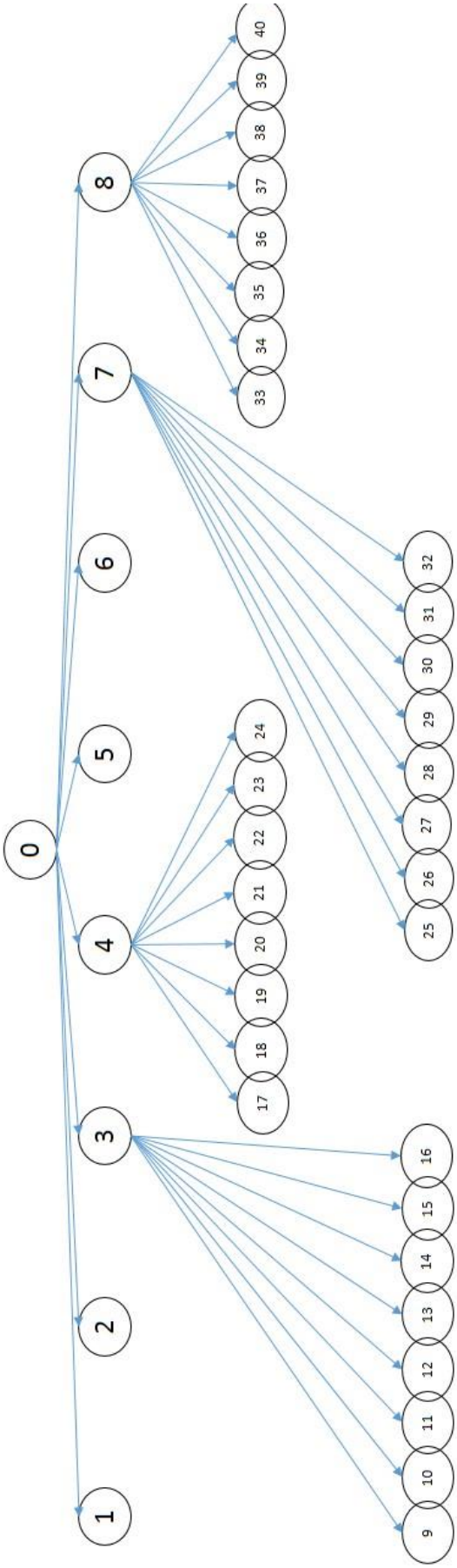


Figure 14. Arborescence de subdivision.

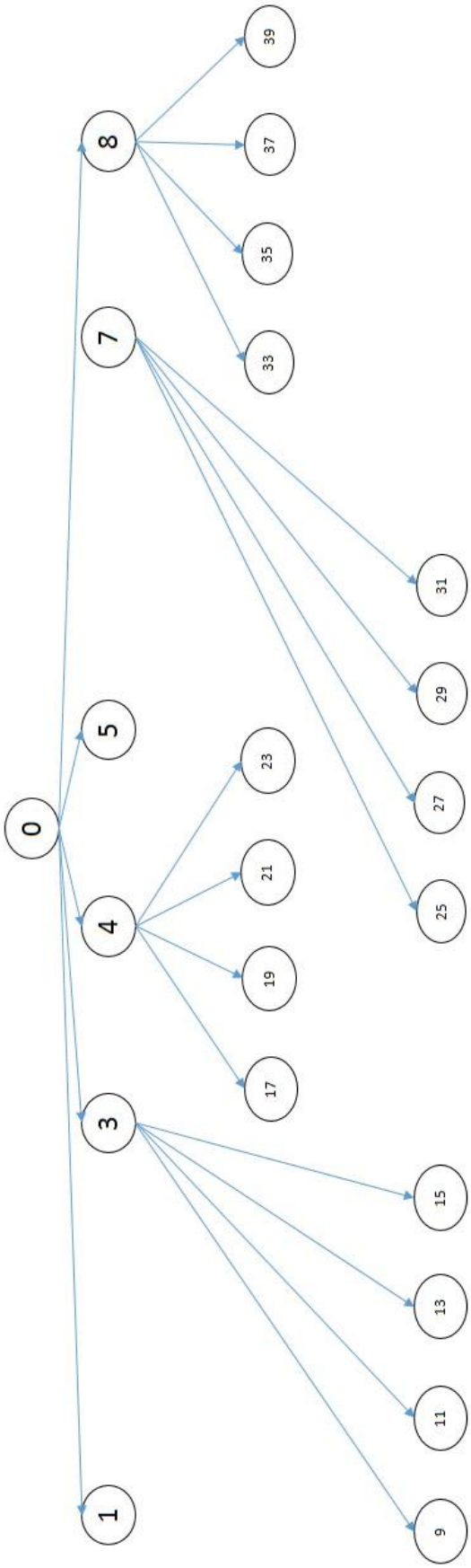


Figure 15. Arborescence après le premier niveau de fusion selon la direction X.

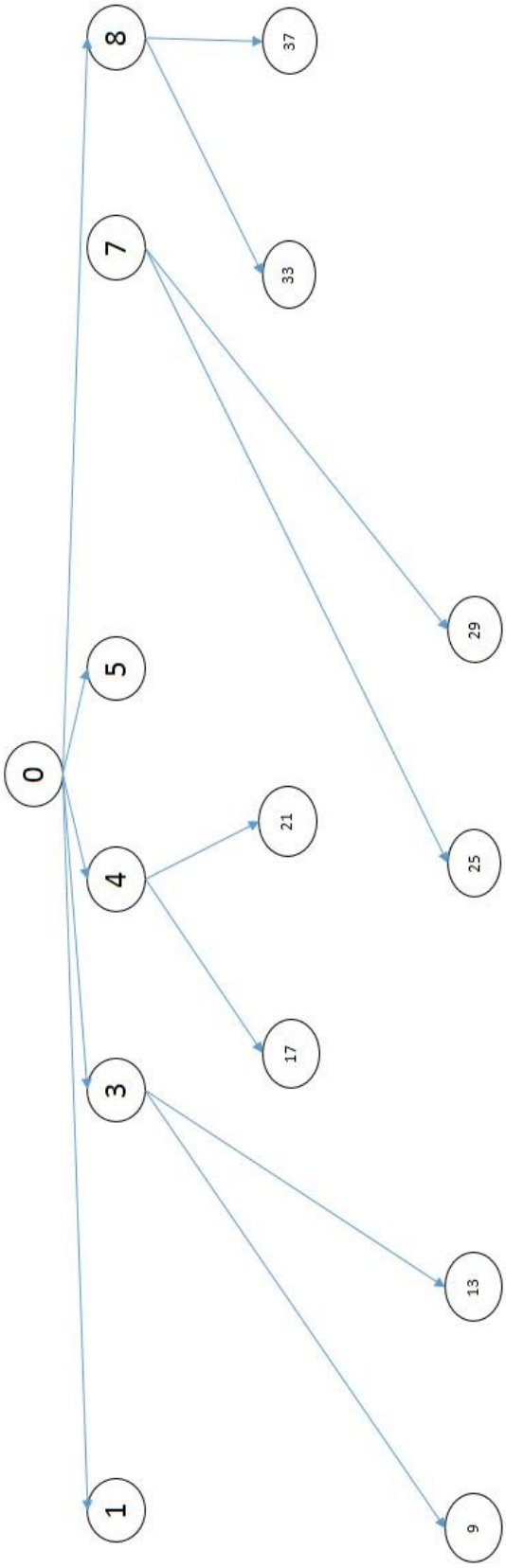


Figure 16. Arborescence après le deuxième niveau de fusion selon la direction Y.

Les fils sont stock

Figure 17. Arborescence après le troisième niveau de fusion selon la direction Z

Lors de la fusion selon l'axe Z, l'Octree fils restant qui regroupe toutes les précédentes fusions est stocké à son tour dans son Octree père (Figure 17).

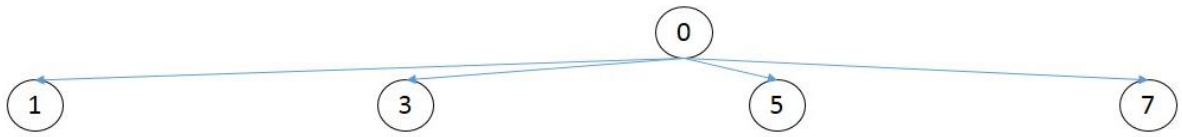


Figure 18. Arborescence après le quatrième niveau de fusion selon la direction X

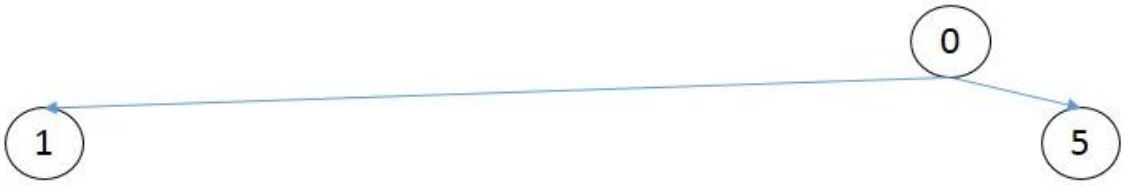


Figure 19. Arborescence après le cinquième niveau de fusion selon la direction Y.

Le tableau suivant décrit les indices des Octrees à fusionner selon chaque direction et chaque niveau en suivant la stratégie de fusion décrite précédemment.

Tableau 1. Indice des Octrees à fusionner selon les différentes directions.

Niveaux / Directions	Indice des octrees à fusionner
Niveau 1 – Direction X	1-2, 5-6, 9-10, 11-12, 13-14, 15-16, 17-18, 19-20, 21-22, 23-24, 25-26, 27-28, 29-30, 31-32, 33-34, 35-36, 37-38, 39-40
Niveau 2 – Direction Y	9-11, 13-15, 17-19, 21-23, 25-25, 29-31, 33-

	35, 37-39
Niveau 3 – Direction Z	9-13, 17-21, 25-29, 33-37
Niveau 4 – Direction X	3-4, 7-8
Niveau 5 – Direction Y	1-3, 5-7
Niveau 6 – Direction Z	1-5

3.4.1. Fusion manuelle

La fusion se fait au clic pour chaque indice de pair d'octrees.

a. Zones affectées

Pour chaque paire d'Octrees à fusionner, il y a deux zones affectées associées aux deux Octrees. La Figure 20 montre ces deux zones pour l'Octree d'indice 1 et 2.

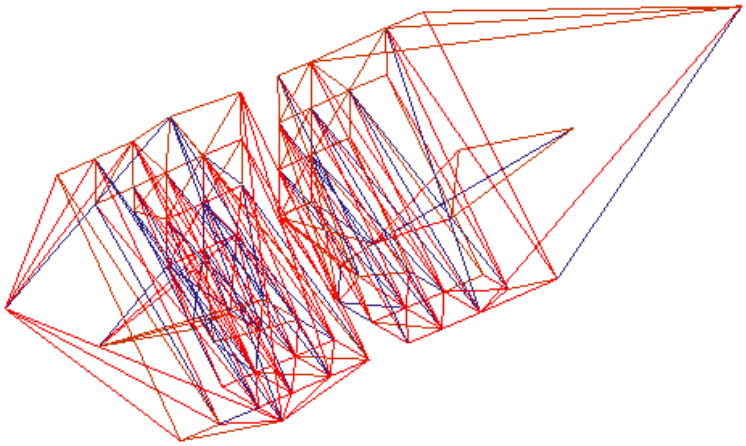


Figure 20. Zones affectées de l'Octree 1 et 2 en 3D.

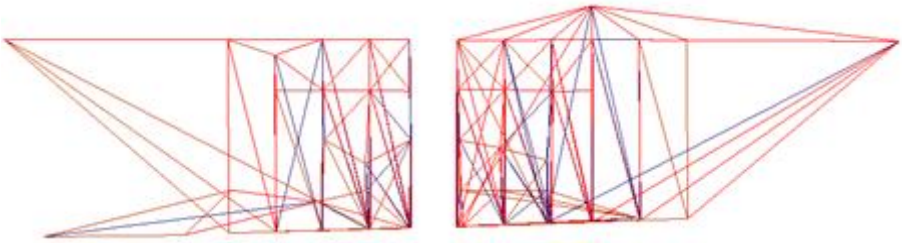


Figure 21. Zones affectées associées aux deux Octrees en projection XY.

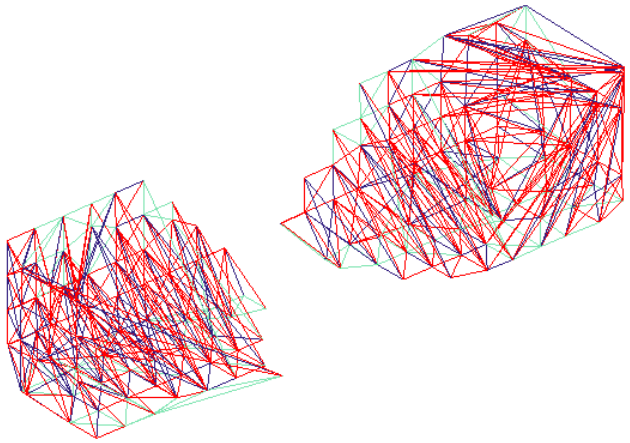


Figure 22. Tétraèdres hors zones affectées

b. Calcul des points proches

Pour effectuer la fusion multidirectionnelle de la triangulation de Delaunay en 3D, il est important de calculer les deux points les plus proches entre les deux Octrees et ce, pour n'importe quelle direction.

Dans le cas des Octrees d'indices 1 et 2, la Figure 23 montre les deux points les plus proches de chaque sous nuage de points ainsi que le segment les reliant.

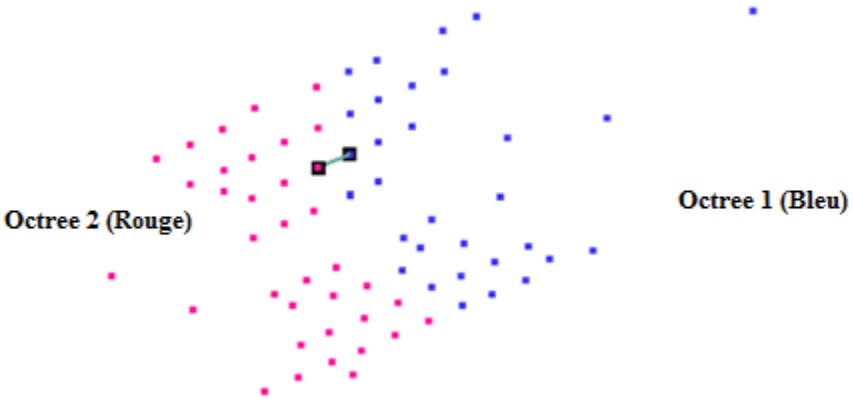


Figure 23. Segment reliant les points les plus proches.

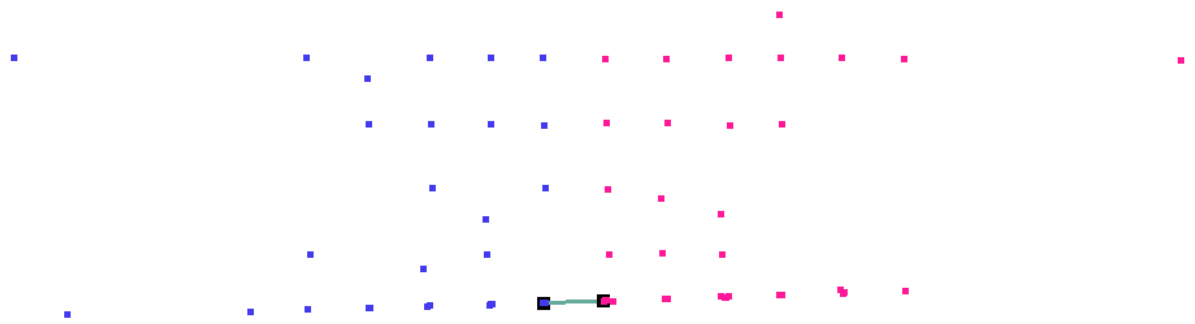


Figure 24. Points les plus proches en projection XY.

c. Premier tétraèdre

La détermination du premier tétraèdre passe d'abord par la récupération des tétraèdres qui ont un sommet commun avec le point le plus proche du même Octree ainsi que les faces des tétraèdres visibles. Les faces visibles sont les faces ou les côtés des tétraèdres sans voisins.

Figure 25. Différentes faces visibles de chaque Octree.

A chaque face visible est associé un premier tétraèdre candidat (Figure 26). Le premier tétraèdre de chaque Octree est choisi par rapport au critère de la sphère vide, ensuite par rapport au rayon de sa sphère. Celui ayant le plus petit rayon est choisi. La Figure 27 présente le premier tétraèdre de la zone affectée 1 et le premier tétraèdre de la zone affectée 2.

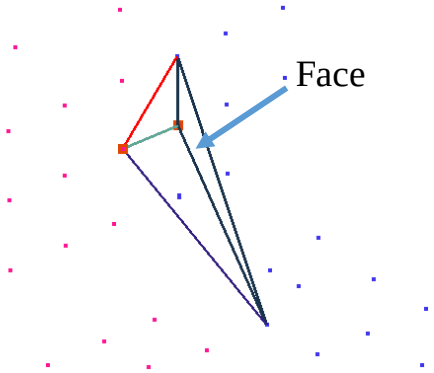


Figure 26. Un des premiers tétraèdres candidats associé à une face visible

Si les deux premiers tétraèdres des deux Octrees sont valides, le même principe est appliqué. Le tétraèdre ayant la plus petite sphère est le premier tétraèdre final (Figure 28).

Figure 27. Premiers tétraèdres de la zone 1 et 2 respectivement.

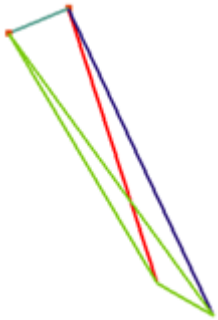


Figure 28. Premier tétraèdre final

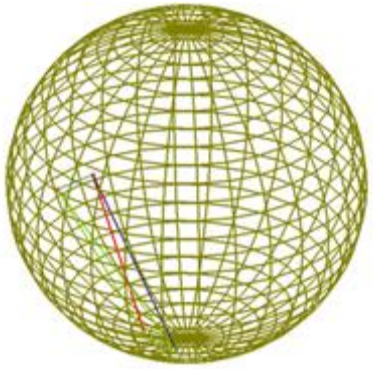


Figure 29. Sphère circonscrite du 1er tétraèdre.

La figure suivante montre les deux zones affectées ainsi que le premier tétraèdre final. C'est la base de la fusion entre les deux octrees.

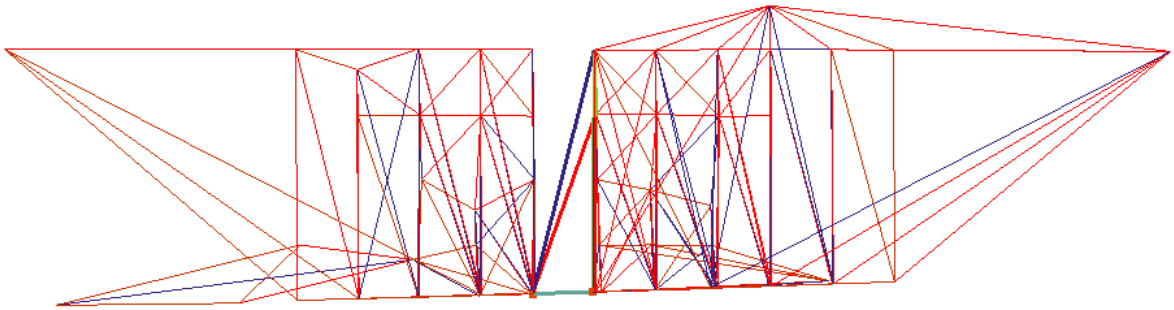


Figure 30. Premier tétraèdre fusionnant les deux Octrees en projection XY.

d. Tétraèdre générique

Chaque face visible du premier tétraèdre est une base pour le tétraèdre générique. Pour chaque face, il pourrait exister un tétraèdre « RBT » et un autre « LBT ».

Le tétraèdre « RBT » est construit avec les trois points de la face de base et un quatrième point de l'Octree opposé. Ce quatrième point appartient à une arête dont le deuxième sommet est commun avec l'un des sommets de la face de base.

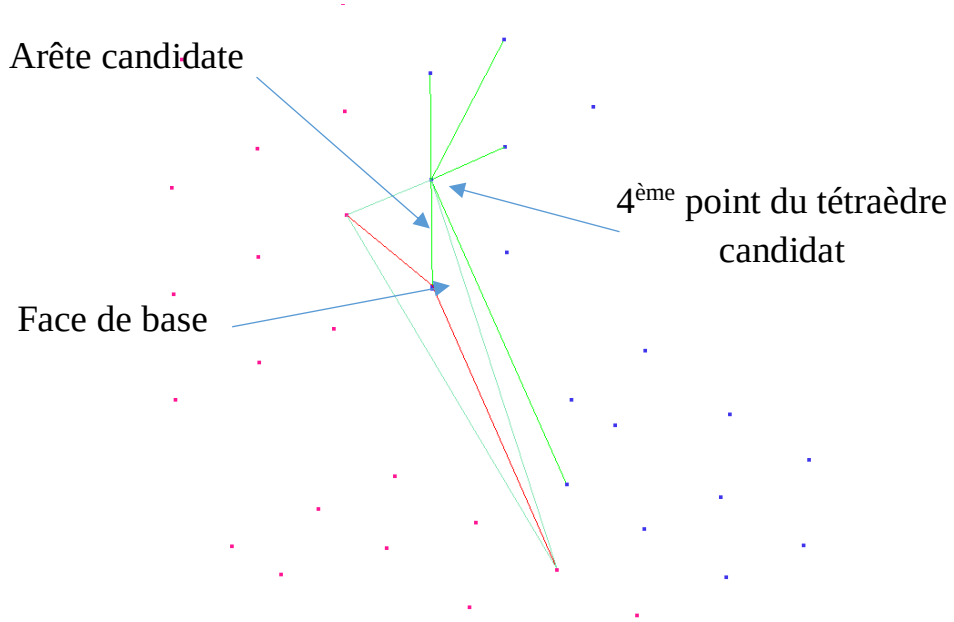


Figure 31. Tétraèdre générique « RBT ».

Le tétraèdre « LBT » est construit avec les trois points de la face de base et un quatrième point d'une face visible de l'Octree, ayant une arête commune avec l'une des arêtes de la face de base.

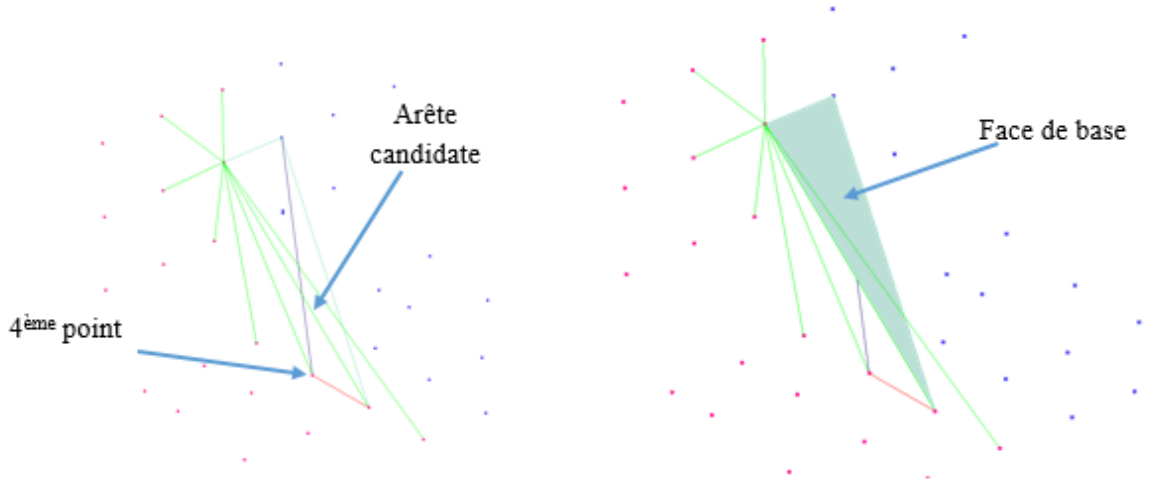


Figure 32. Tétraèdre générique « LBT ».

La technique adoptée pour déterminer laquelle des zones une ou deux est celle de « RBT » ou « LBT » est la suivante :

Pour une face donnée, si deux de ses sommets appartiennent aux points de la zone 1, on applique le test « LBT » sur les tétraèdres de la zone 1, si le troisième point est le seul appartenant à la zone 2, on applique le test « RBT » sur les tétraèdres de la zone 2. Inversement, si deux de ses sommets appartiennent aux points de la zone 2, le test « LBT » est

appliqué sur les tétraèdres de la zone 2 et le troisième point étant le seul appartenant à la zone 1, le test « RBT » est appliqué sur cette dernière.

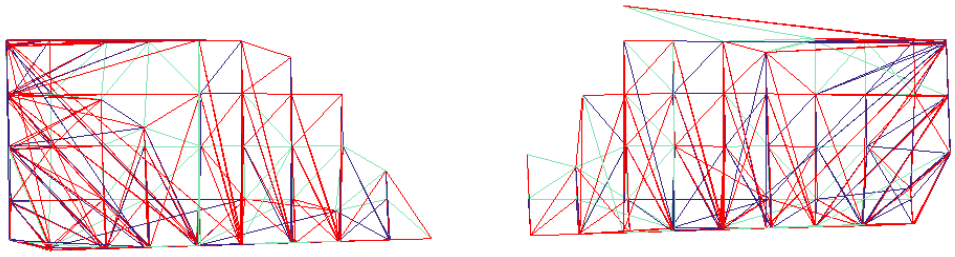


Figure 33. Tétraèdres hors zone affectée en projection XY.

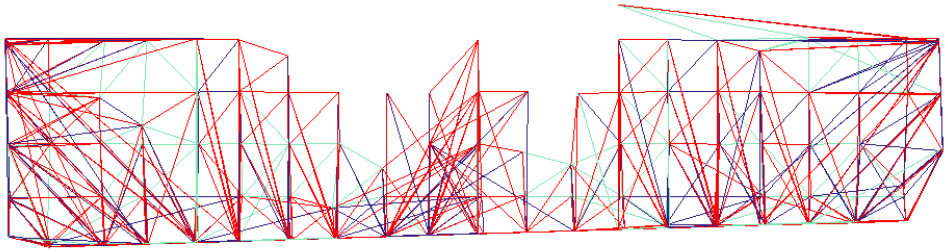


Figure 34. Début de fusion et tétraèdres hors zones affectées en projection XY.

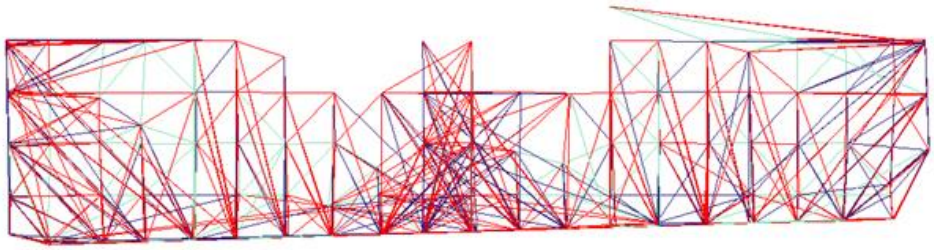


Figure 35. Avancement de la fusion entre les deux Octrees en projection XY.

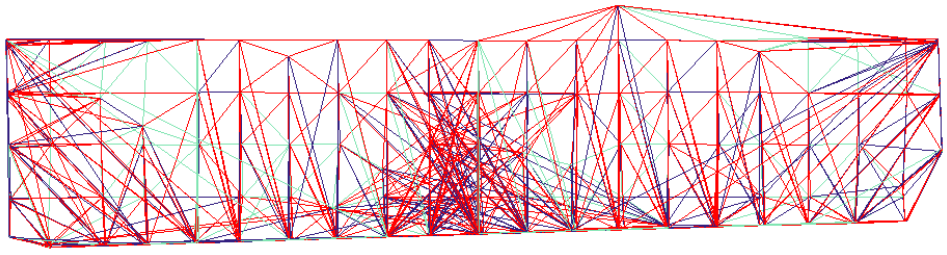


Figure 36. Fin de la fusion entre les deux Octrees en projections XY.

Le résultat de l'approche appliquée pour la fusion de l'Octree d'indice 1 et l'Octree d'indice 2 donnent un nombre de tétraèdres génériques égal à 308 tétraèdres

3.4.2. Fusion automatique

La fusion automatique se divise en deux parties, la fusion séquentielle, qui permet d'effectuer toutes les étapes de la fusion automatiquement. Néanmoins, cette étape n'est pas

achevée, seule la détermination des zones affectées et la génération du premier tétraèdre sont implémentées.

Conclusion

Dans ce chapitre, les différentes étapes de notre application ont été présentées.

Pour des problèmes lors de l'implémentation, la génération des tétraèdres génériques n'est pas encore complètement fiable.

La stratégie proposée est parallélisable, c'est une tâche à considérer en perspectives.

Conclusion générale

Le travail réalisé au sein de l'équipe Conception et Fabrication Assistées par Ordinateur «CFAO » de la Division Productique et Robotique du « CDTA », a pour but, la conception et l'implémentation d'un module logiciel graphique et interactif automatisant le processus de fusion multidirectionnelle des triangulations de Delaunay en 3D en utilisant le paradigme « Diviser pour Régner », pour générer la triangulation finale de l'objet sous l'environnement Embarcadero à partir d'un nuage de points quelconque.

Le premier chapitre de ce travail, a été consacré en premier lieu, à la structure du fichier STL, une étude générale sur le Reverse Engineering, ainsi que les différentes méthodes de reconstruction des modèles d'objets à partir d'un nuage de points quelconque. En deuxième lieu, une étude a été effectuée sur les triangulations de Delaunay ainsi qu'une présentation du paradigme « Diviser pour Régner ». Le deuxième chapitre a résumé la stratégie de concaténation et la conception de la solution proposées, et également l'architecture globale de l'application. Enfin, dans le troisième chapitre, les résultats de l'approche ont été détaillés et le travail a été validé à travers un exemple concret de nuage de points.

Le résultat de notre travail est une application qui répond aux besoins suivants :

- Proposition d'une stratégie de fusion en utilisant les Octrees et le paradigme « Diviser pour Régner ».
- Génération des zones affectées pour chaque Octree sans fils suivant l'ordre de fusion.
- Fusion par un premier tétraèdre pour chaque Octree sans fils selon une quelconque direction.
- Fusion par une succession de tétraèdres génériques pour concaténer les Octrees sans fils.

En perspective de notre travail, nous recommandons le traitement des points suivants :

- Finalisation de l'implémentation des tétraèdres génériques.
- La parallélisation de la stratégie de fusion proposée.
- Calcul parallèle en utilisant les processeurs des cartes graphiques « GPU ».

Références Bibliographiques

[1] «Conception et Fabrication Assistée par Ordinateur » Le matériel utilisé en CFAO : un SYSTÈME AUTOMATISÉ, Subtitel.

[2] Nastase-Dan CIOBOTA, « STANDARD TESSELLATION LANGUAGE IN RAPID PROTOTYPING TECHNOLOGY», 2012, Institut national de recherche et de développement pour la mécatronique et la technique de mesure Sos, l'Université VALAHIA - MATERIALS et MECHANICS, ROUMANIE

[3] Bensalem, Ziane, N. S. (2018). Parallélisme et fusion multidirectionnelle des triangulations de Delaunay. Mémoire. Université Saad Dahlab, Blida.

[4] Vinesh Raja, K. J. (2008). Reverse Engineering - An Industrial Perspective. Londres : Springer.

[5] En ligne : http ://ufrsciencestech.u-bourgogne.fr, OPTorganization infoiem. (Rprésentation des surfaces, date octobre 2017. url)

[6] A. Okabe, B. B. (1992). Spatial Tessellations: Concepts and Applications of Voronoi Diagrams. Chichester, Angleterre: John Wiley.

[7] LEMAIRE, C. (1997). Triangulation de Delaunay et arbres multidimensionnels. France.

[8] C.A. Wang. An optimal algorithm for greedy triangulation of a set of points. In Proceedings of the Sixth Canadian Conference of Computational Geometry, Saskatoon, Canada, pages 332-338, août 1994.

[9] J. D. Boissonnat, M. Teillaud. On the randomized construction of the Delaunay tree. In Theoret. Comput. Sci., 112, pages 339 - 355, 1993.

[10] C. A. R. Hoare, Algorithm 63 and algorithm 65, Communications of the ACM, 4(1961), pp. 321-322.

[11] En ligne : https://www.lucidchart.com/pages/fr/tutoriel-sur-les-diagrammes-de-classes (Tutoriel sur les diagrammes de classes, date 7 août 2019. url)

[12] En ligne : https://www.lucidchart.com/pages/fr/quest-ce-que-le-langage-de-modelisation-unifie (Qu’est-ce que le langage de modélisation unifié ?, date 7 août 2019. utl)

[13] Min-Bin Chen (2011) A parallel 3D Delaunay Triangulation Method. DOI : 10.1109/ISPA.2011.52

[14] Min-Bin Chen (2010). The Merge Phase of Parallel Divide-and-Conquer Scheme for 3D Delaunay Triangulation. DOI : 10.1109/ISPA.2010.71

[15] En ligne : https://whatis.techtarget.com/definition/OpenGL-Open-Graphics-Library (OpenGL Open Graphics Library, 4 septembre 2019)