

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

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

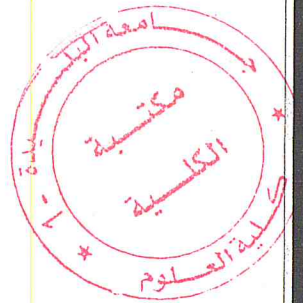


MEMOIRE DE FIN D'ETUDES

Pour l'obtention

D'un diplôme de master en informatique.

Spécialité : Système Informatique et Réseaux



THÈME :

***Parallélisation et Fusion
Multidirectionnelle de la Triangulation
de Delaunay***

Réalisé par :

Bensalem Nour EL-Houda

&&

Ziane Sofia

Soutenue le :28/06/2018

, devant :

Mr. OULD AISSA

Président

Mme. YKHLAFE

Examineur

Mr. KAMECHE Abdallah Hichem

Promoteur

Mr. BENDIFALLAH Mohamed El-Hassène

Encadreur

Mr. BEY Mohamed

Encadreur

Mme. TCHANTCHANE Zahida

Encadreur

2017/2018

Remerciement



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

Nous tenons à exprimer nos vifs remerciements et notre profonde gratitude à Mr. BENDIFALLAH, Mr. BEY et Mme. TCHANTCHANE de nous avoir encadré dans notre mémoire de fin d'études et de nous avoir dirigé dans notre travail.

Nous remercions également Mr Kamèche pour son aide et son encouragement.

Nous tenons à remercier aussi toute l'équipe du CDTA en particulier Mme. BOUAHADJA Khadidja et Melle. FERHAT Sahla.

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

Un grand merci également à nos familles pour leur soutien aussi bien moral que financier et pour leurs sacrifices.

Nous ne pourrions terminer sans remercier tous ceux qui ont participé d'une manière ou d'une autre dans l'élaboration de ce projet de fin d'études.

Merci !!!!



Dédicaces :

Je dédie ce modeste travail à :

*A l'homme de ma vie, mon exemple éternel, mon soutien moral et source de joie et de bonheur, celui qui s'est toujours sacrifié pour me voir réussir,
à toi mon père.*

A la lumière de mes jours, la source de mes efforts, la flamme de mon cœur, ma vie et mon bonheur ; maman que j'adore.

*A celui que j'aime beaucoup et qui m'a soutenu tout au long de ce projet
mon fiancé Chems Eddine*

*À ma très chère sœur Hasna <3
À mes très chères frères Housseem<3 et Hyder<3*

À toute ma famille : mes grands-parents, mes tantes, mes oncles et mes cousins et cousines en particulier Maria, Nour el Houda, Chaima, Marwa , Ghizlane, Meriem ,Yasser , Adem , Ayoub

*À la personne qui m'a soutenu toute l'année Mon binôme et ma sœur
« Nour El-Houda » merci beaucoup ma chérie.*

À tous les membres de l'équipe CFAO.

Mes amis : Nesrine, Imene ,Asmaa Cherif, Sabrina ,Lilia, Chanez, Asmaa ,Imene , Amina , Zienb, Ibtissem

À tous les étudiants de la promotion 2017/2018 SIR.

Enfin, Toute personne qui m'aime et que j'aime ...

SOFIA

Résume :

Les progrès de l'acquisition 3D ont conduit au développement de techniques de modélisation géométrique à partir d'un ensemble de points. Dans ce contexte, 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 sont saisies par un dispositif de mesure. Pour résoudre ce problème, nous avons utilisé le concept de la triangulation parallèle qui consiste à approximer le nuage de points par des éléments géométriques simples tels que les triangles, les tétraèdres, ...etc. Les nuages de points sont très denses et très volumineux, ce qui augmente les temps de traitement et par conséquent les coûts notamment lors de l'utilisation d'un seul processeur.

ملخص

قد أدى التقدم في ميدان ثلاثية الأبعاد إلى تطوير تقنيات النمذجة الهندسية من خلال مجموعة من النقاط التي أخذت كعينات من أسطح الجسم المادي. في هذا السياق، تهدف إعادة تركيب ثلاثي الأبعاد إلى إنشاء نموذج رقمي للسطح الخارجي لجسم مادي من خلال مجموعة نقط حيث يتم التقاط إحداثياته ثلاثية الأبعاد بواسطة جهاز قياس. لحل هذه المشكلة قمنا باستخدام طريقة التثليث المتزامن حيث، يتم تقريب المخطط بعناصر هندسية بسيطة مثل المثلثات ، رباعيات الأوجه ، ... إلخ. تتميز المجموعات لنقطية بالكثافة والضخامة جدًا مما يزيد من وقت المعالجة وبالتالي التكاليف خاصة إذا تم استخدام معالج واحد فقط

Summary :

Devices in 3D acquisition have led to the development of geometric modeling techniques from a set of points sampling the surfaces of a physical object. In this context, 3D reconstruction aims to generate a digital model of the outer surface of a physical object from 3D cloud of points.

To solve this problem, the cloud of points is approximated by simple geometric elements such as triangles, tetrahedra, etc. The point clouds are very dense and very bulky which increases the processing time and therefore the costs especially if only one processor is used.

Sommaire

Liste des figures	1
Liste des tableaux	3
Introduction générale.....	5
Chapitre I :Etat de l'art	
1.Introduction :	7
1.Conception et Fabrication Assistée par Ordinateur « CFAO »	7
1.1 Conception Assistée par Ordinateur(CAO).....	7
1.2Fabrication Assistée par Ordinateur(FAO).....	8
2. Formats d'échanges de données	8
2.1. STL (Standard Tessellation Language)	8
2.2. Caractéristiques de STL	9
2.3. Structure d'un fichierSTL	9
2.4 Différents formats d'échange de données	10
3. Etude des surfaces	11
3.1. Forme paramétrique.....	11
3.1.1. Les caractéristiques.....	11
3.2. Forme non paramétrique	12
4.Triangulation de Delaunay	12
4.1. les Avantages	12
4.2. Propriétés de Delaunay 2D	13
4.3. Structure de données_.....	14
4.4. Algorithmes de Delaunay_	14
4.4.1. Algorithme destruction construction	14
4.4.2. Algorithme de Crust.....	15
4.4.3. Algorithme de basculement (flip).....	15
4.5. Triangulation Delaunay en 3D	16
4.5.1. Principe de la triangulation en 3D	17
4.5.2. Algorithmes de Delaunay en 3D	17
5. Les architecteurs parallèle	19
5.1. Motivation pour le parallélisme.....	20
5.2. Fonctionnalité d'un système	20

5.3. Différents types de système d'exploitation	20
5.4. Etats d'un processus	21
5.5 . Sources de parallélisme.....	22
5.5.1. Parallélisme de contrôle	22
5.5.2. Parallélisme de données.....	22
5.5.3. Parallélisme de flux.....	22
5.6. Mesure de performance	22
Conclusion.....	23
Chapitre II : Conception de l'application	
Introduction :	25
1.Problématique	25
2. Architecture globale de l'application	25
2.1. Description de l'architecture globale de l'application.....	26
2.1.1. Lecture du nuage de points	26
2.1.2. Subdivision du nuage de points.....	27
2.1.3. Triangulation	31
2.1.4. Fusion.....	32
2.1.5. Premier tétraèdre.....	35

2.1.6. Tétraèdre Générique	36
3. Parallélisme	37
4. Modélisation de l'application avec UML.....	38
4.1. Méthode de modélisation	38
Conclusion.....	46
Chapitre III : test et validation	
Introduction	48
1. Implémentation	48
1.1 Langage de programmation	48
1.2 Matériel utilisé.....	49
2. Présentation des fenêtres.....	49
2.1. Lecture du nuage de points	50
2.2 Création des boxes.....	50
2.3 Création des Octrees.....	51
2.4 Triangulation de Delaunay.....	52
2.5 Zone affecter	53
2.6 Fusion.....	53
3 . Validation.....	54
Conclusion :	64
Conclusion générale	65
Références bibliographiques	67

Liste des figures

Figure. 1.1 : Composants de CFAO	p7
Figure. 1.2 : Passage du CAO au FAO	p9
Figure. 1.3 : Exemple d'un modèle STL.....	p9
Figure. 1.4 : Format ASCII	p10
Figure. 1.5 : Format Binaire	p10
Figure. 1.6 : Modélisation de surface	p11
Figure. 1.7 : Localisation d'un point sur une surface paramétrique	p12
Figure. 1.8: Triangulation quelconque et Triangulation de Delaunay	p13
Figure. 1.9 : Cercle circonscrit	p13
Figure. 1.10 : Propriété des angles	p13
Figure. 1.11 : Arête illégale.....	p14
Figure. 1.12 : Arête légale.....	p14
Figure. 1.13 : Insertion d'un point par l'utilisation de la structure de DAG	p14
Figure. 1.14 : Sélection des triangles non Delaunay	p15
Figure. 1.15 : Triangulation de Delaunay	p15
Figure. 1.16 : Diagramme de Voronoï	p15
Figure.1.17: Un côté légal	p16
Figure. 1.18 : Côté illégal (gauche), le flip (droite)	p16
Figure 1.19 : Triangle transformé en tétraèdre.....	p16
Figure. 1.20 : Cercle circonscrit transformé en Sphère circonscrite	p16
Figure. 1.21 : Flip 1-4 et Flip 4-1..14.....	p17
Figure. 1.22 : Flip 3-2 et Flip 2-3	p18
Figure. 1.23 : Flip 2-6 et Flip 6-2	p18
Figure. 1.24: Flip n-2n.....	p18
Figure. 1.25: Flip 4-4.....	p18
Figure.1.26 : Enveloppe convexe	p19
Figure. 1.27 : Diagramme d'état d'un processus.	p21
Figure. 1.28: Parallélisme de contrôle.....	p22
Figure. 1.29: Parallélisme de flux	p22
Figure. 2.1 : Architecture globale de l'application.....	P26
Figure. 2.2 : Organigramme de lecture du nuage de points.	P27
Figure. 2.3 : brut qui englobe le nuage de point.....	p27

Figure. 2.4 : Subdivision de brute en box	p28
Figure. 2.5 : Indexation d'un point dans le plan.....	p28
Figure. 2.6 : Arbre et subdivision de brut en Quadtree.	p29
Figure. 2.7 : Principe de l'octree.	p29
Figure. 2.8 : méthode de calcul des limites de l'octree en 2D et 3D.....	p31
Figure. 2.9 : Arbre de Quadtree (stratégie de triangulation et fusion (2D)).	p32
Figure. 2.10 : Organigramme de création du premier tétraèdre	p35
Figure. 2.11 : Etapes de création de 1er tétraèdre	p36
Figure. 2.12 : Organigramme de création du tétraèdre générique.	P37
Figure. 2.13 : Diagramme cas d'utilisation Générale.	P39
Figure. 2.14 : Diagramme cas d'utilisation Lecture de nuage de point	p40
Figure. 2.15 : Diagramme cas d'utilisation de Subdivision.....	P40
Figure. 2.16 : Diagramme cas d'utilisation de Triangulation	p41
Figure. 2.17 : Création du premier tétraèdre	p41
Figure. 2.18 : Diagramme de classe générale.....	p42
Figure. 2.19 : Classe Point_multi.	p43
Figure. 2.20 : Limite_brute_multi.	p43
Figure. 2.21 : Classe box_multi	p43
Figure. 2.22 : Classe octree	p44
Figure. 2.23 : Classe nuage_point_multi.....	p45
Figure 2.24: Classe Tetraedre_k.....	p45
Figure 3.1: Fenêtre principale de l'application..	p50
Figure 3.2: Onglet lecture du nuage de points	p51
Figure 3.3: Onglet Création des Boxes.	p51
Figure 3.4 : Onglet Création des Octrees..	p52
Figure 3.5: Triangulation de Delaunay.	p53
Figure 3.6: Onglet Zones affectées.....	p54
Figure 3.7 : Onglet Fusion.....	p54
Figure 3.8 : Modèle de validation... ..	p54
Figure 3.9 : Résultat de lecture de nuage de point.	p55
Figure 3.10: Visualisation de nuage de points..	p55
Figure 3.11 : Création des boxes.....	p56
Figure 3.12 : visualisation globale des boxes.....	p56
Figure 3.13 : Visualisation partielle des boxes.	P56

Figure 3.14 : création des Octrees.....	p57
Figure 3. 15 : Visualisation globale des octrees.....	p58
Figure 3. 16 : visualisation partielle d'un Octree.	p58
Figure 3. 17 : Visualisation du nuage de points	p58
Figure 3. 18 : Triangulation du nuage de points.	p59
Figure 3. 19 : Visualisation de la triangulation.	p59
Figure 3. 20 : Zone affectée gauche-droite (triangulation séquentielle).	p60
Figure 3. 21 : Zones affectées avant-arrière (triangulation séquentielle).....	p60
Figure 3. 22 : Zones affectées haut-bas (triangulation séquentielle).....	p61
Figure 3. 23 : Zone affectée mode parallèle.....	p61
Figure 3. 24 : Visualisation du premier tétraèdre.....	p62
Figure 3. 25 : Visualisation du premier tétraèdre (projection X,Y).	p63
Figure 3. 26 : Premier tétraèdre de division par 300 points dans chaque Octree.	p63

Liste des tableaux

Tableau. 1.1 : Caractéristiques des formats d'échange de données.	p10
Tableau .1.1: Tableau 1.2 : Sources du parallélisme.....	p23
Tableau .3.1: Comparaison entre temps de création parallèle et séquentiel.....	p56
Tableau 3.2 : Comparaison entre temps de d'affectation des points parallèles et séquentiels.....	p57
Tableau .3.3: Impact du parallélisme sur la triangulation..	p59

Introduction Générale

Avec le développement informatique, les techniques de la reconstruction des objets 3D deviennent des outils très répandus dans le monde industriel. Ces techniques ou méthodes permettent en effet de modéliser des phénomènes et d'en étudier les caractéristiques en minimisant le recours à l'expérience. À ce titre, les efforts énormes ont été engagés dans le développement de méthodes numériques fiables et efficaces capables de réaliser des simulations à grande échelle.

Cependant, les méthodes de la reconstruction des objets dans l'industrie conduisent le plus souvent à des solutions extrêmement coûteuses. Ce coût se mesure en termes d'espace mémoire et du temps de calcul nécessaires à la réalisation d'un calcul. En effet, il n'est pas rare aujourd'hui, de réaliser des calculs pouvant durer entre quelques heures et plusieurs semaines. Pour ce faire, le recours au calcul massivement parallèle est devenu une nécessité. Cependant, la conception et l'implémentation parallèle d'un outil permettant d'exploiter la puissance d'une machine n'est pas aussi simple que certains le pensent. En effet, le calcul parallèle permet de lever la barrière technologique imposée par les machines conventionnelles d'où ces techniques permettant d'optimiser ce coût tout en assurant la qualité des solutions.

Problématique

La reconstruction des objets 3D avec une bonne précision est une tâche ardue, coûteuse en temps. Elle dépend fortement de la modélisation disponible dans les systèmes CAO et exigeant une grande interactivité entre le logiciel et le concepteur. En outre, le modèle de l'objet qui en résulte est une approximation du nuage de points et nécessite très souvent des ajustements. Pour résoudre ce problème, le nuage de points est approximé par des éléments géométriques simples tels que les triangles, les tétraèdres, ... etc.

Les nuages de points sont très volumineux et nécessitent des calculs intensifs ce qui augmente les temps de traitement de la subdivision, de la triangulation et de la fusion, par conséquent les coûts pour la reconstruction des objets 3D augmentent surtout si un seul processeur est utilisé.

Objectif du travail :

Ce travail s'insère dans le cadre de développement de modules logiciels pour la production des surfaces de formes complexes initié 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 génération d'une triangulation approximant la peau extérieure d'objets de n'importe quelles formes à partir de nuages de points quelconques. Le travail consiste à proposer une méthode qui permet la réduction des temps de traitement en parallélisant les calculs sur plusieurs processeurs en utilisant le paradigme « Diviser pour Régner », Plus le développement d'un module logiciel graphique et interactif sous Windows automatisant cette tâche.

Chapitre I :

Etat de l'art

Chapitre I

Ces outils sont généralement spécialisés : outils aussi divers que la conception et le dessin architectural, le calcul de la résistance des matériaux pour les structures ou les assemblages mécaniques [5]. Nous trouvera des logiciels de CAO dans divers domaines : la mécanique, l'électronique, l'architecture, la construction, l'orthopédie et bien d'autres domaines [2].

1.2. Fabrication Assistée par Ordinateur « FAO » :

La Fabrication Assistée par Ordinateur « FAO » peut être définie comme le processus de passage d'une géométrie théorique spécifiée (pièce souhaitée) à une géométrie usinée conforme (pièce réelle). Il s'agit de fabriquer un produit en respectant des contraintes technologiques, économiques, géométriques et de qualité définies dans la fiche descriptive du produit. Pour ceci, il faut déterminer les méthodes de fabrication les plus performantes pour satisfaire ces contraintes et les exécuter dans un ordre logique et optimal. Les deux modules « Conception » et « Fabrication » communiquent entre eux et échangent des informations de natures complexes. Le module de fabrication, manipule des informations d'ordre géométrique et topologique issues du module de conception pour les transformer en format exploitable par le directeur de la commande numérique de la machine-outil. L'échange d'information entre ces deux modules, nécessite un canal de communication et un format d'échange de données standard [3].

2. Formats d'échanges de données :

Dans un système homogène où la CAO et la FAO sont intégrées dans un seul environnement, l'information se passe simplement. 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 [2]. Le format le plus courant dans le domaine de production est le format STL. (Figure 1.2)

2.1. STL (Standard Tessellation Language) :

STL (STereoLithography) appelé également « Standard Tessellation Language » est un format de fichier natif du logiciel de CAO créé par 3D Systèmes. 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 autre attributs communs du modèle CAO. [4]



Figure. 1.2: Passage de la CAO à la FAO.

2.2. Caractéristiques du fichier STL :

- Décrit la géométrie de la surface d'un objet 3D mais pas d'information concernant la couleur, la texture ou d'autres attributs du modèle de 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).
- 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 [5].

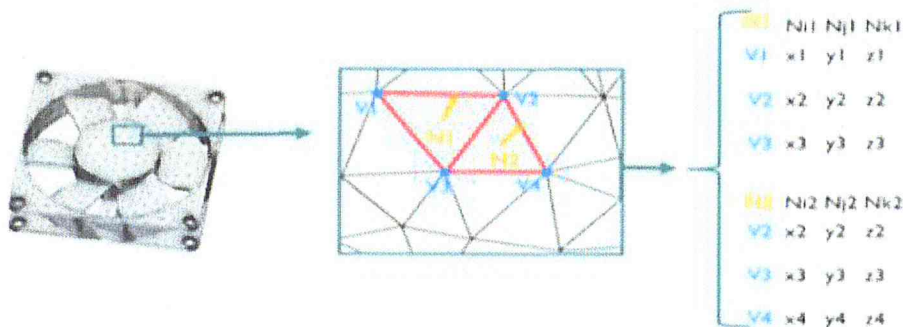


Figure .1.3 : Exemple d'un modèle STL.

2.3. 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 [4].

Chapitre I

- **Format ASCII :** un fichier STL ASCII commence par la ligne « *solid name* » où *name* est une chaîne 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 1.3).

- **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 au 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 [5].

```
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.1.4 : Format 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 .1.5 : Format binaire.

2.4. Différents formats d'échange de données :

Tableau 1.1 : Caractéristiques des formats d'échange de données. [21]

Fichier	Extension	Types	Format	Avantages	Inconvénients
IGES	.IGES, .IGS	ASCII	2D	Echange de plusieurs formes de données.	Sauvegarde de l'extérieur du solide

Chapitre I

STEP	.STEP, .STP	ASCII	3D	Interopérabilité entre logiciels Longévité de données Extensibilité Modularité	Perte d'information Difficultés d'intégration
STL	.STL	Binaire /ASCII	3D	Compact et simple Compatible avec de nombreux logiciels de modelage 3D Très répandu dans l'industrie	Pas d'informations de connectivité Fichiers volumineux

Les formats d'échange de données permettent de ne manipuler que les informations nécessaires lors du passage du modèle géométrique de la pièce au processus de réalisation. Néanmoins, le processus de conception de produits reste une tâche très complexe du fait qu'il fait appel à des techniques de modélisation de surfaces très avancées.

3. Etude des surfaces :

En CAO, la modélisation des surfaces de forme libre est devenue un maillon incontournable dans divers domaines industriels tels que la fabrication des moules, des formes aérodynamiques, des formes esthétiques, etc. La forme de la surface est déterminée à partir de sa représentation mathématique. En CAO, les surfaces sont classées en deux grandes catégories selon leurs représentations.

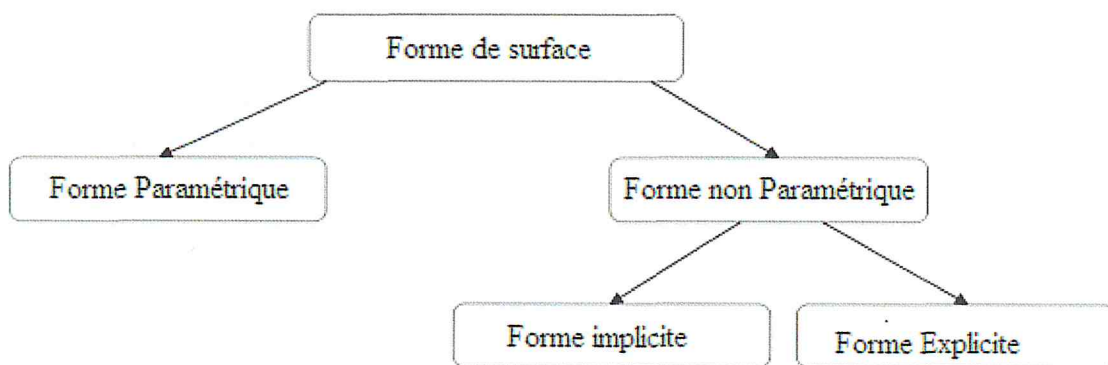


Figure.1.6 : Modélisation de surface.

3.1. Forme paramétrique

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

$$F(u, v) = (X(u, v), Y(u, v), Z(u, v))$$

Chaque point $F(u, v)$ de la surface paramétrique dépend des coordonnées paramétriques u et v (Figure 1.7). 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.

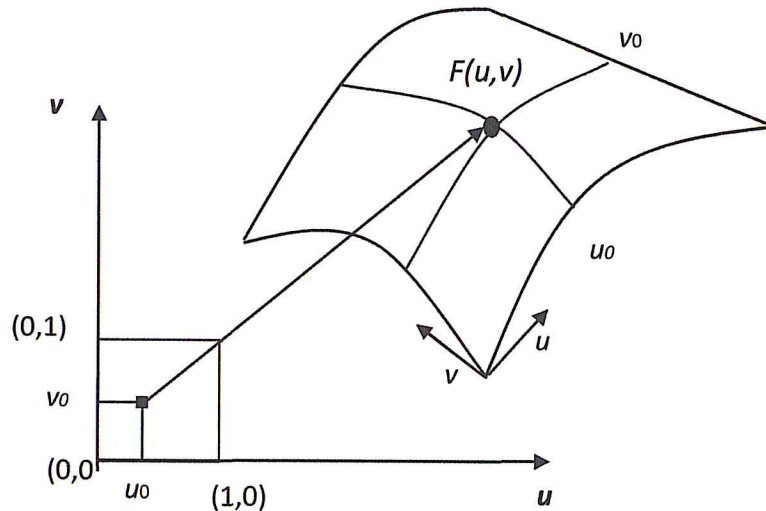


Figure 1.7 : Localisation d'un point sur une surface paramétrique.

3.2. Forme non paramétrique :

Ces surfaces sont représentées sous deux formes :

- **Surface implicite** : cette forme de représentation est très pratique pour les opérations d'intersections. Par contre, elle est terrible pour la visualisation. Elle est représentée par la forme suivante :

$$F(x, y, z) = 0$$

- **Surface explicite** : cette forme est définie comme suite :

$$Z = F(X, Y)$$

C'est Une fonction de deux variables. Cette forme ne permet pas de représenter des surfaces fermées dont l'utilisation en CAO est fréquente [6].

4. Triangulation de Delaunay :

Afin de générer les triangles du modèle STL, nous avons opté pour l'utilisation de la méthode de la triangulation de Delaunay. Cette technique consiste à partitionner un ensemble

de points positionnés dans un espace formé de triangles dont les sommets sont des objets, et qui à eux tous, constituent une partition de l'enveloppe convexe de ces objets [7].

4.1. Avantages :

L'utilité de la triangulation de Delaunay réside dans sa capacité à générer un pavage du plan suivant certaines contraintes. Elle trouve des applications en imagerie telles que la compression de données ou pour l'approximation de surfaces. C'est sur ce concept que nous allons nous baser pour développer une partie de notre travail. Le point commun avec notre application est qu'en choisissant judicieusement la densité du nuage de points, il est possible de concentrer les calculs sur les points d'intérêt (zone à forte densité de points) [7].

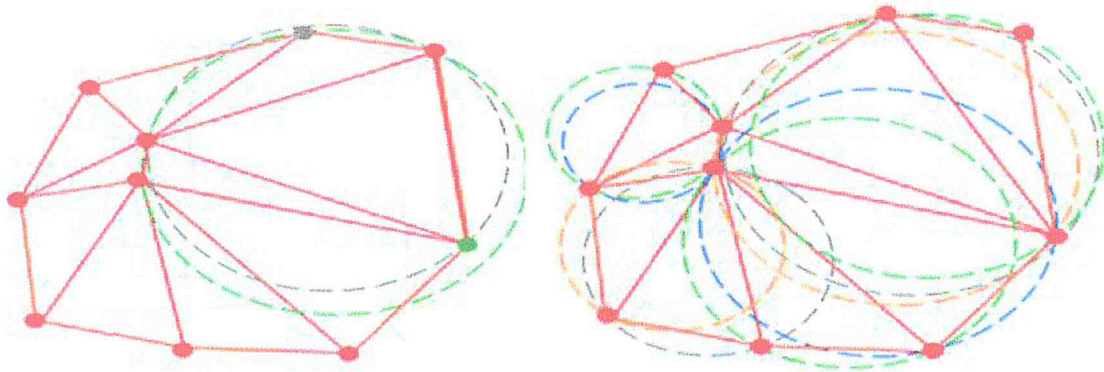


Figure .1.8 : Triangulation quelconque et Triangulation de Delaunay

4.2. Propriétés de Delaunay 2D :

Une triangulation de Delaunay 2D doit répondre à quatre propriétés :

- Chaque triangle est entouré d'un cercle vide qui passe par les sommets du triangle et ne contient aucun autre sommet à l'intérieur (Figure 1.9).

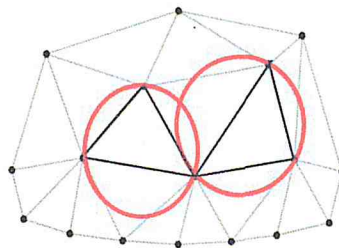


Figure .1.9 : Cercle circonscrit.

- Les sommets et les côtés ouverts de l'ensemble des triangles de Delaunay sont deux à deux disjoints.
- La triangulation de Delaunay maximise le minimum des angles des triangles.

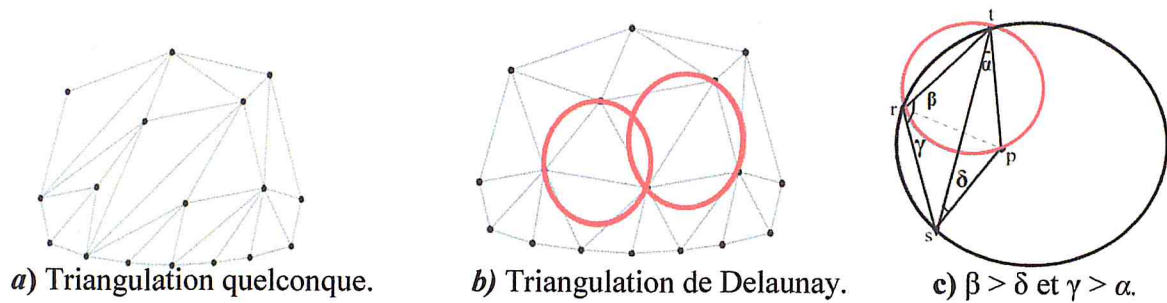


Figure .1.10 : Propriété des angles.

- La triangulation de Delaunay de S est la seule triangulation de S dont toutes les arêtes sont légales.

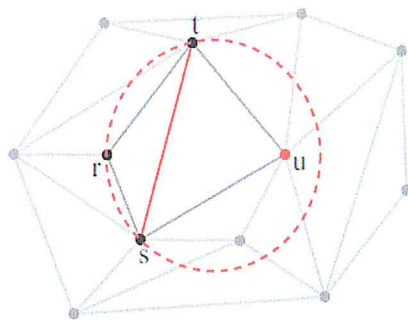


Figure .1.11 : Arête illégale

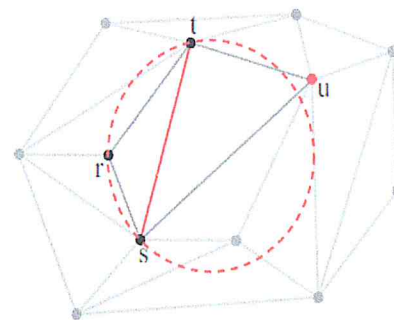


Figure .1.12 : Arête légale.

4.3. Structure de données :

- **D.A.G .(Directed Acyclic Graph) :** c'est une structure de données qui permet de trouver l'emplacement de chaque point et le triangle qui le contient en suivant les étapes suivantes :

- Chercher le triangle de la triangulation déjà construite qui contient x .
- Identifier le nombre k de triangles incompatibles avec un point x .
- Classer les triangles de T (**triangle fictif**) sous forme d'un graphe orienté sans cycles (un arbre avec plusieurs pères possibles) que l'on construit au cours de l'algorithme.
- A chaque ajout de point, le nombre d'un nœud est majoré par k [8].

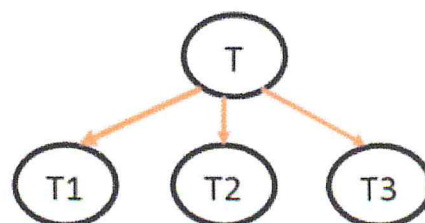
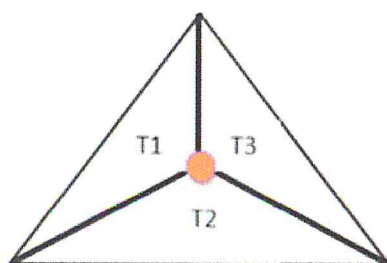


Figure .1.13 : Insertion d'un point par l'utilisation de la structure de DAG.

4.4. Algorithmes de Delaunay :

4.4.1. Algorithme destruction construction :

Après le choix d'un point dans le plan, cet algorithme permet de tester les triangles. S'il trouve un triangle en conflit, il passe aux autres jusqu'à ce qu'il ne reste que les triangles en conflits. Ensuite, il va supprimer tous les triangles en conflits et va relier tous les fins des triangles avec le point inséré (Figures 1.14 et Figure 1.15).

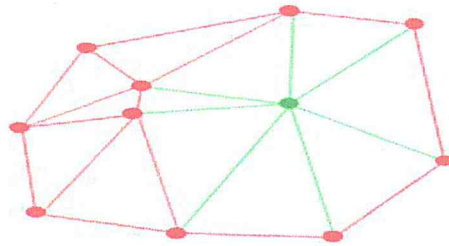
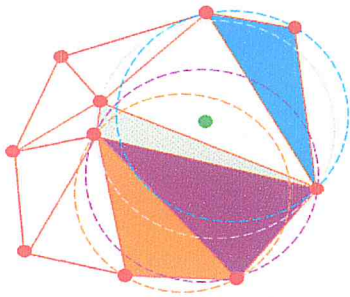


Figure .1.14 : Sélection des triangles non Delaunay. **Figure .1.15 :** Triangulation Delaunay.

4.4.2. Algorithme de Crust

L'algorithme de Crust permet de reconstruire des surfaces. Il s'appuie sur la triangulation de Delaunay et le diagramme de Voronoï [9].

- **Diagramme de Voronoï :** permet de partitionner un plan composé d'objets en zones où chaque zone est définie comme l'ensemble des points les plus proches d'un objet. Cette partition est appelée Diagramme de Voronoï. Chaque zone est appelée zone de Voronoï et elle représente un polygone convexe [10]. Les diagrammes de Voronoï permettent de représenter des relations de distance entre objets.

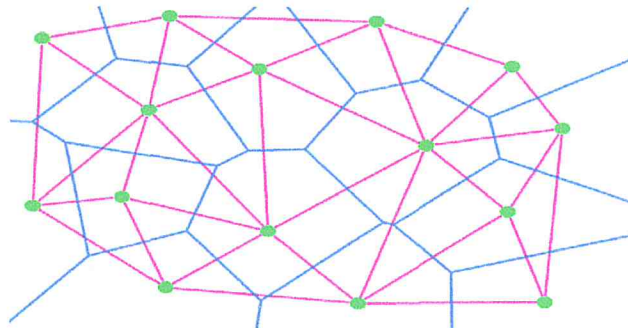


Figure .1.16 : Diagramme de Voronoï.

L'algorithme de Crust se compose de quatre (04) étapes :

Chapitre I

- Construire la triangulation de Delaunay sur le nuage de points.
- Calculer des pôles des cellules de Voronoi.
- Construire la triangulation de Delaunay de l'union des points du nuage et des pôles.
- Garder uniquement les triangles dont les trois sommets sont les points de l'échantillon de départ.

4.4.3. Algorithme de basculement (Flip) :

Cet algorithme est plus rapide que le premier. C'est une grande fonction récursive qui vérifie si les côtés sont légaux. Pour savoir s'il est illégal, nous regardons les points qui ne sont présents que dans un des triangles (donc les points opposés) (Figure 1.17).

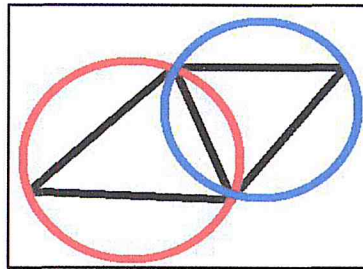


Figure.1.17 : Un côté légal.

Un côté est illégal si au moins un de ces deux points est contenu dans le cercle circonscrit du triangle opposé. Si c'est le cas, le côté illégal est remplacé par un côté relié par les deux points opposés. Par la suite un test est effectué sur les quatre côtés restant pour vérifier si ce changement a provoqué un autre conflit. [11]

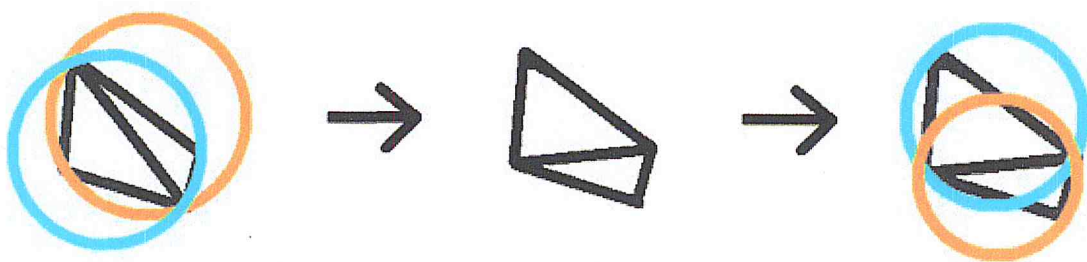


Figure 1.18 : Côté illégal (gauche), le flip (droite).

4.5. Triangulation Delaunay en 3D :

Le passage de la triangulation 2D à la triangulation 3D, consiste à transférer le triangle de Delaunay en un tétraèdre (Figure 1.19) et le cercle circonscrit en une sphère circonscrite (Figure 1.20).

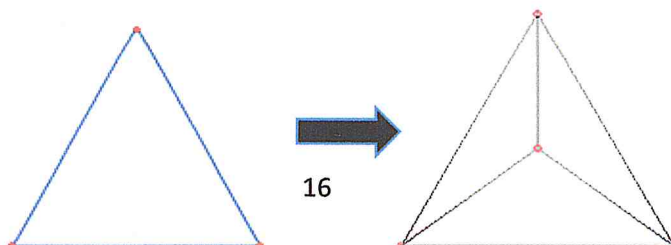


Figure 1.19 : Triangle transformé en tétraèdre.

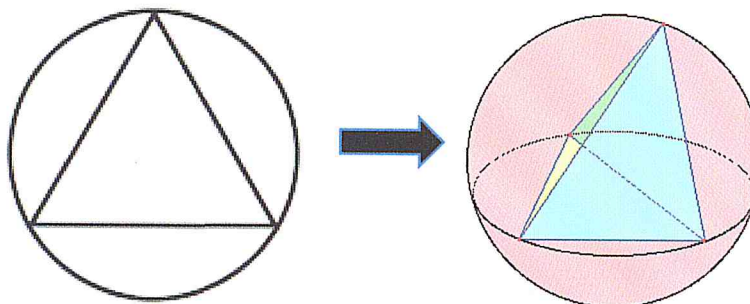


Figure 1.20 : Cercle circonscrit transformé en sphère circonscrite.

4.5.1. Principe de la triangulation en 3D :

La triangulation d'un ensemble donné S de n points dans l'espace est une décomposition d'une enveloppe convexe dans l'espace 3D. Cela signifie que l'enveloppe convexe S se décompose en tétraèdres tels que :

- Les sommets ou tétraèdres appartiennent à S .
- L'intersection de deux tétraèdres de $T(S)$ est soit vide, soit elle partage la même face ou un sommet.
- L'union des tétraèdres dans $T(S)$ remplit entièrement l'enveloppe convexe de S .

4.5.2. Algorithmes de Delaunay en 3D :

➤ Algorithme de Flip :

Cet algorithme a le même principe qu'en 2D et suit les étapes suivantes :

- Choix d'un tétraèdre de la liste des tétraèdres.
- Vérification du critère de Delaunay.
 - Si la sphère circonscrite ne contient aucun sommet appartenant aux tétraèdres adjacents, alors passer au tétraèdre suivant.
 - Si non, les tétraèdres doivent être modifiés par des opérations de flips.

❖ Flips non dégénérés :

L'algorithme commence avec un tétraèdre auxiliaire contenant tous les points de. Les points de S sont insérés dans la triangulation un par un. Dans chaque étape, un tétraèdre T contenant le point à insérer est identifié. Par la suite, le point est connecté dans T et une

Chapitre I

irrégularité de la triangulation résultante est réparée par une séquence de flips. En triangulation 3D, quatre types non-dégénérés des flips existent :

- Flip 1-4 - remplace un tétraèdre par 4 autres.
- Flip 4-1 - remplace 4 tétraèdres par un seul tétraèdre.

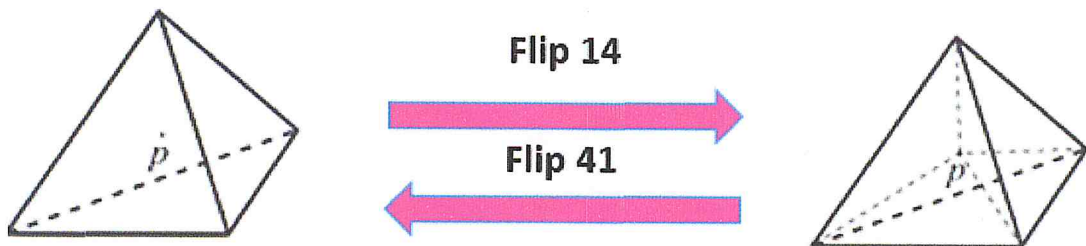


Figure.1.21 : Flip 1-4 et Flip 4-1.

- Flip 2-3 : remplace 2 tétraèdres par 3 autres.
- Flip 3-2 : remplace 3 tétraèdres par 2 autres.

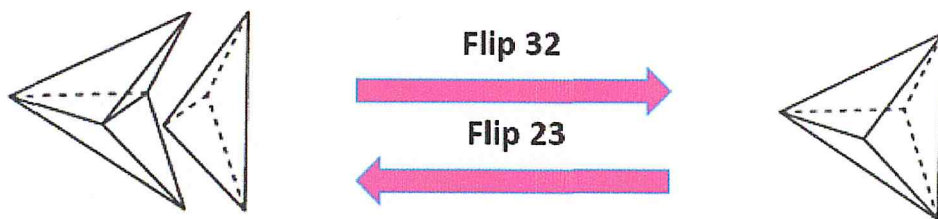


Figure 1.22 : Flip 3-2 et Flip 2-3.

❖ Flips dégénérés :

Il permet d'utiliser des flips dégénérés pour traiter des points en position non-générale. En 3D, trois des bascules dégénérées sont utilisées :

- Flip 26 est utilisé, si le point p à insérer se trouve dans une face f.

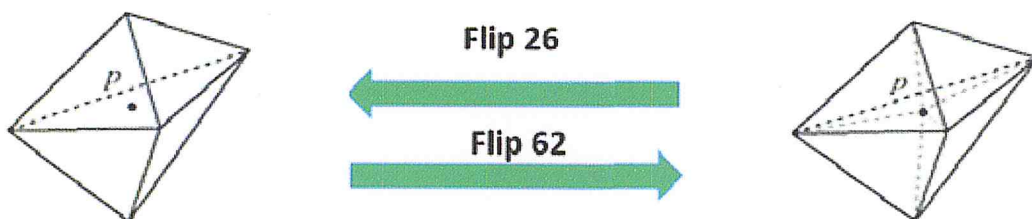


Figure.1.23 : Flip 2-6 et Flip 6-2

- Flip n-2n est utilisé, si le point p à insérer se trouve dans un bord e.

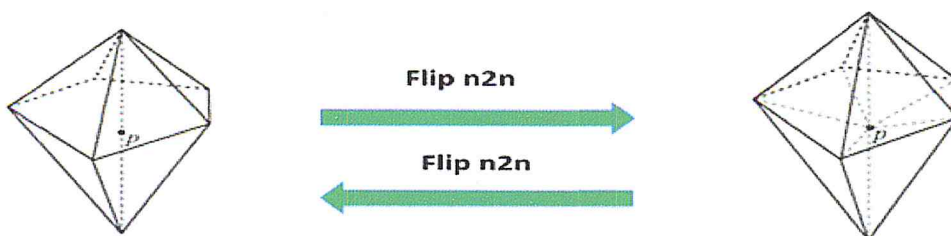


Figure 1.24 : Flip n-2n

- Flip 4-4 : Si deux tétraèdres $abcd$ et $abce$ ont une face partagée abc et si les sommets b, c, d, e sont coplanaires et s'il existe deux tétraèdres voisins bcd et $bcef$, alors la face abc peut être retournée par le flip 4-4. Cette bascule remplace les deux faces abc et bce par deux faces ade et def . [12]

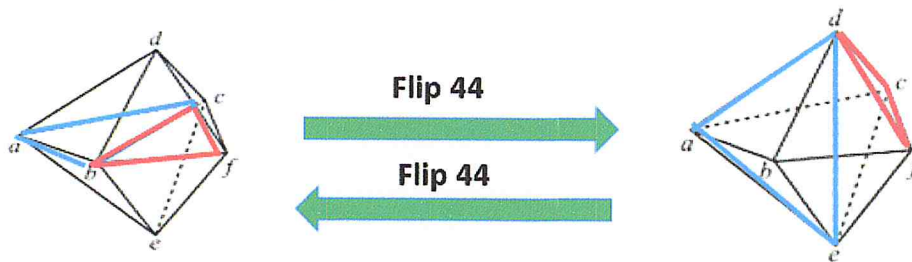


Figure.1.25 : Flip 4-4

• **Algorithme destruction/construction :**

Le principe de l'algorithme destruction/construction est le suivant. Tous les tétraèdres qui sont incidents à un point p à supprimer sont retirés. Ceci crée un "trou" dans la triangulation. L'objectif est de retriangler le trou de telle sorte que la triangulation résultante soit régulière.

Enveloppe convexe :

L'enveloppe convexe d'un ensemble de points du plan est le plus petit domaine convexe du plan qui contient tous les points de l'ensemble. Il s'agit en fait d'un polygone convexe dont les sommets sont des points de l'ensemble [24].

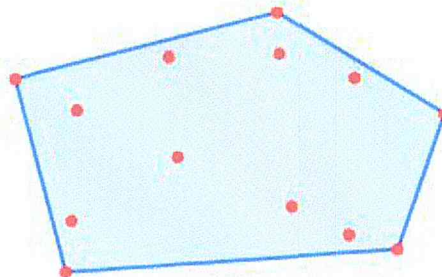


Figure.1.26 : Enveloppe convexe

5. Architectures parallèles :

Chapitre I

La notion de parallélisme, qui consiste à utiliser plusieurs processeurs ou opérateurs matériels pour exécuter un ou plusieurs programmes est ancienne. Les multiprocesseurs datent des années 1960. De cette période jusqu'à la fin des années 1990, des architectures parallèles ont été utilisées pour les applications nécessitant des besoins de calcul que les monoprocesseurs étaient incapables de fournir. Étaient concernés, les mainframes et serveurs d'une part et les machines vectorielles puis parallèles utilisées pour le calcul scientifique hautes performances d'autre part. L'utilisation massive du parallélisme se limitait aux très grandes applications de simulation numérique avec les architectures massivement parallèles. Le début des années 2000, avec les limitations des monoprocesseurs et le « mur de la chaleur », a complètement changé la situation. Les processeurs multicœurs sont présents en 2016 dans les architectures matérielles pour tous les types de composants : appareils mobiles (smartphones, tablettes), systèmes embarqués, télévisions, PC portables et PC de bureau, et jusqu'aux machines parallèles et super-ordinateurs pour la très haute performance [14].

5.1. Motivation pour le parallélisme :

L'exploitation du parallélisme dans l'architecture des ordinateurs est née de la conjonction de trois éléments : les besoins des applications, les limites des architectures séquentielles et l'existence dans les applications de la propriété de parallélisme [14]. Deux facteurs importants contribuent à l'amélioration des performances des processeurs modernes :

- Technologie rapide des circuits.
- Caractéristique architecturale :
 - Cache de plus en plus grand.
 - Bus multiple de plus en plus rapide.
 - Le pipe line.
 - Architecture super scalaire [15].

5.2. Fonctionnalité d'un système :

La fonctionnalité du système concerne les points suivants :

- Gestion de processus.
 - Déroulement de l'exécution.

- Gestion des évènements.
- Gestion des informations.
 - Gestion de la mémoire.
- Gestion des communications.
 - Gestion des entres/sorties
 - Gestion des réseaux.
- Sécurité et protection.

5.3. Différents types de système d'exploitation :

Ils existent cinq types de système d'exploitation :

- Systèmes multi-tâches :
 - Plusieurs programmes s'exécutent simultanément sur l'ordinateur.
 - Multiprogrammation.
- Systèmes multiprocesseurs :
 - Exploitation de plusieurs CPUs
 - Gestion des activités parallèles
- Systèmes multi-utilisateurs :
 - Réseau (serveur web...)
- Systèmes distribués :
 - Plusieurs ordinateurs autonomes interconnectés en réseau coopèrent pour réaliser des activités bien définies
- Systèmes Grille (G R D) :
 - Grand nombre de ressources interconnectées.

5.4. Etats d'un processus :

L'évolution des processus passe par différents états : de la naissance à la destruction au bout d'un temps fini (Figure 1.27). Les différents états sont les suivants :

- **Nouveau** : processus en cours de création,

Chapitre I

- **Prêt** : attend son affectation au processus,
- **Exécution** : processus en cour d'exécution sur le processeur,
- **Attente** : si le processus ne dispose pas du processeur et d'autres ressources ou de données nécessaires à son exécution.

La création et la destruction d'un processus est souvent faite par d'autres processus qui déclenchent des opérations en conséquence d'une commande directe de l'utilisateur [18].

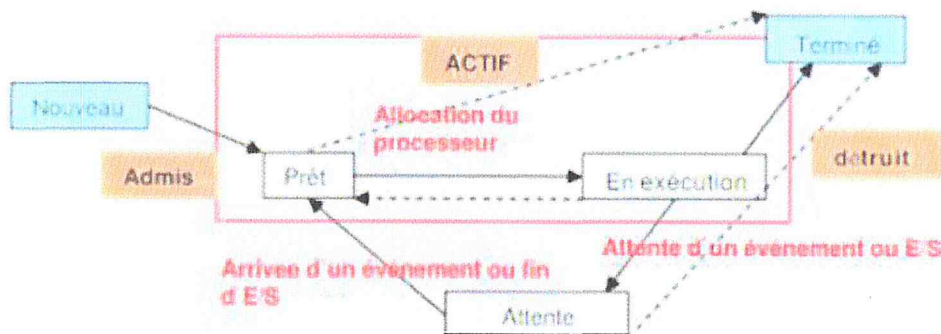


Figure .1.27 : Diagramme d'état d'un processus.

5.5. Sources de parallélisme :

Ils existent trois principales sources de parallélisme :

5.5.1. Parallélisme de contrôle :

Le principe consiste à exécuter plusieurs actions (tâches) en même temps. Chaque action est allouée à une ressource de calcul et les actions sont exécutées d'une façon plus ou moins indépendante, ce qui offre l'avantage d'un gain en temps linéaire. Toutefois, les dépendances qui existent entre les tâches ralentissent l'exécution parallèle (Figure 1.28).

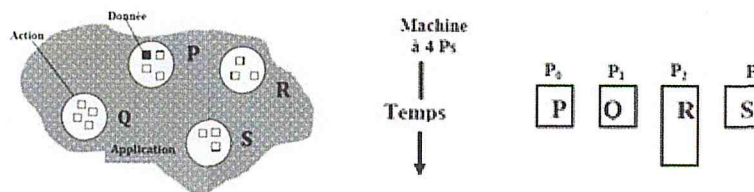


Figure.1.28 : Parallélisme de contrôle.

5.5.2. Parallélisme de données :

Chapitre I

Ce concept s'applique à des applications composées de données identiques sur lesquelles est effectuée une action répétée (tableaux de données). Les ressources de calcul sont associées aux données.

5.5.3. Parallélisme de flux

Ce principe s'applique sur des applications fonctionnant selon le mode de travail à la chaîne. On dispose d'un flux de données, généralement similaires, sur lesquelles est effectuée une suite d'opérations en cascade (Figure 1.29). Les ressources de calcul sont associées aux actions et chaînées tels que les résultats des actions exécutées au temps T sont passés au temps T+1 au processeur suivant (mode de fonctionnement Pipe-Line).[19]

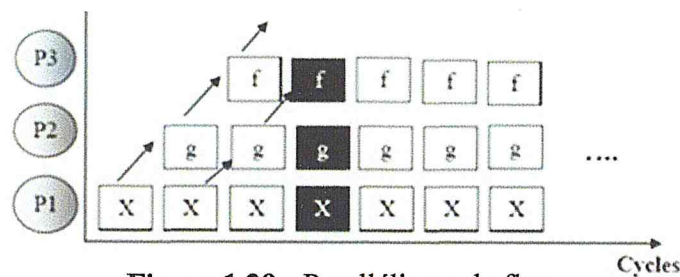
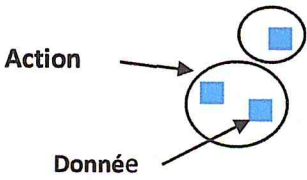
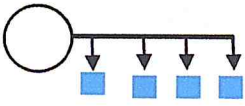
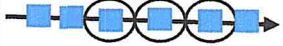


Figure.1.29 : Parallélisme de flux.

5.6. Mesure de performance :

Le but essentiel de la parallélisation est de gagner en performance de calcul. Des facteurs ont été mis aux points pour mesurer ces performances.[20]

Tableau 1.2 : Sources du parallélisme

	Parallélisme de Contrôle	Parallélisme de Données	Parallélisme de Flux
Devise	« Faire plusieurs chose au même temps »	« Repérer des actions sur des données similaires »	« Travailler à la chaîne »
Schéma de principe			
Domaine d'application	Toute application	Application de type "Élément finis"	Application de type "Contrôle_Commande"
Degré moyen de parallélisme	3	1000 000	30
Mode de pensée	Concurrent	Parallèle	Concurrent (parallèle)

Chapitre I

Extraction de parallélisme	Implicite Explicite	Explicite	Implicite Explicite
Données	Scalaires	Vectérielles	Scalaires Vectérielles
Modèles d'Architecture	Microprocesseurs Passage de Messages	Cellulaires	Vectérielles
Programme	Décentralisé	Centralisé	Décentralisé Centralisé
Mémoire	Décentralisé	Centralisé	Décentralisé Centralisé
Unité de contrôle	Asynchrone	Synchrone	Synchrone (Asynchrone)
Processeur associer aux	Actions	Données	Actions

Conclusion :

L'étude effectuée dans cette partie, portait dans un premier temps, sur la présentation du concept de la CFAO. Par la suite, une présentation des formats d'échange de données est faite et l'accent est mis sur le format STL. Dans un deuxième temps, les différentes étapes de reconstruction d'objets 3D à partir de nuage de points ont été présentées et la triangulation de Delaunay ; étape cruciale dans la reconstruction 3D; avec ces multiples algorithmes est introduite. Enfin, afin d'optimiser le processus de reconstruction d'objet 3D, les architectures parallèles ainsi que leurs caractéristiques et leurs applications ont été explicitées. Dans le prochain chapitre, la conception de l'application est abordée.

Chapitre II

Conception de l'application

win

Introduction :

La reconstruction 3D permet l'obtention d'une représentation numérique en trois dimensions d'un objet à partir d'un ou de plusieurs nuages de points. L'objectif visé est de produire un modèle d'objet approchant le mieux le modèle réel ;en général le modèle STL ; largement utilisé par les systèmes de FAO .L'approche préconisée consiste à subdiviser le nuage de points récupéré de la phase de digitalisation en plusieurs sous nuages. Puis, distribuer le traitement de la triangulation et de la fusion sur plusieurs processeurs afin de produire en sortie un modèle fiable et approchant le mieux le modèle réel. Pour cela, une approche combinant la triangulation de Delaunay et le concept des octrees est adopté.

1. Problématique :

La création des modèles 3D avec une bonne précision dans un temps optimal n'est pas une tâche facile. Plusieurs chercheurs se sont penchés sur le problème pour apporter des solutions. Il apparait clairement que la complexité de la reconstruction 3D dépend en grande partie de la taille du nuage de points qui sont souvent volumineux et dès lors, il devient très délicat de les traiter avec un seul processeur. Une des solutions apportées au problème consiste à paralléliser le traitement du nuage de points.

2. Architecture globale de l'application :

Le but global de l'application est de diminuer le temps de la triangulation de Delaunay d'un nuage dense de points puis minimiser le temps de reconstruction du modèle. Pour se faire, l'approche adoptée divise le problème en deux grandes parties. Chacune est spécialisée dans une phase de résolution :

- La première partie : concerne la subdivision du nuage de points
- La deuxième partie : concerne la fusion des sous-nuages après traitement.

La première étape prend en entrée un nuage de pointset génère en sortie des triangulations des sous-nuages après la création des Octrees.La deuxième étape prend en entrée les différentes triangulations générées lors de la première étape pour les fusionner selon un ordre bien défini en utilisant le concept de zone affectée afin de générer un modèle STL.

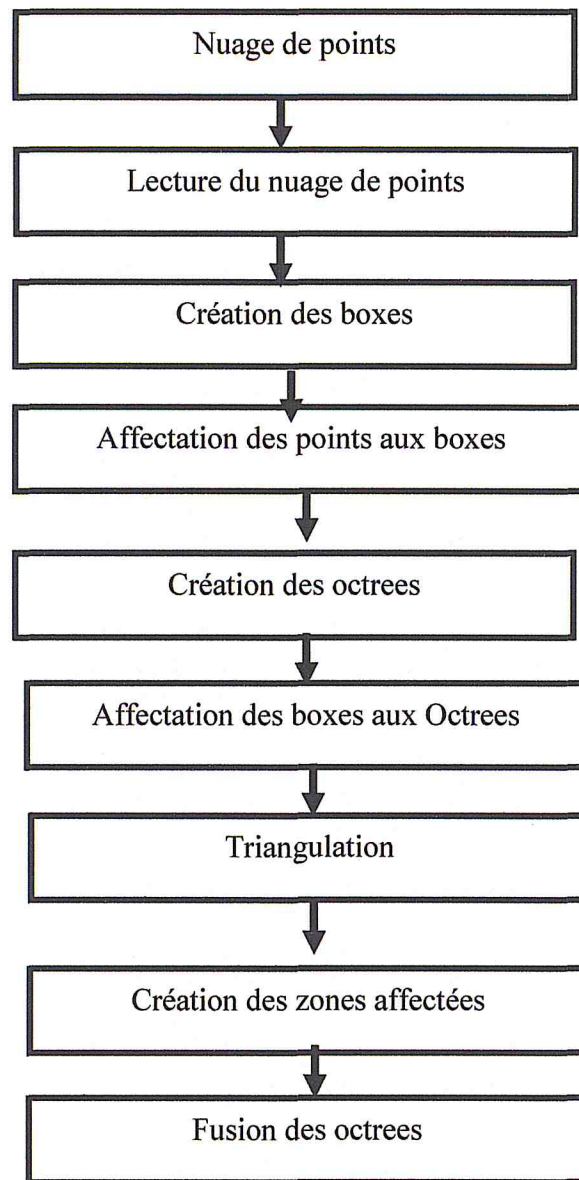


Figure.2.1 :Architecture globale de l'application.

2.1 Description de l'architecture globale de l'application :

Le processus général de génération du modèle STL est composé de plusieurs parties. Chacune est composée de plusieurs étapes.

2.1.1 Lecture du nuage de points :

La Lecture du nuage de points consiste à lire un fichier de points nettoyé où chaque point est représenté par ses coordonnées cartésiennes (x. y. z) dans le but de créer un brut englobant le nuage de points.

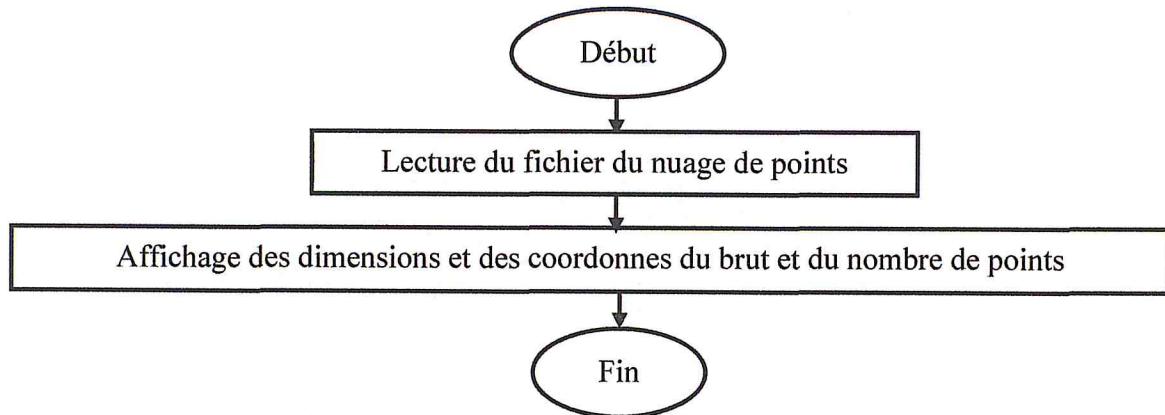


Figure.2.2: Organigramme de lecture du nuage de points.

2.1.2 Subdivision du nuage de points :

La lecture du fichier de points permet d'un côté, d'identifier les paramètres du brut englobant et de l'autre côté la création des octrees support de la subdivision du nuage de points.

- ✓ **Etape 1 : Création du brut** : la création du brut consiste à définir les paramètres d'un brut englobant : coordonnées des points limites du brut et ses dimensions (longueur, largeur et hauteur).

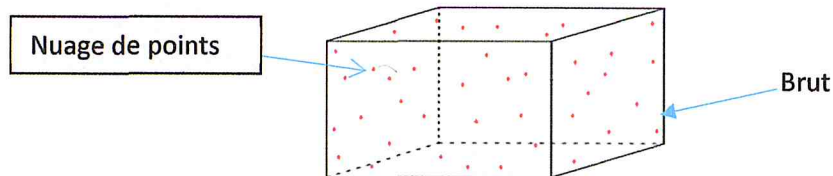


Figure 2.3 : Brut du nuage de points.

- ✓ **Etape 2 : Création des boxes** : pour faciliter la manipulation des points lors de l'étape de la triangulation, le brut est subdivisé en petites cellules de même taille appelées boxes. L'approche adoptée dans la phase de parallélisme, impose un nombre pair de boxes afin de pouvoir pratiquer d'autres subdivisions si nécessaire.
- calcul des tailleX, tailleY et tailleZ :

tailleX= 2 puissance nombre de divisions selon X

tailleY= 2 puissance nombre de divisions selon Y

tailleZ= 2 puissance nombre de divisions selon Z

- Calcul des dimensions des boxes :

$$pasX = \frac{X_{max} - X_{min}}{tailleX}$$

$$pasY = \frac{Y_{max} - Y_{min}}{tailleY}$$

$$pasZ = \frac{Z_{max} - Z_{min}}{tailleZ}$$

- Affectation des points aux boxes :

L'affectation des points aux boxes se fait par indexation des boxes suivant les trois directions. Ainsi, chaque point sera affecté à la zone appropriée en identifiant les indices du boxe cible. La Figure 2.4 illustre le processus appliqué à un nuage de points dans le plan.

$$I = \frac{x - X_{min}}{PasX} \quad J = \frac{y - Y_{min}}{PasY} \quad K = \frac{z - Z_{min}}{PasZ}$$

I, J, K : indice du boxe.

Xmin, Ymin, Zmin : limites minimales du boxe.

pasX, pasY, pasZ : dimensions du boxe.

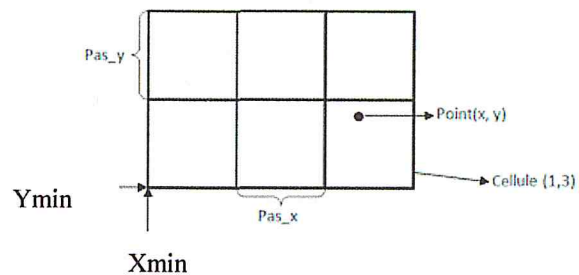
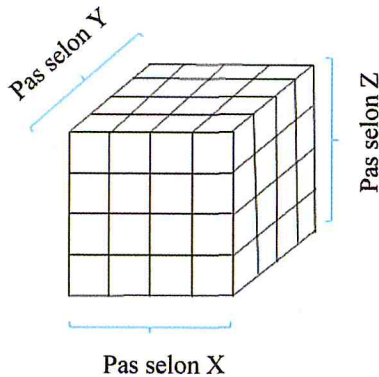


Figure.2.4 : Subdivision du brut en boxes.

Figure.2.5: Indexation d'un point dans le plan.

- ✓ **Etape 3 : Création des Octrees :** le but recherché est de trouver une méthode pour minimiser le temps de subdivision du nuage points tout en gardant une traçabilité de cette subdivision pour pouvoir les sous-nuages de points sans altérer la forme de l'objet. Pour atteindre cet objectif, la méthode des Octrees est utilisée.

La manipulation du concept de l'Octree passe par la maîtrise du Quadtree, qui n'est autre que l'octree appliqué dans le plan.

- **Principe du Quadtree :**

Le Quadtree «région» représente une partition de l'espace en deux dimensions en décomposant la région (le brut) en quatre quadrants égaux. Puis, chaque quadrant en quatre sous-quadrants et ainsi de suite avec chaque «nœud terminal» comprenant des données correspondant à une sous-région spécifique. Chaque nœud de l'arbre a exactement soit quatre enfants soit aucun (cas d'un nœud terminal). [23]

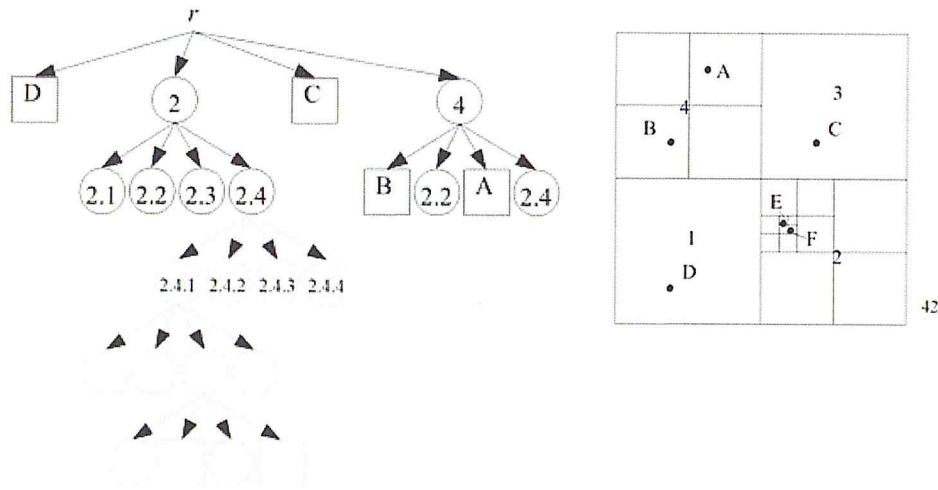
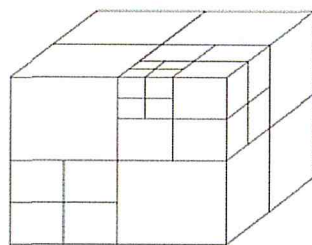


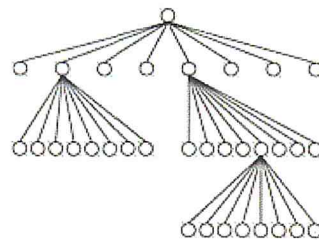
Figure.2.6: Arbre et subdivision du brut en Quadtree.

- **Principe de l'Octree :**

Un octree est un partitionnement binaire de l'espace sous forme d'arbre au sein duquel chaque nœud interne possède huit fils. Cette structure permet de retrouver rapidement les relations de voisinage entre ces nœuds [22]. L'Octree est récursivement subdivisé en fonction de la planéité des points contenus afin d'obtenir une structure anisotrope. Par conséquent, l'octree sera récursivement subdivisée jusqu'à ce que toutes ses feuilles décrivent une surface localement plane dont la normale est déduite. Puis, dans la phase de fusion, la même structure sera utilisée afin de retrouver le voisinage de chacune des feuilles et de fusionner au sein d'un même segment les feuilles voisines et de normales proches.



a) Partitionnement d'un espace par Octree.



b) Structure de donnée associée.

Figure.2.7 : Principe de l'Octree.

Chapitre II

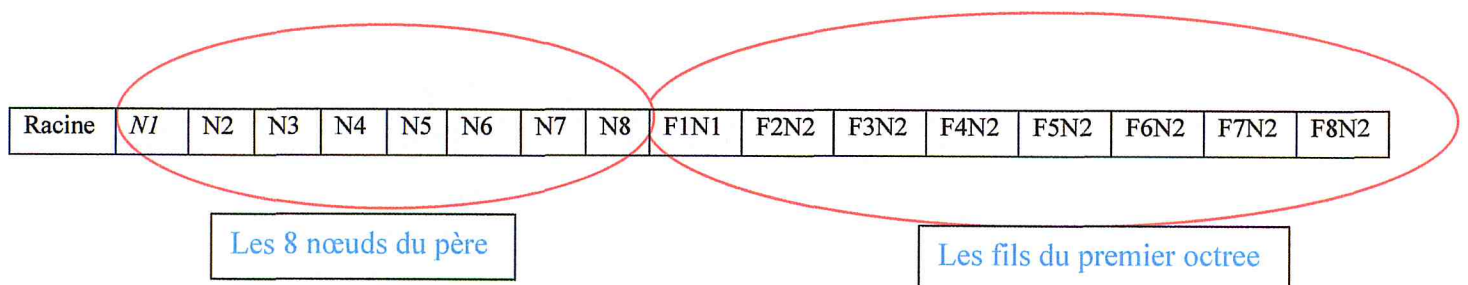
- **Intégration de l'octree dans notre architecture :**

Dans cette méthode ,les mêmes notions de base de la méthode du Quadtree sont appliquées sauf que la subdivision se fait en 3D. Ainsi, l'enveloppe carrée devient un cube et la subdivision en 4 devient une subdivision en 8. La création de l'arbre reste inchangée avec l'augmentation du nombre de nœuds de 4 à 8 nœuds. Ceci permet d'éviter le calcul des propriétés géométriques en chaque point et l'accès rapide au voisinage des points. La subdivision de l'Octree est faite comme suit :

- Fixer un nombre N qui représente le nombre maximum de points dans chaque Octree.
- Tester si le nombre de points dans chacun des 8 octrees est inférieur ou égale à N,
 - ✓ Si oui, pas de subdivision
 - ✓ Si non, division de l'Octree est appliquée.
- Créer un tableau à partir de l'arbre obtenu de l'Octree.

Ce tableau comporte tous les nœuds de l'arbre. Au début, seuls les huit premiers nœuds sont insérés et le tableau est parcouru :

- Si un nœud respecte la contrainte, alors passer au prochain nœud .Sinon, ses fils seront insérés au niveau de la queue.



Chaque octree est défini par :

- Ses fils
- Son père
- Ses 3 voisins
- Sa position
- Le nombre de ses points.
- Limites de l'octree
- Indice maximum et minimum du premier et du dernierboxe.

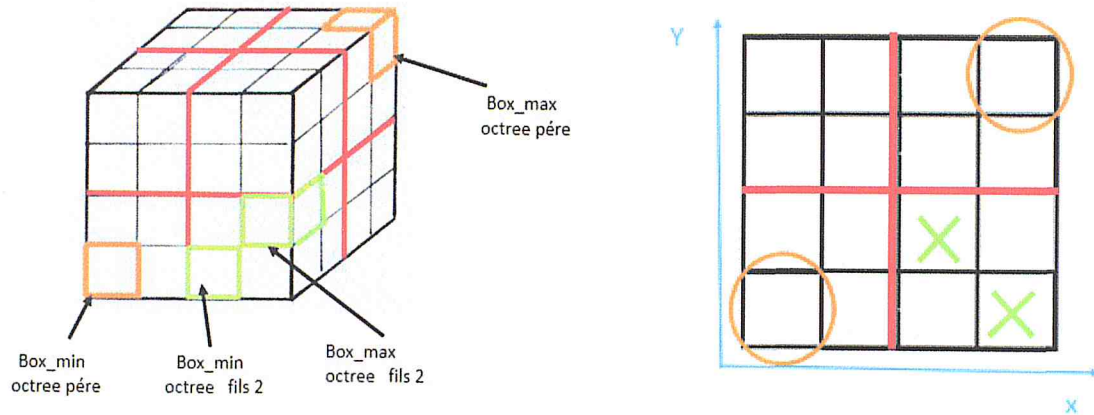


Figure.2.8 : Méthode de calcul des limites de l'octree en 2D et en 3D.

A partir de la position de l'octree et les limites de l'octree père, on peut calculer à chaque fois, les l'indices du box_min (**Imin,Jmin,Kmin**) et du box_max (**Imax,Jmax,Kmax**).

Pour l'octree 2, l'indce box_min_père et box_max_père sont donnés par :

Pour le box_min :

$$\mathbf{Imin} = \frac{\text{Imin de père} - \text{Imax de père}}{2} + 1$$

$$\mathbf{Jmin} = \text{Jmin de père}$$

$$\mathbf{Kmin} = \text{Kmin de père}$$

Pour le box_max :

$$\mathbf{Imax} = \text{Imax de père}$$

$$\mathbf{Jmin} = \frac{\text{Jmax de père} - \text{Jmax de père}}{2} + 1$$

$$\mathbf{Kmin} = \frac{\text{Kmin de père} - \text{Kmax de père}}{2} + 1$$

2.1.3 Triangulation :

Ils existent plusieurs algorithmes qui génèrent la triangulation de Delaunay. La triangulation proposée dans ce travail vise à optimiser le temps de génération des triangles par la manipulation des octrees.

- **Stratégie de triangulation :**

Dans le processus de génération du modèle STL, l'étape de la triangulation est la plus consommatrice en termes de temps de traitement. A cet effet, cette fonction est la première à

être paralléliser. La stratégie adoptée consiste à traiter uniquement les octrees qui ne possèdent pas de fils.

2.1.4 Fusion :

La fusion constitue l'étape clé du processus de reconstruction du nuage de points subdivisé. La stratégie de fusion des Octrees adoptée dépend de la forme de l'objet. La stratégie de concaténation adoptée dans ce travail est décrite ci-dessous :

- Sélectionner les nœuds concernés par la concaténation : seul quatre nœuds ou leurs nœuds fils sont traités à la fois.
- La concaténation se fait deux à deux dans une direction bien définie.
- La concaténation commence par le niveau le plus bas. Le résultat de la concaténation est placé dans le nœud père et on remonte au niveau supérieur jusqu'à arriver aux nœuds les plus hauts.
- La fusion débute par la création des zones affectées. Ces zones sont utilisées pour la création des tétraèdres supports de fusion des octrees :

La zone affectée est la région de la triangulation de chaque Octree particulièrement sensible à des modifications lors de la fusion des triangulations. Ces régions représentent les portions de zones voisines entre les différents Octrees dans les trois directions X, Y et Z.

La Figure 2.9 illustre le processus de sélection des nœuds candidats à la concaténation. Les nœuds 9,8,7,6,5,3,2, sont candidats pour le premier niveau de concaténation. Le résultat est stocké dans le nœud 4.

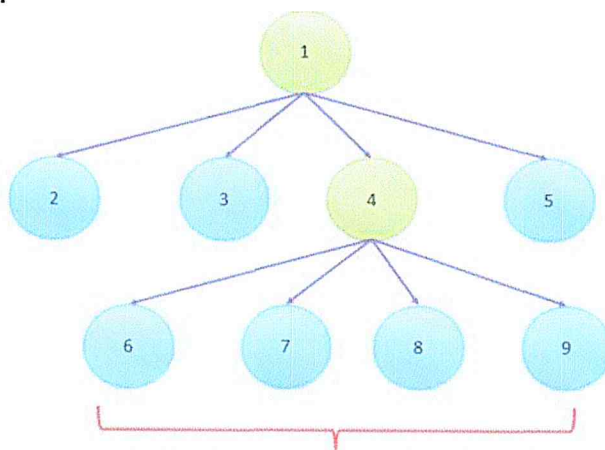


Figure.2.9 : Arbre de Quadtree (stratégie de triangulation et fusion (2D)).

➤ Création des zones affectées :

La zone affectée est la région de la triangulation de chaque sous-nuage de points qui peut être modifiée lors de la fusion. Ces régions représentent les portions de zones mitoyennes entre les différents sous-nuages dans les trois directions. Six zones peuvent être créées, deux (02) dans chaque direction.

Zones affectées gauches et droites :

La génération de la zone affectée droite (gauche) suit le processus suivant

- Vérification de l'intersection entre la sphère circonscrite et le plan X_{max} (X_{min}).
- Si oui, alors calcul du rayon du cercle d'intersection entre la sphère et le plan.
- Vérification du rayon s'il est compris entre Z_{min} et Z_{max} .
- Vérification du rayon s'il est compris entre Y_{min} et Y_{max} .

Zone affectée avant et arrière :

La génération de la zone affectée arrière (avant) suit le processus suivant :

- Vérification de l'intersection entre la sphère circonscrite et le plan Y_{max} (Y_{min}).
- Si oui, alors calculer le rayon du cercle d'intersection entre la sphère et plan.
- Vérification du rayon s'il est compris entre X_{min} et X_{max} .
- Vérification du rayon s'il est compris entre Z_{min} et Z_{max} .

Zone affectée haut et bas :

La génération de la zone affectée haut (bas) suit le processus suivant :

- Vérifier s'il y a intersection entre la sphère circonscrite et le plan Z_{max} (Z_{min}).
- Si oui, alors calculer le rayon du cercle d'intersection entre la sphère et plan.
- Vérifier si le rayon est compris entre Y_{min} et Y_{max} .
- Vérifier si le rayon est compris entre X_{min} et X_{max} .

Chaque octree contient trois zones affectées selon sa position au niveau de l'octree père. Afin d'optimiser le processus de fusion et éviter le stockage inutile des trois zones pour chaque Octree, une méthode en trois étapes a été adoptée. Cette façon de faire, permet de stocker seulement les zones utilisées dans l'étape de la stratégie de fusion. Les trois étapes sont décrites ci-dessous :

➤ **La première étape** : consiste à affecter une zone droite de chaque octree impair et gauche pour chaque octree pair de la manière suivante :

- Affecter la zone droite pour le premier Octree et gauches pour le deuxième.
- Affecter la zone droite pour le troisième Octree et gauches pour le quatrième.
- Affecter la zone droite pour le cinquième Octree et gauches pour le sixième.
- Affecter la zone droite pour le septième Octree et gauches pour le huitième.

➤ **La deuxième étape** : cette étape consiste à affecter les zones avant et arrière aux Octrees impair de la manière suivante :

- Affecter la zone arrière pour le premier octree et avant pour le troisième.
- Affecter la zone arrière pour le cinquième octree et avant pour le septième.

➤ **La troisième étape** : cette étape consiste à affecter les zones hautes et bas aux Octrees impair de la manière suivante :

- Affecter la zone haute pour le premier octree et bas pour le cinquième.

La fusion consiste à concaténer les Octrees subdivisés et triangulés dans le but de la reconstruction des objets en 3D selon une approche prédéfinie. Le principe de la fusion est comme suit :

Dans une première étape, les octrees triangulés deux à deux, pair et impair, sont fusionnés de telle sorte à utiliser la zone affectée droite des octrees impaires avec la zone affectée gauches des octrees paire set le résultat est stocké au niveau de l'Octree impair. La chronologie de fusion est la suivante :

- Fusionner le premier et le deuxième Octree et stoker les résultats au niveau du premier Octree.
- Fusionner le troisième et le quatrième Octree et stoker les résultats au niveau du troisième Octree.
- Fusionner le cinquième et le sixième octree et stoker les résultats au niveau du cinquième Octree.
- Fusionner le septième et le huitième Octree et stoker les résultats au niveau du septième Octree.

Par la suite, les octrees sont fusionnés par l'utilisation des zones affectées avant et arrière, ce qui donne une fusion octree impaires – impaire selon l'ordre suivant :

- Fusionner le premier et le troisième octree et stoker les résultats au niveau du premier octree.

- Fusionner le cinquième et le septième octree et stoker les résultats au niveau du cinquième octree.

La dernière étape consiste à fusionner les Octrees en utilisant les zones affectées haut et bas. La fusion sera aussi impair – impair.

- Fusionner le premier et le cinquième octree et stoker les résultats au niveau du premier octree.

2.1.5 Premier tétraèdre:

La fusion des Octrees consiste à générer un premier tétraèdre qui servira de support pour la génération des autres tétraèdres. Le processus de la création du premier tétraèdre se fait par la méthode suivante :

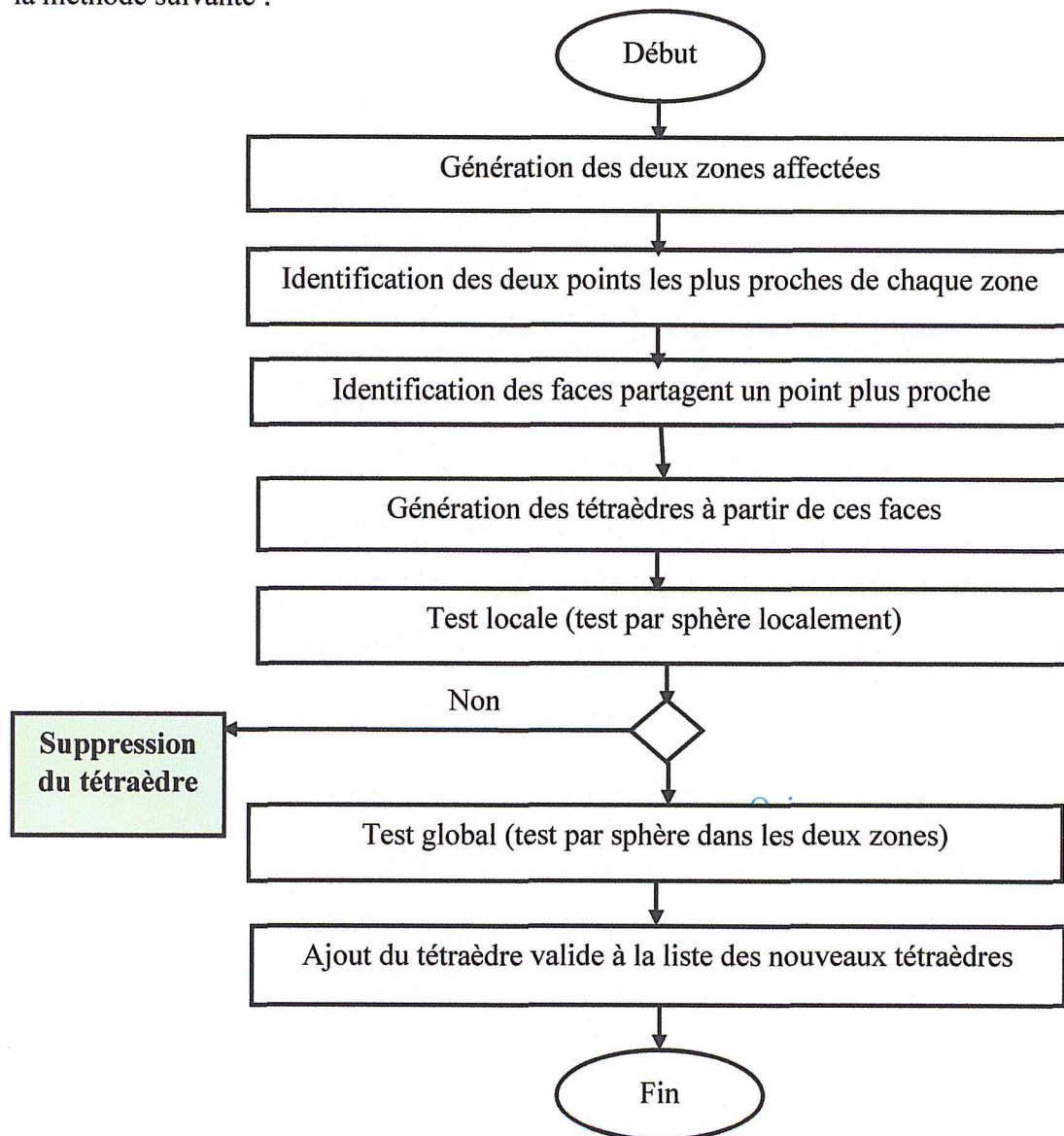


Figure.2.10 : Organigramme de création du premier tétraèdre.

La création du premier tétraèdre commence par la génération des zones affectées. Le test local de la sphère consiste à tester la sphère avec les points de la zone affectée gauche si la face se trouve dans la zone affectée droite et vice versa. Si la sphère n'est pas vide, on supprime le tétraèdre. Sinon, on passe au test global qui consiste à tester l'ensemble des points des deux zones avec les tétraèdres affectés à fusionner.

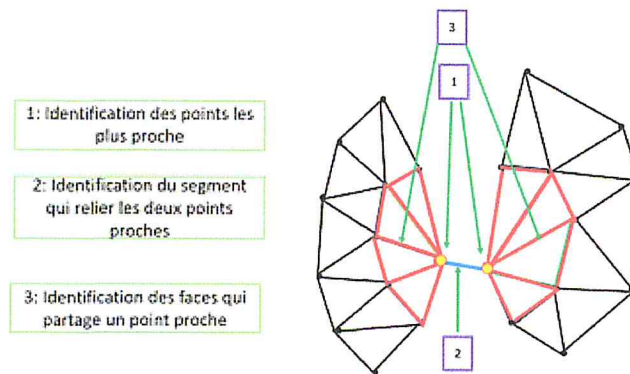


Figure.2.11: Etapes de création de 1^{er} tétraèdre.

2.1.6 Tétraèdre générique :

La génération du tétraèdre générique obéit au test de la sphère circonscrite. Deux tests sont pratiqués :

- Un test local de la sphère : consiste à tester la sphère avec les points de la zone affectée gauche si la face se trouve dans la zone affectée droite et vice versa.
- Un test global de la sphère : consiste à tester l'ensemble des points des deux zones affectées à fusionner.

Le processus de création du second tétraèdre est illustré dans l'organigramme suivant.

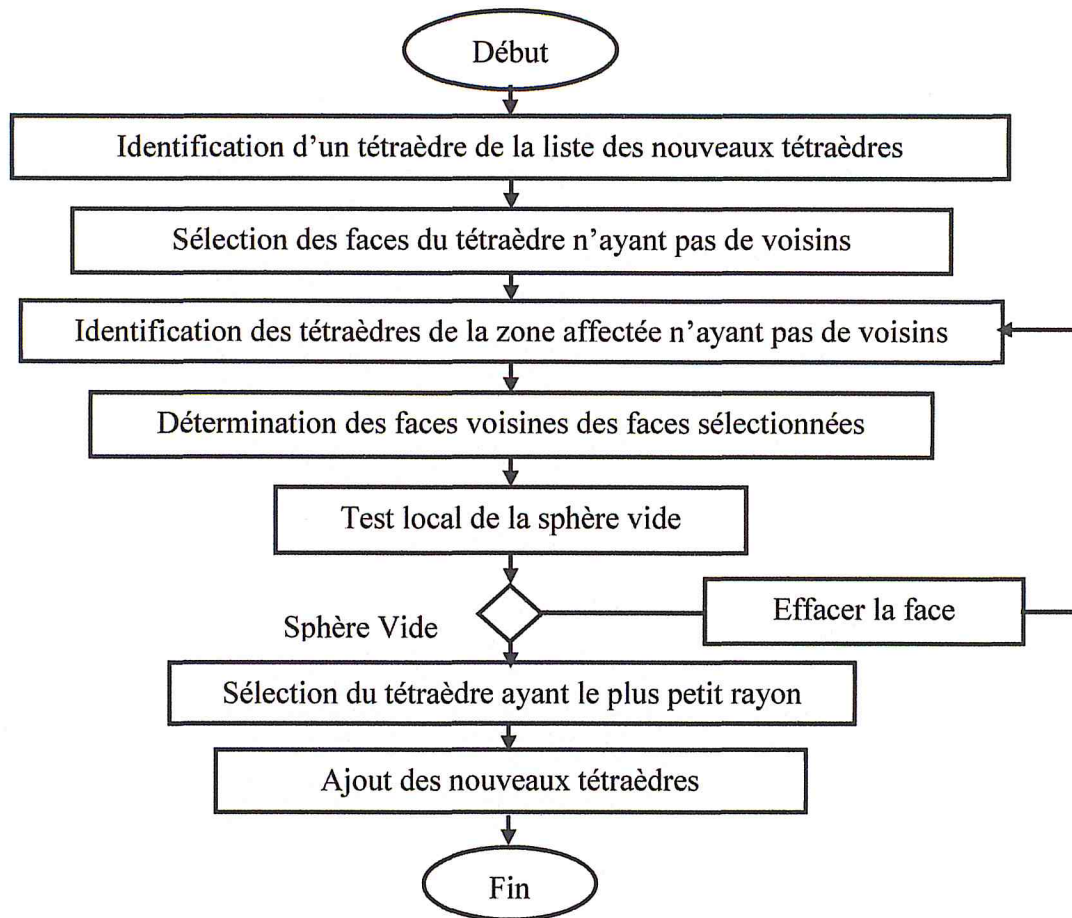


Figure.2.12: Organigramme de création du tétraèdre générique.

3. Parallélisme :

Le parallélisme consiste à traiter des informations de manière simultanée. Ces techniques ont pour but d'optimiser les tâches consommatrices de temps et de mémoire. Dans notre travail, les tapes de la subdivision, de la triangulation de Delaunay et de la fusion sont parallélisées. Ils existent plusieurs prototypes de parallélismes, mais celui de l'EMBERCADERO est le mieux adapté à notre application du fait que ce type de parallélisme vise les boucles. Donc, la fonction du traitement contient toutes les instructions sans boucle et la fonction d'appel contient l'appel selon le prototype.

suivant : TParallel::For(NULL,min,Max,MyIteratorEvent)

La mise en œuvre du parallélisme exploite les trois bibliothèques offertes par EMBERCADERO.

```
#include <System.Threading.hpp>
```

```
#include <System.Diagnostics.hpp>
```

```
#include<System.SyncObjs.hpp>
```

Ils existent deux types de parallélismes sur Embercadero.

- **TTask** :la bibliothèque de programmation parallèle fournit une classe TTask pour exécuter une ou plusieurs tâches en parallèle. Une tâche est une unité de travail à effectuer. La PPL fait l'association entre la tâche et le thread qui effectue la tâche de sorte que plusieurs tâches peuvent être exécutées en parallèle sans avoir à créer de propres threads personnalisés ni à les gérer.[31]
- **TParallel.For** :la PPL inclut une fonction boucle TParallel.For qui permet d'exécuter des boucles en parallèle. Dans les boucles parallèles, les opérations sont transformées en opérations thread-safe.[31]

4. Modélisation de l'application avec UML :

4.1 Méthode de modélisation :

La modélisation en informatique est la conception d'un modèle selon son objectif et les moyens utilisés. C'est une étape très importante pour la réalisation du système. L'objectif de la modélisation d'un système informatique est la maîtrise de la complexité, décrire son déroulement, son fonctionnement et ses résultats. UML « Unified Modeling Language » est une méthode de modélisation orientée objet. Elle est développée en réponse à l'appel à propositions lancé par l'OMG (Object Management Group) dans le but de définir la notation standard pour la modélisation des applications construites à l'aide d'objets. Elle est héritée de plusieurs autres méthodes telles que OMT (Object Modeling Technique) et OOSE (Object Oriented Software Engineering) et Booch. Dans notre projet, les diagrammes suivants ont été utilisés:

- ❖ **L'expression des besoins** : pour l'expression des besoins, le diagramme de cas d'utilisation a été utilisé.
- ❖ **Conception** :pour la conception de l'application, le diagramme de classe a été utilisé.
- **Le diagramme des cas d'utilisation** : (Use Case Diagram) constitue la première étape de l'analyse UML en :
 - Modélisant les besoins des utilisateurs.
 - Identifiant les grandes fonctionnalités et les limites du système.

- Représentant les interactions entre le système et ses utilisateurs.[26]

Les diagrammes de cas d'utilisation sont des diagrammes UML utilisés pour donner une vision globale du comportement fonctionnel d'un système logiciel. UML définit une notation graphique pour représenter les cas d'utilisation. Cette notation est appelée diagramme de cas d'utilisation. Dans ce qui suit, les différentes fonctionnalités du système développé ainsi que les offres proposées aux utilisateurs avec un diagramme de cas d'utilisation sont présentées.[25] La vue globale du système développé est présentée dans la figure suivante :

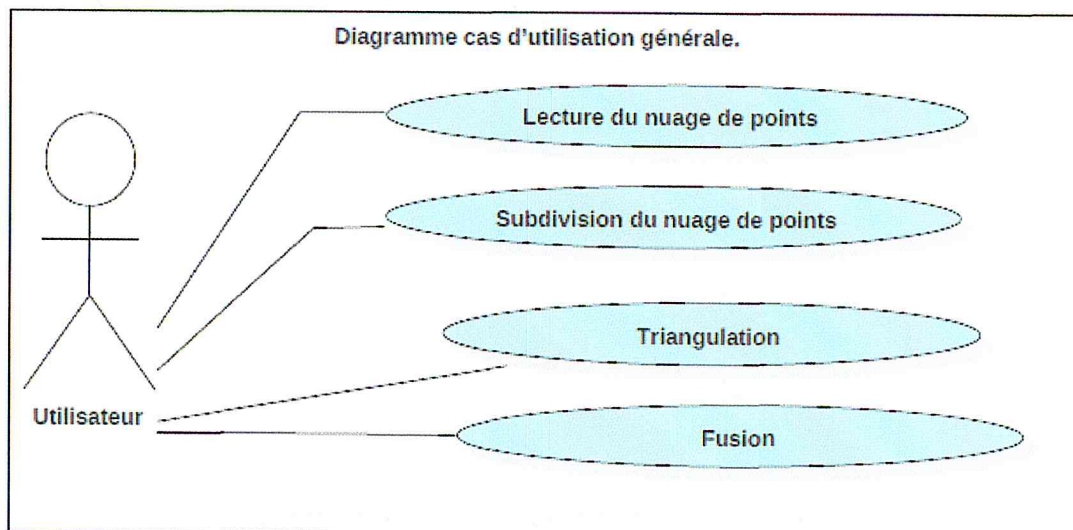


Figure.2.13: Diagramme cas d'utilisation générale

La première étape du cas d'utilisation consiste à lire le fichier du nuage de points. Cette tâche est décrite par le diagramme suivant :

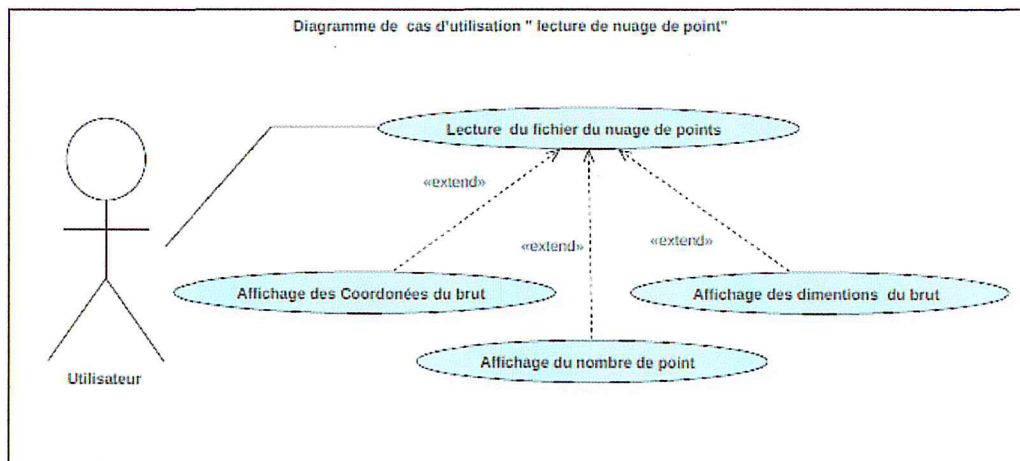


Figure.2.14: Diagramme cas d'utilisation lecture du nuage de points.

La lecture du nuage de points est suivie par la subdivision du nuage de points en boxes puis en octrees. Le diagramme ci-dessous illustre cette procédure.

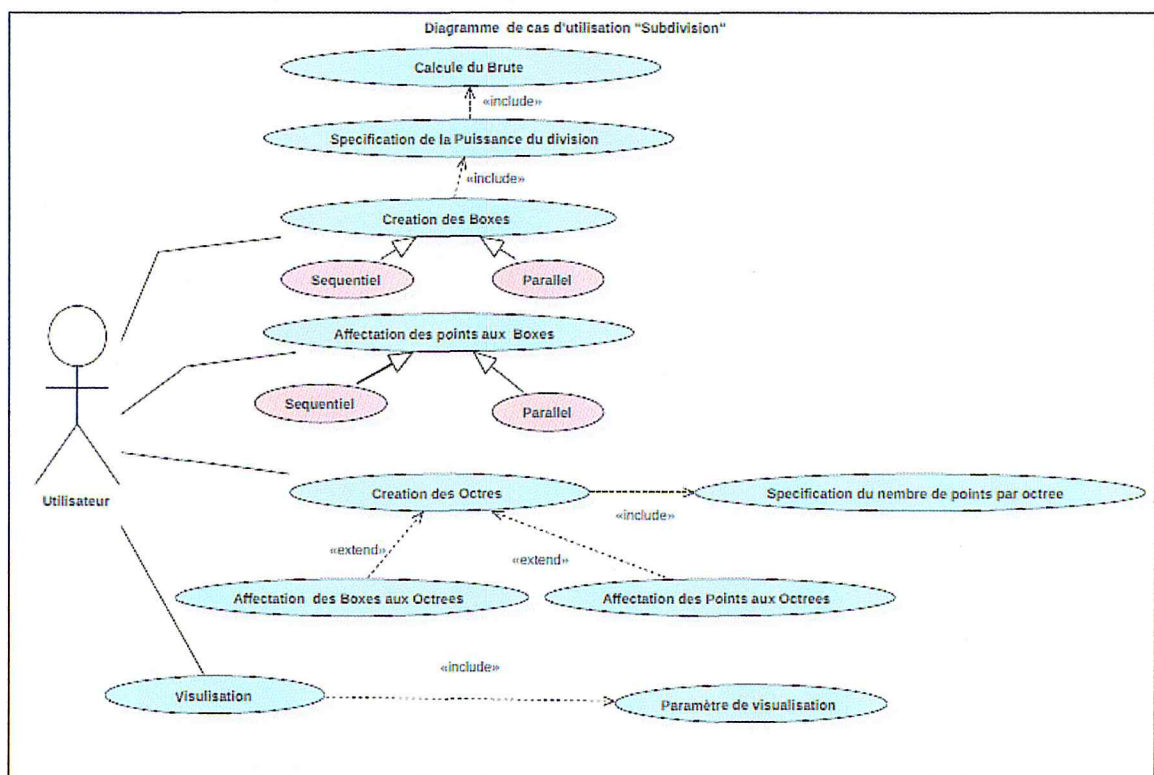


Figure.2.15:Diagramme cas d'utilisation de subdivision

Chapitre II

Le résultat de la subdivision du nuage de points est un ensemble d'octrees contenant chacun un nombre spécifique de points. Chaque octree subit une triangulation de Delaunay.

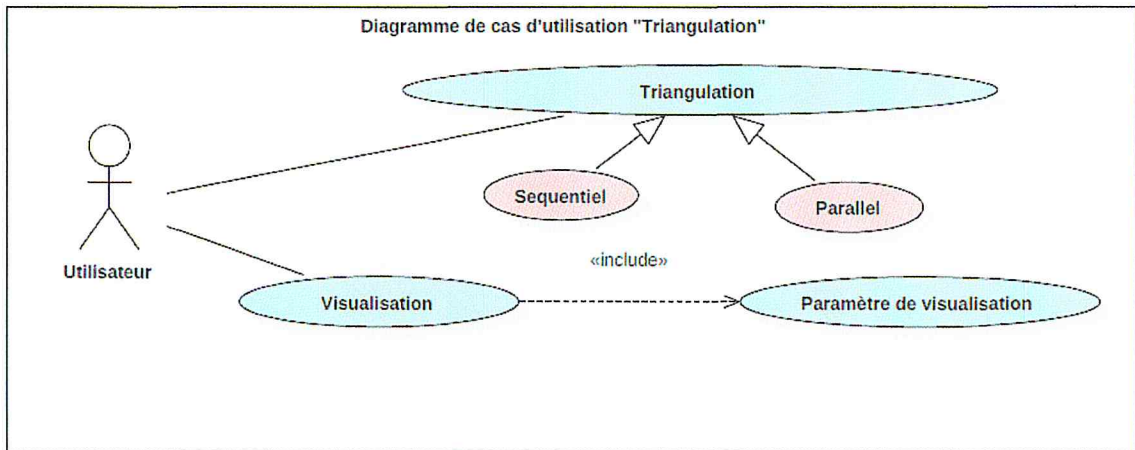


Figure.2.16: Diagramme cas d'utilisation de triangulation.

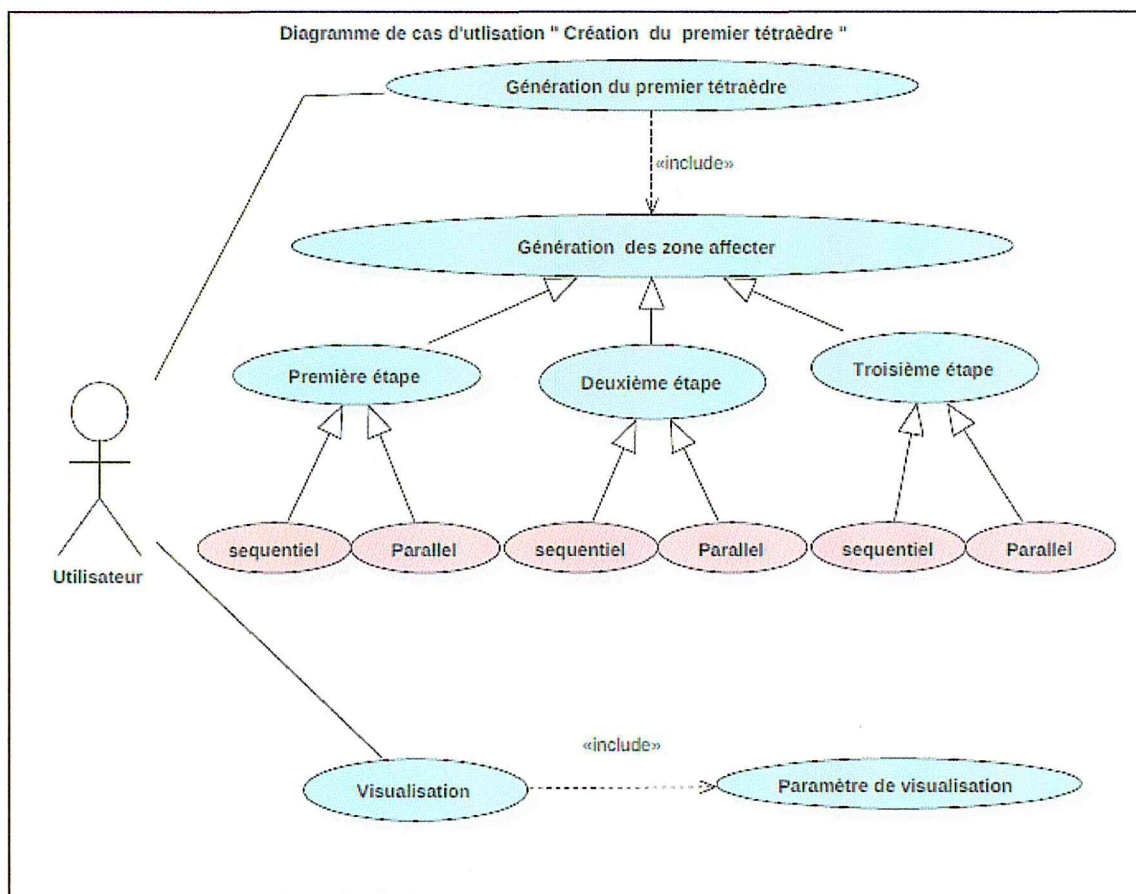


Figure.2.17: Création du premier tétraèdre.

➤ **Diagramme de classes** : le diagramme de classes fournit une vue d'ensemble du système à modéliser en décrivant les objets et les classes à l'intérieur du système et les relations entre eux. Il offre une grande variété d'usages à partir de la modélisation de la structure de données spécifique au domaine de la conception détaillée du système. Avec les installations du modèle de l'action, le modèle de classe peut être réutilisé dans les diagrammes d'interaction pour la conception détaillée du comportement dynamique. Le diagramme de classe représente une vue statique du système [27]. Les classes composant le système sont :

- Classe couleur_multi
- Classe point_multi
- Classe Brute
- Classe nuage_point_multi
- Classe box_multi
- Classe octree
- Classe Face
- Classe Tetraedre_k

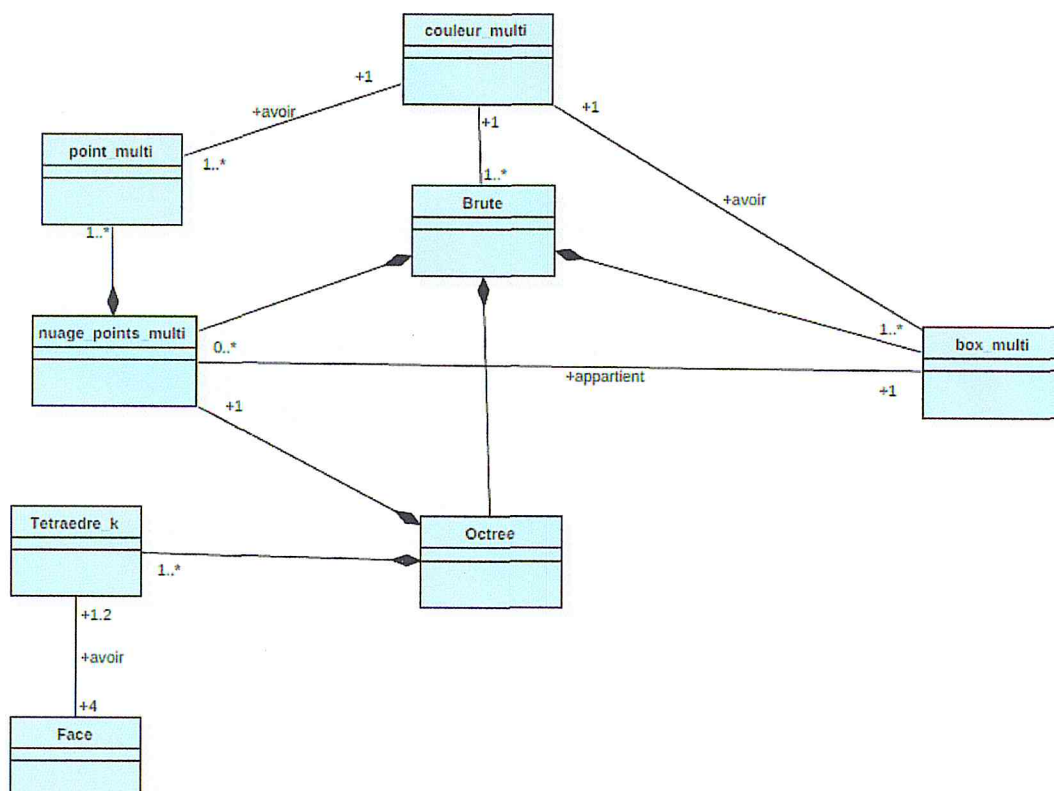


Figure.2.18: Diagramme de classe générale

Chapitre II

La représentation détaillée (attributs et fonctions) de chaque classe Implémentée est décrite dans la suite.

- ❖ **Classe point multi:** c'est la classe qui regroupe les coordonnées X, Y, Z des points après l'opération de l'acquisition ainsi que les indices des boxes pour pouvoir positionner chaque point dans son boxe.

Point_multi
-x: double -y: double -z: double -ii: int -jj: int -kk: int
+dessiner_point_multi(): void

Figure.2.19: Classe Point_multi

- ❖ **Classe limite_brute_multi :** regroupe les limites du nuage de points. Pour une visualisation claire du brut, on utilise des fonctions de dessin du brut.

Limite_brut_multi
-xmax: double -ymax: double -zmax: double -xmin: double -ymin: double -zmin: double
+dessiner_brut(): void +dessiner_brut_rendu(): void

Figure.2.20: Classe Limite_brute_multi

- ❖ **Classe box_multi :** c'est une classe de la première subdivision du nuage de points. Elle contient un tableau d'indice des points, d'où chaque boxe est combiné à un ensemble de points

Box_multi
-indice_point: vector<int>
+ajouter_element_indice_point(): void

Figure.2.21: Classe box_multi

❖ **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 boxe. Parmi les fonctions de la classe **Octree**, nous avons :

- `calcule_nb_point()` : retourner le nombre de points de chaque Octree,
- `calcule_voisin(int i, int j)` : retourne les voisin 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 affectée.

octree
-pere: int -voisin: int[3] -fils: int[8] -nbre_points_octree: int -position: int -indice box_min: indice +indice box_max: indice -tab_tetraedre: vector<Tetraedre_k> -tab_id_tetraedre_affecte: vector<int> -Point_Nuage point_proche -face_point: vector<Face> -tetra_resultant: vector<Tetraedre_k> -tetra_resultant2: vector<Tetraedre_k> -tetraedre_candidat: Tetraedre_k -point_proche_octree_voisin: Point_Nuage
+points_proches(): void +calcule_nb_point(): int +calcule_voisin(int i, int j): void +triangulation_noeud_sans_fils(): void +zone_affectee_droite(): void +zone_affectee_haut(): void +zone_affectee_bas(): void +zone_affectee_avant(): void +zone_affectee_arriere(): void +choix_zone_affectees_etape_1(): void +choix_zone_affectees_etape_2(): void +choix_zone_affectees_etape_3(): void +test_locale(Point_Nuage *point): void +test_global(int indice_octree): void

Figure.2.22: Classe Octree.

❖ **Classe nuage_points_multi** : elle contient trois tableaux ; un tableau des points, un tableau des octrees et un troisième tableau pour les boxes. Ainsi, cette classe regroupe tous les composantes du nuage du point. Parmi les fonctions de cette classe :

- `cree_box()` : retourne des boxes
- `cree_octree()` : retourne des Octrees
- `affecter_points()` : affecte chaque point à son boxe approprié.

nuage_points_multi
-tab_points: vector<point_multi> -tab_box_multi: ***box_multi -tab_octree: vector<octree> -nbre_max_points_octree: int -taillex: unsigned int -tailley: unsigned int -taillez: unsigned int -pasx: double -pasy: double -pasz: double
+calculer_limites(): void +cree_box(): void -affecter_points(): void +affecter_points_parallel(): void +cree_octree(): void +affecter_box_octree(): void +affecter_points_octree(): void +cree_box_parallel(): void +My_creation_boxk2(TObject* Sender, int s): void __fastcall +affecter_points_indice(): void +affecter_points_indice_parallel(): void +My_affecter_points_indice_parallel(TObject* Sender, int b): void __fastcall +triangulation_parallel_noeud_sansfils(): void +MyIterator_noeud_sans_fils(TObject* Sender, int l): void __fastcall +determiner_zone_affecter(): void +determiner_zone_affecter2(): void +determiner_zone_affecter3(): void

Figure.2.23: Classe nuage_point_multi.

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

Tetraedre_k
-point1: Point_Nuage* -point2: Point_Nuage* -point3: Point_Nuage* -centre_face1: Point_2018 -centre_face2: Point_2018 -centre_face3: Point_2018 -centre_face4: Point_2018 -v1: int -v2: int -v3: int -v4: int -centre_tetraedre: Point_Nuage
+creer_tetraedre(Point_Nuage *p1, Point_Nuage *p2, Point_Nuage *p3, Point_Nuage *p4): void +getcentre_sphere(): Point_Nuage +getcentre_tetraedre(): Point_Nuage +calculer_centre_rayon(): void

Figure.2.24: Classe Tetraedre_k

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 détaillées 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 :

test et validation

Chapitre III

Introduction :

Nous consacrons ce dernier chapitre à la présentation de divers résultats obtenus grâce à notre méthode adaptative de la reconstruction des objets 3D. Nous cherchons tout d'abord à démontrer les possibilités offertes par notre implémentation, en présentant les résultats obtenus via notre application graphique, afin d'illustrer les différentes fonctionnalités disponibles. Nous mettons, ensuite en évidence le bon comportement du parallélisme de la reconstruction d'objets 3D en termes de fiabilité, temps de traitement et de performances.

Nous présentons l'architecture logicielle que nous avons conçue pour implémenter la reconstruction des objets 3D à partir d'un nuage de points.

Enfin, nous nous proposons de comparer le comportement de notre méthode en parallélisme avec les résultats obtenus en mode séquentiel.

Il est à noter, que notre application logicielle graphique et interactive est intégrée dans l'environnement de production d'objets de formes complexes développé par l'équipe CFAO du CDTA

1. Implémentation :

La réalisation des solutions proposées a été faite par l'utilisation d'Embarcadero C++ Builder 10 Seattle et la plateforme Windows. Ce travail est réalisé à travers un ensemble de fenêtres Windows proposées à l'utilisateur dans le but d'interagir avec l'application et de visualiser les résultats de notre application à travers la bibliothèque graphique OpenGL.

1.1. Langage de programmation :

Afin d'intégrer notre application dans l'environnement de reconstruction d'objet développée par l'équipe CFAO du CDTA, le travail réalisé a été implémenté en utilisant le langage C++. Ceci est motivé par le fait que toutes les applications de l'équipe CFAO ont été réalisées en utilisant ce même langage.

- **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

Chapitre III

fonctionnalités puissantes, comme par exemple la notion de classe, qui permet d'appliquer les techniques de la programmation-objet.[28]

C++Builder est un logiciel de développement rapide d'applications(RAD) conçu par Borland qui reprend les mêmes concepts, la même interface et la même bibliothèque que Delphi en utilisant le langage C++. Il permet de créer rapidement des applications Win32 ou Win64 ainsi qu'une interface graphique avec son éditeur de ressources. Il est compatible avec la version de norme ISO C++ de 2011 (C++11).[29]

Embarcadero C ++ Builder 10Seattle 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.[31]

OpenGL (Open Graphics Library) est un ensemble normalisé de fonctions de calcul d'images 2D ou 3D lancé par Silicon Graphic en 1992. Cette interface de programmation est disponible sur de nombreuses plateformes où elle est utilisée pour des applications qui vont du jeu vidéo jusqu'à la CAO en passant par la modélisation.[30]

1.2. Matériel utilisé

Pour la réalisation et le développement de notre application , nous avons utilisé le matériel suivant :

Un PC sous windows 8.1, Intel® core™ I7-5500U CPU @ 2.40GHz 2.40 GHz 8Go de RAM.

La configuration minimale pour l'exécution de l'application développée est la suivante :

- RAM : 04 Go
- Carte graphique : NVIDIA
- Processeur multi-cœurs.

Chapitre III

2. Présentation des fenêtres :

Notre application est composée d'un environnement qui englobe dans ses onglets toutes les fonctionnalités du projet. La fenêtre principale de notre projet est composée de six (06) onglets (Figure 3.1). Chaque onglet est dédié à une des tâches du processus général de l'application:

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

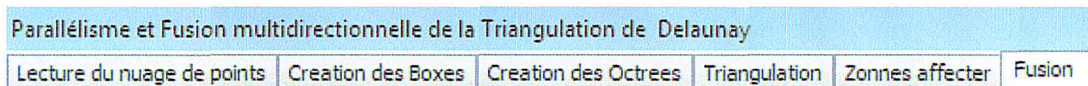


Figure3.1::Fenêtre principale de l'application.

2.1. Lecture du nuage de points :

Le premier onglet « Lecture du nuage de points » est divisé en trois parties.

La première partie permet la lecture du fichier contenant le nuage de points par simple clic sur le bouton « Ouvrir le fichier ». Par la suite, le nuage de points et le brut sont affichés.

La seconde partie consiste à calculer les coordonnées du brut (X_{min} , X_{max} , Y_{min} , Y_{max} , Z_{min} , Z_{max}).

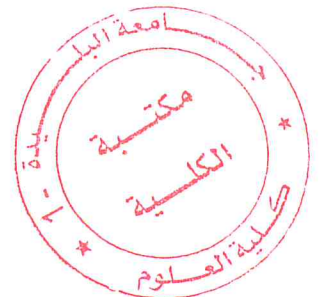
La troisième partie est consacrée au calcul des dimensions du brut (Hauteur, Largeur et Longueur) et même le nombre de points du nuage est calculé et affiché.

2.2. Création des boxes :

Cet onglet est dédié à la création des boxes. L'utilisateur a la possibilité de choisir entre deux modes de création :

- Création des boxes séquentielle.
- Création des box parallèle.

Dans les deux cas, l'utilisateur peut faire deux modes de visualisation :



Chapitre III

- Visualisation de tous les boxes.
- Visualisation d'un seul boxe.

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

- Affectation des points séquentielle (inclut dans le bouton de la création)
- Affectation des points parallèle (bouton séparer).

Une fois l'opération est faite, le temps du traitement est afficher.

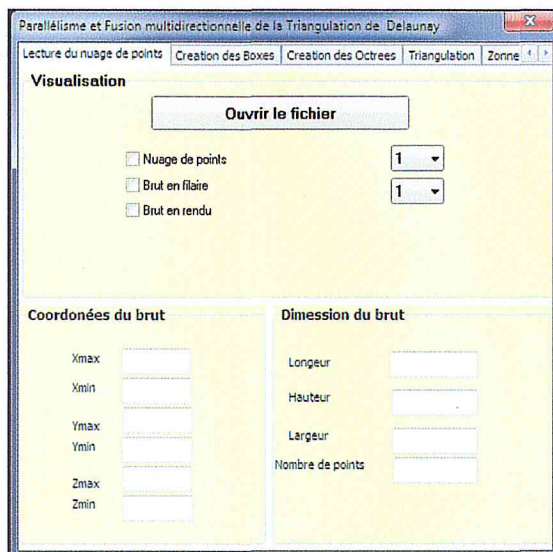


Figure 3.2 : Onglet lecture du nuage de points.

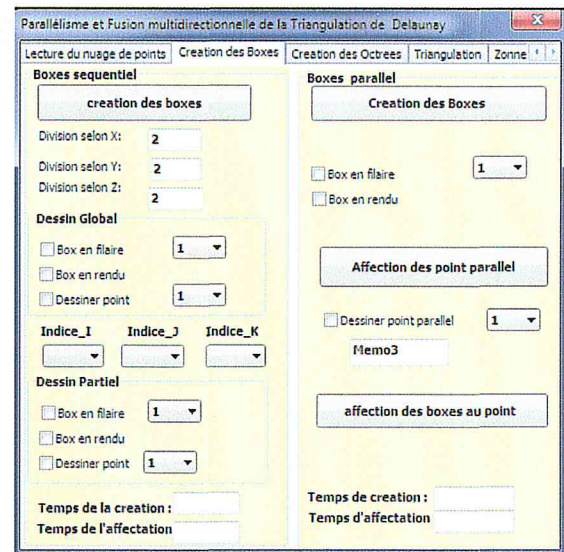


Figure 3.3: Onglet Création des Boxes.

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 Octreepère, fils, voisin

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

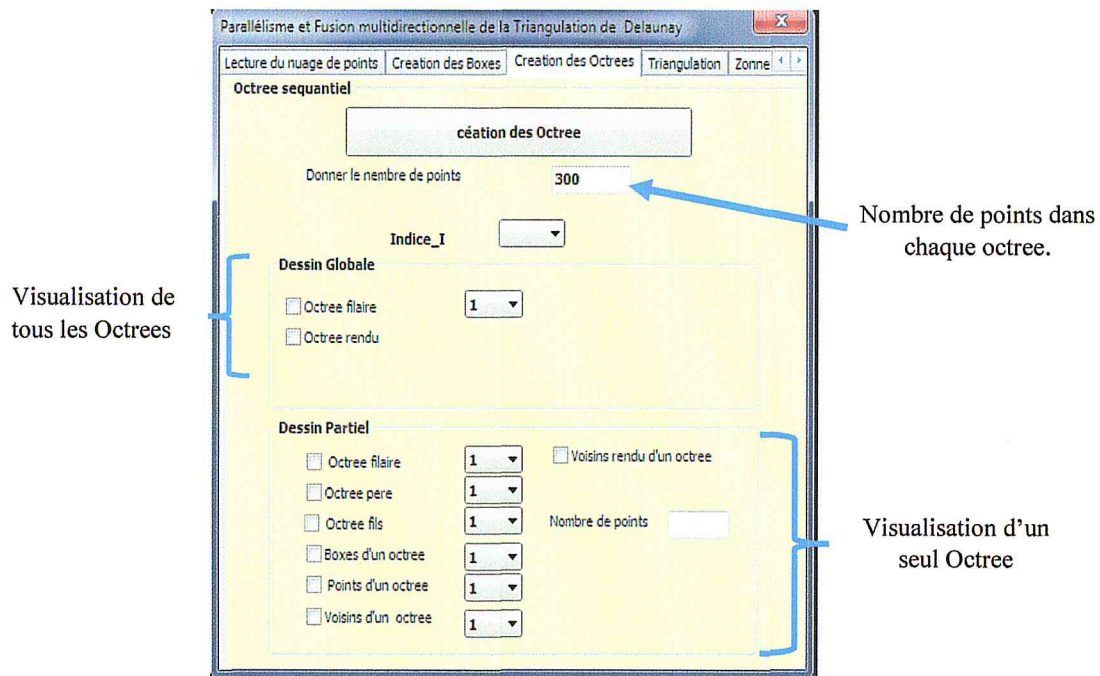


Figure 3.4: Onglet Création des Octrees.

2.4. Triangulation de Delaunay :

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

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

Une fois la triangulation est faite, le temps de traitement est calculé. Dans les deux cas, l'utilisateur a le choix entre deux types de visualisation :

- Visualisation globale de la triangulation de tous les octrees.
- Visualisation partielle de la triangulation d'un seul octree selon son indice.

Chapitre III

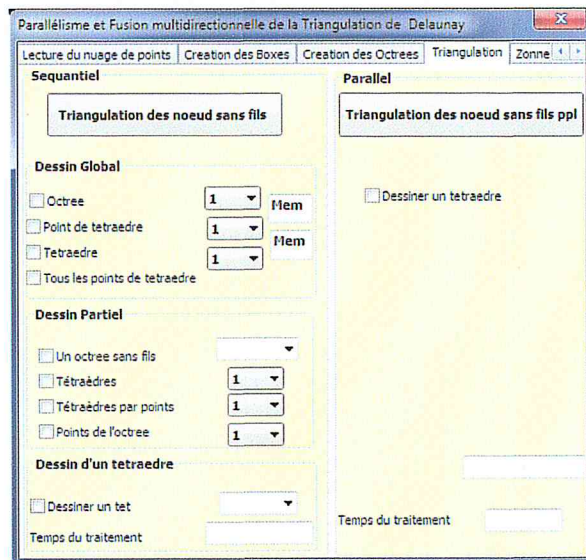


Figure 3.5 : Triangulation de Delaunay.

2.5. Zone affectée :

Cet onglet est consacré pour la création des zones affectées. Il est divisé en deux grandes parties et l'utilisateur a le choix entre deux types de zones affectées :

- Création des zones d'une manière séquentielle.
- Création des zones en mode parallèle.

Une fois la création est faite le temps de traitement est calculé. Dans les deux cas de création, l'utilisateur doit passer par les trois étapes de la création de la zone affectée par un simple clic sur les Boutons « Création des zones étape 1 », « Création des zones étape 2 », « Création des zones étape 3 » selon le besoin.

2.6. Fusion :

Cet onglet est dédié à la fusion des sous-nuages triangulés. L'utilisateur doit déterminer les indices des octrees à fusionner avec un simple clic sur le bouton « Création du premier tétraèdre »

Chapitre III

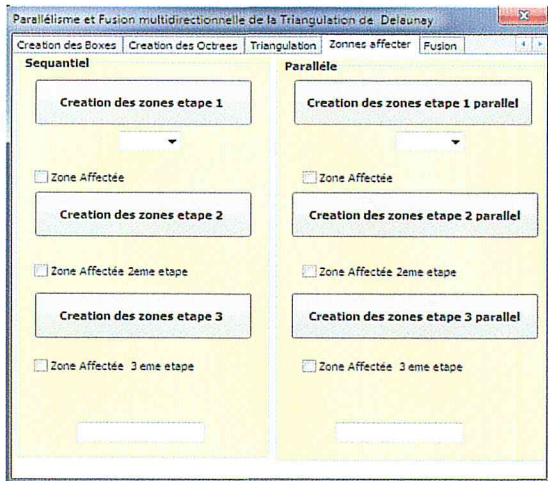


Figure 3.6: Onglet Zones affectées.

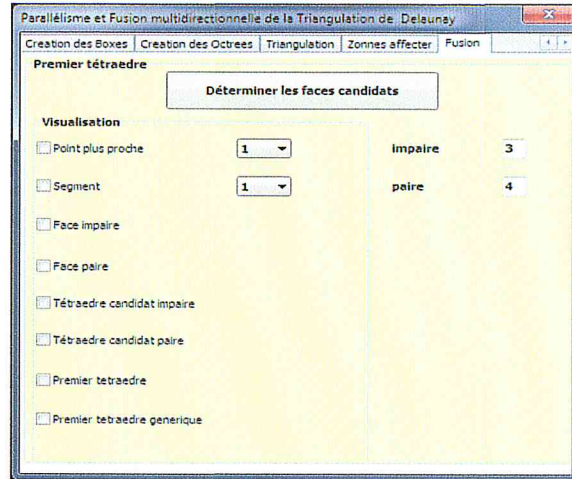


Figure3.7 : Onglet Fusion.

3. Validation :

Nous avons validé notre travail sur l'exemple du nuage de points présenté dans la (Figure3.8). 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.



Figure3.8:Modèle de validation.

- 4. Lecture de nuage de points :** la première étape inclut la lecture du nuage de points du modèle présenté, le calcul, l'affichage et la visualisation de ses limites (X_{min} , X_{max} , Y_{min} , ...) et ses dimensions (Longueur, Largeur et Hauteur). Ce fichier contient un nombre de points égal à 3323 566 points.

Chapitre III

Figure.3.9 :Résultat de lecture de nuage de points.

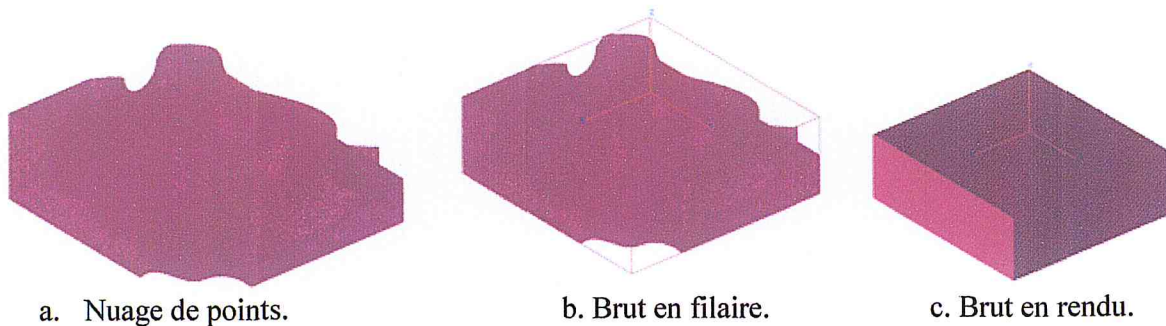


Figure.3.10 :Visualisation de nuage de points.

➤ **Subdivision** : cette étape contient deux parties :

- **Création des Box** :du fait que le fichier est très volumineux, nous avons choisi de répartir le nuage de point sur 07boxes suivant les trois axes (X, Y, Z) (nombre maximum de puissance possible).Ceci, implique 2^7 boxes dans chaque direction et le nombre total de box est de **2097152** box. Lors de la création séquentielle, on peut utiliser les fonctionnalités de visualisation globale ou de la visualisation partielle .Les mêmes fonctionnalités de visualisation peuvent être utilisées lorsqu'on clique sur le bouton de la création parallèle, sauf que la tâche est divisée en 3 boutons. Le bouton de la « Création des boxes » est activé en premier lieu, ensuite « Affectation des point parallèle » et « Affectation des boxes au point » en dernier lieu car les fonctions dans le parallélisme doivent être indépendantes.

Chapitre III

Boxes séquentiel

creation des boxes

Division selon X: 7

Division selon Y: 7

Division selon Z: 7

Dessin Global

☐ Box en filaire 1

☐ Box en rendu

☐ Dessiner point 1

Indice_I 0 Indice_J 0 Indice_K 0

Dessin Partiel

☐ Box en filaire 1

☐ Box en rendu

☐ Dessiner point 1

Temps de la creation : 0.531

Temps de l'affectation : 0.625

Boxes parallèle

Creation des Boxes

☐ Box en filaire 1

☐ Box en rendu

Affectation des point parallèle

☐ Dessiner point parallèle 1

affectation des boxes au point

Temps de creation : 0.484

Temps d'affectation : 0.297

a. Création séquentielle b. Création parallèle

Figure.3.11 : création des boxes

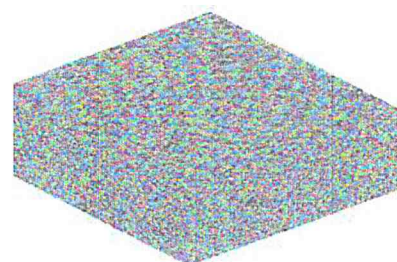
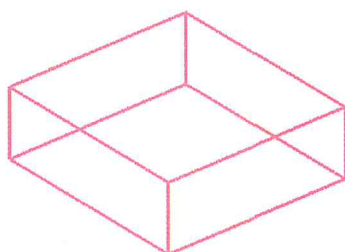
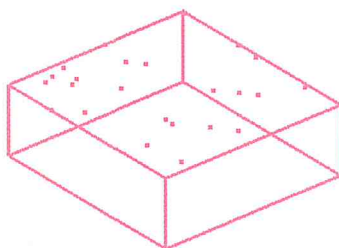


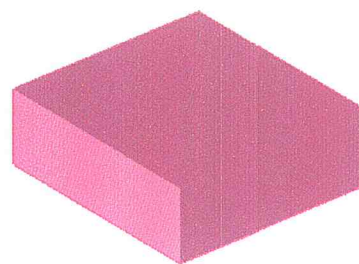
Figure.3.12 : visualisation globale des boxes.



a. Octree filaire.



b. Points d'un boxe



.c. Boxe en rendu

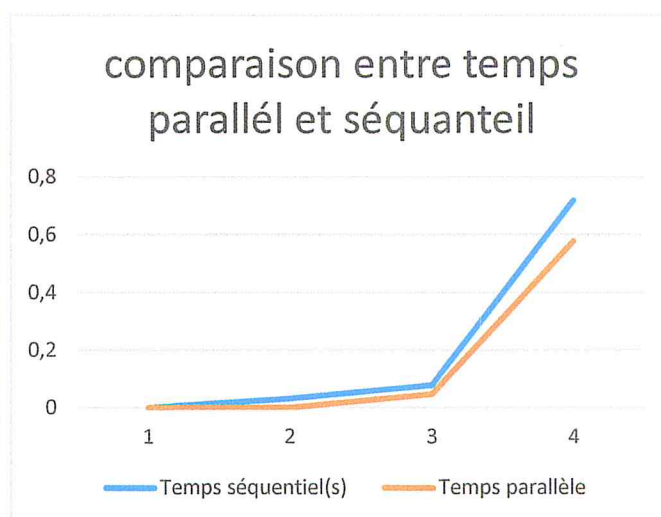
Figure.3.13 : Visualisation partielle des boxes.

Le tableau suivant contient quelques tests des temps de traitement et d'affectation en fonction de la variation des pas de création des boxes.

Chapitre III

Tableau 1.3 : Comparaison entre temps de création parallèle et séquentielle.

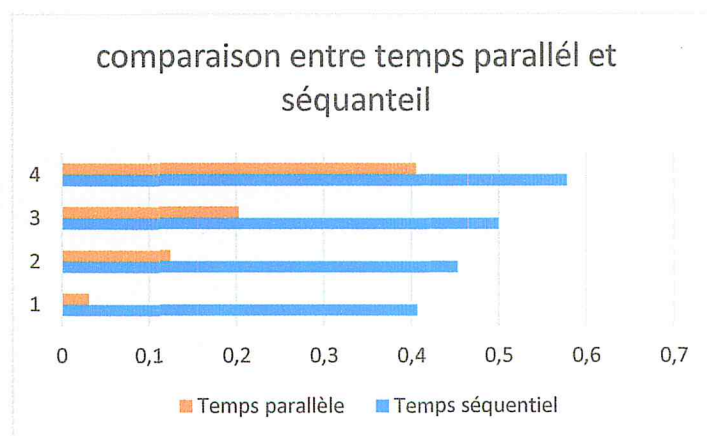
Nombre de Box	Temps séquentiel(s)	Temps parallèle (s)	Gain (%)
32768	Instantané	Instantané	Instantané
131072	0.031	Instantané	100
262144	0.078	0.047	39.7
2097152	0.719	0.578	19.6



a. diagramme relié au tableau

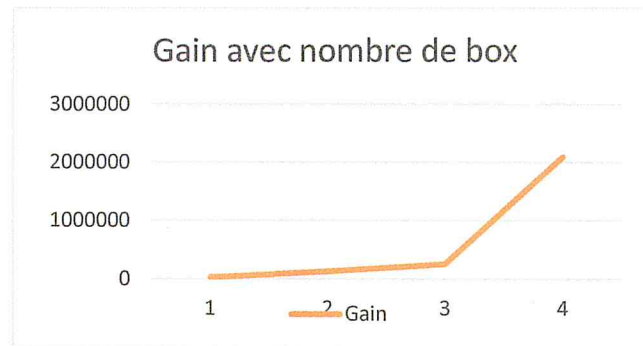
Tableau 1.2 : Comparaison entre les temps d'affectation des points parallèle et séquentielle.

Nombre de Box	Temps séquentiel (s)	Temps parallèle (s)	Gain (%)
32768	0.407	0.031	92.3
131072	0.453	0.125	75.4
262144	0.5	0.203	59.4
2097152	0.578	0.406	29.7



a. diagramme relié au tableau

Chapitre III



b. le gain du tableau

La visualisation des points de tous les boxes ou les boxes en filaire n'est pas possible à cause de la taille du nombre de points.

- **Création des Octree :** nous avons choisi de spécifier un nombre de points égal à 10 000 points dans chaque octree. Ceci implique 1 536 octrees, et la visualisation globale (tous les octrees) ou partielle (chaque octree) devient possible.

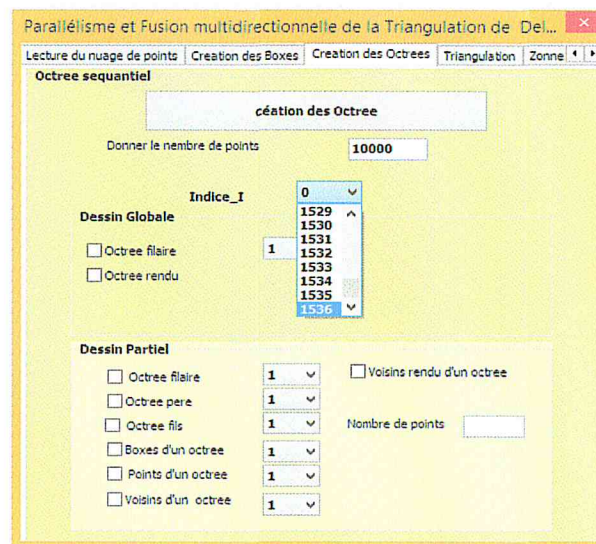
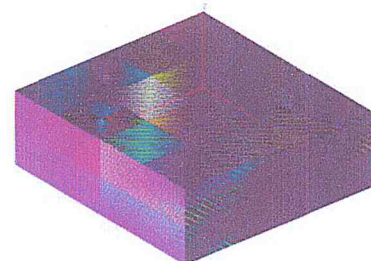
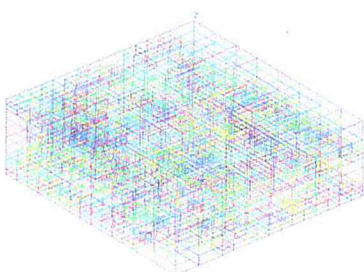


Figure.3.14 : création des Octrees.



Chapitre III

a. Octree en filaire.

b. Octree en rendu.

Figure.3.15 : Visualisation globale des Octrees.

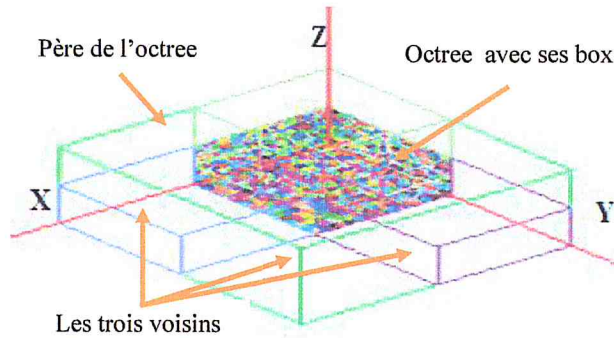
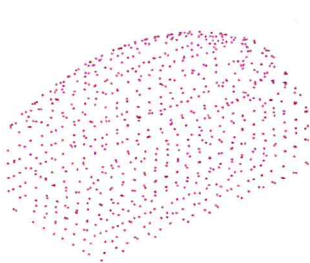


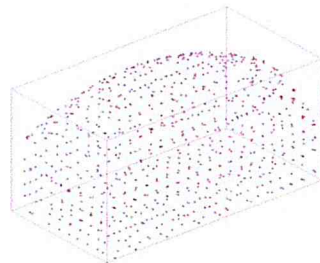
Figure.3.16 : visualisation partielle d'un Octree.

On peut remarquer que le nombre de points dans chaque octree ne dépasse pas 10 000.

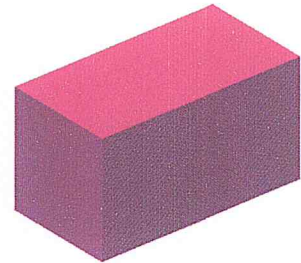
➤ **Triangulation :** dans la triangulation nous avons choisi un fichier moins volumineux (809 points) pour pouvoir visualiser les triangles créés dans chaque octree soit dans le mode séquentiel ou dans le mode parallèle.



a. Nuage de point.

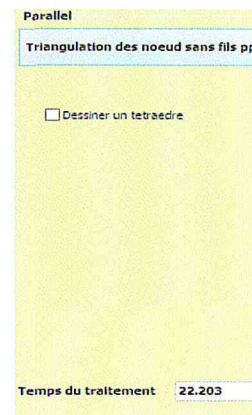


b. Brut en filaire.



c. Brut en rendu.

Figure.3.17 : Visualisation du nuage de points.

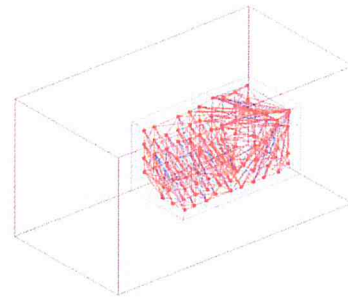
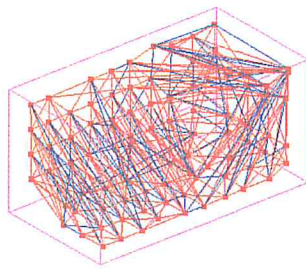


Chapitre III

a) Mode séquentiel.

b) Mode parallèle.

Figure.3.18:Triangulation du nuage de points.



a. Octree triangulé.

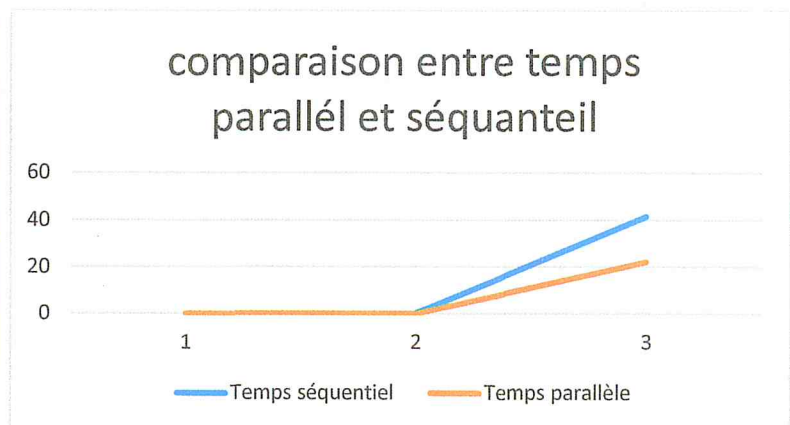
b. Octree dans le brut.

Figure.3.19:Visualisation de la triangulation

Pour voir l'effet du parallélisme dans notre travail, nous avons testé quelques fichiers avec des nombres de points différents.⁷

Tableau 3.3 :Impact du parallélisme sur la triangulation.

Nuage de point	Temps séquentiel (s)	Temps parallèle (s)	Gain (%)
55	0.047	0.032	31.9
84	0.125	0.06	52.0
809	41.563	22.203	46.0

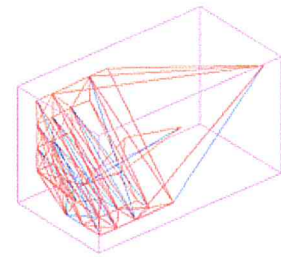
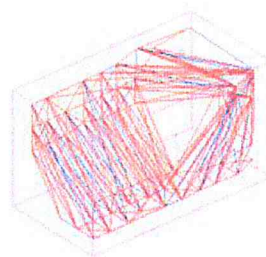
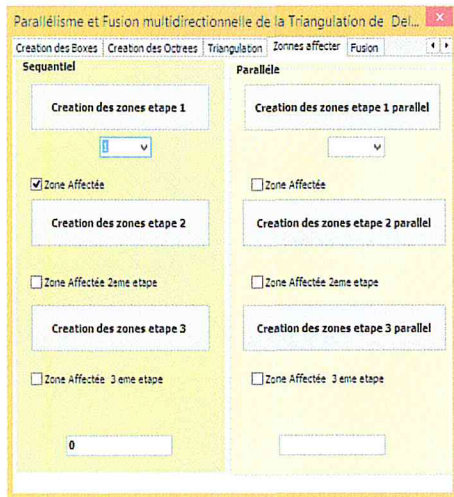


a. diagramme relié au tableau

Chapitre III

- **Zone Affecté :** la validation de la tâche 'Zone Affectée' a été faite selon les deux modes, mode séquentiel et mode parallèle.

- Etape 1 : Visualiser les parties gauche _ droite (Figure3.20.)



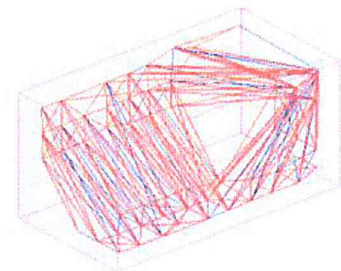
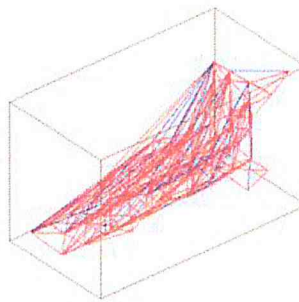
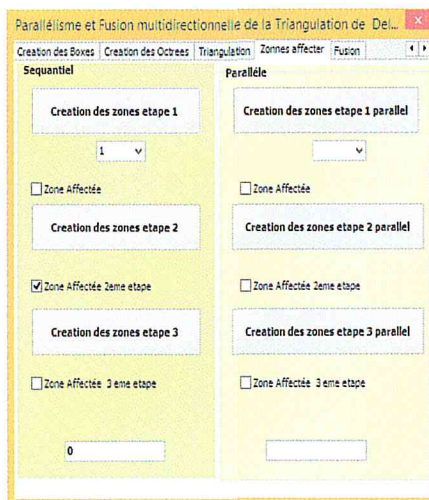
a) Zone affectée gauche-droite

b) zone affectée droite

c) zone affectée gauche

Figure.3.20 :Zone affectée gauche-droite (triangulation séquentielle).

- Etape 2 : Visualisation des parties avant _ arrière (Figure3.21)



a) Zone affectée avant-arrière

b)Zone affectée avant

c). Zone affectée arrière

Figure.3.21:Zones affectées avant-arrière (triangulation séquentielle).

- Etape 3 : Visualisation des parties haut _ bas.

Figure.3.22 :Zones affectées haut-bas (triangulation séquentielle)

Les mêmes fonctionnalités sont utilisées dans le mode parallèle.

➤ **Fusion** : Dans cette partie, le système sélectionne les deux points les plus proches des deux indices d'octree inséré (indice 3 et 4). Le segment entre les deux points, les faces de chaque octree (impaire et paire) des nœuds sans fils de l'étape 1 si elles existent. À la fin, les tétraèdres candidats sont identifiés.

Chapitre III

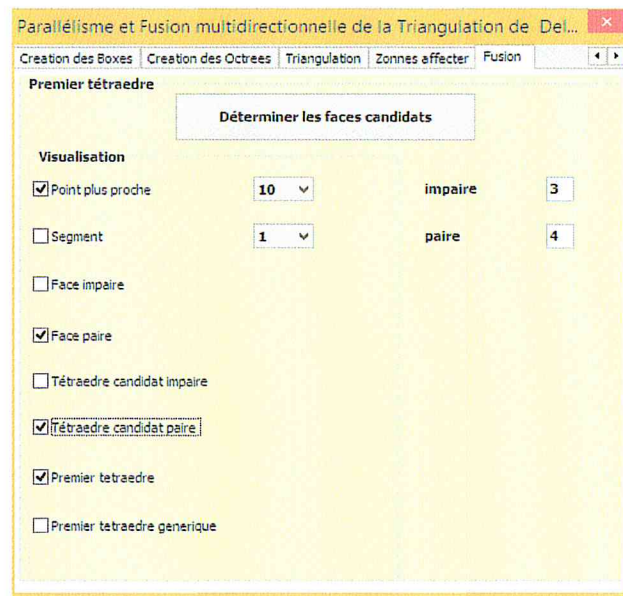
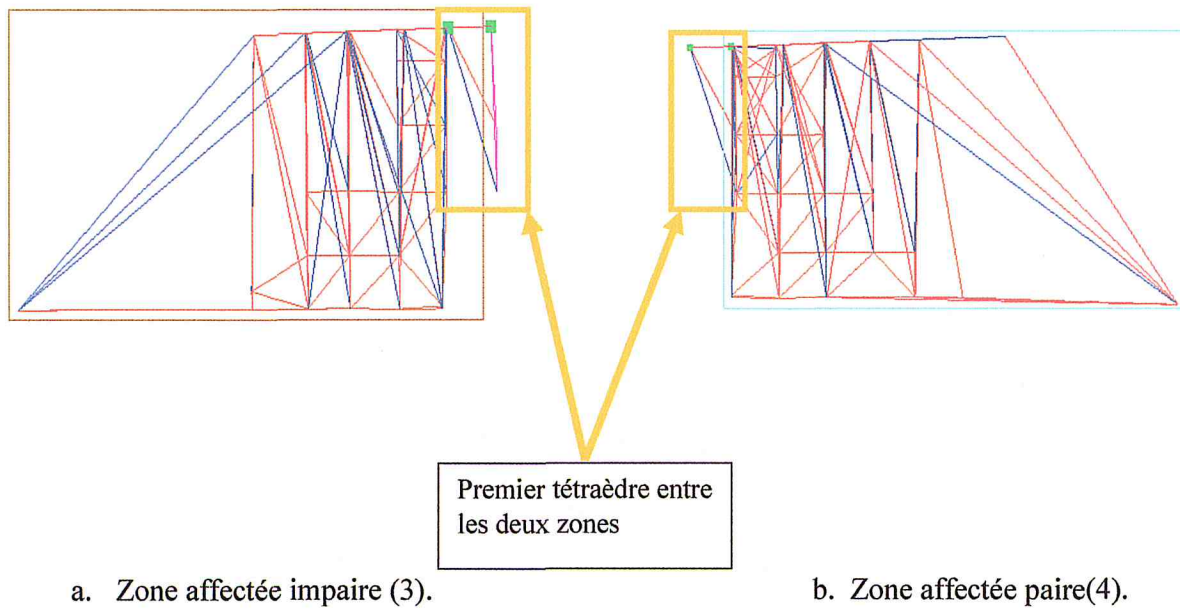
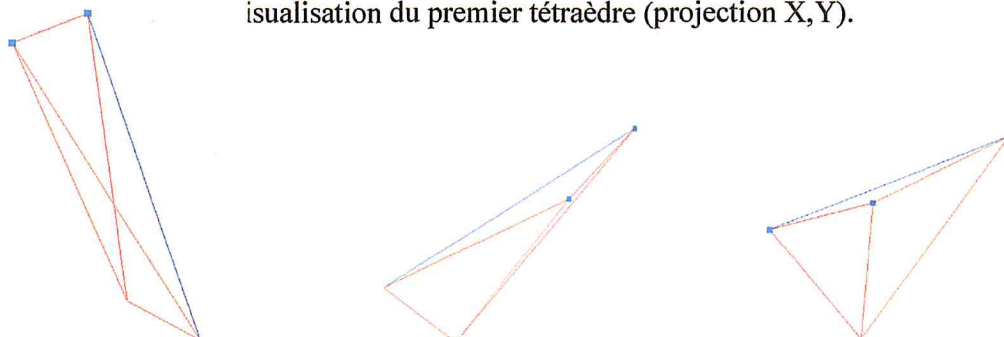


Figure.3.24: Visualisation du premier tétraèdre.



Visualisation du premier tétraèdre (projection X,Y).



Chapitre III

a) Zones 1_2 b. Zones 3_4 c. Zones 5_6

Figure.3.26:Premier tétraèdre de division par 300 points dans chaque Octree.

Conclusion :

Dans ce chapitre, les différentes étapes de notre application ont été présentées. Nous pouvons dire qu'une partie du travail a été validée à travers des exemples réels de pièces très complexes.

La validation à travers d'exemples réels reflète l'intérêt de l'application dans le domaine de la reconstruction d'objet en 3 dimensions.

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'une approche parallèle pour la reconstruction des objets en 3D sous l'environnement Embercadéro à partir d'un nuage de points quelconque en appliquant la triangulation de Delaunay.

Le premier chapitre de ce travail, a été consacré en premier lieu, à une étude générale sur les surfaces complexes, les différents algorithmes de triangulation de Delaunay en 2D et 3D et à la structure du fichier STL. En deuxième lieu, une étude préalable sur les différentes architectures du parallélisme a été effectuée. Le deuxième chapitre a résumé la conception de la solution proposée, l'architecture globale ainsi que la modélisation de l'approche adoptée. Enfin, dans le troisième chapitre, les résultats de l'approche ont été détaillés et le travail a été validé à travers d'exemples concrets de nuages de points.

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

- Lecture du nuage de points.
- Subdivision du nuage de points en boxes séquentielle et parallèle.
- Application d'une deuxième subdivision par la création des Octrees.
- Génération d'une triangulation 3D pour chaque octree en utilisant la triangulation de Delaunay 3D séquentielle et parallèle.
- Génération des zones affectées pour chaque Octree sans fils en séquentielle et en parallèle.
- Fusion par un tétraèdre entre deux Octrees.

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

- Utilisation de la GPU dans la phase de parallélisme.
- Développement d'une approche distribuée sur plusieurs PC.

Bibliographie :

- [1] Céline ROUDET, « Contribution de la reconstruction 3D à la compression de maillages surfaciques triangulaires », Rapport de stage, Master M2 Recherche Informatique, Université Claude Bernard Lyon 1.
- [2] « Conception et Fabrication Assistée par Ordinateur » Le matériel utilisé en CFAO : un SYSTÈME AUTOMATISÉ, Subtitel.
- [3] Alain Bernard « fabrication assister par ordinateur » 02/2003, Collection Traité IC2 Série Productique Editeur : HERMES / LAVOISIER
- [4] Nastase-Dan CIOBOTA, « STANDARD TESSELLATION LANGUAGE IN RAPID PROTOTYPING TECHNOLOGY », 2012, Institut national de recherche et de développement pour la mécanique et la technique de mesure Sos, l'Université VALAHIA - MATERIALS et MECHANICS, ROUMANIE
- [5] En ligne : https://fr.wikipedia.org/wiki/Fichier_de_st%C3%A9r%C3%A9olithographie
- [6] En ligne : <http://ufrsciencestech.u-bourgogne.fr>, OPTorganization infoiem.
(Représentation des surfaces, date octobre 2017. url)
- [7] Cécilia Damon, Francisco Sánchez Vega, « La triangulation de Delaunay : application au problème de la superre solution », Décembre 2004, Journal Mastère de Recherche Mathématiques
- [8] Massicot, Pillet, « Triangulation de Delaunay », Avril 2010, Ecole Normale Supérieure de Cachan, antenne de Bretagne.
- [9] Nasser yacine , ouyous Mina, « Reconstruction de surfaces d'objets 3D à partir de nuage de points », 16 mars 2015.
- [10] Raphaël Berthon et Joan Thibault, « Triangulation de Delaunay », date 8 mars 2015, ENS Rennes
- [11] En ligne : <http://fearyourself.developpez.com/tutoriel/jeu/Delaunay/#L2.2.b>.
- [12] Michal Zemek, « Regular Triangulation in 3D and Its Applications, The State of the Art and Concept » Avril 2010, Ph.D, Thesis, Université de West Bohemia,.

- [11] En ligne : http://www.datakit.com/fr/step_qu_est_ce_que_step.php octobre 2017.
- [14] Franck CAPPELLO et Jean-Paul SANSONNET « Introduction au parallélisme et aux architectures parallèles », 05/08/2017, la version actualisée de l'article [H 1 088].
- [15] Manser, « les architecture parallèle », cour system d'exploitation, Université de Saad Dahleb ,Blida, Algérie.
- [18] Dr.N.Bounour « Notion de processus, Parallélisme, coopération et de compétition » Chapitre 1.SE II, 3ème LMD.
- [19] Rédha LOUCIF « Mémoire Magister Parallélisation d'Algorithmes d'Optimisation Combinatoire », 13/01/2014, Rapp. tech. université colonel Hadj Lakhdar-Batna,.
- [20] : Dr. Olfa Hamdi-Larbi « Sources du parallélisme », 2010-2011, Cours pour Ingénieur
- [21] 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
- [22] Joris Ravaglia, Alexandra Bac, Richard Fournier. « Segmentation de nuages de points par octreeset analyse en composantes principales ». GTMG 2014, Mar 2014, Lyon, France.
- [23] En ligne : <https://fr.wikipedia.org/wiki/Quadtrees>
- [24] Frédéric Legrand, « Enveloppe convexe », Licence Creative Commons. Licence Creative Commons.
- [26] Bruno Bouzy « UNIFIED MODELING LANGUAGE (UML) » , 1 juin 2014.
- [25] « Langage de modélisation objet unifié Diagramme des cas d'utilisation » unified modeling language TM .
- [27] « diagramme de classe notation graphique » Université de Ouargla, 2013/2014
- [28] O. Marguin « C++ : LES BASES » , 2003/2004
- [29] : En ligne <https://fr.wikipedia.org/wiki/C%2B%2BBuilder>

[30] : En ligne <https://fr.wikipedia.org/wiki/OpenGL>

[31] : En ligne : <https://www.embarcadero.com/>

