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

C.D.T.A

MEMOIRE DE FIN D'ETUDES

Pour l'obtention

D'un diplôme de master en informatique.

Spécialité : **Ingénierie logiciel (IL)**

THÈME :

Optimisation des performances et parallélisme pour la Reconstruction de surface d'objet 3D à Partir d'un Nuage de Points

Réalisé par :

Mr AIT MESSAOUDENE Boualem

Mr LEKRIM Mohamed

Soutenue le : 31-10-2017, devant :

Mme ZAHRA F.Zohra	Promoteur
Mme TCHANTCHANE Zahida	Encadreur
Mr BENDIFALLAH Mohamed El Hassène	Encadreur
Mr BEY Mohamed	Encadreur
Mr CHRIF-ZAHAR Sid-Ahmed Amine	Président
Mr KAMECH Abdellah Hicham	Examinateur

2016/2017

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 Mme. TCHANTCHAN, Mr. BENDIFALLAH et Mr. BEY de nous avoir encadrés dans notre mémoire de fin d'études et de nous avoir dirigés dans notre travail.

Nous remercions également Mme. ZAHRA pour son aide précieuse et son encouragement qu'elle n'a cessé de nous communiquer.

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 pourrons 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

A vous très chers mère et père,

Mes Frères Abderezak et Tarek

A toute ma famille plus particulièrement mes grands parents

A nos encadreurs Mme. TCHANTCHANE, Mr. BENDIFALLAH

ET Mr. BEY

A ma promotrice Mme. ZAHRA

A tous mes collègues, plus particulièrement Karim et Lotfi

En témoignage de mon amitié sincère;

A tous mes amis, plus particulièrement Amine, Amir, Cherif, Hocine, Marouane et Nassim

A tous ceux qui m'ont soutenu, qu'ils trouvent ici l'expression de mon amour et ma profonde gratitude

MR. BOUALEM AIT MESSAOUDENE

Dédicaces

Je dédie ce modeste travail
A vous très chers mère et père,
Mon frère Yacine
A toute ma famille
A nos encadreurs Mme. TCHANTCHANE, Mr. BENDIFALLAH
et Mr. BEY
A ma promotrice Mme. ZAHRA
A tous mes collègues, plus particulièrement Karim et Lotfi

A tous mes amis
En Témoignage de mon amitié sincère;
A tous ceux qui m'ont soutenu, qu'ils trouvent ici l'expression de
mon amour et ma profonde gratitude

MR. MOHAMED LEKRIM

Résumé

Ce travail s'insère dans le cadre du développement d'une plateforme logicielle pour la production des surfaces de formes complexes initiée 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 méthodologie permettant la réduction des temps de traitement en parallélisant les calculs sur plusieurs processeurs. Le but de ce travail est le développement d'une application logicielle distribuée graphique et interactive sous Windows automatisant cette tâche.

Mots-Clés : Application Distribuée, Reconstruction, Triangulation de Delaunay, Modèle STL, Nuage de Points

Abstract

This work is a part of the software developed by "CAD/CAM" team of the Center for the Development of Advanced Technologies "CDTA" dedicated to the machining of free form surfaces on CNC milling machines.

In this project, we are interested in generating a triangulation approximating the outer skin of objects of any shape from any clouds of points. The work consists in proposing a methodology allowing the reduction of the processing times by parallelizing the calculations on several processors. The goal of this work is the development of a graphical and interactive distributed software application on Windows automating this task.

Keywords: Distributed Application, Reconstruction, Delaunay Triangulation, STL Model, Cloud of points

Sommaire

Chapitre IV : Implémentaion et validation

Liste des figures

Liste des tableaux

Introduction générale

Avec l'évolution technologique dans les différentes industries tels que l'industrie automobile, robotique ou médicale, le besoins de produire des objets plus complexes est toujours présent avec plus de détails et un degré de précision plus élevé.

Avec l'apparition de la 3D, une nouvelle vision dans l'espace a donné la possibilité de visualiser le modèle sous toutes les perspectives, ce qui permet une meilleure représentation et approximation de l'objet surtout quand il s'agit d'objets très complexes.

Néanmoins, les techniques de construction des surfaces restent limitées ou insuffisantes pour permettre la réalisation des formes souhaitées. Souvent, le concepteur est tributaire des fonctionnalités du logiciel mis à sa disposition. La forme résultante ne correspond donc pas forcément à l'intention du concepteur mais est la plus proche représentation géométrique que le modeleur permette d'obtenir. En général, ces surfaces sont conçues selon deux approches :

L'une dite classique, basée sur la modélisation de l'objet selon un cahier des charges en utilisant des fonctions de conception et de manipulation offertes par les logiciels de conception assistée par ordinateur « CAO ». L'inconvénient de cette approche réside dans la difficulté d'obtenir les formes désirées sans recourir à des phases de modifications et de validations successives, afin de respecter les contraintes géométriques (tangentes, courbures, etc.) et fonctionnelles imposées, ce qui conduit à des temps de traitement très importants.

L'autre approche dite ingénierie inverse, est utilisée dans le cas où les formes des objets sont très complexes et ne peuvent être conçues dans un logiciel de CAO, ou si le modèle de description de l'objet n'est pas disponible. Elle est basée sur la modélisation de l'objet à partir de nuages de points, Cette approche est devenue de nos jours une pratique très courante dans le monde industriel, puisqu'elle permet de réduire les temps de développement de nouveaux produits à partir de prototypes.

Problématique

La reconstruction des modèles en trois dimensions avec une bonne précision est une tâche ardue, coûteuse en temps, dépendant des possibilités de modélisations disponibles 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 et par conséquent les coûts pour la génération d'une triangulation surtout si un seul processeur est utilisé.

Objectifs et besoins du travail

Ce travail s'inscrit dans le cadre de développement d'une application logicielle distribuée graphique et interactive sous Windows automatisant la tâche de 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 en premier lieu à la génération d'une triangulation en 3D, plus précisément un modèle STL approximant la peau extérieure d'objets de n'importe quelles formes à partir de nuages de points quelconques, puis la proposition d'une méthodologie permettant la réduction des temps de traitement en parallélisant les calculs sur plusieurs processeurs ou plusieurs cœurs. Afin de remédier aux problèmes cités ci-dessus, nous avons assigné à notre étude les objectifs suivants :

- Génération d'une triangulation 3D (modèle STL) approximant la peau extérieure d'objets de n'importe quelles formes à partir de nuages de points quelconques.
- Proposition d'une méthodologie permettant la réduction des temps de traitement.
- Parallélisme des calculs sur plusieurs processeurs.

Structure du mémoire

Le présent mémoire est composé des chapitres suivants :

Le premier chapitre est réservé à l'étude de la modélisation des surfaces et reconstruction d'objet. Le processus de conception d'objet, le processus de l'ingénierie et les méthodes de reconstruction des objets 3D sont explicités. Le deuxième chapitre traite des architectures parallèles. Plusieurs classifications sont introduites ainsi que la programmation séquentielle et parallèle. Dans le troisième chapitre, la conception de l'application et les différentes approches et méthodes de résolution du problème sont détaillées. Enfin, le dernier chapitre conclue le travail à travers son implémentation et sa validation sur une pièce réelle.

Chapitre I

Modélisation des surfaces et reconstruction d'objets

Introduction

Depuis quelques années, les modèles géométriques tridimensionnels sont de plus en plus employés. Ils sont apparus avec le développement de l'infographie et de la vision tridimensionnelle, qui ont permis une modélisation 3D d'objets de formes complexes. Ces modèles peuvent être créés par des systèmes de modélisation comme les logiciels de conception assistée par ordinateur « CAO » ou obtenus grâce à des procédés de numérisation tridimensionnelle d'objets physiques. Ils sont généralement représentés sous forme de maillages surfaciques ou volumiques. Ces modèles peuvent aussi être décrits par des nuages de points, des formes à pôles (Bézier, B-Spline, NURBS, ...) ou des surfaces implicites [1].

I. Processus de conception d'objets de formes complexes

I.1 Modélisation 3D des objets complexes

La modélisation est la représentation des formes et des dimensions des objets. Elle peut être filaire, surfacique ou volumique (Figure 1). Dans ce qui suit, nous allons détailler la modélisation surfacique des objets de notre étude.

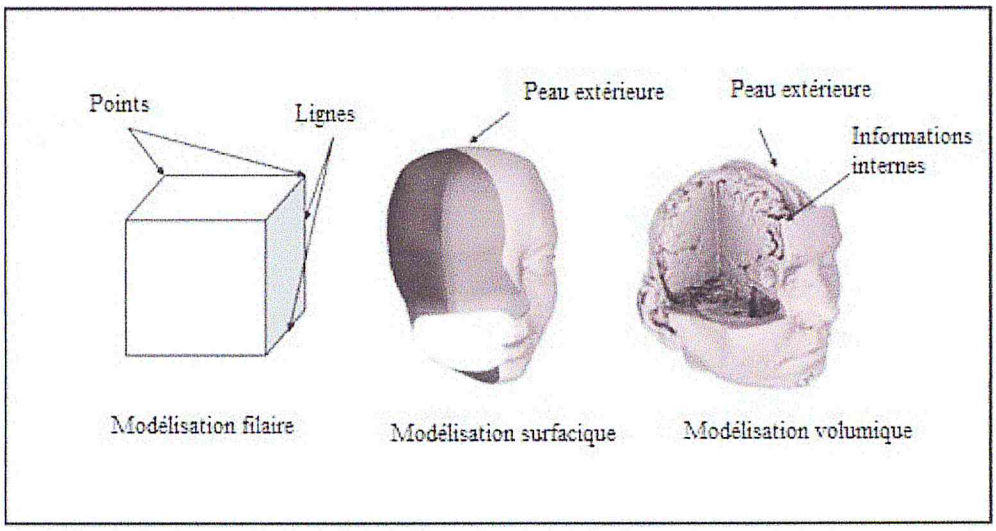


Figure 1: Différents niveaux de modélisation.

I.1.1 Modélisation filaire

Elle est basée sur des points et des segments, l'objet est décrit par ses sommets (points) et ses arêtes (segments reliant ces sommets) [2].

I.1.2 Modélisation surfacique

La description de ce modèle est basée sur des concepts mathématiques d'approximation et prend en compte la notion de surfaces simples et de surfaces complexes.

L'objectif de cette modélisation est la définition de l'enveloppe et des surfaces frontières de l'objet. Il existe deux types de modélisation surfacique [3] :

a). Surfacique à facette plane : c'est une forme géométrique simple inscrite ou conçue dans un plan, souvent le triangle est considéré comme la facette la plus simple.

b). Surfacique gauche : cette méthode est employée quand la surface à définir est trop complexe pour être définie par des surfaces simples (planes, cylindriques, etc.).

Une surface est dite gauche dans les cas suivants [4] :

- Une surface qui ne peut pas être définie avec une équation mathématique exacte.
- Une surface qui est obtenue par raccordement de plusieurs morceaux de surfaces (Figure 2).

a. Surface simple. *b. Surface facétisée.*

Figure.2: Différence entre une surface simple et une surface gauche.

I.1.3 Modélisation volumique

C'est la modélisation la plus complète. Elle englobe la modélisation filaire et surfacique. Elle permet la représentation de l'objet dans l'espace en intégrant la notion de matière [4].

I.2 Méthodes de modélisation

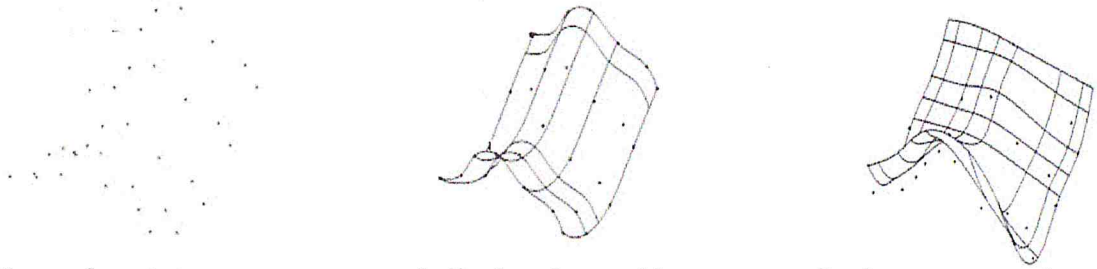
Pratiquement, dans la plupart des situations de conception de nouvelles surfaces, le concepteur n'a en tête que la forme de la surface qu'il veut obtenir sans connaitre ni sa représentation mathématique ni ses propriétés géométriques. Il existe plusieurs méthodes de modélisation des surfaces. Celle qui nous intéresse, c'est la méthode basée sur l'utilisation des points.

I.2.1 Méthodes basées sur l'utilisation des points

Trois approches peuvent être utilisées pour la génération d'une surface à partir d'un ensemble de points décrivant la forme de la surface (Figure 3.a) [5] :

a). Interpolation : c'est la génération d'une surface qui passe par l'ensemble de points (Figure 3.b).

b). Approximation : génération d'une surface qui approxime le mieux un ensemble de points avec une précision imposée mais qui ne passe pas nécessairement par ces points (Figure 3.c)



a. Nuage de points. *b. Surface interpolée.* *c. Surface approximée.*

Figure 3: Modélisation de surface.

c). Surface paramétrique : c'est une surface continue définie par un réseau de points de contrôle représentant la forme de la surface (Figure 4).

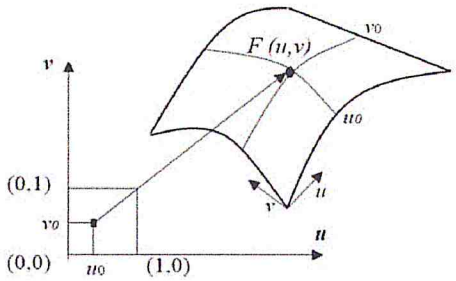


Figure 4: Surface paramétrique.

II. Ingénierie inverse

L'ingénierie inverse ou Reverse Engineering consiste à dupliquer un produit existant pour lequel les dessins, la documentation ou les modèles informatiques sont inexistants. Dans une procédure d'ingénierie directe, le produit est fabriqué sur la base d'une fiche de spécifications techniques et technologiques détaillées dans un cahier des charges. Par contre, le Reverse Engineering suit la procédure inverse. L'objet est réalisé à travers l'analyse de sa structure et des fonctions qu'il réalise. Il est très recommandé dans certaines situations telles que [6, 7] :

- Absence du produit original.
- Arrêt de la production du produit original.
- Perte de la documentation de conception du produit original.
- Complexité de la surface pour être conçue par la méthode directe.

II.1 Définition et domaines d'application

L'ingénierie inverse appliquée à la conception d'objets, est une technique qui permet d'associer une représentation numérique à un modèle physique existant. Cette technique peut être utilisée comme un outil de conception pour produire une copie d'un objet. Son principe

repose sur le traitement à travers des logiciels de reconstruction de surfaces, de nuages de points acquis par digitalisation des surfaces de l'objet [6, 7]. Ce processus est une discipline relativement jeune, qui intéresse plusieurs domaines d'application et de recherches tels que le génie mécanique, l'ingénierie médicale, la vision artificielle, la robotique, l'automobile ou encore l'aérospatial [6].

II.2 Processus de l'ingénierie inverse

Le processus de l'ingénierie inverse appliqué à la reconstruction d'objets nécessite le passage par trois principales étapes (Figure 5) [8] :

Étape 1 : digitalisation ou la numérisation de l'objet pour l'acquisition de nuages de points.

Étape 2 : traitement des nuages de points.

Étape 3 : reconstruction du modèle CAO à partir de nuages de points.

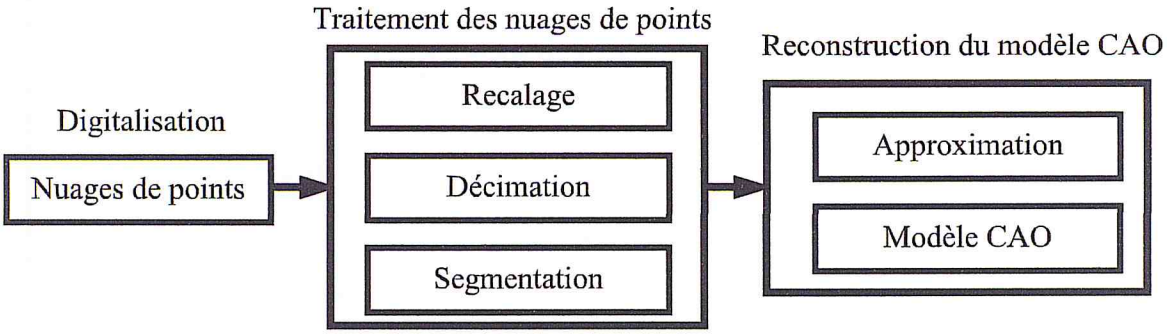


Figure 5: Processus de reconstruction d'un objet [9].

II.2.1 Numérisation de l'objet

La numérisation est l'opération qui consiste à palper l'objet pour récupérer un nuage de points. Cette mesure se traduit par un ou plusieurs ensembles de points 3D représentés par les coordonnées cartésiennes qui, une fois assemblés, forment l'image en profondeur de l'objet. L'ensemble de ces triplets offre une définition de la, ou des surfaces de l'objet mesuré sans indication de sa topologie, de sa géométrie, de ses frontières ou de ses lignes de séparation.

Il existe un grand nombre de méthodes d'acquisition de formes simples ou complexes. Chaque méthode utilise un mécanisme ou un phénomène physique afin d'interagir avec la surface de l'objet à numériser. Ces méthodes sont classées principalement en deux types :

a). Méthodes à contact : les méthodes à contact agissent directement sur l'objet. Au touché de l'objet, un signal est émis, l'évènement point se produit et la prise des coordonnées est exécutée. Dans ce type de méthodes, les machines à mesurer tridimensionnelles (MMT) sont les plus fréquemment utilisées (Figure 6).

b). Méthodes sans contact : ces méthodes utilisent soit la lumière, soit le son ou alors les champs magnétiques comme agent de mesure à partir des entités physiques tels que le « capteur CCD source laser »..

a. Objet réel *b. Machine MMT* *c. Objet numérisé.*

Figure 6: Numérisation d'objets.

II.2.2 Recalage

Un objet de forme complexe ne peut pas être numérisé en une seule passe. Donc, plusieurs points de vue sont nécessaires. Chaque positionnement de l'objet requiert une nouvelle passe de numérisation. Cette numérisation donne naissance à plusieurs nappes de points qui se superposent partiellement (Figure 7). Cette étape consiste à aligner les images de profondeur dans un même repère pour ensuite les fusionner pour former un seul nuage de points réparti sur toute la surface numérisée de l'objet [7].

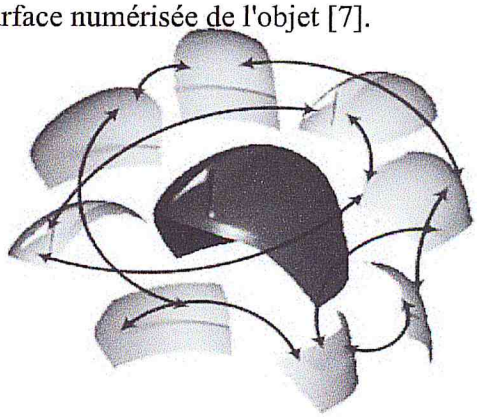


Figure 7: Recalage des nuages de points.

II.2.3 Décimation

Une simplification s'avère parfois nécessaire afin de rendre le nuage de points manipulable et traitable sur ordinateur avec une puissance de calcul et des ressources mémoires raisonnables. L'objectif de la décimation est de réduire le nombre de points du nuage tout en garantissant une définition satisfaisante de l'objet (Figure 8) [9].

a. Avant décimation. *b. Après décimation.*

Figure 8: Décimation d'un nuage de points [14].

II.2.4 Segmentation

La segmentation du nuage de points est l'une des étapes les plus critiques et qui consiste à subdiviser le nuage de points en un nombre fini de régions distinctes (régions caractéristiques) en se basant sur un certains ensemble de critères (Figure 9) [6].

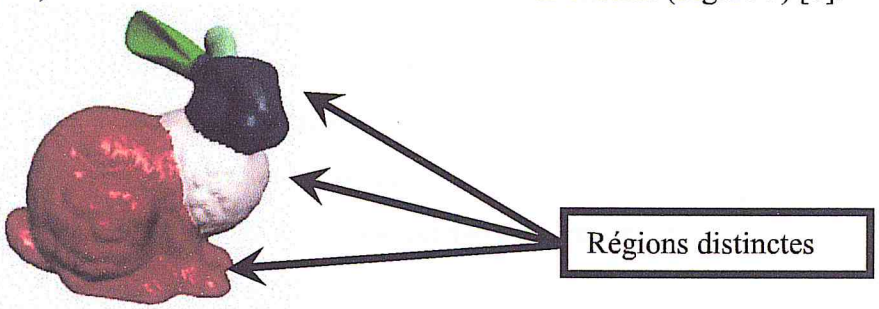


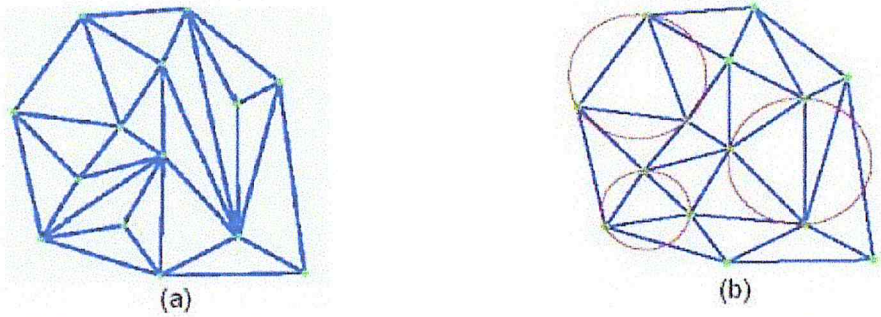
Figure 9 : Segmentation d'un nuage de points.

II.2.5 Approximation des surfaces

Elle consiste à reconstituer la surface numérisée à partir de l'échantillon de points. Cette étape de reconstruction est au centre de la problématique de notre projet. Une fois que les régions sont déterminées, il faut les approximer par des modèles géométriques de surfaces adéquats, en utilisant des techniques de lissage comme l'interpolation et l'approximation tout en assurant une certaine continuité entre les frontières de ces surfaces. Les surfaces résultantes sont des surfaces qui approximent le nuage de points (Figure 10).

a. Nuage de points d'un objet. *b. Approximation du nuage par des triangles.*

Figure 10: Approximation par des triangles.



a. Triangulation quelconque. *b. Triangulation de Delaunay.*

Figure 11 : Triangulation d'un ensemble de points [2].

III.2 Triangulation de Delaunay

La triangulation consiste à mailler un nuage de points avec des triangles ayant pour sommets les points du nuage et des arêtes qui relient ces derniers pour une représentation 2D ou 3D. Ce maillage doit répondre à certaines propriétés de la triangulation, à savoir que l'intersection de deux triangles est soit un ensemble vide, un point ou une arête.

III.2.1 Dérivés de Delaunay

Parmi les méthodes basées sur l'utilisation de la triangulation de Delaunay:

- **Boissonnat** : elle utilise la triangulation de Delaunay pour reconstruire une forme 3D unique en sculptant cette triangulation. Mais, cette méthode reste trop vague [11].
- **Alpha Shape** : cette méthode permet de reconstruire à partir de la triangulation de Delaunay une famille de formes qui dépendent d'un paramètre alpha [11].

III.2.2 Propriétés de la triangulation de Delaunay

La triangulation de Delaunay est caractérisée par plusieurs propriétés [12] :

Propriété 1 : chaque triangle est entouré d'un cercle vide qui passe par les sommets du triangle et ne contient aucun autre site ou sommet à l'intérieur (Figure 12).

Propriété 2 : les sommets et les côtés ouverts de l'ensemble des triangles de Delaunay sont deux à deux disjoints.

Propriété 3 : la triangulation de Delaunay maximise le minimum des angles des triangles (Figure 13.b).

Propriété 4 : la triangulation de Delaunay d'un nuage de points *S* est la seule triangulation de *S* dont toutes les arêtes sont légales (Figure 15).

Pour générer la triangulation de Delaunay, plusieurs méthodes peuvent être utilisées telles que Destruction-Construction et Flip.

Propriété 5 : selon *flip* $\beta > \delta$ et $\gamma > \alpha$ (Figure 16). Où β, δ, γ et α sont des angles.

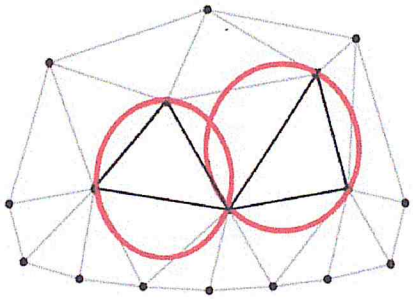


Figure 12 : Cercle circonscrit.

a. Quelconque. *b. Delaunay.*

Figure 13 : Triangulation de Delaunay

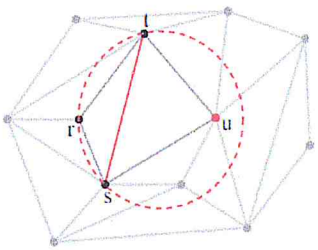


Figure 14 : Arête illégale.

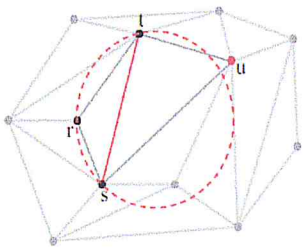


Figure 15 : Arête légale.

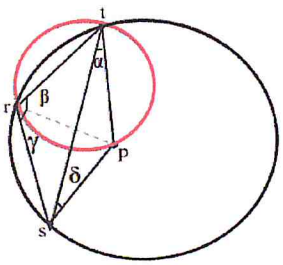


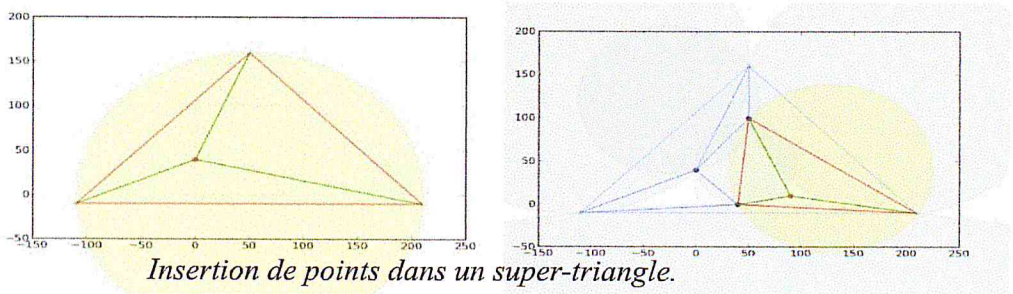
Figure 16 : Propriété des angles.

III.2.3 Méthodes basées sur la triangulation de Delaunay

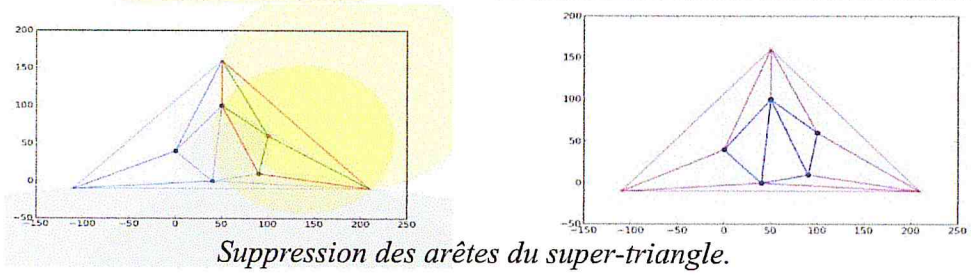
Il existe plusieurs algorithmes permettant la génération de façon incrémentale la triangulation de Delaunay. La technique incrémentale consiste à insérer des points à une triangulation initiale:

a) Triangulation par Destruction/Construction : elle se fait à partir d'une triangulation partielle du nuage de points et utilise deux fonctions: une récursive pour la suppression des triangles (destruction), et une autre pour la création de nouveaux triangles (construction). Elle passe par les étapes suivantes (Figure 17) [13] :

1. Création du super triangle à partir de trois points de départ.
2. Identification des triangles dont le cercle circonscrit contient le point ajouté.
3. Stockage de toutes les arêtes de ces triangles
4. Suppression des arêtes dupliquées.
5. Suppression des triangles dont le cercle circonscrit contient le point ajouté.
6. Création de nouveaux triangles en utilisant le point inséré courant et les arêtes stockées
7. Ajout de nouveaux triangles à la triangulation
8. Suppression des triangles ayant au moins une arête en commun avec le super triangle.



Insertion de points dans un super-triangle.



Suppression des arêtes du super-triangle.

Figure 17 : Insertion de points et suppression des arêtes.

b) Méthode Divide-Conquer: le principe de la méthode consiste à imposer un ordonnancement aux points initiaux selon l'axe des abscisses puis selon l'axe des ordonnées. Par la suite, cet ensemble est divisé en deux sous-ensembles en gardant le même nombre de

points (à l'unité près) dans les deux sous-ensembles (Figure 18). La division est appliquée sur les deux sous-ensembles jusqu'à l'obtention des ensembles d'au plus trois (03) points. Chaque couple d'ensembles est fusionné par la suite pour obtenir la triangulation de Delaunay relative à ces deux ensembles en supprimant certaines arêtes en cas de nécessité. Ainsi, un nouvel ensemble constitué de triangles est obtenu. On réitère la fusion entre deux ensembles et cela jusqu'à ce qu'il n'y ait plus d'ensemble à fusionner [14].

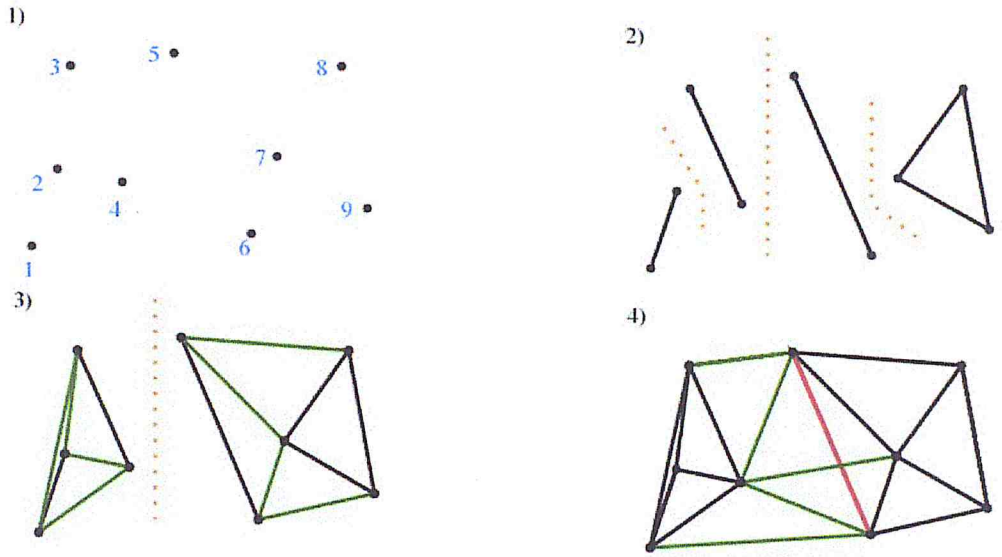


Figure 18 : Méthode Divide-Conquer.

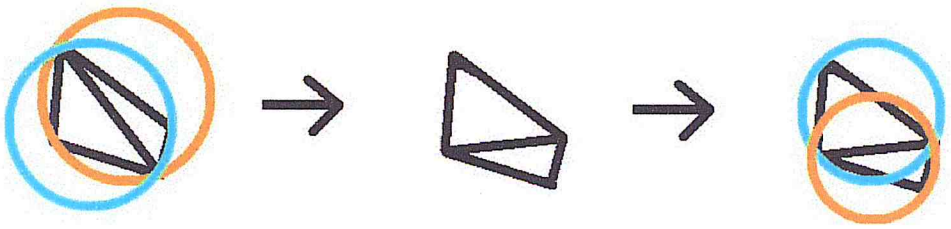
Le schéma ci-après résume la structure de l'algorithme de la méthode Divide-Conquer :

```
Entrée : E, ensemble considéré
Division(E) :
        Si taille(E) <= 3 Alors
                Renvoyer Triangle(E);
        Sinon
                Diviser E en deux :  E1, E2;
                Renvoyer( Fusion( Division(E1), Division(E2) ) );
Fin;
```

Schéma 1: Algorithme diviser pour régner

c) Triangulation de Flip : l'algorithme de Flip est très rapide par rapport aux deux méthodes précédentes. C'est une grande fonction récursive, qui peut être implémentée en itératif, qui vérifie si les côtés sont légaux. Un côté est défini comme étant l'arête commune à deux triangles. Pour savoir s'il est illégal, seuls les points qui sont présents dans un des

triangles subissent le test du cercle circonscrit et donc les points opposés. Si le cercle circonscrit contient des sommets d'un triangle voisin, l'arête commune (illégale) est remplacée par une arête côté relié par les deux points opposés et les quatre côtés qui restent sont testés pour voir si ce changement a provoqué un autre conflit (Figure 19).



Un côté illégal *Flip*

Figure 19 : Triangulation par Flip.

D'autres problèmes peuvent surgir tels que les triangles prolongés. Dans ce cas, un point est ajouté et la triangulation localement est refaite. Ainsi, l'algorithme de Flip peut être décrit d'une façon simple, comme suit:

- Insérer le point, détruire le triangle le contenant.
- Créer trois triangles qui ont ce nouveau point comme sommet.
- Vérifier chaque côté externe pour une illégalité [15].

.III.3 Format d'échange de données « STL »

STL est l'abréviation de « Standard Tessellation Language ». C'est un format d'échange de données utilisé dans les logiciels de **Stéréo lithographie**. Il Permet de décrire un maillage surfacique par des triangles, où chaque triangle est défini de manière unique par sa normale unitaire dirigée vers l'extérieur (Figure 20), et les coordonnées X, Y et Z de ses trois sommets répertoriés dans le sens trigonométrique (Figure 21). Chaque triangle doit partager deux (2) sommets avec chacun des triangles le juxtaposant (Figure 22). Il n'y a pas d'information d'échelle et l'unité de longueur est arbitraire. Le modèle STL est déjà une approximation du modèle théorique de départ [16].

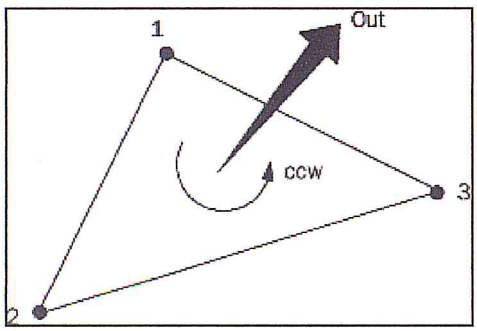


Figure 20 : Orientation de la normale

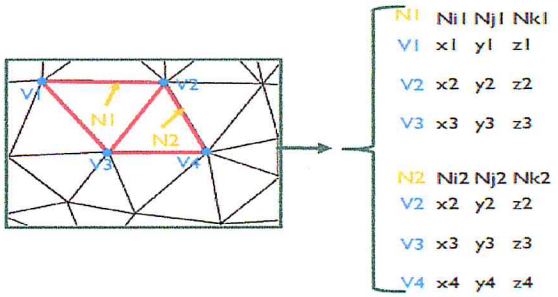


Figure 21 : Paramètres d'un triangle.

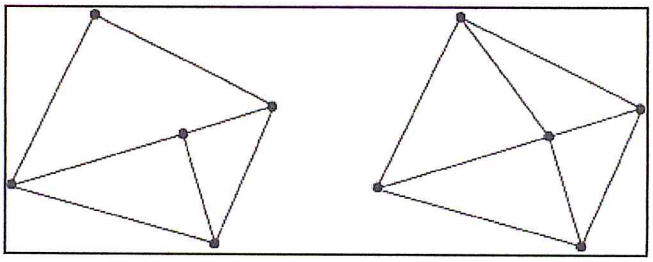


Figure 22 : Règle des sommets.

Conclusion :

L'étude effectuée dans cette partie portait sur les différentes activités inhérentes au processus de rétro-conception de produits manufacturés. Dans un premier temps, les différentes méthodes de modélisation de surfaces en CAO ont été explicitées, s'en est suivie une présentation des phases de la rétro-conception où l'accent a été mis sur le processus de reconstruction d'objets 3D à partir de nuages de points et la description des formats d'échange de données et en particulier le format STL ont été introduits.

Le prochain chapitre sera consacré à la présentation des architectures parallèles, objets de notre étude.

Chapitre II

Architectures et programmation parallèle

Introduction :

Les architectures parallèles sont devenues le paradigme dominant pour tous les ordinateurs depuis les années 2000. C'est un concept qui regroupe plusieurs processeurs qui coopèrent et communiquent entre eux pour résoudre des problèmes dans un court délai. De même, le calcul parallèle est l'utilisation simultanée de plusieurs ressources de calcul pour résoudre un problème donné (Figure 23). Ainsi:

a) Le problème est donc exécuté en utilisant plusieurs processeurs.

b) Le problème est divisé en parties distinctes qui peuvent être résolues en même temps.

c) Chaque partie est subdivisée en une série d'instructions.

d) Les Instructions de chaque partie sont exécutées simultanément sur des processeurs différents.

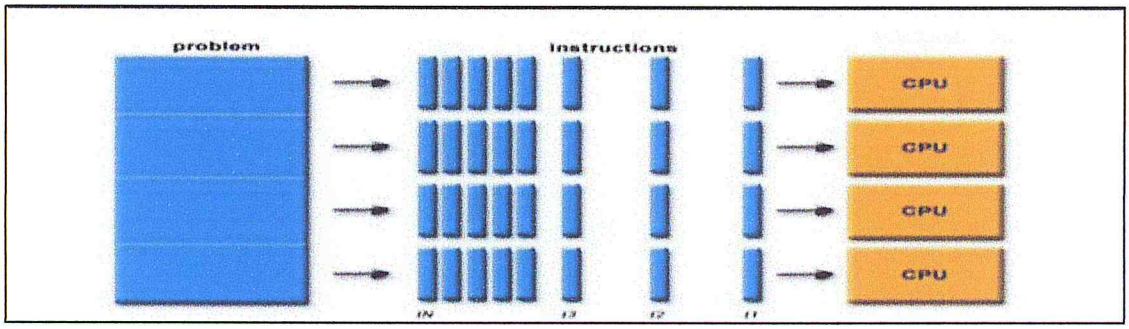


Figure 23: Principe du calcul parallèle [17].

I. Sources de parallélisme

Trois principales sources de parallélisme sont considérées [18].

I.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 24).

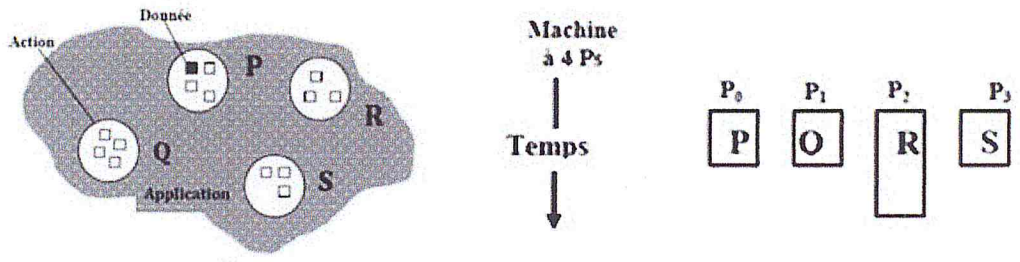


Figure 24 : Parallélisme de contrôle.

I.2 Parallélisme de données

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.

I.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 25). 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).

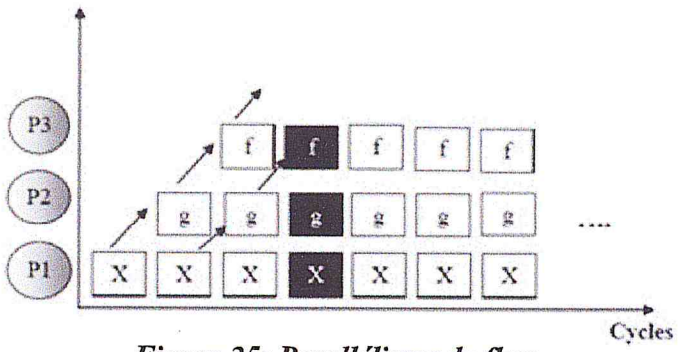


Figure 25: Parallélisme de flux.

II. Classification des architectures parallèles :

Il existe plusieurs classifications des architectures parallèles. Les plus utilisées sont :

II.1 Classification de Michael J. FLYNN (1972)

Flynn propose une classification suivant deux paramètres à savoir le flux d'instructions et le flux de données.

- **Flux d'instructions :** séquence d'instructions exécutées par la machine.
- **Flux de données :** séquence des données appelées par le flux d'instructions.

Tableau 1 : Classification de FLYNN.

	Flux d'instructions	Flux d'instructions
Flux de données	SISD	MISD (pipeline)
Flux de données	SIMD	MIMD

S : Single (une seule), M : Multiple (plusieurs)

D : Data (données), I : Instruction (instruction)

➢ **SISD :** « Single Instruction Stream, Single Data Stream », une seule instruction avec une seule donnée en entrée. Cette catégorie correspond aux machines séquentielles conventionnelles (modèle de Von Neumann), pour lesquelles chaque opération s'exécute sur une donnée à la fois. L'unité de contrôle (UC) recevant son flot d'instructions (FI) de l'unité mémoire (UM) envoie les instructions à l'unité de traitement (UT) qui effectue ces opérations sur le flot de données (FD) provenant de l'unité mémoire (Figure 26) [19].

➢ **MISD :** « Multiple Instruction Stream, Single Data Stream », plusieurs instructions pour une seule donnée. Cette catégorie regroupe les machines spécialisées de type « systolique » dont les processeurs, arrangés selon une topologie fixe, sont fortement synchronisés (Figure 27) [19].

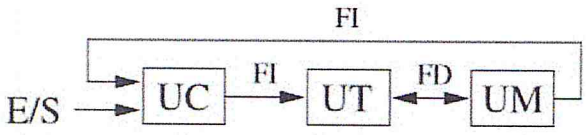


Figure 26: Architecture SISD [19].

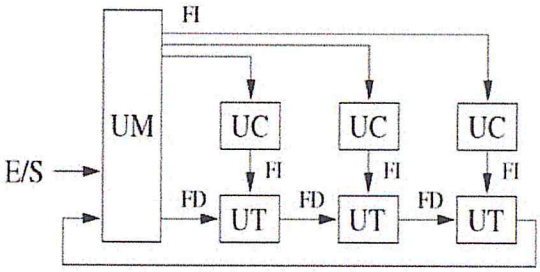


Figure 27: Architecture de MISD [20].

➢ **SIMD :** « Single Instruction Stream, Multiple Data Stream », une seule instruction pour plusieurs données. Plusieurs données sont traitées en même temps par une seule instruction. Elle est utilisée dans les gros ordinateurs vectoriels. Une machine SIMD exécute à tout instant une seule instruction, mais qui agit en parallèle sur plusieurs données. Dans ce cas, on parle en général de parallélisme de données (Figure 28) [17].

➢ **MIMD :** « Multiple Instruction Stream, Multiple Data Stream », plusieurs instructions sur plusieurs données. Exécution d'une instruction différente sur chaque processeur pour des données différentes. Il désigne les machines multiprocesseurs où chaque processeur exécute son code de manière asynchrone et indépendante. Pour assurer la cohérence des données, il est souvent nécessaire de synchroniser les processeurs entre eux ; les techniques de synchronisation dépendent de l'organisation de la mémoire qui peut être une mémoire partagée ou bien une mémoire distribuée (Figure 29) [20].

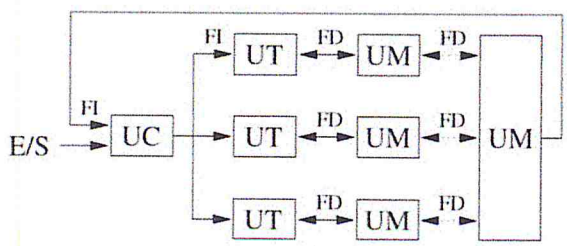


Figure 28 : Architecture SIMD [19].

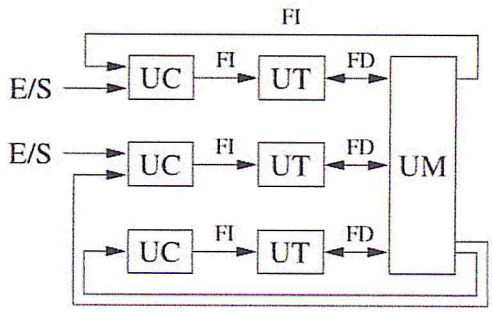


Figure 29 : Architecture MIMD [19].

II.2 Classification de Raina

Raina propose une classification des machines MIMD selon deux critères, à savoir l'organisation de l'espace d'adressage et le type d'accès mémoire mise en œuvre (Figure 33).

- **Organisation de l'espace d'adressage [19] :**

SASM : « Single Address Space, Shared Memory », une seule mémoire partagée entre tous les processeurs. Le temps mis pour accéder à la mémoire est identique pour tous les processeurs. L'accès simultané à la même zone mémoire par plusieurs processeurs n'est pas possible; un seul accède à la fois.

DADM : « Distributed Address Space, Distributed Memory », appelée aussi les architectures distribuées dont chaque processeur a sa propre mémoire et aucune mémoire partagée. L'échange de données entre processeurs s'exécute nécessairement par passage de messages au moyen d'un réseau de communication.

SADM : « Single Address Space, Distributed Memory », mémoire distribuée, avec espace d'adressage global, autorisant éventuellement l'accès aux données situées sur d'autres processeurs. Le temps d'accès pour les divers bancs de mémoires, devient par conséquent différent.

- **Types d'accès mémoire mis en œuvre :**

NORMA « No Remote Memory Access », ce type n'a pas de moyens d'accès aux données distantes, ce qui nécessite le passage de messages.

UMA « Uniform Memory Access » : accès symétrique à la mémoire avec un coût identique pour tous les processeurs (Figure 30).

NUMA « Non-Uniform Memory Access », dans ce cas, les performances d'accès à la mémoire dépendent de la localisation des données.

CC-NUMA : « Cache-Coherent NUMA », type d'architecture NUMA intégrant la mémoire caches et implémentation de protocole cohérence de cache (Figure 31).

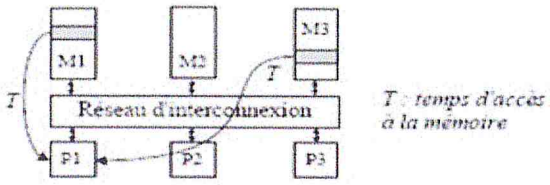


Figure 30: Accès mémoire UMA [17]

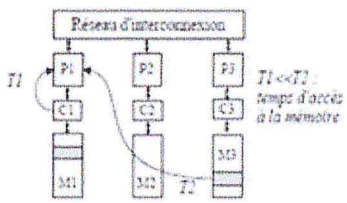


Figure 31: Accès mémoire CC-NUMA [17]

OSMA : « Operating System Memory Access », les accès aux données distantes sont gérés par le système d'exploitation, qui traite les défauts de page au niveau logiciel et gère les requêtes d'envoi/copie de pages distantes.

COMA : « Cache Only Memory Access » : les mémoires locales se comportent comme des caches, de telle sorte qu'une donnée n'a pas de processeur propriétaire ni d'emplacement déterminé en mémoire (Figure 32).

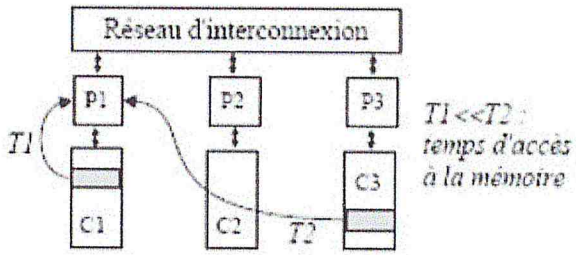


Figure 32: Accès mémoire COMA [17].

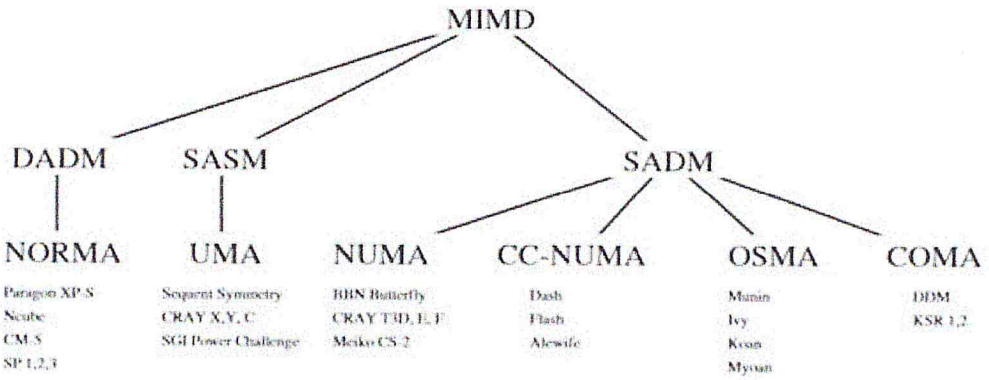


Figure 33: Classification MIMD de Rania [19].

III. Classification selon la mémoire

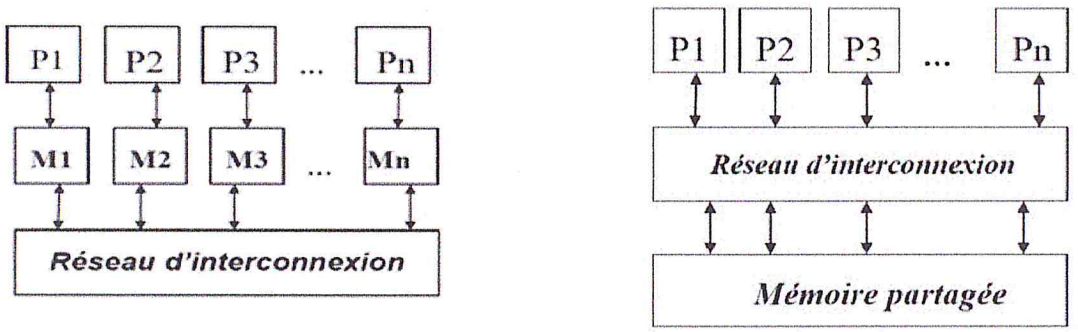
III.1 Machines parallèles à mémoire partagée

Ces machines sont caractérisées par une horloge indépendante pour chaque processeur. Mais une seule mémoire partagée entre ces processeurs, où tous les processeurs lisent et écrivent dans le même espace d'adressage mémoire, ce qui permet de réaliser un parallélisme de données et de contrôles. Le programmeur n'a pas besoin de spécifier l'emplacement des données, il définit seulement la partie du programme que doit exécuter chaque processeur en plus de la gestion de la synchronisation [18]. Ce type d'architecture possède plusieurs avantages dont la simplicité, le passage à l'échelle et la parallélisation de haut niveau. Son inconvénient majeur est lié principalement à la limite de la bande passante du réseau d'interconnexion.

III.2 Machines parallèles à mémoire distribuée

Dans ce type de machine, chaque processeur possède sa propre mémoire locale, où il exécute des instructions identiques ou non aux autres processeurs. Les différents nœuds définis par l'ensemble mémoires plus les processeurs sont reliés entre eux par un réseau d'interconnexion. Le parallélisme est implémenté par échange de messages (Figure 34) [18]. L'avantage de ce type de machine est l'augmentation du nombre de processeurs avec des moyens simples, tels que les clusters. Seulement, elles présentent plus de difficulté dans la programmation.

Différents types de réseaux d'interconnexion ont été mis en place pour relier les différents nœuds des machines parallèles. On peut citer principalement l'anneau, grille torique, fat tree, hypercube et d'autres architectures hybrides (Figure 35).



a. Mémoire partagée. *b. Mémoire distribuée.*

Figure 34: Machine parallèle [18].

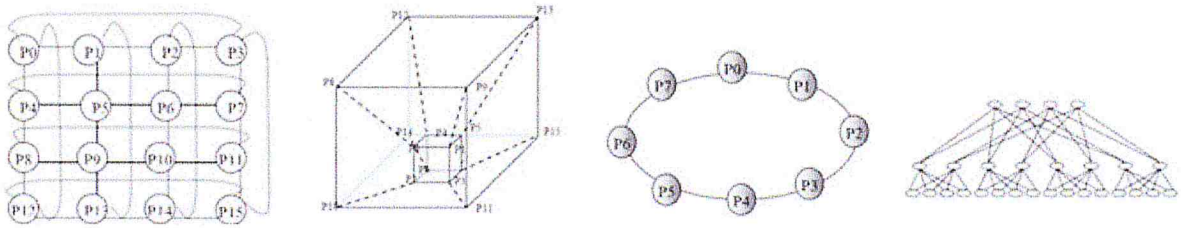


Figure 35: Différentes architectures de réseau d'interconnexion [18].

IV. Les mesures de performances des architectures parallèles

IV.1 Le temps d'exécution :

Le temps d'exécution, c'est le temps écoulé entre le début du calcul parallèle et l'exécution de la dernière instruction par le plus long processus. Théoriquement $Te = Nc*Tc$ où :

- **Te :** temps d'exécution.
- **Nc :** Nombre de cycles de programme.
- **Tc :** Temps de cycle d'un processeur.

IV.2 Accélération (Speed Up) :

Le gain en performance peut s'exprimer en termes d'accélération. Suivant la loi d'Amdahl, l'accélération est le rapport du temps d'exécution séquentielle sur le temps d'exécution parallèle. Plus la partie parallélisable d'un programme est grande, plus l'accélération est meilleure.

$$S_{(p)} = \frac{T_s}{T_p}$$ avec :

S (p) : Accélération, **Ts** : Temps séquentiel, **Tp :** Temps parallèle.

La loi d'Amdahl s'exprime aussi avec (Figure 36) [17] :

$$S_p = \frac{1}{(1-\alpha)+P}$$ avec

P : Nombre de processeurs.

α: Fraction de la partie parallélisable du programme.

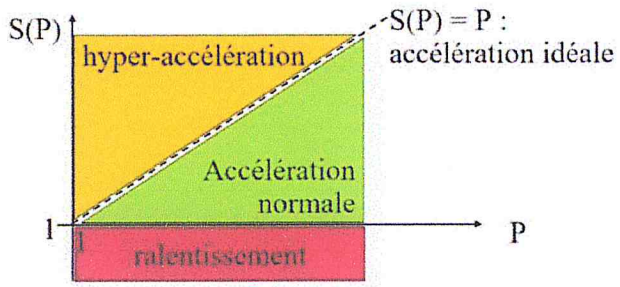


Figure 36 : Analyse du résultat de l'accélération [17].

a) **S(p) < 1**, On ralentit ! mauvaise parallélisation.

b) **1< S(p) < p**, Normal.

c) **S(p) > p**, Hyper-accélération analyser & justifier.

d) **S(p)=p,** Accélération idéale.

IV.3 Efficacité :

L'efficacité est une valeur métrique qui représente le taux d'utilisation des ressources, ou la fraction de l'accélération idéale dans l'architecture parallèle. L'efficacité est inférieure ou égale à 1.

$$E = \frac{S_{(p)}}{P}$$ Avec :

S(p) : Accélération (Speed Up)

P : Nombre de processeurs

V. Architecture parallèle actuelle

Dans cette section, nous allons présenter quelques architectures parallèles actuelles.

V.1 Processeur graphique

Un processeur graphique, ou GPU (Graphics Processing Unit), est un circuit intégré présent sur une carte graphique mais pouvant aussi être intégré sur une carte-mère ou dans un CPU et assurant les fonctions de calcul de l'affichage. Un processeur graphique a généralement une structure hautement parallèle qui le rend efficace pour une large palette de tâches graphiques tels que le rendu 3D, Direct3D, OpenGL, la gestion de la mémoire vidéo, le traitement du signal vidéo, la décompression Mpeg ...etc. [20]. Peu d'entreprises conçoivent de tels processeurs. Les plus connues sont NVIDIA, AMD et Intel (Figure 37).

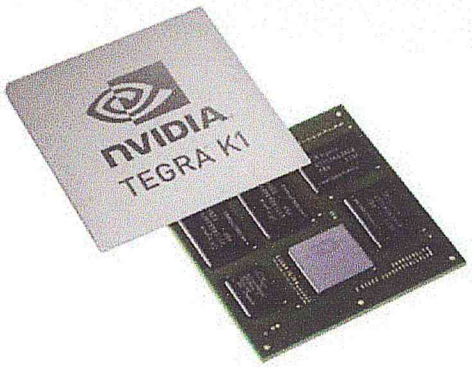


Figure 37: Processeur graphique NVIDIA avec un CPU [21].

V.2 Cloud

Cloud est l'exploitation de la puissance de calcul ou de stockage de serveurs informatiques distants par l'intermédiaire d'un réseau, généralement l'internet. Ces serveurs sont loués à la demande, le plus souvent par tranche d'utilisation selon des critères techniques (puissance, bande passante, etc.) mais également au forfait. Il se caractérise par sa grande souplesse. Selon le niveau de compétence de l'utilisateur client, il est possible de gérer soi-même son serveur ou de se contenter d'utiliser des applicatifs distants [22]. Les grandes entreprises du secteur informatique font la promotion du Cloud Computing, qui constitue un nouveau paradigme des systèmes informatiques, jusque-là constitués de serveurs situés au sein même de l'entreprise.

V.2.1 Modèles de service de Cloud

Il existe trois modèles de service de Cloud [23] :

- **SasS** (Software as a Service), Logiciel en tant que service.
- **PaaS** (Platform as a Service), Plateforme en tant que service.
- **Iaas** (infrastructure as a Service), Infrastructure en tant que service.

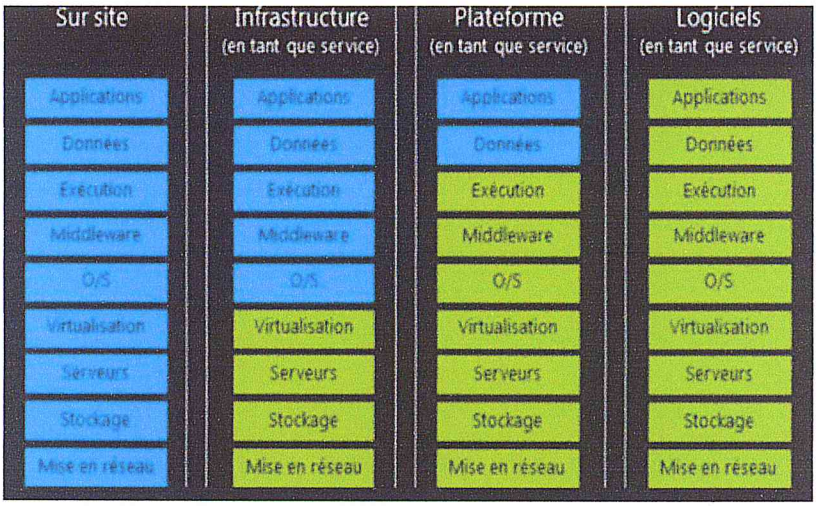


Figure 38: Différents modèles de Cloud.

V.3 Cluster

Le cluster « grappe en français » est une architecture de groupe d'ordinateurs, utilisée pour former de gros serveurs, chaque machine est un nœud du cluster (Figure 39). L'ensemble est considéré comme une seule et unique machine. Toutes les machines du cluster travaillent ensemble sur des tâches communes en s'échangeant des données (l'échange est restreint dans une zone géographique étroite), ce type de travail est appelé calcul parallèle [24].

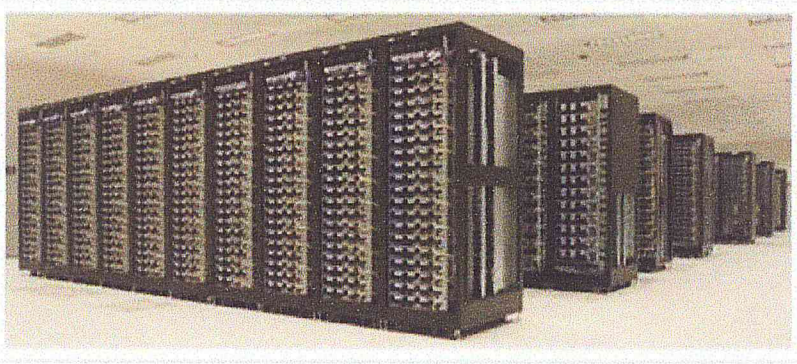


Figure 39: Cluster de machines chez Yahoo [25].

V.3.1 Concept d'un cluster informatique

Les concepts d'un cluster informatique sont les suivants [26] : partage de l'exécution d'une application entre plusieurs machines du cluster, la redondance, la haute disponibilité, l'accélération des temps d'exécution de façon significative, le problème de charge réseau, l'échange de messages, le temps de latence, la communication par échange de message, l'évolutivité, la délégation des calculs et la Sécurité.

V.4 Grille informatique :

La grille informatique est une architecture réseau qui utilise le calcul distribué en regroupant des ressources géographiquement distribuées, et se caractérise par un transfert important de données, des logiciels de coordination et d'ordonnancement (Figure 40). La grille informatique consiste également à mettre en commun la puissance de toutes les machines d'une même entreprise ou d'un réseau plus vaste, et de redistribuer la puissance de calcul en fonction des besoins spécifiques de chaque client.

Figure 40:Architecture d'une grille informatique [27].

V.4.1 Type de grille :

On distingue trois types de grille [28] :

- **Grille d'information :** partage de l'information comme le site web.
- **Grille de stockage :** partage des données, musique, vidéo, applications à succès, données scientifiques ...etc.
- **Grille de calcul :** agrégation de puissance de calcul, dont on distingue trois sous-types :

  - ✓ Supercalculateur Virtuel.
  - ✓ Internet Computing.
  - ✓ MetaComputing.

V.4.2 Caractéristique d'une grille informatique

Les grilles informatiques sont caractérisées par [29] :

- Existence de plusieurs domaines administratifs.
- Hétérogénéité des ressources.
- Passage à l'échelle.
- Nature dynamique des ressources.

V.5 Programmation séquentielle et parallèle

V.5.1 Programmation séquentielle

La programmation séquentielle est un type de programmation existant actuellement et le plus ancien. Il s'agit d'écrire un programme en utilisant un ou des algorithmes séquentiels. Le nombre d'algorithmes séquentiels dans un programme dépend de la fonctionnalité et de la complexité de la conception du programme. Un algorithme séquentiel est une suite d'instructions code machine exécutables par le processeur une par une ; une instruction n'est exécutable que si l'exécution de la précédente est terminée. Il est entendu que l'exécution de l'algorithme (programme) dépendra de l'ordre des instructions de ce dernier. Le programme séquentiel est exécuté par un seul et unique processeur.

Certains algorithmes séquentiels ont un temps d'exécution très long, comme les algorithmes de calcul des factoriels ou bien les algorithmes de recherche de graphes sur des grandes données à complexité temporelle et exponentielle ou bien factorielle ...etc. Ces algorithmes nécessitent le recours aux architectures parallèles pour paralléliser les algorithmes pour les rendre plus efficaces en termes de temps d'exécution et de performances de coût et d'exploitation des ressources.

V.5.2 Programmation parallèle

Le parallélisme est omniprésent dans les ordinateurs d'aujourd'hui. Au niveau microscopique, les processeurs multiplient les unités arithmétiques pipelinées sur un même circuit intégré. Au niveau macroscopique, on interconnecte les stations de travail en grappes pour construire des supercalculateurs à peu de frais. Dans les deux cas, l'algorithmique parallèle permet de comprendre et de maîtriser les concepts fondamentaux à mettre en œuvre pour l'utilisation de plates-formes distribuées. Elle emprunte beaucoup à l'algorithmique classique dans sa problématique (conception, analyse, étude de complexité), mais s'enrichit d'une nouvelle dimension avec l'exploitation simultanée de plusieurs ressources.

On écrit des programmes parallèles qui seront exécutés sur plusieurs processeurs répartis géographiquement dans le monde, connectés via un réseau de télécommunication dans une architecture parallèle à mémoire distribuée où la communication entre les processus, se fait par échange de messages. Aussi, les algorithmes parallèles peuvent s'exécuter sur une seule machine avec un processeur multi-cœurs dans une architecture parallèle à mémoire partagée où le parallélisme se fait par des threads s'exécutant chacun, sur de différents cœurs.

La programmation parallèle est basée sur les algorithmes parallèles, dont le développeur doit considérer plusieurs facteurs tels que la partie du programme qui peut être traitée en parallèle, la manière de distribuer les données, les dépendances des données, la répartition de charges entre les processeurs et les synchronisations entre les processeurs.

Il y a essentiellement deux méthodes pour concevoir un algorithme parallèle, l'une consiste à détecter et à exploiter le parallélisme à l'intérieur d'un algorithme séquentiel déjà existant. L'autre consistant à inventer un nouvel algorithme dédié au problème donné.

Conclusion

Dans ce chapitre, l'architecture parallèle a été présentée et les différentes sources de parallélisme ont été introduites tels que le parallélisme de contrôle ou celui de données ou encore celui de flux. Diverses méthodes de classification des architectures parallèles ont été présentées et les modèles de service récents tels que les Cloud ou encore les Clusters ont été introduits. Ce chapitre a été terminé par une présentation de la programmation séquentielle et de la programmation parallèle. Le prochain chapitre, sera consacré à la présentation de la conception de notre application.

Chapitre III

Conception de l'application

Introduction

La reconstruction d'objets est une discipline relativement jeune qui est introduite dans un large spectre de domaines tels que le génie mécanique, la vision artificielle, la robotique, la médecine, le génie civil, l'environnement, …etc. 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. Le but recherché est la génération en sortie d'un modèle numérique tels que STL, B-Spline, NURBS, …etc. approchant au mieux la forme de l'objet physique avec un temps de calcul optimal. Pour cela, la triangulation de Delaunay combinée avec le paradigme « Diviser pour Régner » a été appliquée.

I. Problématique :

La reconstruction des modèles 3D avec une bonne précision est une tâche ardue et consommatrice de temps. Par conséquent, la solution à ce problème n'est pas triviale. Plusieurs méthodes ont été développées pour tenter d'y apporter une solution. Pour résoudre ce problème, le nuage de points est approximé par des éléments géométriques simples tels que les triangles dans le plan et les tétraèdres dans l'espace, …etc. Les nuages de points sont souvent très volumineux pour être traités avec un seul processeur. Cette situation augmente les temps de traitement et par conséquent, les coûts.

II. Architecture globale de l'application

L'architecture générale de l'application logicielle à développer est composée de neuf (09) étapes (Figure 41). Elle prend en entrée le nuage de points pour générer des cellules à partir du brut englobant le nuage de points. Par la suite, chaque point du nuage est affecté à la cellule correspondante selon une technique préétablie. Chaque ensemble de cellules est affecté à un processeur qui prend en charge la triangulation de Delaunay et la définition de la zone affectée pour fusionner les différentes cellules selon un ordre bien défini. En sortie, génération du modèle STL.

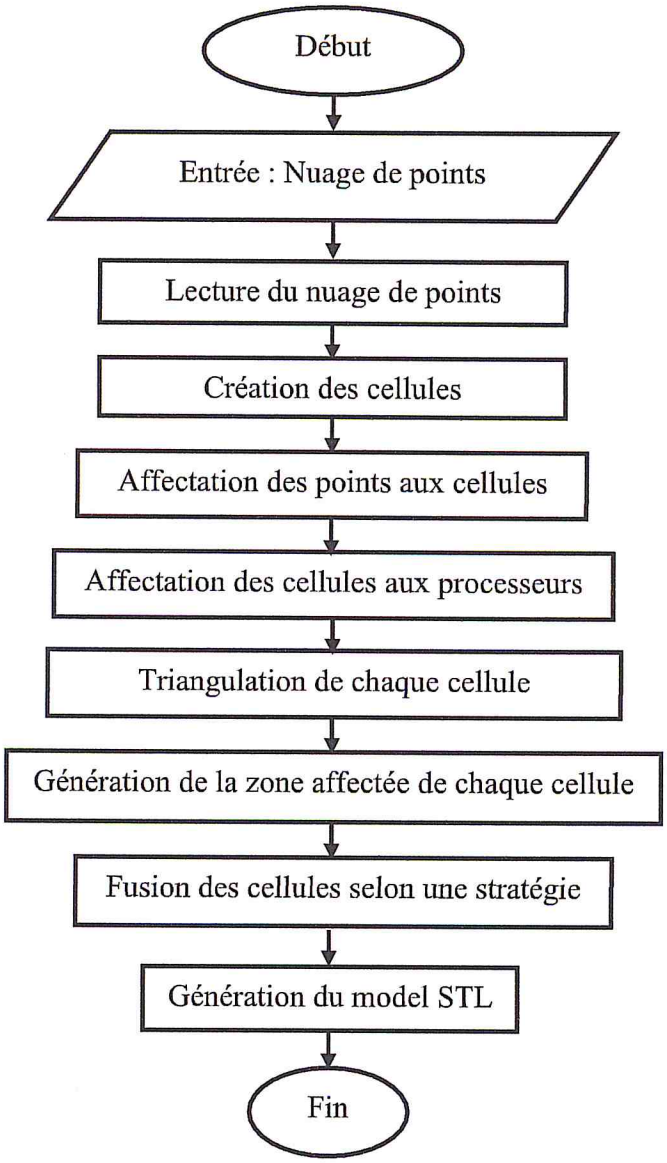


Figure 41: Architecture globale de l'application.

III. Description de l'organigramme global de la reconstruction

L'organigramme global de la reconstruction est divisé en huit tâches. Chaque tâche est composée de plusieurs sous-tâches.

III.1. Lectures du nuage de points :

Le traitement du nuage de points dépend de sa conformité. Cette conformité se traduit par une représentation unique et correcte de l'objet digitalisé. Afin de faciliter la triangulation, les coordonnées (x, y, z) de tous les points du nuage non structuré sont récupérées et stockées après élimination des point redondants. L'algorithme de la vérification de la conformité du nuage de points est donné par la Figure 42.

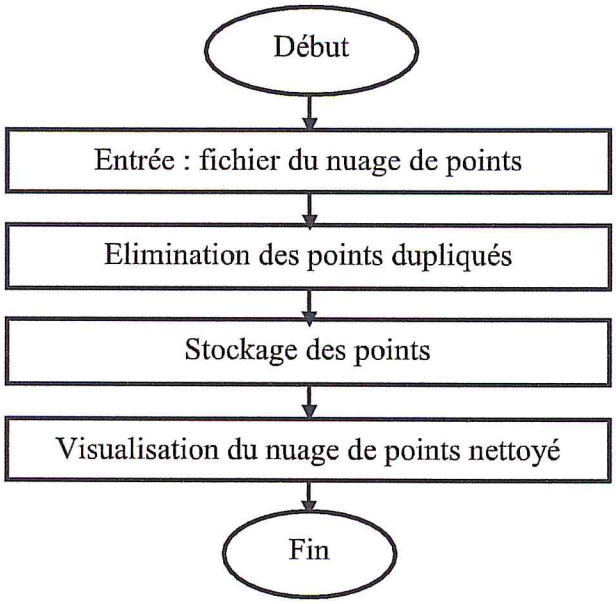


Figure 42: Organigramme de lecture du nuage de points.

III.2. Subdivision du nuage de points

Après lecture du fichier et élimination des points redondants, celui-ci est restructuré pour optimiser le temps de reconstruction des surfaces sans modifier la forme du modèle. Pour atteindre cet objectif, une solution en deux étapes est proposée :

- Première étape : consiste en la création des cellules.
- Deuxième étape : consiste en l'affectation de chaque point à la cellule appropriée.

La création des cellules consiste à subdiviser le brut englobant le nuage de points (Figure 43) en cellules de même taille (Figure 44). Le processus de subdivision est illustré ci-dessous :

1. Calcul des limites du brut (Figure 43).
   - Définition des coordonnées du brut englobant $(X_{min}, X_{max}, Y_{min}, Y_{max}, Z_{min}, Z_{max})$.
   - Calcul des dimensions du brut : longueur, longueur et hauteur.
2. Création des cellules en fixant soit le nombre de cellules suivant les trois axes X, Y et Z, soit les dimensions des cellules.

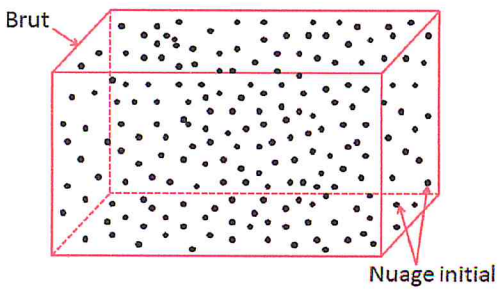


Figure 43 : Calcul du brut.

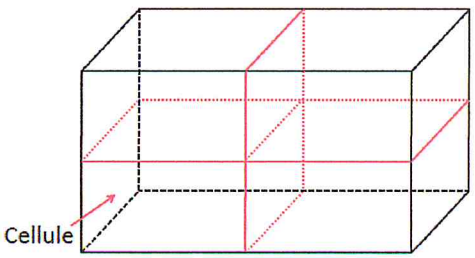


Figure 44 : Création des cellules.

III.3. Affectation des points aux cellules

L'affectation des points aux cellules se fait par indexation des cellules selon les trois directions (Figure 46). Cette façon de faire permet d'affecter directement chaque point à la cellule appropriée sans avoir à tester ces trois coordonnées. La Figure 45 illustre le processus appliqué à un nuage de points dans le plan.

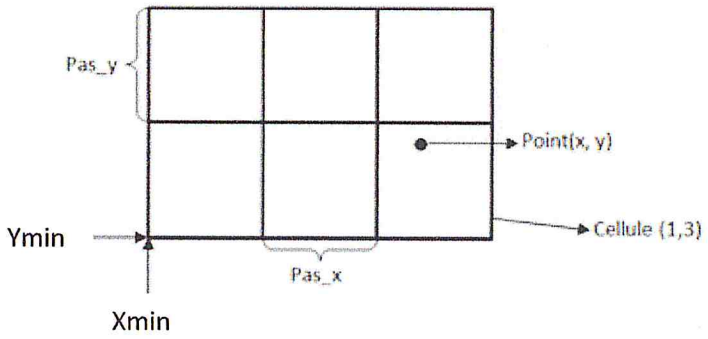


Figure 45: Indexation des cellules appliquée à un point dans le plan.

Les indices de la cellule contenant un point quelconque de coordonnées x, y et z sont donnés par :

$$I = \frac{x - Xmin}{Pas\_x} \qquad J = \frac{y - Ymin}{Pas\_y} \qquad K = \frac{z - Zmin}{Pas\_z}$$

Avec :

I, J et K : indices de la cellule.

Xmin, Ymin et Zmin : limites minimales du brut englobant le nuage de points.

Pas_x, Pas_y et Pas_z : dimensions de la cellule.

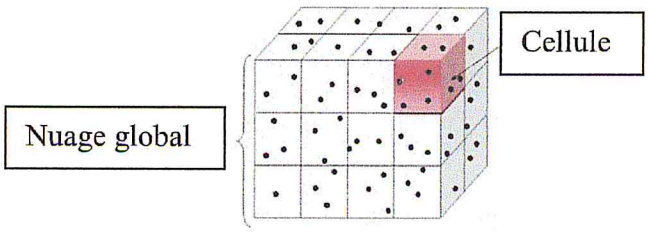


Figure 46 : Affectation des points aux cellules correspondantes.

III.4. Triangulation de Delaunay

La triangulation de Delaunay est une étape clé du processus de reconstruction de surfaces à partir d'un nuage de points. Cependant, notre travail porte sur le parallélisme du processus de reconstruction d'objet pour optimiser le temps de traitement. Ainsi, le processus de la triangulation de Delaunay sera présenté globalement.

La triangulation de Delaunay en 3D consiste à créer des tétraèdres à partir du nuage de points non structuré pour générer à la fin un fichier STL représentant la peau extérieure de l'objet digitalisé. Le processus de la triangulation est illustré par la Figure 47.

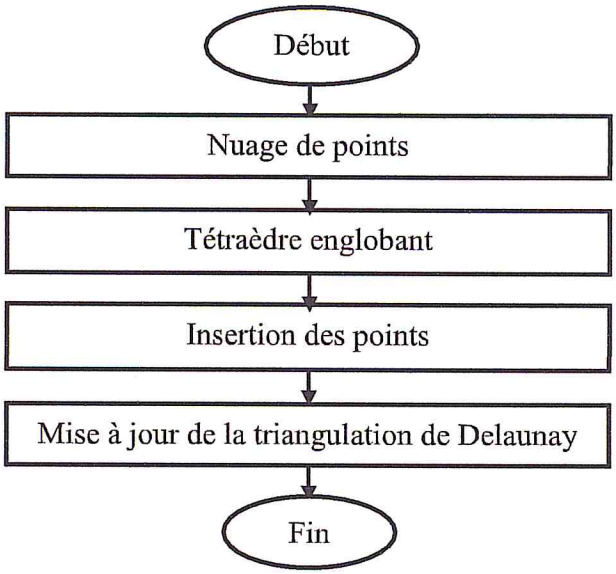


Figure 47: Organigramme de la triangulation de Delaunay.

III.5. Triangulation de Delaunay parallèle

Elle consiste à créer un fichier exécutable « .exe » qui ne fait que la triangulation de Delaunay. Cet exécutable est appelé autant de fois que nécessaire en fonction du nombre de cellules pour récupérer les nuages de points des cellules qui seront stockés sous forme de fichiers textes. Cet appel se fait d'une manière automatique et non visible pour l'utilisateur (fonctionnement en arrière-plan). Le processus est illustré par un exemple de deux cellules suivant chaque axe, ce qui donne huit (08) cellules (Figure 48).

a. Nuage de points de chaque cellule. *b. Triangulation de chaque cellule.*

Figure48: Fichiers textes utilisés par le fichier exécutable.

La fonction ***spawnl*** permet de faire appel à l'exécutable de la triangulation. Cette fonction qui charge et exécute un nouveau processus enfant, elle peut attendre la fin de l'exécution de ce dernier avant de lancer l'exécution d'un autre processus ou exécute les processus simultanément [30].

Le prototype de cette fonction est le suivant :

```
int spawnl (int mode, char *path, char *arg0, ...);
```

- char *path : pour spécifier le chemin de l'exécutable. Dans notre cas, c'est la Triangulation_Delaunay_3D.exe.
- char *arg : pour spécifier les arguments nécessaires qui sont les fichiers textes.
- int mode : il y a plusieurs modes d'exécution (P_OVERLAY, P_WAIT, P_NOWAIT, P_DETACH). Le mode qui nous intéresse est P_NOWAIT. Ce mode nous permet de continuer à exécuter le processus d'appel simultanément avec un nouveau processus sans attendre la fin du processus en cours. Donc, c'est une triangulation en parallèle des nuages de points des cellules.

Si par exemple huit (08) cellules sont créées, alors huit (08) fichiers textes seront créés et par la suite l'exécutable « Triangulation_Delaunay_3D.exe» sera appelé huit (08) fois simultanément et en parallèle pour faire la triangulation de Delaunay [30].

III.6. Zone affectée

La zone affectée est la région de la triangulation de chaque sous nuage de points qui est susceptible d'être modifiée lors de la fusion des triangulations partielles. Ces régions représentent les portions de zones mitoyennes entre les différents sous nuages dans les trois directions X, Y et Z (Figure 50).

Afin d'éviter le traitement de tous les tétraèdres des cellules, seuls les tétraèdres dont la sphère circonscrite entre en contact avec les limites des cellules voisines, selon les trois directions X, Y et Z, seront considérés lors du traitement. Pour chaque cellule, six (06) zones affectées doivent être déterminées ; deux selon chaque axe. Les zones déterminées sont : zone affectée droite, zone affectée gauche, zone affectée haut, zone affectée bas, zone affectée avant et zone affectée arrière [31]. La Figure 51 résume le processus de création des zones affectées.

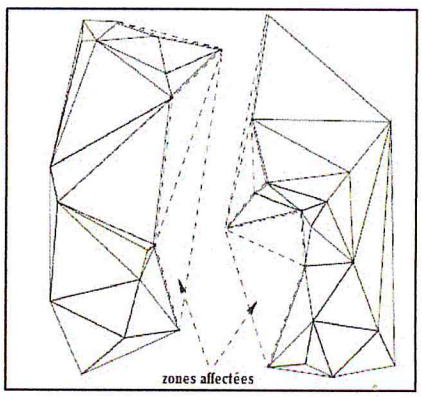


Figure 49: Exemple de deux zones affectées.

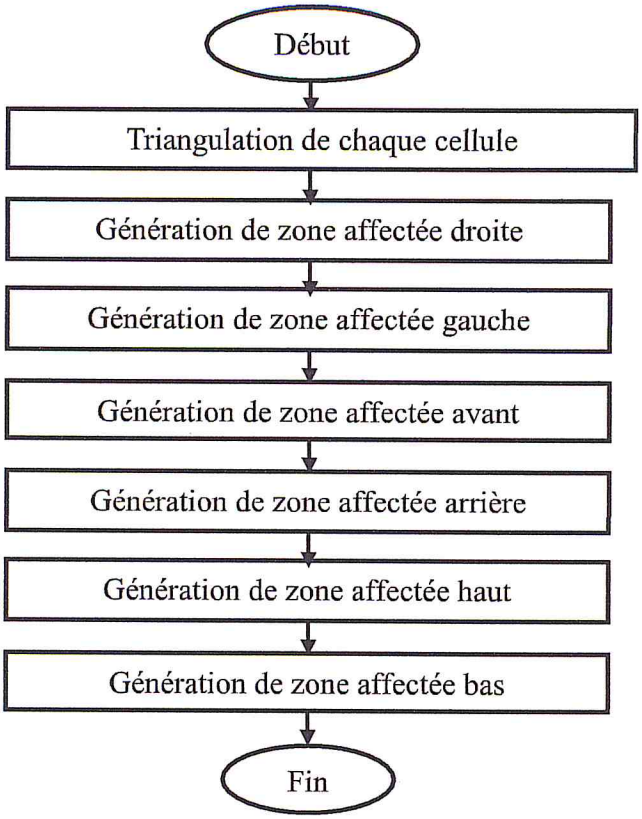
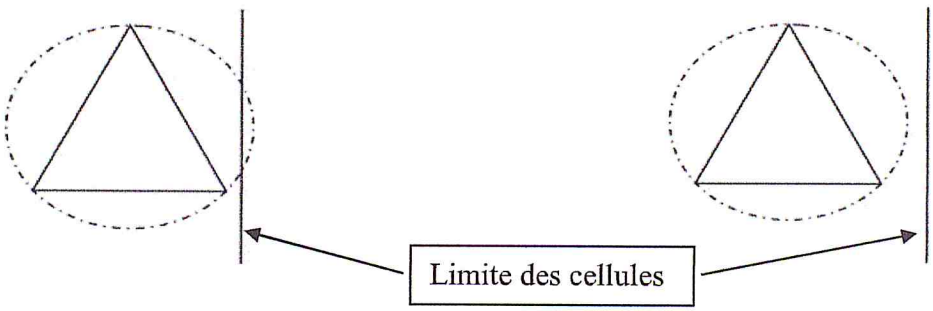


Figure 50: Organigramme de la génération des zones affectées.

III.6.1. Test limite-cercle circonscrit

Ce test est utilisé pour la génération des différentes zones affectées décrites précédemment. Le but est de savoir si la sphère circonscrite d'un tétraèdre entre en contact avec un plan représentant la limite de la cellule contenant le tétraèdre. Plusieurs cas peuvent apparaitre lors du test. Pour mieux comprendre la complexité de la définition des zones affectées, nous appliquerons le processus à un exemple dans le plan où les tétraèdres seront remplacés par des triangles, les sphères par des cercles et le plan par une droite (Figure 52).



a. Triangle affecté. *b. Triangle non affecté.*

Figue 51 : Test du cercle circonscrit.

Parfois, il y a des cas où le test de la sphère circonscrite ne donne pas de résultats exacts du fait que les limites des cellules soient des plans et sont donc infinis. On aura alors des cas mathématiquement correctes mais pas cohérents comme illustré par la Figure 53.

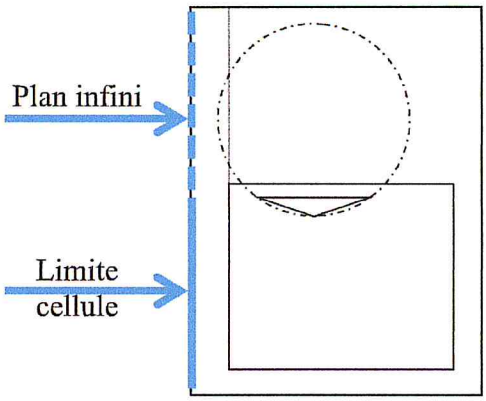


Figure 52 : Exemple de tétraèdre à ne pas considérer.

Il existe d'autres cas qui s'avèrent corrects par rapport au résultat souhaité (Figure 54).

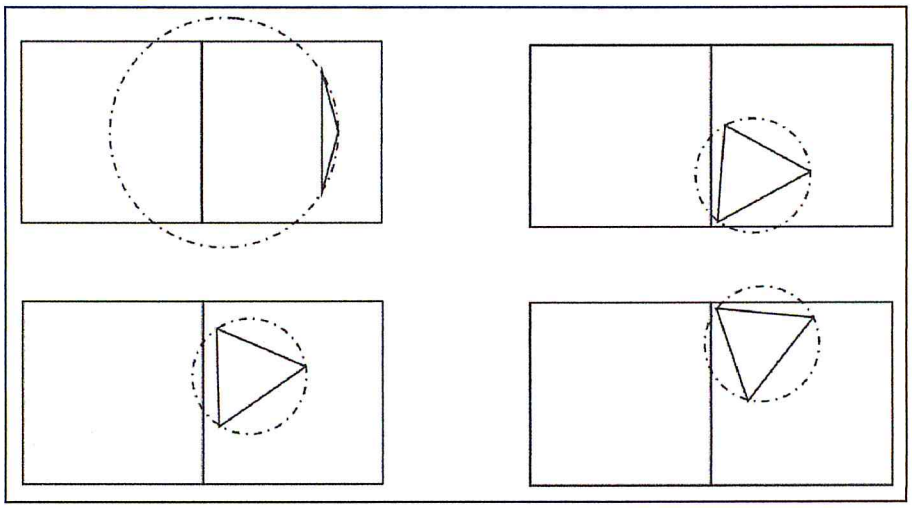


Figure 53 : Quelques exemples de cas rencontrés.

Pour avoir des résultats plus cohérents par rapport à l'approche adoptée et éviter les faux cas, il est recommandé d'appliquer les tests du cercle circonscrit plutôt que d'appliquer la sphère circonscrite. L'utilisation du test du cercle circonscrit est justifiée par le fait que l'intersection de la sphère d'un tétraèdre avec un plan (la limite de la cellule) est un cercle dont le rayon est facilement calculable (Figure 55).

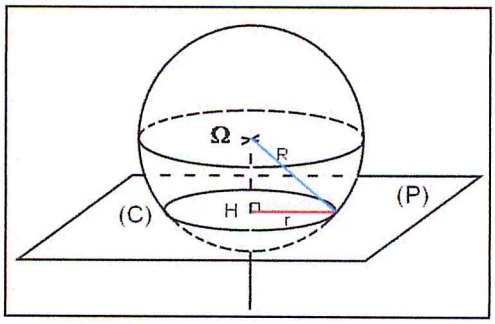
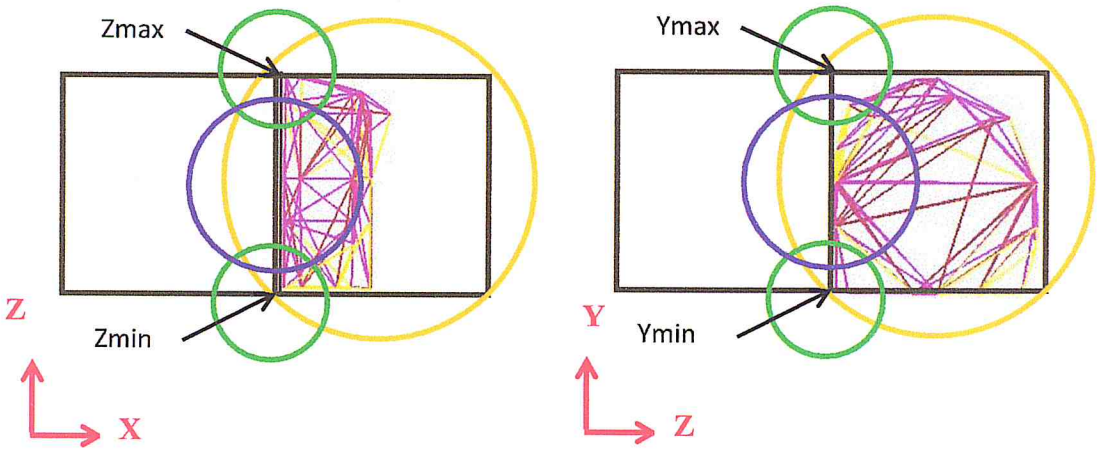


Figure 54 : Intersection d'une sphère avec un plan.

III.6.2. Zone affectée droite (gauche) :

La génération de la zone affectée droite (gauche) suit le processus suivant (Figure 56) :

- Vérification de l'intersection entre la sphère circonscrite et le plan Xmax (Xmin).
- Si oui, alors calcul du rayon de cercle suite au calcul de l'intersection entre la sphère et le plan.
- Vérification du rayon s'il est compris entre Zmin et Zmax.
- Vérification du rayon s'il est compris entre Ymin et Ymax.



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

Figure 55: Génération de la zone affectée droite.

III.6.3. Zone affectée avant (arrière) :

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

- Vérification de l'intersection entre la sphère circonscrite et le plan Ymax (Ymin).
- Si oui, alors calculer le rayon du cercle d'intersection entre la sphère et plan.
- Vérification du rayon s'il est compris entre Xmin et Xmax.
- Vérification du rayon s'il est compris entre Zmin et Zmax.

III.6.4. Zone affectée haut (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 Zmax (Zmin).
- Si oui, alors calculer le rayon du cercle d'intersection entre la sphère et plan.
- Vérifier si le rayon est compris entre Ymin et Ymax.
- Vérifier si le rayon est compris entre Xmin et Xmax.

III.7. Fusion

La fusion est le cœur du processus de reconstruction du nuage de points subdivisé. De la stratégie de fusion des cellules adoptée, dépend la forme de l'objet reconstitué. Dans notre approche, la stratégie adoptée consiste à concaténer les sous nuages deux à deux en utilisant les zones affectées des cellules adjacentes. La reconnaissance des cellules adjacentes se fait par comparaison des indices des cellules selon les trois directions. La fusion se fait d'abord selon la direction X, puis selon la direction Y et enfin selon la direction Z (Figure 57 et Figure 58). Elle consiste à générer un premier tétraèdre qui servira de support pour la génération des autres tétraèdres [31].

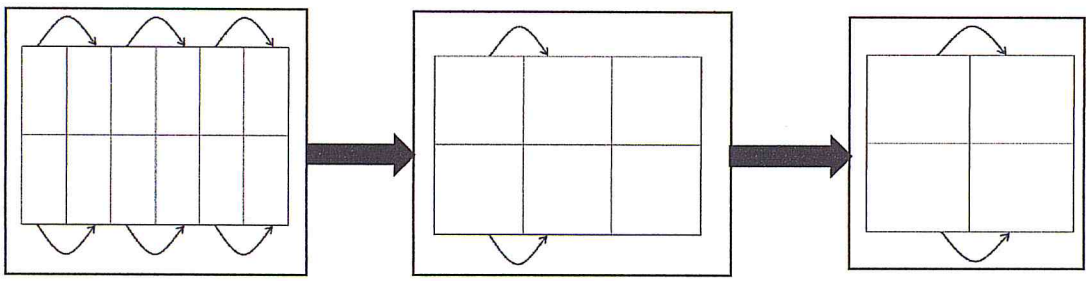


Figure 56 : Fusion suivant la première direction.

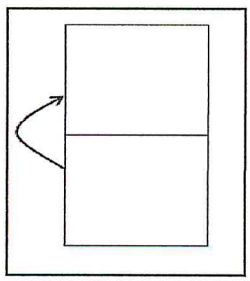


Figure 57 : Fusion suivant la deuxième direction.

III.7.1. Création du premier tétraèdre :

La création du premier tétraèdre est basée essentiellement sur deux concepts [31] :

- Identification des deux points les plus proches de chacune des deux zones affectées adjacentes selon la direction de fusion (Figure 60).
- Test de la sphère circonscrite vide.

La création du premier tétraèdre est décrite par l'organigramme suivant (Figure 59) :

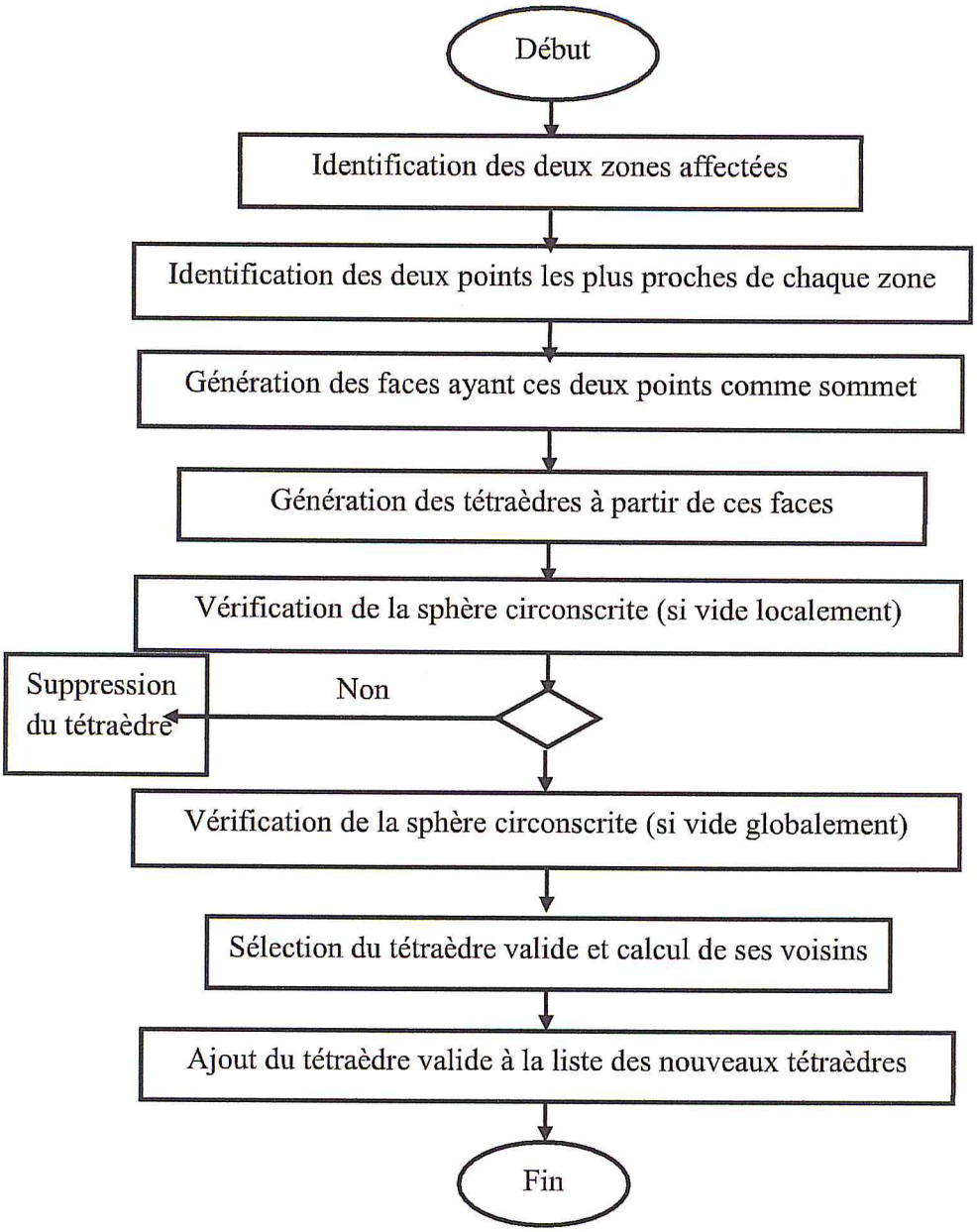


Figure 58 : Organigramme de création du premier tétraèdre.

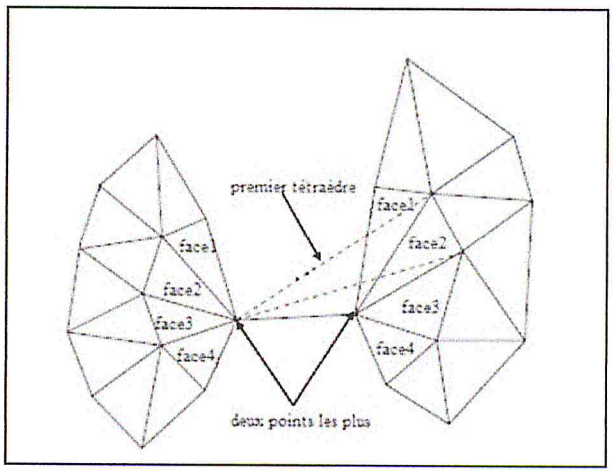


Figure 59 : Sélection des deux points et création du premier tétraèdre.

III.7.2. Tétraèdre générique

La création de tétraèdre générique est basée sur le test de la sphère circonscrite en deux étapes : un test local et un test global [31].

- 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 gauche et vice versa.
- Le 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 par les Figure 61 et Figure 62.

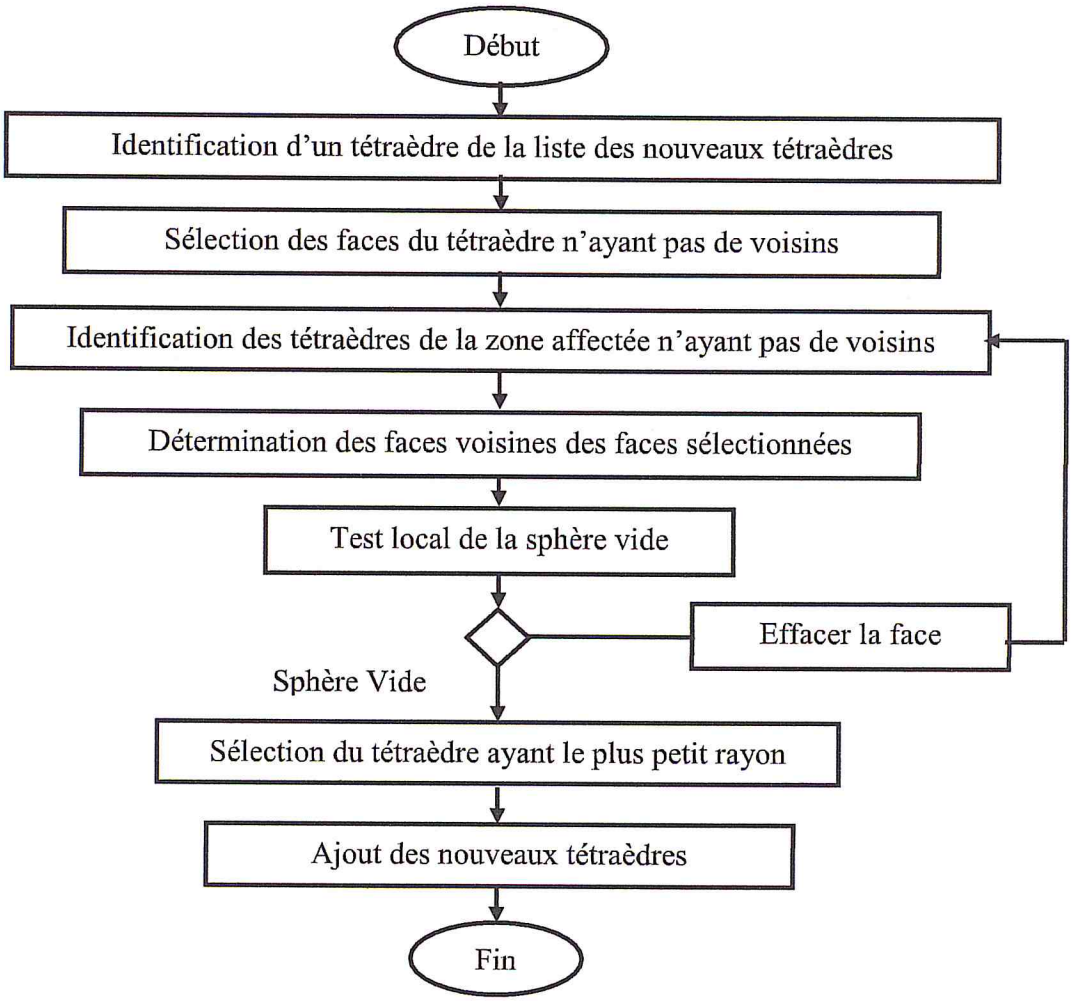


Figure 60 : Organigramme de création d'un tétraèdre générique.

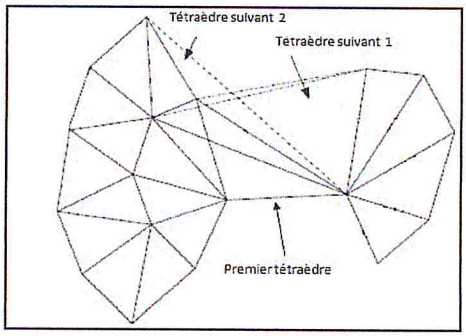


Figure 61 : Création du deuxième tétraèdre.

La création du tétraèdre dit 'tétraèdre générique' peut paraître simple à réaliser. En réalité, elle demande beaucoup de réflexion et impose le respect de plusieurs conditions, notamment lors des mises à jour après la suppression des faces (triangles) ou bien des arêtes.

La création du tétraèdre générique se fait en se basant sur le premier tétraèdre ce qui permet de recenser deux sortes de tétraèdres.

Le premier cas est décrit par la Figure 63.

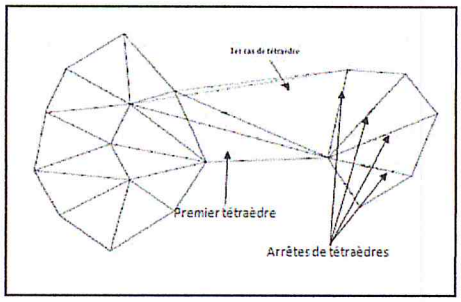


Figure 62 : Premier cas du tétraèdre générique.

Ce premier cas du tétraèdre générique, consiste à faire le test de la sphère vide entre une des faces du premier tétraèdre qui n'a pas de voisins et les arêtes qui sont reliées à l'un de ses sommets. Si la sphère est vide, alors le tétraèdre est créé. Dans le cas contraire, l'arête est supprimée. A noter que la suppression d'une arête peut engendrer la suppression d'un ou plusieurs tétraèdres de la zone affectée, ce qui rend la mise à jour de la zone affectée qui respecte les critères de Delaunay très délicate et difficile à réaliser [31].

Le deuxième cas est décrit par la Figure 64.

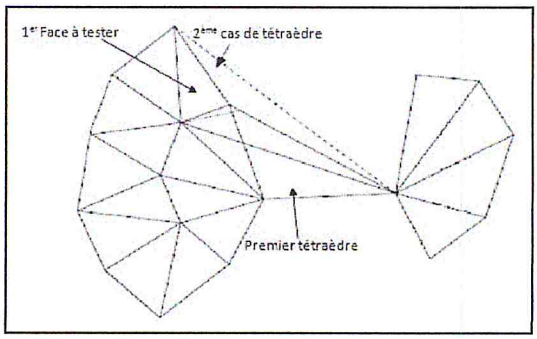


Figure 63 : Deuxième cas du tétraèdre générique.

Ce cas est aussi important que le cas précédent, sauf qu'ici, le test de la sphère vide se fait en considérant l'une des faces du premier tétraèdre et l'une des faces de la zone affectée qui est reliée avec le tétraèdre cible. Si la sphère n'est pas vide, alors la face est supprimée ainsi que le tétraèdre associé.

Dans le cas d'une existence simultanée de deux tétraèdres génériques appartenant aux deux cotés différents des deux zones affectées adjacentes et qui vérifient les conditions citées précédemment, alors le tétraèdre ayant la sphère avec le plus petit rayon est sélectionné. Ce dernier devient la base pour la création d'un autre tétraèdre générique [31].

III.8. Parallélisme

Le parallélisme consiste à mettre en œuvre des architectures d'électronique numérique permettant de traiter des informations de manière simultanée, ainsi que les algorithmes conçus pour celles-ci. Ces techniques ont pour but de réaliser le plus grand nombre d'opérations en un minimum de temps. Dans certaines applications, la recherche de l'optimalité absolue, consiste à paralléliser uniquement les étapes gourmandes en terme de temps et de mémoire. Dans notre travail, seules les étapes de la **subdivision** et la **triangulation de Delaunay** sont parallélisées. Cette décision de paralléliser ces deux étapes est inspirée d'un exemple concret de fabrication de jus de pomme illustré par les Figure 65 et Figure 66:

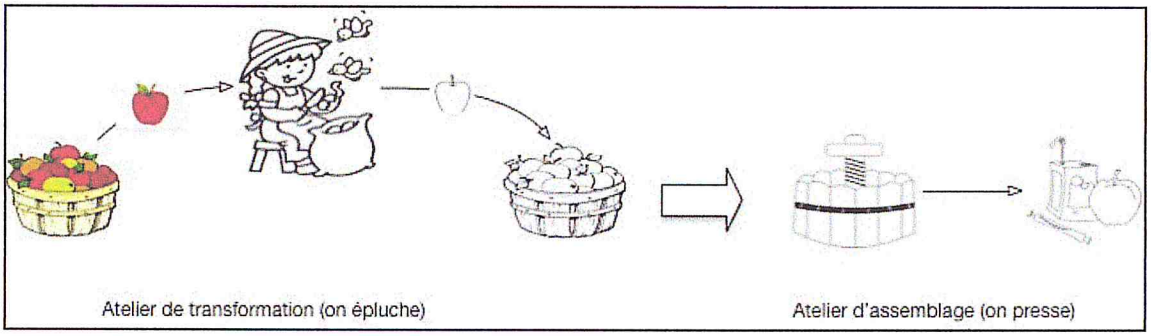


Figure 64 : Fabrication de jus de pomme d'une manière séquentielle.

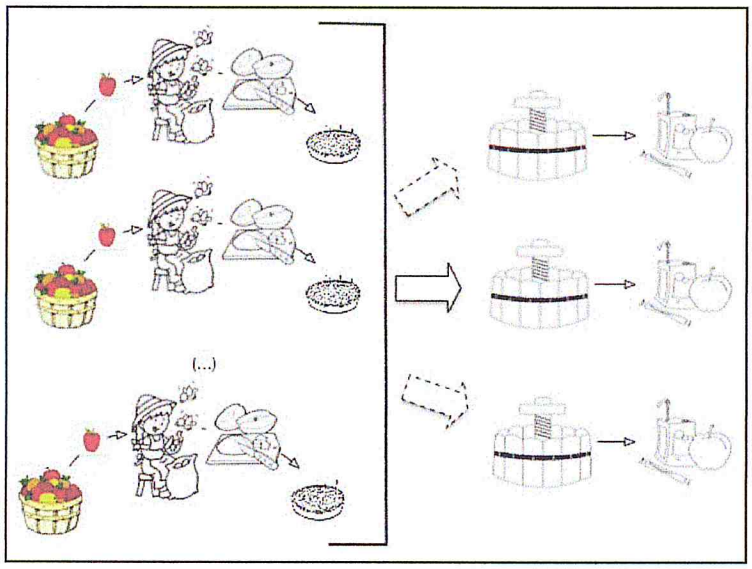
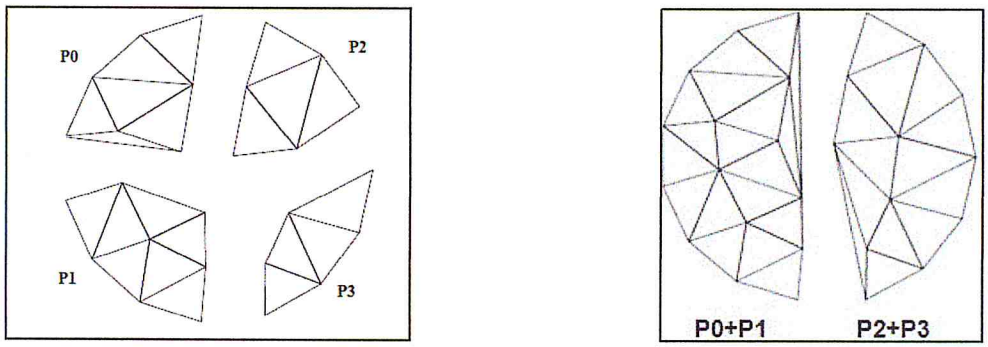


Figure 65 : Fabrication de jus de pomme d'une manière parallèle.

La projection de cet exemple sur notre application donne la solution suivante (Figure 67 et Figure 68) :

- Ouvrières représentant les processus.
- Epluchage et pressage des pommes représentant la subdivision et la triangulation de Delaunay.



a. Triangulation de Delaunay parallélisée. *b. Fusion deux à deux des triangulations.*

Figure 66 : Triangulation de Delaunay.

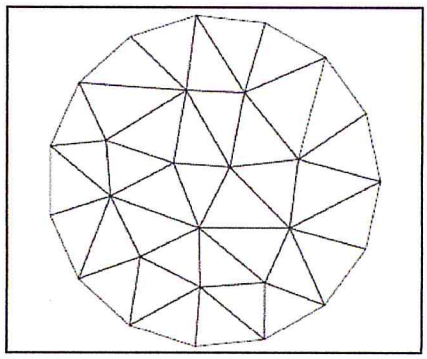


Figure 67 : Résultat final de la fusion.

III.9. Génération du modèle STL

Le modèle STL est une représentation surfacique de l'objet, il permet la définition de la peau extérieure du modèle. La triangulation réalisée est une triangulation volumique qui représente tout l'objet en profondeur. Pour cela, nous allons suivre quelques étapes présentées dans l'organigramme suivant (Figure 69).

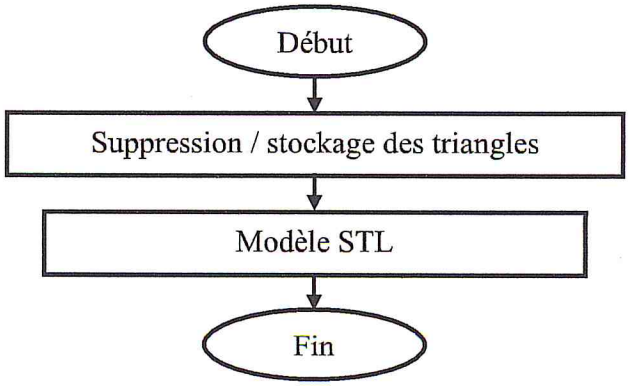


Figure 68: Génération du modèle STL.

Afin d'obtenir la surface désirée de l'objet ou le modèle STL, nous allons garder uniquement les faces qui n'ont pas de voisins. Pour cela, nous allons suivre les étapes suivantes :

1. Suppression de toutes les faces des tétraèdres qui ont au moins un voisin.
2. Stockage des faces triangulaires qui n'ont pas de voisins.
3. Visualisation des faces triangulaires (modèle STL).

Conclusion

Dans ce chapitre, nous avons présenté notre conception de l'application parallélisée et nous avons détaillé le processus de reconstruction d'objets à partir d'un nuage de points en mettant en évidence les organigrammes de chaque étape de reconstruction (lecture du nuage de points, subdivision, triangulation de Delaunay, génération des zones affectées, fusion). Les activités parallélisées ont été identifiées et présentées en se basant sur un exemple concret.

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

Chapitre IV

Implémentation et validation

Introduction

Dans le but d'optimiser le temps de traitement du processus de reconstruction d'objets en 3D à partir d'un nuage de points non structuré, une application logicielle graphique et interactive intégrée à l'environnement de production d'objets de formes complexes développé par l'équipe CFAO du CDTA, a été développée. La mise en œuvre de ce travail s'achève par une présentation de l'application conçue, de l'environnement de développement et des outils utilisés ainsi qu'une une présentation de l'interface utilisateur et une évaluation du système à travers un exemple de validation.

Ce chapitre a pour but de présenter les informations manipulées à travers les sessions d'utilisation appliquées à un exemple de validation. Certes, un exemple ne peut à lui seul présenter les performances de l'application développée, mais il peut donner une idée générale sur son intérêt surtout sur la complexité de la distribution des tâches dans un système distribué dédié à la reconstruction d'objets en 3D et les différentes techniques utilisées pour le respect de la contrainte de Delaunay.

Dans la première partie, le langage utilisé et les différentes fenêtres de notre application sont présentés. Quand à la deuxième partie, les scénarios les plus généraux appliqués sur une pièce donnée illustrés par des captures d'écrans, commençant par un nuage de points jusqu'à la génération du modèle STL, constituent la partie validation.

L'implémentation a été réalisée en utilisant un langage évolué doté d'une interface graphique qui facilite à l'utilisateur la manipulation et l'exécution des différentes tâches.

I. Implémentation

La réalisation des solutions proposées a été faite par l'utilisation d'Embarcadero C++ Builder 10 Seattle et la plateforme Windows 10. Ce travail est un ensemble de fenêtres Windows qui permettent à l'utilisateur d'interagir avec l'application et de visualiser tous les objets manipulés via la bibliothèque graphique OpenGL.

II. Langage de programmation utilisé

Pour permettre l'intégration de 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++.

Crée en 1983 par Bjarne Stroustrup le langage C++ est actuellement l'un des langages les plus utilisés dans le monde, aussi bien pour les applications scientifiques que pour le

développement de logiciels. En tant qu'héritier du langage C, le C++ est d'une grande efficacité. De plus, il a des fonctionnalités puissantes, comme par exemple la notion de classe, qui permet d'appliquer les techniques de la programmation orientée objet.

II.1. C++ Builder

C++ Builder est un RAD (Rapid Application Development) développé par Borland mais racheté par Embacadero Technologies en 2009 pour la création de programmes en C++ visant plusieurs plates-formes en utilisant la technologie appelée «Drag and Drop » qui facilite et accélère la programmation.

II.2. C++ Builder 10 Seattle

Sorti fin août 2015, il reprend tous les avantages des versions précédentes avec un compilateur plus rapide et plus stable et d'une mémoire plus grande pour les grands projets. Le support multi-moniteur est étendu et l'Inspecteur d'objets est amélioré. C ++ Builder permet aux développeurs de fournir des applications jusqu'à 5 fois plus rapides sur plusieurs plates-formes de bureau, de mobile, de base de données et de base de données incluant Windows 10 32 bits et 64 bits .

II.3. OpenGL

OpenGL (Open Graphics Library) est une bibliothèque graphique très complète qui permet aux programmeurs de développer des applications graphiques en 2D et en 3D assez facilement.

II.4. Matériel utilisé

Le matériel utilisé dans la phase de développement et de validation de l'application se résume en un PC portable HP sous Windows 10, Intel® Core™ I5-4200 CPU @1.60 GHz 2.30 GHz 4Go de RAM.

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

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

III. Présentation des fenêtres

L'application principale est composée d'un environnement qui englobe dans ses onglets toutes les applications du projet.

La fenêtre principale de notre projet est composée de quatre (04) onglets. Chaque onglet est dédié à une/des phase(s) du processus de génération de la triangulation Delaunay en parallèle :

- Onglet 1 : Lecture et subdivision du nuage de points.
- Onglet 2 : Triangulation de Delaunay.
- Onglet 3 : Zones affectées.
- Onglet 4 : Fusion des triangulations.

Figure 69 : Fenêtre principale de l'application développée.

III.1. Lecture et subdivision du nuage de points

Le premier onglet « Lecture et subdivision du nuage de points » (Figure 71) est divisé en deux parties :

La première partie permet la lecture du fichier contenant le nuage de points par simple clic sur le bouton « Lecture du nuage de points ». Par la suite, le nuage de points est filtré par suppression des points doubles.

Après lecture du fichier et filtrage du nuage de points, les paramètres du brut (X_min, X_max, Y_min, Y_max, Z_min, Z_max, Hauteur, Largeur, Longueur) sont calculés et affichés ainsi que le nombre de points du nuage.

La seconde partie est consacrée à la subdivision du nuage de points filtré en cellules. Cette partie permet à l'utilisateur de choisir le nombre de cellules en spécifiant soit le nombre de cellules, soit les dimensions des cellules suivant les trois directions X, Y et Z.

Après la phase de subdivision, les points du nuage de points sont affectés aux cellules correspondantes.

III.2. Triangulation de Delaunay

Cet onglet est dédié à la triangulation de Delaunay (Figure 72). L'utilisateur a la possibilité de choisir entre plusieurs modes de triangulation :

- Triangulation de Delaunay d'une cellule donnée.
- Triangulation de toutes les cellules en séquentiel.
- Triangulation de toutes les cellules en parallèle.

Une fois la triangulation terminée, l'utilisateur peut visualiser la triangulation complète d'une/des cellule(s) en filaire et en rendu ou bien visualiser chaque tétraèdre d'une cellule donnée en filaire ou en rendu ainsi que la sphère circonscrite et les quatre voisins de ce dernier.

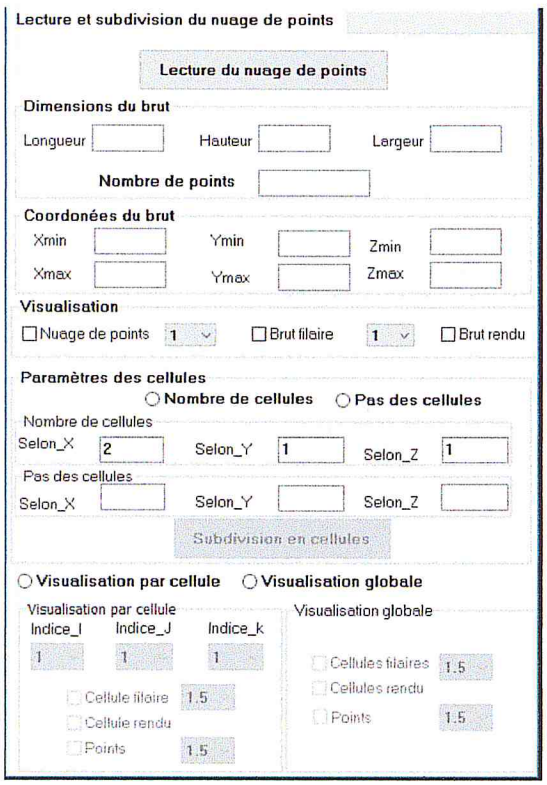


Figure 70 : Lecture et subdivision du nuage.

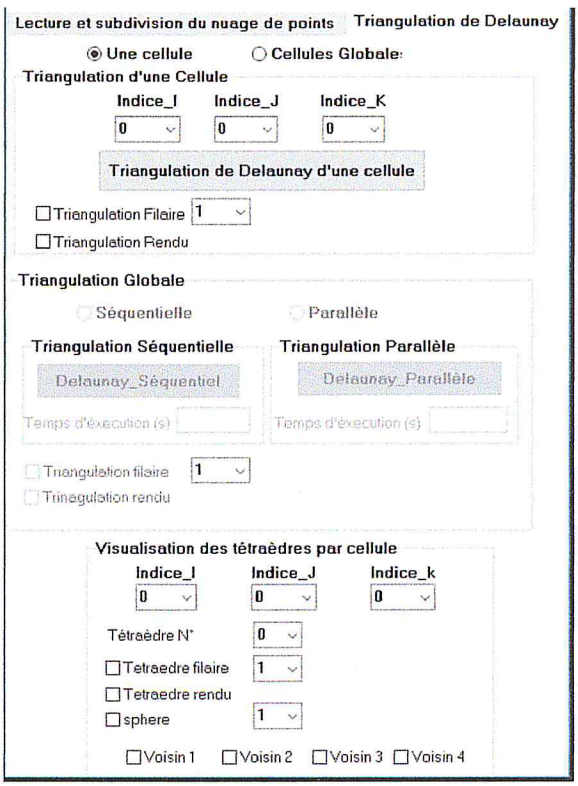


Figure 71 : Triangulation de Delaunay.

III.3. Zones affectées

Cet onglet est divisé en deux parties (Figure 73). La première partie est réservée à la génération de l'enveloppe convexe de chaque cellule. L'enveloppe convexe est constituée des faces extérieures de la triangulation (peau externe de l'objet).

La deuxième partie est dédiée à la génération des zones affectées de chaque cellule, ainsi qu'à la visualisation de chaque tétraèdre affecté par cellule. Les zones affectées sont générées selon le nombre de cellule(s) voisine(s) que possède une cellule donnée.

III.4. Fusion des triangulations

Cet onglet est dédié à la phase de fusion entre les triangulations (Figure 74). Le premier bouton permet de déterminer les deux points les plus proches.

Le bouton détermination des tétraèdres candidats sert à déterminer les deux tétraèdres candidats pour la fusion. L'utilisateur peut visualiser les faces à partir desquelles ont été construits les tétraèdres candidats ainsi que le tétraèdre candidat de chaque cellule.

Le bouton déterminer premier tétraèdre de fusion permet de générer le premier tétraèdre dans le processus de fusion entre des deux triangulations.

Enfin, le bouton fusion des triangulations permet de générer la triangulation finale après la fusion des deux cellules.

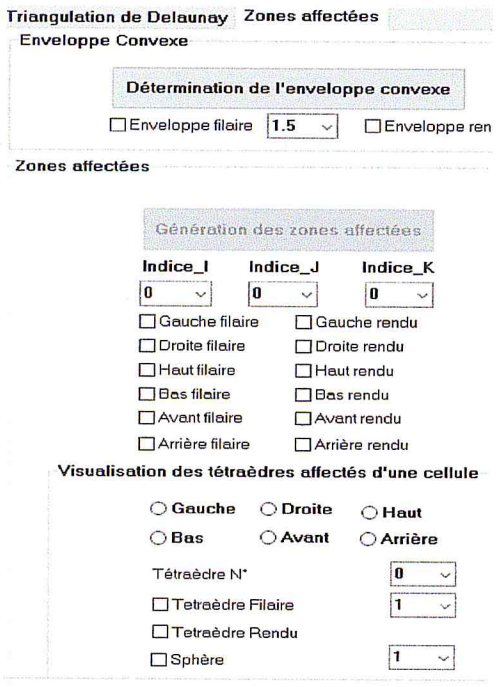


Figure 72 : Zones affectées.

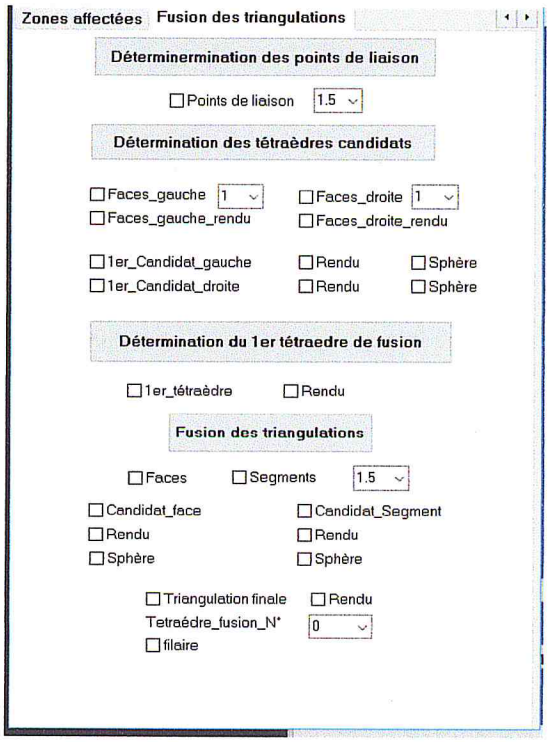


Figure 73 : Fusion des triangulations.

IV. Validation

La validation de notre travail est effectuée sur la pièce présentée par la Figure 75. Le traitement est effectué sur le nuage de points issus de la digitalisation de l'objet jusqu'à la génération d'un modèle STL, l'objet cible est présenté par la suite. Le nuage de points généré suite à la digitalisation de la pièce est représenté par la Figure 76.

Figure 74 : Pièce de test.

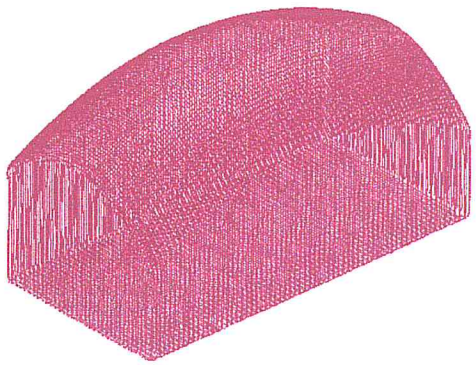


Figure 75 : Nuage de points de la pièce.

IV.1. Lecture et subdivisons du nuage de points

IV.1.1. Lecture du nuage de points

La première étape consiste à lire le nuage de points, à calculer les limites de la pièce brute ainsi que ses dimensions (longueur, largeur et hauteur). La simplification du nuage de points permet de générer un nouveau nuage de points de faible densité composé de 809 points (Figure 77). Le nuage de points avec différents attributs de visualisation est représenté par la Figure 78.

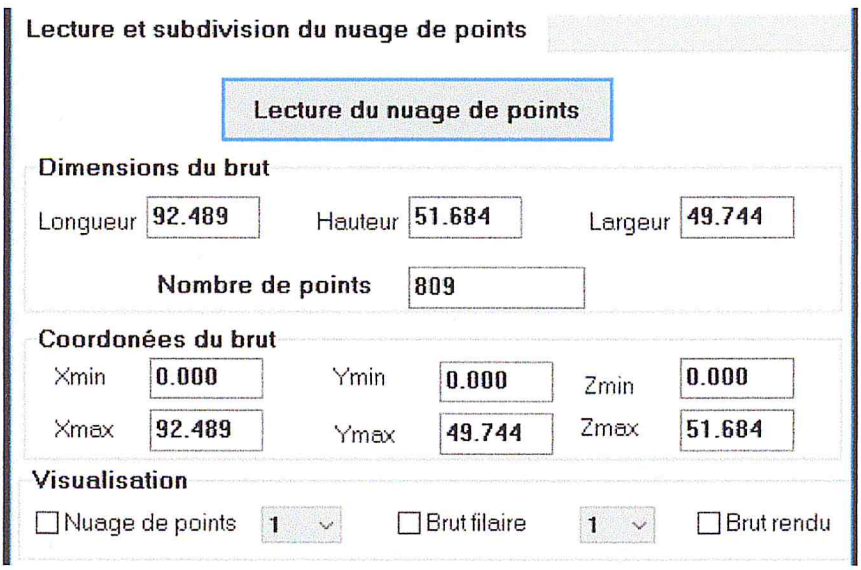


Figure 76 : Lecture du nuage de points.

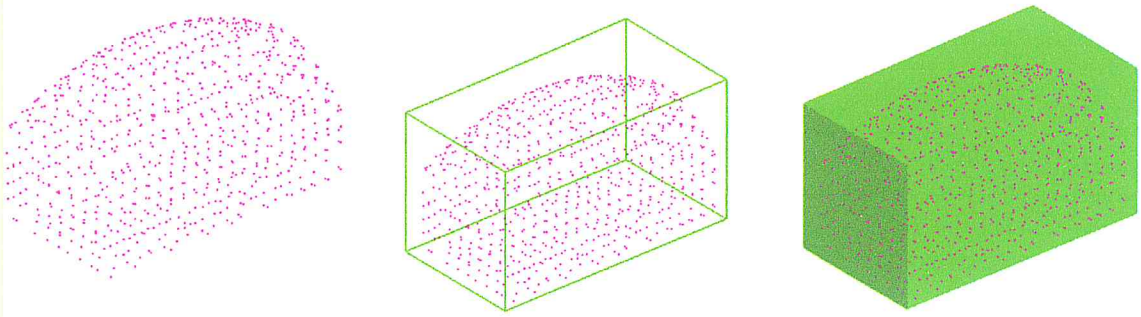


Figure 77 : Visualisation du nuage de points.

IV.1.2. Subdivision du nuage de points

Le test de la subdivision a été effectué pour deux (02) cellules selon une direction (X), une cellule selon Y et une cellule selon Z.

Le bouton Subdivision en cellules permet de générer des cellules en fonction des paramètres introduits par l'utilisateur. Le choix, est fait en sélectionnant ; soit un nombre de cellules selon X, Y et Z, soit les dimensions des cellules en activant le RadioButton associé (Figure 79). En choisissant de créer deux cellules suivant chaque direction, les cellules générées ainsi que les points affectés à ces cellules sont représentés par la Figure 80. La Figure 81 montre les paramètres d'une cellule donnée. Le Tableau 2 donne le nombre de points de chaque cellule.

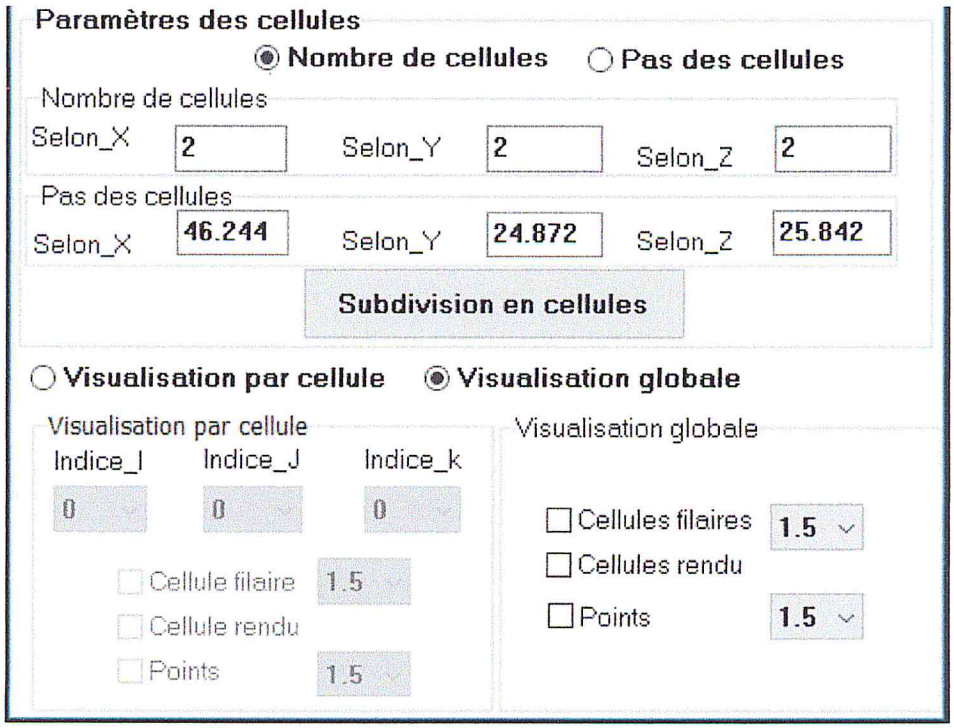
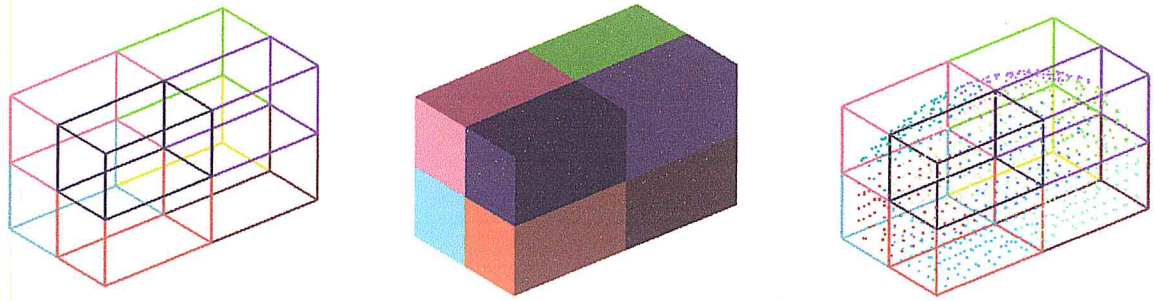
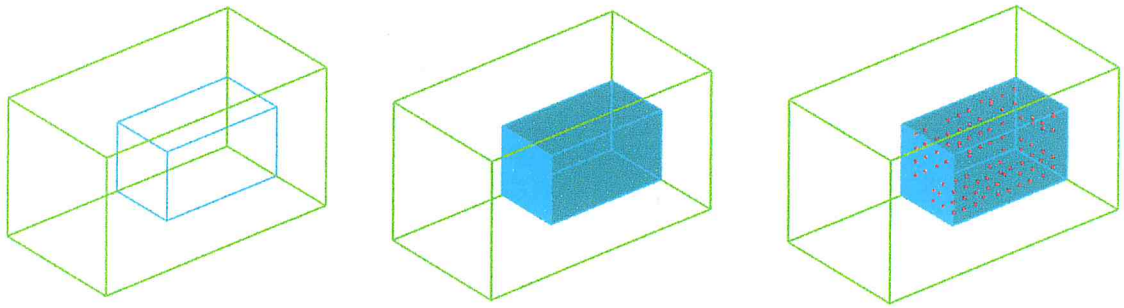


Figure 78 : Subdivision du nuage de points.



a. Cellules en filaire. *b. Cellules en rendu.* *c. Points des cellules.*

Figure 79 : Visualisation des cellules et de leurs points.



a. Une cellule en filaire. *b. Une cellule en rendu.* *c. Points dans une cellule.*

Figure 80 : Visualisation des paramètres d'une cellule.

Tableau 2 : Nombre de points de chaque cellule.

Indice de la cellule	Nombre de points
Cellule 0_0_0	89
Cellule 0_1_0	108
Cellule 0_0_1	88
Cellule 1_0_0	93
Cellule 1_1_0	108
Cellule 1_0_1	92
Cellule 0_1_1	118
Cellule 1_1_1	113

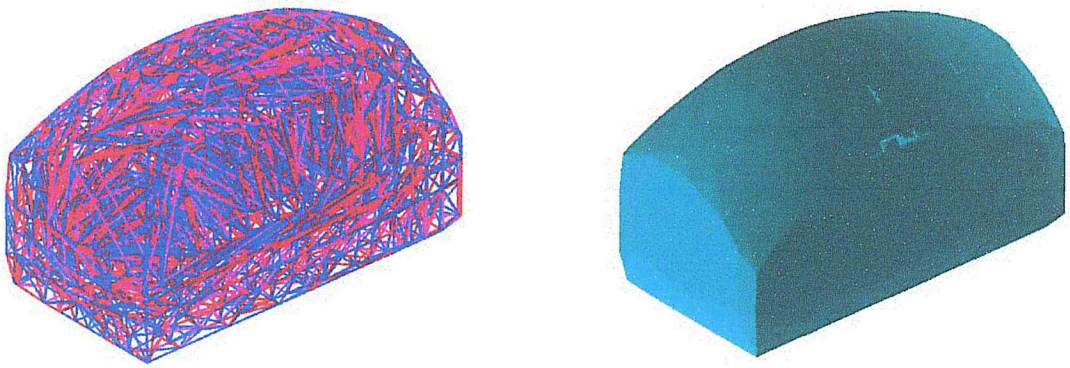
On remarque que le nombre de points dans chaque cellule est plus ou moins homogène. Cependant, ce nombre peut varier en fonction de la taille, de la géométrie et des paramètres de digitalisation de l'objet considéré.

IV.2. Triangulation de Delaunay

Afin de comparer les résultats, une série de tests est menée sur un nuage de points donné. Dans ce qui suit, trois cas sont considérés :

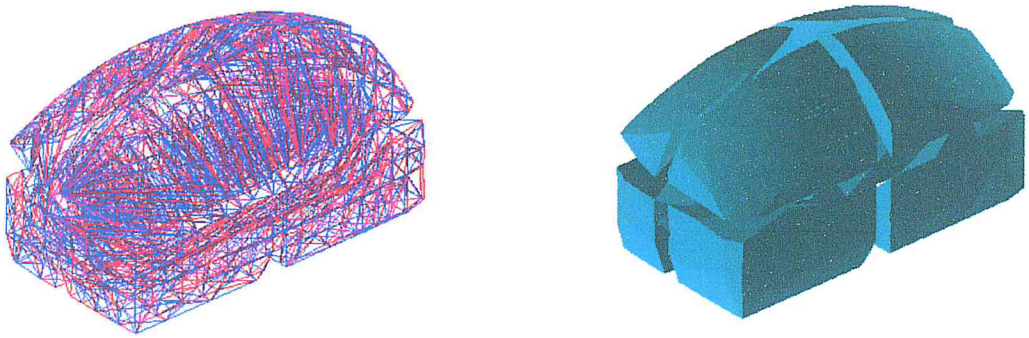
1. Premier cas : triangulation du nuage de points global sans sa subdivision en cellules (Figure 82).
2. Deuxième cas : triangulation séquentielle du nuage de points après sa subdivision en huit cellules, deux cellules dans chaque direction (Figure 83).
3. Troisième cas : triangulation parallèle des huit cellules.

Le Tableau 3 donne le nombre de tétraèdres générés pour chaque cellule. Une comparaison entre les temps de génération de la triangulation de Delaunay pour les trois cas est donnée par le Tableau 4. Il ressort des résultats obtenus que la triangulation en parallèle des cellules est la plus optimale où le pourcentage de réduction des temps est très important comparé à celui obtenu par la triangulation de Delaunay sans subdivision. D'où l'intérêt du calcul parallèle.



a. Triangulation globale en filaire. *b. Triangulation globale en rendu.*

Figure 81 : Visualisation de la triangulation du nuage de points sans subdivision.



a. Triangulation des huit cellules en filaire. *b. Triangulation des huit cellules en rendu.*

Figure 82 : Visualisation de la triangulation des huit cellules.

Tableau 3 : Nombre de tétraèdres dans chaque cellule.

Cellules	Nombre de tétraèdres
Cellule_0_0_0	249
Cellule_0_1_0	309
Cellule_0_0_1	242
Cellule_1_0_0	265
Cellule_1_1_0	296
Cellule_1_0_1	257
Cellule_0_1_1	350
Cellule_1_1_1	336

Tableau 4 : Comparaison entre les trois modes de triangulation.

Mode de triangulation	Temps d'exécution
Global sans subdivision	1h 04min
Nuage subdivisé en huit cellules en séquentiel	2min 13s
Nuage Subdivisé en huit cellules en parallèle	1min 12s

IV.3. Zones affectées

IV.3.1. Génération de l'enveloppe convexe

Avant de lancer la génération des zones affectées, nous devons déterminer l'enveloppe convexe (faces extérieures) de chaque triangulation (Figure 84).

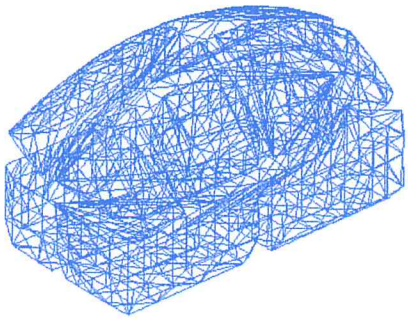
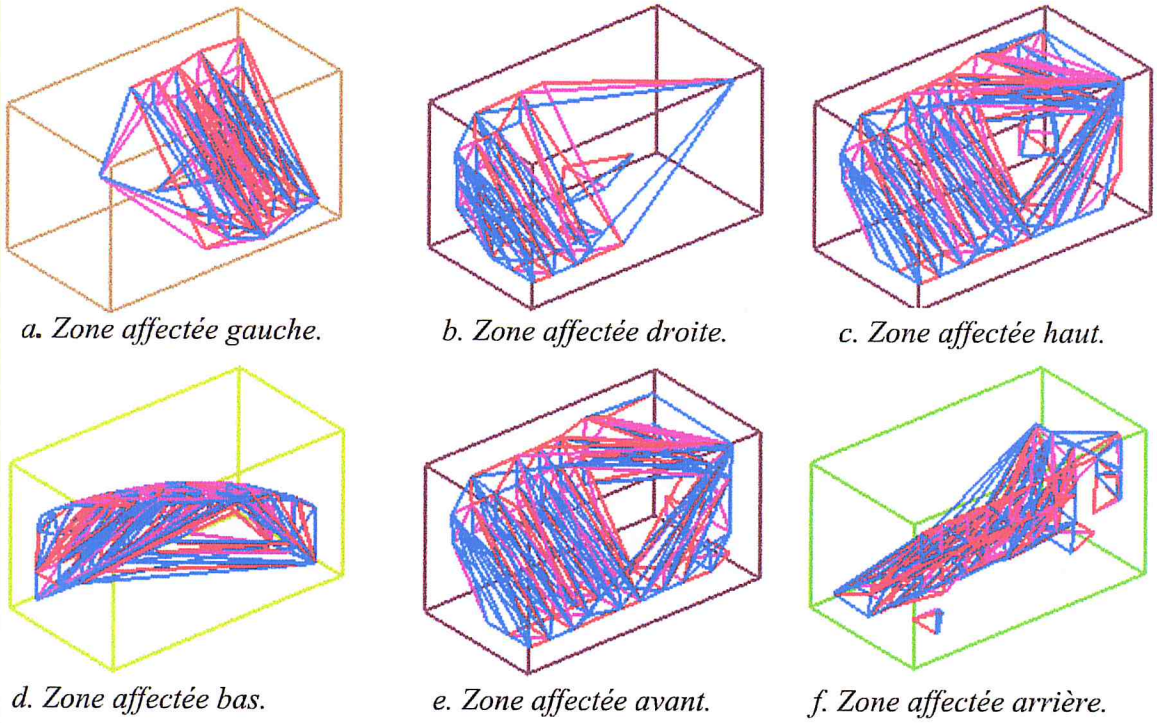


Figure 83 : Visualisation de l'enveloppe convexe des huit cellules.

IV.3.2. Générations des zones affectées

Lors du processus de génération de la zone affectée, nous avons pris en compte les indices des cellules pour générer les bonnes zones affectées. A titre d'exemple, pour la cellule 0_0_0, seules les zones affectées droite, haut et avant seront générées, les zones affectées gauche, bas et arrière ne seront pas prises en compte car la cellule 0_0_0 n'a aucune autre cellule voisine sur ces côtés (Figure 85). Le Tableau 5 donne le nombre de tétraèdres affectés dans les cellules.



a. Zone affectée gauche. *b. Zone affectée droite.* *c. Zone affectée haut.*

d. Zone affectée bas. *e. Zone affectée avant.* *f. Zone affectée arrière.*

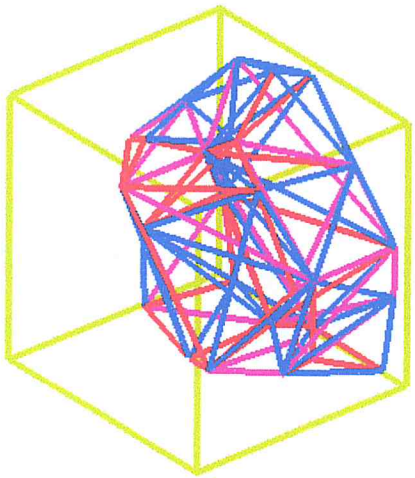
Figure 84 : Visualisation des zones affectées.

Tableau 5 : Nombre de tétraèdres affectés dans des cellules données.

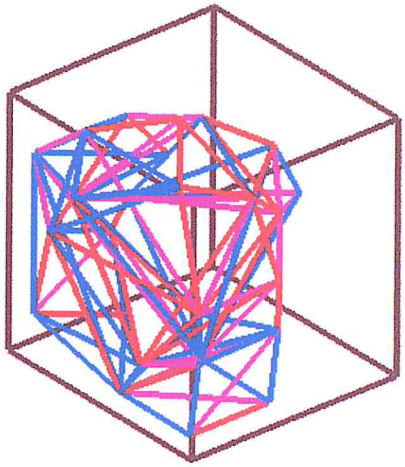
Cellules	Nombre de tétraèdres affectés
Cellule_0_0_0 droite	44
Cellule_1_0_0 gauche	69
Cellule_0_0_0 haut	103
Cellule_0_0_1 bas	196
Cellule_0_0_0 avant	114
Cellule_0_1_0 arrière	149

Etant donné que la densité du nuage de points traité n'est pas grande, les zones affectées ne sont pas bien visibles. Afin de bien expliciter les zones affectées, un nuage de 84 points est utilisé. Ce nuage est subdivisé en deux sous nuages selon trois manières :

- Deux (02) cellules selon X, une (01) selon Y et une (01) selon Z (Figure 86).
- Une (01) cellule selon X, deux (02) selon Y et une (01) selon Z (Figure 87).
- Une (01) Cellule selon X et une (01) selon Y et deux (02) selon Z (Figure 88).

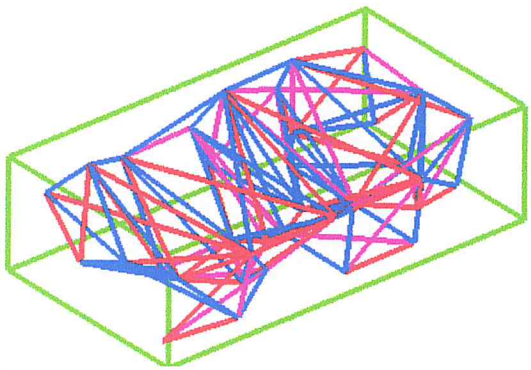


a. Zone affectée gauche.

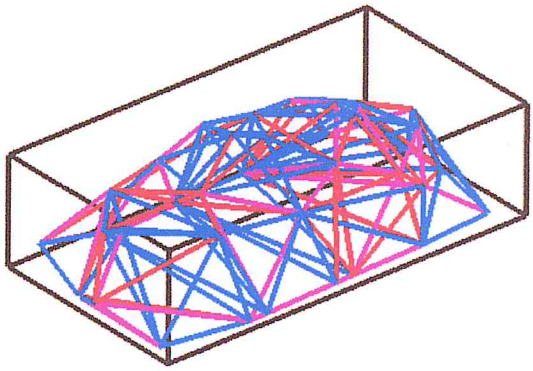


b. Zone affectée droite.

Figure 85 : Zones affectées du nuage après subdivision en deux cellules selon X.

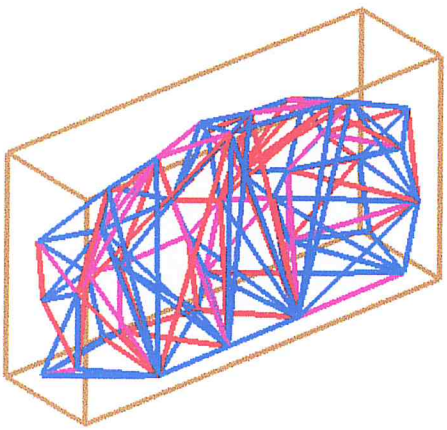


a. Zone affectée haut.

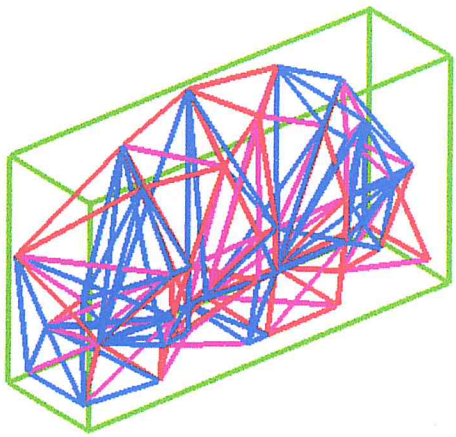


b. Zone affectée bas.

Figure 86 : Zones affectées du nuage après subdivision en deux cellules selon Z.



a. Zone affectée avant.



b. Zone affectée arrière.

Figure 87 : Zones affectées du nuage après subdivision en deux cellules selon Y.

V. Fusion des triangulations

Dans cette partie, le test a été effectué entre deux cellules à savoir la cellule_0_0_0 et la cellule_1_0_0 avec une fusion gauche-droite. Le processus est divisé en quatre parties :

- Détermination des points les plus proches : détermination des points les plus proches entre la cellule 0_0_0 et la cellule 1_0_0 (Figure 89).

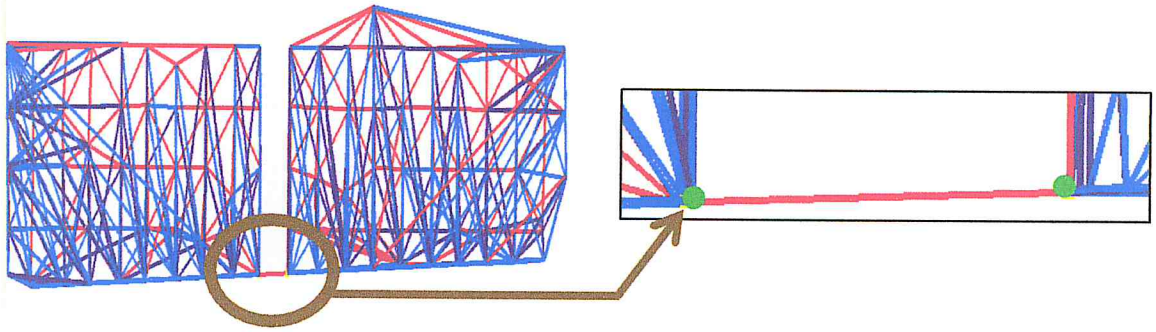


Figure 88 : Visualisation des deux points les plus proches.

- Détermination des tétraèdres candidats : les faces qui contiennent les deux points trouvés précédemment sont déterminées et combinées avec le point situé dans la cellule adjacente pour construire les tétraèdres candidats de chaque cellule (Figure 90).

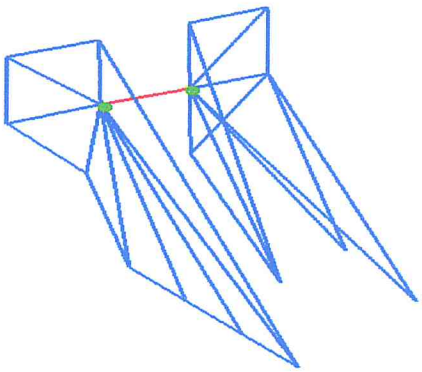
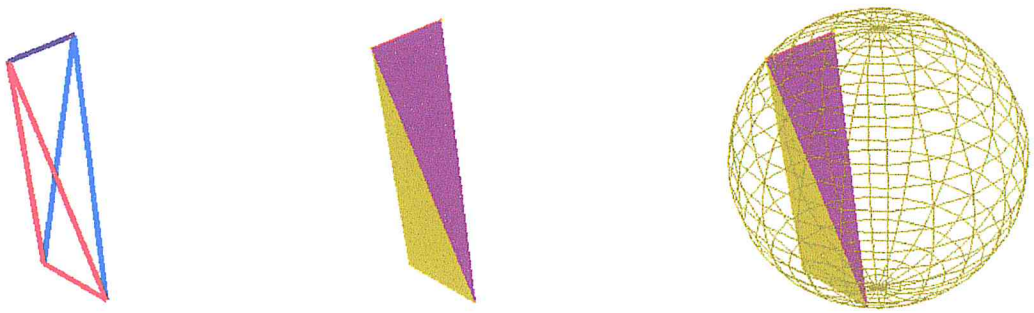


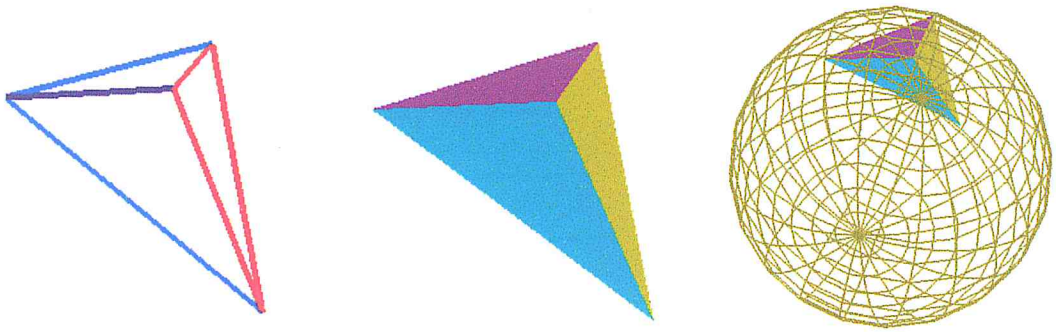
Figure 89 : Visualisation des faces qui contiennent les deux points les plus proches.

Deux tétraèdres candidats sont générés, un pour chaque cellule (Figure 91 et Figure 92).



a. Tétraèdre en filaire. *b. Tétraèdre en rendu* *c. Sphère circonscrite.*

Figure 90 : Visualisation du tétraèdre candidat issu de la cellule_1_0_0.



a. Tétraèdre en filaire. *b. Tétraèdre en rendu* *c. Sphère circonscrite.*

Figure 91 : Visualisation du tétraèdre candidat issu de la cellule_0_0_0.

Après l'application de l'algorithme présenté dans le chapitre précédent, le tétraèdre retenu est le tétraèdre issu de la cellule_0_0_0 (Figure 93).

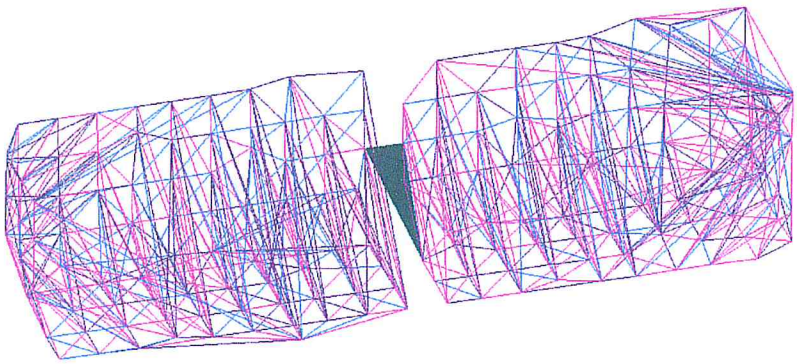


Figure 92 : Visualisation du premier tétraèdre après fusion.

VI. Etude comparative

Dans cette partie, une comparaison est menée entre les résultats obtenus en changeant le nombre de cœurs/cellules sur un même nuage de points de 809 points. Les tests sont réalisés sur trois (03) PC différents :

- PC portable HP sous Windows 10, Intel® Core™ I5-4200 CPU @1.60 GHz 2.30 GHz, 4Go de RAM.
- PC portable ACER sous Windows 7 Intel® Core™ I3–3217U 1.8, 4Go de RAM.

Le Tableau 6 donne les temps de traitement pour les trois cas en fonction du nombre de cœurs.

Tableau 6 : Temps d'exécution en fonction du nombre de cœurs.

Nombre de cœurs	2	4	6
Nombre de points	809	809	809
Nombre de cellules	2	2	2
Temps d'exécution	1h 10min	24min 07s	17min 55s

Le Tableau 7 résume les temps d'exécution pour une machine à quatre (04) cœurs.

Tableau 7 : Temps d'exécution en fonction du nombre de cellules pour une machine à quatre cœurs.

Nombre de cœurs	4	4	4
Nombre de points	809	809	809
Nombre de cellules	2	3	4
Temps d'exécution	32min 46s	13min 57s	5min 06s

Le tableau 8 résume les temps d'exécution pour une machine à deux (02) cœurs.

Tableau 8 : Temps d'exécution en fonction du nombre de cellules pour une machine à deux cœurs.

Nombre de cœurs	2	2
Nombre de points	809	809
Nombre de cellules	2	3
Temps d'exécution	1h 10min	45min 17s

On remarque une différence très importante entre les temps d'exécution. Plus le nombre de cellules est élevé, plus le temps d'exécution est réduit. L'utilisateur doit prendre en compte le nombre de points initiaux et la complexité de l'objet scanné pour faire un bon choix du nombre de cellules afin d'obtenir une réduction considérable des temps d'exécution.

Conclusion

Dans ce chapitre, les différentes étapes de notre application ont été présentées et le travail réalisé a été validé à travers un exemple réel qui est une pièce de forme complexe. L'application a été déroulée depuis la lecture du fichier du nuage de points jusqu'à la fusion.

La validation à travers un seul exemple ne peut refléter l'intérêt de l'application dans le domaine de la reconstruction d'objets surtout pour les applications du Réverse Engineering, mais elle constitue un nouveau maillon dans l'environnement de reconstruction d'objets développé par l'équipe CFAO du CDTA.

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 s'articule autour du développement d'une application logicielle graphique et interactive sous Windows, pour la reconstruction des surfaces gauches à partir d'un nuage de points quelconque. Il s'agit précisément de la réalisation d'une approximation des nuages de points par des éléments géométriques (tétraèdres).

Le premier et le deuxième chapitre évoqués dans ce mémoire, ont été réservés à l'étude bibliographique qui permet en premier lieu de maitriser les méthodes de conception et de modélisation des surfaces gauches, ensuite d'étudier les différentes architectures parallèles et les différentes techniques utilisées pour la parallélisation des applications gourmandes en temps de calcul en appuyant notre étude sur quelques exemples réels d'architectures parallèles.

A la fin, le troisième et le quatrième chapitre ont porté sur la description explicite et l'implémentation de l'approche adoptée ainsi qu'aux solutions proposées.

Le résultat de notre travail est une application logicielle qui prend en charge les points suivants :

- ✓ Subdivision du nuage de points en cellules pour accélérer la reconstruction des surfaces.
- ✓ Affectation des points aux cellules.
- ✓ Génération d'une triangulation 3D pour chaque cellule en utilisant la triangulation de Delaunay 3D en parallèle.
- ✓ Génération des zones affectées pour chaque cellule.
- ✓ Fusion des cellules selon une seule direction.

En perspective, nous proposons de considérer les thèmes suivants :

- Fusion multidirectionnelle des triangulations.
- Gestion de l'affectation des cellules aux processeurs.
- Exécution de la triangulation de Delaunay sur Cluster (MapReduce/MPI).

- Implémentation de l'approche sur la carte graphique NVIDIA en utilisant CUDA pour accélérer des calculs.
- Parallélisation de tous les calculs séquentiels pour minimiser le temps de la triangulation.
- Proposition d'une nouvelle approche pour la concaténation des triangulations.
- Optimisation de la détermination des zones affectées.
- Subdivision optimale du nuage de points.

Références bibliographiques

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, p. 13-14.
2. Philippe Vanackère, Les modélisations géométriques utilisées dans les logiciels de DAO/CAO, revue, Lycée technique d'Armentières, Titre In http : //www.epi.asso.fr/revue/dossiers/d12p123.html, 2004
3. Franck Rolland. Représentation tridimensionnelle et reconstruction 3D à partir de coupes 2D. Modélisation et simulation. Université Joseph-Fourier - Grenoble I, 1991. France. P.11-33
4. Christine Potier, "Modélisation - Mastère Multimédia", Ecole nationale supérieure des télécommunications, Janvier 2002.p.1-24.
5. Bey Mohamed, Modélisation des courbes et des surfaces, Laboratoire de Robotique et d'Intelligence Artificielle Equipe Systèmes Robotisés de Production, Avril 2000. P.21-27.
6. Maria, A. N. Optimisation de l'opération de finition des surfaces de formes libres par l'intégration de différentes formes d'outils et de stratégies d'usinage à partir d'un modèle STL. centre de développement des technologies avancées (2011).p. 27-35.
7. Rémi ALLÈGRE, Thèse de doctorat, "Contributions à l'introduction de l'exibilité dans la reconstruction et l'édition de modèles 3D", l'université Claude Bernard Lyon 1 décembre 2006, p.2-32.
8. Mohanad Makki, Christophe Tournier, François Thiébaud, Claire Lartigue, Reverse Engineering et Copiage rapide de nuages de points numérisés, IUT de Cachan, Université Paris-Sud 11, août 2007.p.3-5.
9. M.O. Challali, I. B. Magister Challali Reconstruction d'objets discrets 3 D Application de la triangulation de Delaunay. p. 13-104
10. V.T.Rajan. Optimality of the Delaunay triangulation in Rd. Discrete Computational Geometry 12; 1994.p. 189-202
11. Seng Poh Lim, Habibollah Haron, Artificial Intelligence Review, June 2012,
12. P.L.George, H. Bourouchaki. Triangulation de Delaunay et maillage, application aux éléments finis. Editions Hermes, Paris 1997.p 7-15
13. C.A.Arens, The Bowyer-Watson algorithm, Faculty of Civil Engineering and Geosciences Delft University of Technology, 2002,p.7-9.
14. Description de l'algorithme diviser pour régner appliqué à Delaunay, par Samuel Peterson : http://www.geom.uiuc.edu/~samuelp/del_project.html
15. P.L.George, H. Bourouchaki. Triangulation de Delaunay et maillage, application aux éléments finis. Editions Hermès, Paris 1997.p 7-15
16. Vérification des fichiers STL, Titre In http://www.drim3d.com/STL

17. BOUGACI Dalila. *Introduction au calcul parallèle*. Rapp. Tech. université des sciences et de la technologie Houari-Boumédiène, 2015/2016.

18. Rédha LOUCIF. *Mémoire Magister Parallélisation d'Algorithmes d'Optimisation Combinatoire*. Rapp. tech. université colonel Hadj Lakhdar-Batna, 13/01/2014.

19. "François Pellegrini Enseirb". "Livre Architectures et Systèmes des Calculateurs Parallèles". In : (2003).

20. *Processeur graphique*. https://fr.wikipedia.org/wiki/Processeur_graphique. Dernière mise à jour : 13-03-2016.

21. *GPU, puces graphiques : qui sont-elles et à quoi servent-elles ?* http://www.frandroid.com/hardware/processeurs/235635_gpu-puces-graphiques-servent. Dernière consultation : 20-03-2016.

22. *Cloud computing*. https://fr.wikipedia.org/wiki/Cloud_computing. Dernière mise à jour : 20-03-2016.

23. "MICHAEL ULRYCK". *Les modèles de service du Cloud : SaaS/PaaS/IaaS*. http://www.michaelulryck.com/les-modeles-de-service-du-cloud-saas-paasiaas. Dernière mise à jour : 8-06-2014.

24. *Grappe de serveurs*. https://fr.wikipedia.org/wiki/Grappe_de_serveurs. Dernière mise à jour : 30-12-2015.

25. *Running Hadoop on Ubuntu Linux (Single-Node Cluster)*. http ://www.michael-noll.com/tutorials/on-ubuntu-linux-single-node-cluster/. Dernière consultation : 01-05-2016.

26. Cédric LHERM. *Présentation : Les clusters*. 2000.

27. Nezha EL GOURII. *Grid Computing*. http://www. supinfo.com / articles /single/281-grid-computing. Dernière mise à jour : 13-09-2015.

28. Smaël Laskri" Aurélien Jolly. *Les grilles de calcul*. http://igm.univ-mlv.fr/~dr/XPOSE2006/Jolly_Laskri/index.html. Années :2006/2007.

29. MEHDI Malika. *Cour Gestion et Administration de grilles de calcul*. Rapp. tech. université des sciences et de la technologie Houari-Boumédiène.

30. http://docwiki.embarcadero.com Dernière mise à jour 25-08-2015

31. Min-Bin Chen , The Merge Phase of Parallel Divide-and-Conquer Scheme for 3D Delaunay Triangulation , China University of Technology,2011