République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad Dahlab, Blida
USDB.

Faculté des sciences.
Département informatique.

جامعة الوليدة
المكتبة
كلية العلوم

Mémoire pour l'obtention
D'un diplôme d'ingénieur d'état en informatique.
Option : Système d'Information (SI)

Sujet :

Conception et Développement d'une Application de Reconstruction des Surfaces Gauches à Partir d'un Nuage de Points et Recherche des Outils Usinage Optimums

Présenté par : BOUTASSOUNA Mohamed **Promoteurs :** BEY Mohamed

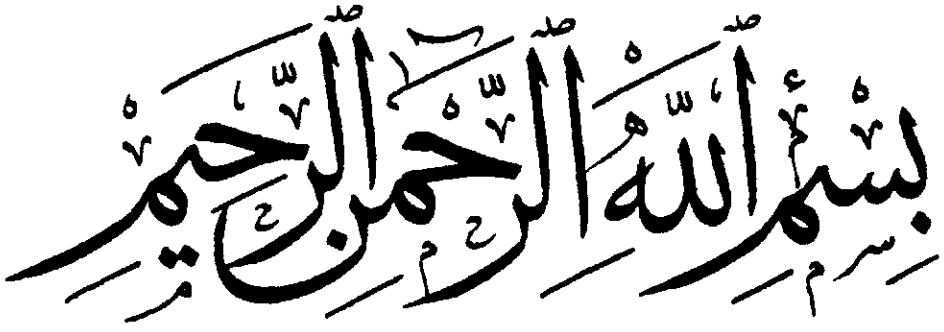
Organisme d'accueil : CDTA (Centre de Développement des Technologies Avancées).
Division Productique et Robotique

Soutenue le: date soutenance, devant le jury composé de :

Nom. Président du jury, grade, organisme	**Président**
Nom examinateur 1, grade, organisme	**Examinateur**
Nom examinateur 2, grade, organisme	**Examinateur**

- 2006/2007-

MIC-004-194-1


بسم الله الرحمن الرحيم

ملخص

إنّ هذا العمل يندرج في إطار تطوير وسيلة، قادرة على تصميم و تصنيع الأسطح ذات الأشكال الحرّة و التي يقوم بتطويرها فرقة البحث الخاصّة بالتّصميم و التّصنيع عن طريق الكمبيوتر على مستوى فرع التّصنيع الآلي الخاص بمركز تطوير التكنولوجية المتقدّمة.

هذا العمل يدرس إعادة هندسة الأجسام المعرفة بمجموعة نقاط, و لقد اخترنا طريقة تثليث "دلوني"، بإضافة الى إيجاد نصف قطر القاطعة الملائم و الذي يجنبنا تداخل مساحات التقطيع في ما بينها، تقسيم المساحات و تشكيلها بمساحات من النقط متباينة مع إيجاد نصف قطر القاطعة الملائم لكل مساحة .
الهدف من هذا العمل هو تطوير برنامج آلي تصويري قابل للاستعمال على نظام التشغيل ويندوز.

الكلمات المفتاحية: مجموعة نقاط ، تثليث دلوني، إعادة الهندسة.

Résumé

Ce travail s'insère dans le cadre de développement d'outils de conception et de fabrication des surfaces gauches initié par l'équipe Conception et Fabrication Assistées par Ordinateur (CFAO) au niveau de la Division Robotique et Productique du Centre de Développement des Technologies Avancées (CDTA).

Dans ce projet on s'intéressera en premier lieu à la triangulation du nuage de points et en second lieu au choix automatique des dimensions des outils hémisphériques permettant d'éviter les problèmes d'interférences et de collisions lors de l'usinage des surfaces gauches en finition sur des fraiseuses à commande numérique à 3 axes. En dernier lieu, le partitionnement du nuage de points en des régions distinctes (concave, convexe, ...etc.) dans le but de sélectionner l'outil optimum pour chaque région. Le but de ce travail est le développement d'une application logicielle graphique et interactive sous Windows permettant d'automatiser toutes les tâches.

Mots clés: **Nuage de points, triangulation, triangulation de Delaunay, reconstruction.**

Summary

This work is included in development of tools for the design and manufacture of free form surfaces initiated by the team Computer Aided Design and Manufacture of the Robotics and Production Division in the Advanced Technology Development Center (CDTA).

In this project, we are interested firstly in the automation of the triangulation from a point cloud data and secondly in choosing automatically the optimum tools in order to avoid the problems of interferences and collisions while finishing free form surfaces on 03-axes CNC milling machines. Finally, the determination of the different regions (concave, convex, etc.) in order to select the optimum tool for each region. The objective of this work is the development of software permitting the automation of these tasks.

Keywords: ***Point data, triangulation, Delaunay triangulation, reconstruction.***

Remerciements

D'aéroports en aéroports

De gare en gare

De péages d'autoroutes en péages d'autoroutes . . .

Que de kilomètres parcourus mais enfin cette thèse se termine[1] *. . .*

*Tout d'abord, Je tiens à remercier en premier lieu « **ALLAH** » le tout puissant de ma avoir donné le courage, la volonté et la chance de mener ce projet à terme.*

Je ne voyais pas comment commencer autrement.

Je remercier très chaleureusement, nos très chers parents, pour leur soutient moral et financier tout au long de mes études.

*Il est agréable d'exprimer notre sincère gratitude, à notre promoteur M **Bey Mohamed** qui s'est occupé de la réalisation technique et qui n'a jamais fui devant mes idées les plus folles et impossibles, Il m'a non seulement encouragé et conseillé, mais j'ai aussi bénéficié de sa constante disponibilité et de ses remarques pertinentes tout au long de ce travail. mais aussi sans les qualités humaines et professionnelles de **ZEMMOURI CHABANE** et **ALAA DINNE**.*

*Je tiens à remercier **M CHERIF ZAHAR** qui a accepté de présider cette thèse, je lui suis infiniment reconnaissant et je suis très fier de le compter parmi mon jury. Je remercie également **Mme ARKAM** pour avoir accepté de faire parti du jury comme examinateurs*

*Un grand merci à **M BEN GUERBA** pour avoir accepté de participer au jury de ma thèse*

1Laissons le temps au temps, la lumière au soleil et les âmes en paix. . .

Je remercie ***M BALA*** et ***Mr TOUMI*** *pour ses grandes qualités de pédagogie et pour l'aide qu'ils ont me donner.*

Je termine en remerciant toutes les personnes, que je n'ai pas citées ici mais qui se reconnaîtront, qui de près ou de loin m'ont fait confiance et ont participé à ce modeste travail à leur façon.

Je remercie encore et toujours mes parents pour tout le soutien qu'il m'ont donné,
merci, merci, mille et deux fois merci. . .

DEDICACE

Je dédie le fruit de mes études :

A mes très chers parents qui m'ont toujours encouragé et qui ont éclairé ma route par leur compréhension, leur sacrifice et leur affection.

A mes frères : Yacine, Lahcen, Hocine, Ahmed, Tahat Abdelkader , et surtout abdelrahmen.

A mes sœur : Nassima, Assuma, Maria et a ma petite fleure Iftissane.

A ma grande famille.

A mes amis : Ismaïl, Chaban, Hmida ,Sadek, Toufik. Mohamed

A tous les amis de ma promotion. A tous mes amis et collègues sans restriction.

A Mr Bey Med et Bendifallah Hassen.

Sommaire

sommaire

Chapitre III : Reverse Engineering « Reconstruction »

Chapitre IV : Triangulation de Delaunay

Chapitre V : Conception de l'application

Chapitre VI : Implémentation De L'application

Chapitre VII : *Tests Et Validations*

Liste des Figures

Liste des figures

Chapitre I : Méthodes de conception et de représentation des surfaces

Chapitre II : Machine outils et usinage des surfaces gauches

Chapitre III : Reverse Engineering « Reconstruction »

Chapitre IV : Triangulation de Delaunay

Chapitre V : Conception de l'application

Chapitre VI : Implémentation De L'application

Liste des abréviations

CAO : Conception Assistée par Ordinateur

FAO : Fabrication Assistée par Ordinateur

CFAO : Conception et Fabrication Assistées par Ordinateur

MMT : Machine à Mesurer Tridimensionnelle

MOCN : Machine Outil à Commande Numérique

TD : Triangulation de Delaunay

NURBS : Non Uniform Rational B-Spline

ISO : International Standardization Organization

C_C : Cutter Contact (point de contact)

C_L : Cutter Location (point extrémité outil)

RE : Reverse Engineering

Notations

Π	Plan
T_u	Vecteur tangent d'un point dans la direction u
T_v	Vecteur tangent d'un point dans la direction v
$\vec{n}$	Vecteur normal
$p(u,v)$	Coordonnées d'un point dans le plan paramétrique
$f(u,v)$	Coordonnées d'un point sur la surface paramétrique
K	Courbure gaussienne
K_1	Courbure minimale
K_2	Courbure maximale
R	Rayon du cercle de courbure
H	Courbure moyenne d'une surface en point P
$P_{i,j}$	Point de contrôle
$W_{i,j}$	Poids d'une point de contrôle
T	Triangulation

Introduction Générale

INTRODUCTION GENERALE

I. SITUATION DU PROBLEME

Les systèmes de conception et de fabrication assistés par ordinateur sont utilisés dans diverses industries afin de créer des objets physiques à partir de modèles numériques. De tels systèmes ont énormément attirés l'attention des chercheurs et des développeurs dans ce domaine et continuent à le faire. Cependant, le problème inverse, qui consiste à conférer un modèle numérique à un objet physique, n'a pas suscité le même engouement si ce n'est depuis ces quelques dernières années. Ce problème inverse est désigné par "la reconstruction" ou par "reverse engineering". Les avantages et l'importance des systèmes de CFAO ne sont plus à démontrer, à telle enseigne qu'il est devenu presque impensable de concevoir des objets industriels en utilisant une table de dessin. L'existence d'un modèle numérique d'un objet permet un gain substantiel. Celui-ci permet, entre autres, d'apprécier le produit, sous son format numérique, sur différents plans avant d'envisager sa production. Ainsi, une amélioration de la qualité, une augmentation de l'efficacité du design, des simulations virtuelles et des analyses de comportement fonctionnel ainsi qu'une optimisation des conditions de production sont réalisées avant de décider de la validité et de la conformité du produit.

Les pièces de formes gauches sont utilisées dans la conception et la réalisation des moules, des matrices, des formes esthétiques, ...etc. En raison des formes géométriques très complexes, ces pièces sont usinées sur des fraiseuses à commande numérique à 3, 4 et 5 axes après la génération de la trajectoire d'usinage à partir des modèles CAO des formes des surfaces à usiner. Ces modèles peuvent être obtenus par deux méthodes. La première méthode se base sur l'utilisation d'un logiciel de CAO. Cependant, il n'est pas toujours évident d'obtenir les formes voulues. Dans le cas où les formes des pièces sont très complexes et ne peuvent pas être conçues dans un logiciel de CAO ou le modèle CAO n'est pas disponible, une deuxième méthode est utilisée « Reverse Engineering » qui repose sur l'acquisition d'un nuage de points avec une machine à mesurer tridimensionnelle « MMT », et par suite faire des traitements appropriés pour générer le trajet d'usinage. Avant la génération du trajet d'usinage, une étape très importante qu'il faut considérer c'est l'étape de choix des formes et des dimensions des outils à utiliser pour minimiser les temps d'usinage et pour éviter les problèmes d'interférences et de collisions. Dans le cas où les modèles CAO sont donnés, ce choix est simplifié par la possibilité de calculer les propriétés géométriques des surfaces, mais dans le cas où les modèles CAO ne sont pas disponibles et la surface est représentée par un nuage de points très dense, cette tâche devient très complexe et prend beaucoup de temps.

Le travail que nous présentons dans ce mémoire s'inscrit dans le cadre de développement d'outils de conception et de fabrication des surfaces de formes libres (surfaces gauches) initié par l'équipe CFAO de la Division Productique et Robotique du Centre de Développement des Technologies

Avancées (CDTA). Notre projet est une continuité des travaux précédents traitant de la :

- Modélisation et conception des courbes et des surfaces B-Spline et NURBS;
- Reconstruction des courbes et des surfaces B-Spline et NURBS par interpolation et par approximation à partir d'un nuage de points;
- Usinage des surfaces gauches par les courbes isoparamétriques,
- Usinage des surfaces gauches par la stratégie des plans parallèles,
- Usinage des surfaces gauches par la stratégie Z-Constant,
- Ebauchage des surfaces gauches,
- Simulation de l'usinage des surfaces gauches.

II. OBJECTIF DU TRAVAIL

Dans ce projet, nous nous s'intéresserons à l'automatisation de la génération d'une triangulation dite de Delaunay à partir d'un nuage de points qui approxime le mieux la surface d'un objet, le groupement des triangles en des régions distinctes en fonction des formes locales des points et en dernier lieu la détermination du rayon d'outil optimum à associer à chaque région pour éviter les problèmes d'interférence et cela par le développement d'une application logicielle graphique et interactive permettant d'automatiser ces tâches.

III. DESCRIPTION DU TRAVAIL

Le présent mémoire est composé de sept chapitres :

- Dans le premier chapitre, nous allons étudier les méthodes de représentation et de conception des surfaces et en particulier les surfaces NURBS et B-Spline.
- Le deuxième chapitre est consacré à la présentation des machines outil à commande numérique (MOCN) et en particulier les fraiseuses ainsi que l'usinage des surfaces gauches sur ces machines.
- Dans le troisième chapitre, nous allons présenter les différentes étapes du processus du reverse engineering.
- Dans le quatrième chapitre, nous allons étudier la triangulation de Delaunay, ses propriétés et les méthodes de génération de cette triangulation.
- La conception de notre application logicielle est présentée dans le cinquième chapitre.
- L'implémentation de notre application logicielle est détaillée dans le sixième chapitre.
- Le dernier chapitre présente les tests et la validation des résultats.

Chapitre I

Méthodes de Conception et de Représentation des Surfaces

I. INTRODCTION :

La génération, la représentation et la modification des surfaces jouent un rôle important en conception de formes et en calcul de structures. Donc, nous avons intérêt à disposer pour les surfaces de modèles mathématiques concis associés à des règles mathématiques simples permettant d'obtenir génériquement des détails et des propriétés d'un élément de surface. Les surfaces complexes sont souvent construites par l'assemblage d'éléments simples de type carreaux, avec des contraintes de continuité. Dans ce chapitre, nous allons considérer les méthodes de conception et de représentation des surfaces utilisées dans la conception des pièces mécaniques.

II. METHODES DE REPRESENTATION DES SURFACES [2, 5] :

Les méthodes de représentation des surfaces peuvent être classées en deux grandes familles :

- Les surfaces non paramétriques.
- Les surfaces paramétriques.

II.1. Surfaces non paramétriques :

Ces surfaces peuvent être représentées sous deux différentes formes :

II.1.1.Forme explicite :

Une surface explicite est donnée par une équation algébrique de la forme :

$$Z = f(x, y) \tag{1.1}$$

Où à chaque valeur de x et y correspond une et une seule valeur de Z. Donc, cette surface ne peut pas être fermée puisque des valeurs multiples ne sont pas permises. La forme explicite n'est pas utilisée en CAO pour les raisons suivantes :

- La complexité de la modification interactive et de la modélisation
- L'impossibilité de représenter toutes les surfaces.

II.1.2. Forme implicite :

Une surface implicite est donnée par l'équation :

$$f(x, y, z) = 0 \tag{1.2}$$

Cette représentation nécessite la résolution d'une équation pour obtenir les triplets de coordonnées. La forme n'a pas la limitation de la représentation explicite, mais il peut être difficile de représenter une surface en 3D sous une forme implicite, ainsi que la manipulation interactive est difficile pour ces surfaces.

II.2. Surfaces paramétriques :

Les surfaces paramétriques sont très utilisées dans les systèmes de modélisation de surfaces. Une surface paramétrique est définie par un ensemble de trois fonctions, une pour chaque coordonnée. Ces fonctions dépendent de deux paramètres *u* et *v*. Une surface paramétrique est donnée par [1] :

$$P(u,v) = (x(u,v), y(u,v), z(u,v)) \qquad (1.3)$$

Où les paramètres *u* et *v* sont dans l'intervalle [0, 1]. Donc, (u,v)est un point d'un carré défini par les sommets (0,0), (1,0), (0,1) et (1,1) dans le plan des coordonnées (u,v). Pour chaque (u,v) lui correspond un point $P(u,v)$sur la surface paramétrique (voir Fig.I.1).

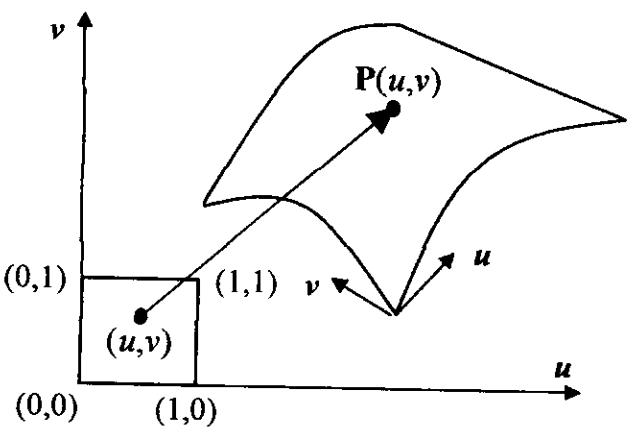


Fig.I.1. Localisation d'un point sur une surface paramétrique.

Dans la pratique de conception, les surfaces paramétriques ne sont pas utilisées individuellement, mais plusieurs morceaux de surfaces paramétriques sont joints ensemble pour former une forme plus complexe.

II.3. Propriétés géométriques des surfaces paramétriques [2-4] :

Les principales propriétés géométriques des surfaces paramétriques sont :

II.3.1. Vecteurs tangents et vecteur normal :

Le calcul des vecteurs tangents et du vecteur normal en un point (*u, v*) nécessite la détermination des dérivées partielles. Ainsi, les deux vecteurs tangents à la surface au point $P(u,v)$ dans les deux directions *u et v* sont donnés par les deux formules suivantes :

$$T_u = \frac{\partial P}{\partial u} = \left(\frac{\partial x}{\partial u}, \frac{\partial y}{\partial u}, \frac{\partial z}{\partial u}\right) \qquad (1.4)$$

$$T_v = \frac{\partial P}{\partial v} = (\frac{\partial x}{\partial v}, \frac{\partial y}{\partial v}, \frac{\partial z}{\partial v}) \qquad (1.5)$$

Le vecteur normal à la surface $n(u,v)$ est le produit vectoriel des deux vecteurs tangents et il est donné par l'équation suivante :

$$n = \frac{\frac{\partial P}{\partial u} \times \frac{\partial P}{\partial v}}{\left|\frac{\partial P}{\partial u} \times \frac{\partial P}{\partial v}\right|} \qquad (1.6)$$

La figure suivante représente les vecteurs tangents T_u et T_v, le vecteur normal n et le plan tangent Π en un point $P(u,v)$d'une surface paramétrique.

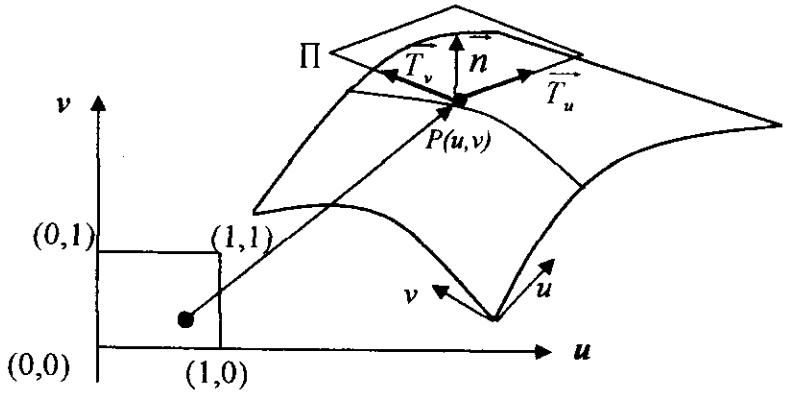


Fig.I.2. Vecteurs tangents et vecteur normal.

II.3.2. Courbes isoparamétriques :

Une surface paramétrique peut être considérée comme l'union d'un nombre infini de courbes. Ils existent plusieurs méthodes pour former cette union, mais la plus simple est appelée courbe isoparamétrique. La surface paramétrique *P(u, v)* est obtenue en fixant l'un des deux paramètres et en faisant varier l'autre ce qui génère une courbe isoparamétrique dans la direction de la variable variée. La figure suivante montre deux courbes isoparamétriques (une pour chaque direction).

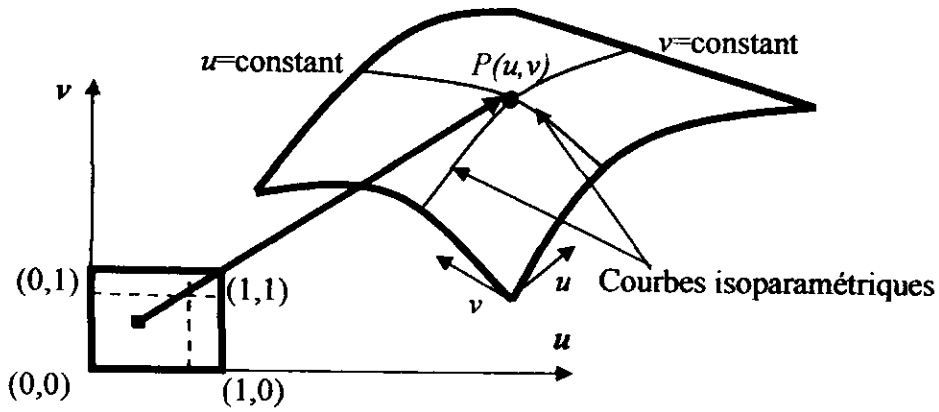


Fig.I.3. Courbes isoparamétriques d'une surface paramétrique.

II.3.3. Notion de courbure [1] :

En un point d'une courbe, la courbure est l'inverse du rayon du cercle osculateur (le cercle qui approxime le mieux la courbe que tout autre cercle), ce rayon est appelé rayon de courbure. Cette courbure est donnée par l'équation suivante :

$$K = 1/R \tag{1.7}$$

R : est le rayon du cercle de courbure.

K : est la courbure.

T : est le vecteur tangent

N : est le vecteur normal au point considéré.

La figure suivante montre le cercle osculateur d'une courbe en un point :

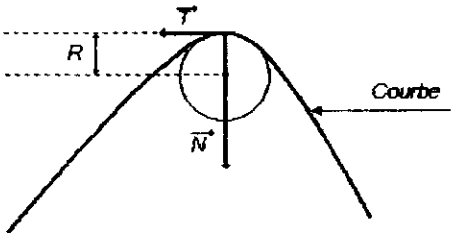


Fig.I.4. Courbure d'une courbe en un point.

Une surface possède plusieurs types de courbures et le calcul d'un type de courbure d'une surface permet la détection des anomalies extérieures possibles de cette surface. Pour une surface, en chaque point nous pouvons calculer deux courbures principales k_1 et k_2. Les rayons de courbures principaux d'une surface sont les solutions de l'équation du second degré suivante :

$$(LN - M^2)R^2 + (2MF - GL - EN)R + EG - F^2 = 0 \tag{1.8}$$

Où L, E, N, G, M et F sont les paramètres utilisés pour calculer les deux formes fondamentales et sont donnés par les relations suivantes :

$$E = \frac{\partial P}{\partial u} \times \frac{\partial P}{\partial u} \tag{1.9}$$

$$F = \frac{\partial P}{\partial u} \times \frac{\partial P}{\partial v} \tag{1.10}$$

$$G = \frac{\partial P}{\partial v} \times \frac{\partial P}{\partial v} \tag{1.11}$$

$$L = n \times \frac{\partial^2 P}{\partial u^2} \tag{1.12}$$

$$M = n \times \frac{\partial^2 P}{\partial u \partial v} \tag{1.13}$$

$$N = n \times \frac{\partial^2 P}{\partial v^2} \tag{1.14}$$

La courbure moyenne H d'une surface en point P est donnée par :

$$H = \frac{K_1 + K_2}{2} \tag{1.15}$$

La courbure gaussienne *K* d'une surface en un point est donnée par :

$$K = K_1 \times K_2 \tag{1.16}$$

Les valeurs de ces deux courbures (H, K) en un point x sur la surface, nous permettent de définir la forme locale de la surface :

- Si $K > 0$ alors *x* est un point elliptique (concave ou connexe).
- Si $K < 0$ alors *x* est un point hyperbolique (selle de cheval).
- Si $K = 0$ et $H \neq 0$ alors *x* est un point parabolique (développable).
- Si $K = 0$ et $H = 0$ alors *x* est un point plat.

III. METHODES DE CONCEPTION DES SURFACES [1] :

Les méthodes de conception des surfaces sont classées en deux grandes classes :

Méthodes basées sur les courbes : dans ces méthodes, la reconnaissance de quelques courbes clés permet la détermination de la surface. Les types de surfaces de cette classe sont :

- Surface réglée.
- Surface de révolution.
- Surface balayée.
- Surface oscillante.
- Surface lissée.
- Surface de Gordon.
- Surface de Coons.
- Surface extrudée.

Méthodes basées sur les points : dans ces méthodes, l'information élémentaire est le point. Les types de surfaces de cette classe sont :

- Interpolation et approximation d'un nuage de points.
- Surfaces de Bézier.
- Surfaces de Bézier Rationnelle.
- Surfaces B-Spline.
- Surfaces NURBS (Non-Uniform Rational B-Spline).

Dans notre travail, nous allons considérer les surfaces les plus utilisées en CAO (Conception Assistée par Ordinateur) à savoir les B-Spline et les NURBS.

III .1. Surface B-Spline :

Une surface B-Spline est définie à partir des informations suivantes (voir Fig.I.5) :

- Un réseau de $(m+1)\times(n+1)$ points de contrôle $\mathbf{p}_{i,j}$, où $0 \le i \le m$ et $0 \le j \le n$,
- Un vecteur des nœuds de $h+1$ nœuds dans la direction u.
- Un vecteur des nœuds de $k+1$ nœuds dans la direction v.
- Le degré p de la surface dans la direction u.
- Le degré q de la surface dans la direction v.

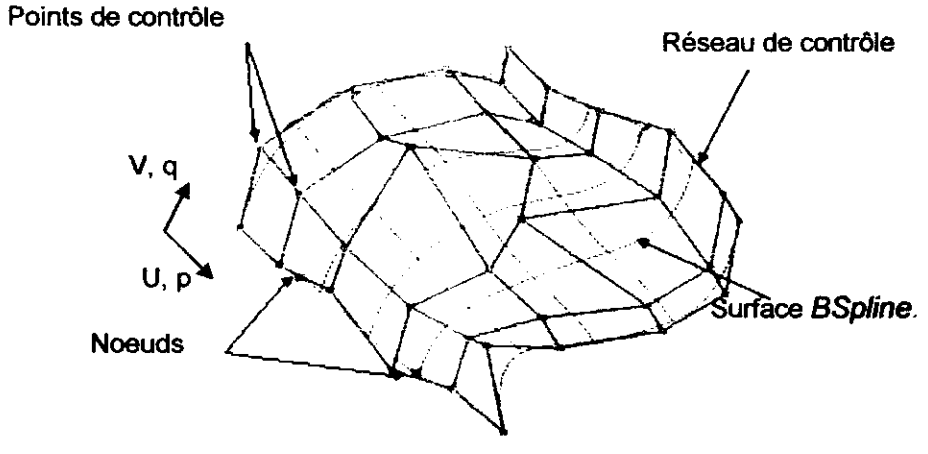


Fig.I.5. Paramètres de définition d'une surface B-Spline.

La surface B-Spline définie par ces informations est donnée par [5,8] :

$$p(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u) N_{j,q}(v) p_{i,j} \qquad (1.19)$$

Où $N_{i,p}(u)$ et $N_{j,q}(v)$ sont les fonctions base B-Splines de degré p et q respectivement et sont données par :

$$N_{i,0}(v)=\begin{cases}1 & si \quad u_i \le u < u_{i+1}\\ 0 & sinon\end{cases}$$

$$N_{i,p}(u)=\frac{u-u_i}{u_{i+p}-u_i}N_{i,p-1}(u)+\frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(u) \qquad (1.20)$$

$$N_{j,0}(v)=\begin{cases}1 & si \quad v_i \le v < v_{i+1}\\ 0 & sinon\end{cases}$$

$$N_{j,q}(v)=\frac{v-v_j}{v_{j+p}-v_j}N_{j,q-1}(v)+\frac{v_{j+q+1}-v}{v_{j+q+1}-v_{j+1}}N_{j+1,q-1}(v) \qquad (1.21)$$

Les identités fondamentales une pour chaque direction doivent être satisfaites :

$$\begin{cases}h=m+p+1\\ k=n+q+1\end{cases} \qquad (1.22)$$

III.1.1. Surfaces B-Spline pincées, fermées et ouvertes :

Une surface B-Spline de degré (p, q) de h+1 nœuds dans la direction u et k+1 nœuds dans la direction v, peut se présenter sous trois types dans chaque direction (pincée, ouverte ou fermée).

- Si la surface B-Spline est ***pincée*** dans les deux directions, alors cette surface ne passe pas par les points de contrôle $p_{0,0}$, $p_{m,0}$, $p_{0,n}$ et $p_{m,n}$. et elle est tangente aux huit segments du réseau de contrôle en ces points
- Si la surface B-Spline est ***fermée*** dans une direction, alors toutes les courbes isoparamétriques dans cette direction sont fermées.
- Si la surface B-Spline est ***ouverte*** dans les deux directions, alors la surface ne passe pas par les points de contrôle $p_{0,0}$, $p_{m,0}$, $p_{0,n}$ et $p_{m,n}$.

III.1.2. Importantes propriétés des surfaces B-Splines :

Les importantes propriétés des surfaces B-Splines sont les suivantes :

- La représentation des formes en surfaces B-Spline est définie par morceaux.
- Les surfaces de Bézier sont des cas particuliers des surfaces B-Spline sous certaines conditions.
- La surface B-Spline doit satisfaire les égalités fondamentales suivantes :

$$\begin{cases}h=m+p+1\\ k=n+q+1\end{cases}$$

- La surface B-Spline vérifie la propriété de l'enveloppe convexe.
- La surface B-Spline n'est pas aussi complexe que son réseau de contrôle.
- Les courbes S(0,v), S(1,v), S(0,u) et S(1,u) sont les courbes limites.
- La surface est indépendante du système de coordonnées.

L'inconvénient des surfaces B-Spline, c'est l'impossibilité de représenter les surfaces coniques (sphère, ellipsoïde, ...etc.) puisque ces surfaces utilisent des fonctions polynomiales, d'où la nécessité de passer aux surfaces qui utilisent des fonctions rationnelles et qui sont les surfaces NURBS.

III.2. Surface NURBS :

Une surface NURBS est définie à partir des informations suivantes (Fig.I.6) :

- ❖ Un réseau de (m+1)×(n+1) points de contrôle $p_{i,j}$, où $0 \le i \le m$ et $0 \le j \le n$.
- ❖ Pour chaque point de contrôle est associé un poids $w_{i,j} \ge 0$.
- ❖ Un vecteur des nœuds de h+1 nœuds dans la direction u.
- ❖ Un vecteur des nœuds de k+1 nœuds dans la direction v.
- ❖ Le degré p de la surface dans la direction u.
- ❖ Le degré q de la surface dans la direction v.

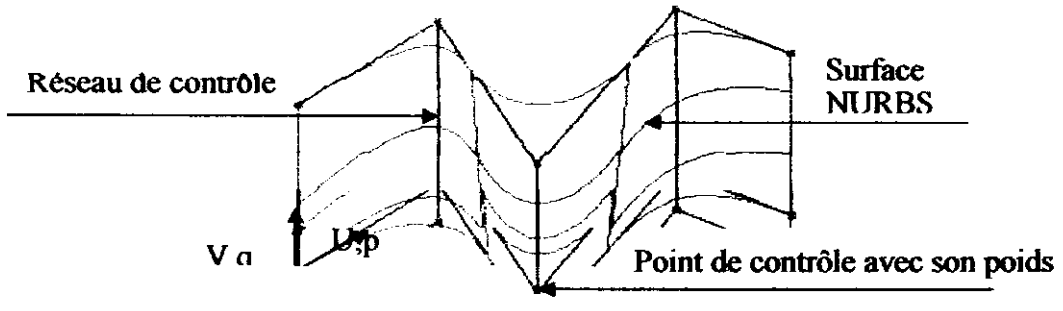


Fig.I.6. Paramètres de définition d'une surface B-Spline.

La surface NURBS définie par ces informations est donnée par [6, 7, 8] :

$$p(u,v) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)N_{j,q}(v)w_{i,j}p_{i,j}}{\sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)N_{j,q}(v)w_{i,j}} \qquad (1.23)$$

Où $N_{i,p}(u)$ et $N_{j,q}(v)$ sont les fonctions base B-Spline de degré p et q respectivement.

III.2.1.Propriétés d'une surface NURBS [7] :

En plus des propriétés des surfaces B-Spline, une surface NURBS a les propriétés suivantes :

1- Le déplacement d'un point de contrôle cause le déplacement d'une portion de la surface dans la même direction (schéma de modification locale).

2- Augmenter (resp. diminuer) la valeur du poids $w_{i,j}$ du point de contrôle $\mathbf{p}_{i,j}$ attire (resp. repousse) une portion de la surface vers (resp. loin de) ce point de contrôle.

3- Les surfaces B-Spline et les surfaces de Bézier sont des cas particuliers des surfaces NURBS.

4- N'importe quelle courbe isoparamétrique sur la surface NURBS est une courbe NURBS définie par un ensemble de points de contrôle.

5- Les courbes isoparamétriques $\mathbf{p}(0,v)$, $\mathbf{p}(1,v)$, $\mathbf{p}(u,0)$ et $\mathbf{p}(u,1)$ sont les courbes de frontières.

IV. CONCLUSION :

La conception des surfaces est faite en utilisant différentes surfaces de représentation. La représentation la plus puissante est celle des surfaces NURBS. Cette dernière est caractérisée par un schéma de modification locale et par l'existence des poids qui donnent plus de flexibilité lors de la conception, et en plus elle englobe les surfaces de Bézier et B-Spline comme des cas particuliers.

Chapitre II

Machine Outils et Usinage des Surfaces Gauches

I. INTRODUCTION :

L'usinage en général et sur machines outil à commande numérique « MOCN » en particulier nécessite la prise en compte de plusieurs éléments :

- Quelle est la pièce à fabriquer, dans quel matériau, à partir de quel brut ?
- Quelle quantité de pièces, en lots de combien de pièces, pour quand ?
- Quelle gamme va-t-on suivre ?
  - Phase (quelle machine ?)
  - Sous-Phases (quelle mise en positions ?)
  - Opérations (outils, stratégie d'usinage, paramètres de coupe, etc.)

Dans ce chapitre, nous allons présenter les différents composants d'une machine outil à commande numérique et en particulier la fraiseuse ainsi que la structure générale des programmes d'usinage et nous terminons par l'usinage des surfaces gauches.

II. MACHINE OUTIL A COMMANDE NUMERIQUE (MOCN) [9,10 ,14] :

La commande numérique d'une machine-outil est un processus impliquant :

- Une certaine automatisation du processus (outils multiples, axes asservis,... etc.)
- Une définition symbolique des commandes (un programme)
- Des mouvements outils/pièces définis numériquement

L'usinage par commande numérique était destiné à la réalisation de surfaces gauches.

II.1 Domaine d'utilisation de MOCN

Les MOCN sont employées dans de nombreux secteurs industriels et les principaux procédés de fabrication sont :

a. Perçage, taraudage
b. Tournage, alésage
c. Fraisage
d. Rectification, ... etc.

II.2 Avantages liés aux MOCN

Les avantages des MOCN sont les suivants :

a. Réalisation d'usinages impossibles sur les machines conventionnelles
b. Favorise les très petites séries et les pièces unitaires
c. Précision de l'usinage
d. Fidélité de reproduction

III. ETUDE ORGANIQUE D'UNE MOCN [10] :

La machine outil doit avoir la structure suivante :

- Les axes de la machine : qui assurent la mise en position de l'outil par rapport à la pièce et les mouvements d'avance.
- La broche : qui assure le mouvement de coupe donné à l'outil ou à la pièce en fonction de la machine.
- Un système de contrôle – commande : qui permet le bon fonctionnement des axes et des fonctions auxiliaires.
- Le bâti : qui assure le lien entre ces systèmes.

IV. CLASSIFICATION DES MACHINE OUTILS [10] :

Traditionnellement, les machines sont classées en fonction des formes de surfaces à réaliser. On a trois types de machine :

Machine point à point : seule la position est garantie (voir Fig.II.1).

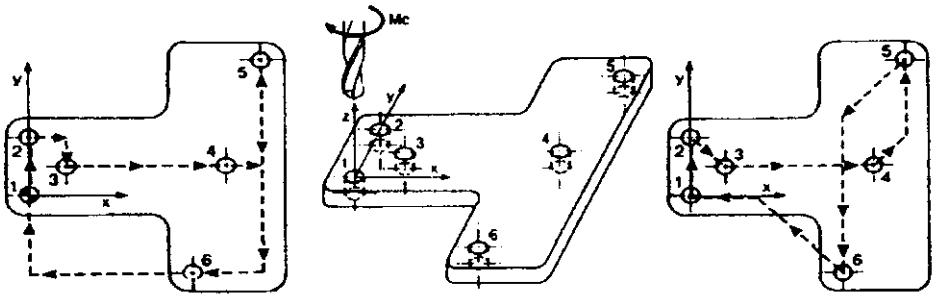


Fig.II.1. Opération point par point.

Machines paraxiales : à chaque instant, un unique axe de déplacement est asservi en vitesse et en position et donc un usinage dans des directions parallèles aux axes de la machine outil (voir Fig. 8).

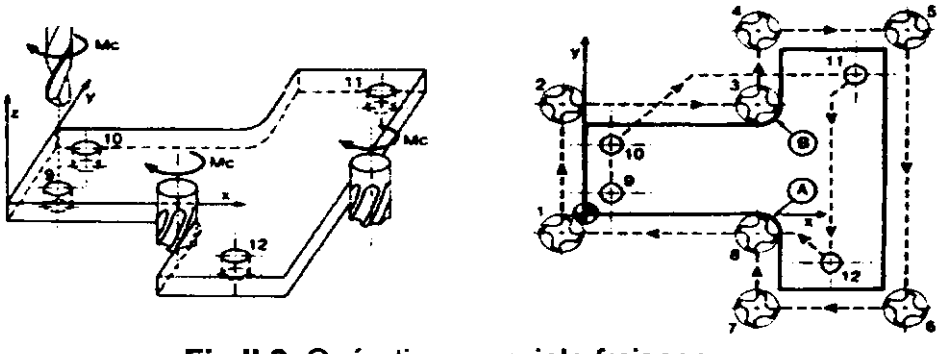


Fig.II.2. Opération paraxiale fraisage.

Machines en contournage : plusieurs axes sont asservis simultanément en vitesse et en position (usinage des surfaces complexes) (voir Fig.II.3).

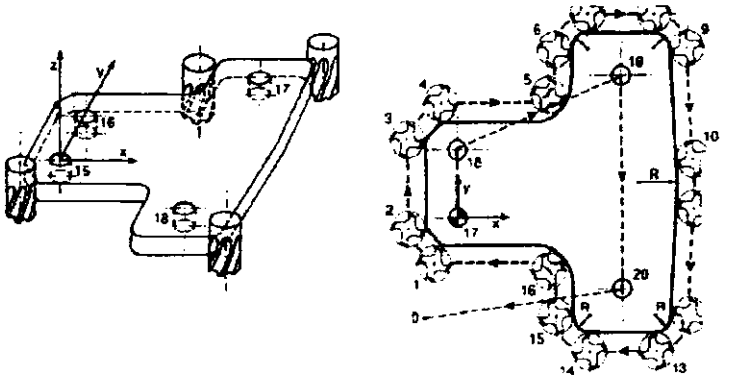


Fig.II.3. Opération contournage fraisage.

Dans les ateliers de mécanique, les machines les plus utilisées dans la fabrication des pièces de formes libres (moules, matrices, ...etc.) sont les fraiseuses.

V. PRESENTATION DES FRAISEUSES [9,12] :

Les fraiseuses sont des machines utilisées pour la réalisation des formes prismatiques et peuvent réaliser des opérations de contournage. L'outil (fraise) est fixé à une broche qui le fait tourner (mouvement de coupe) et peut se déplacer en translation par rapport à la pièce suivant trois directions (mouvement d'avance).

Les différentes parties d'une fraiseuse sont les suivantes :

a. Bâti : est la plateforme de la machine.
b. Broche : elle porte l'outil (fraise) et transmis ainsi le mouvement de rotation nécessaire à l'opération de fraisage.
c. Porte outil : il assure la liaison entre l'outil et la broche.
d. Outil d'usinage : l'outil utilisé dans le fraisage est appelé fraise.
e. Pupitre de commande : pour le dialogue avec la commande de la machine.
f. Armoire électronique : elle englobe tous les composants électroniques et des cartes de gestion.

VI. AXES DE DEPLACEMENT [10,12] :

Les axes de déplacement mettent en mouvement les parties mobiles des machines avec de fortes accélérations. Les axes sont constitués d'un guidage, d'un système d'entraînement, d'une motorisation et d'un système de mesure. Ces constitutions se sont reparties dans deux parties principales :

- Une partie opérative : qui est essentiellement de nature matériel.
- Une partie commande : constituée elle même de deux parties :
  1. Une partie matérielle : dispositifs électroniques (numérique et analogique).
  2. Une partie logicielle.

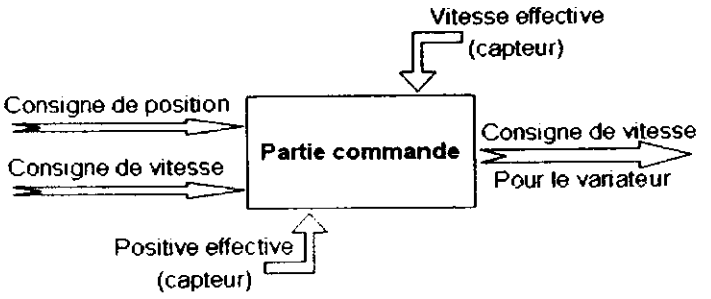


Fig.II.4. Schéma fonctionnel (entrée et sortie) d'un axe numérique [13].

Axes commandés [12] : on appelle axe commandé une liaison équipée d'une motorisation qui permet à un instant donné de fournir une valeur à la coordonné articulaire (voir Fig.II.4). La commande d'axe permet d'asservir en position et en vitesse le déplacement des mobiles. A chaque cycle de calcul, le directeur de commande numérique délivre une consigne de position. L'asservissement a pour fonction de faire suivre au mobile la succession de la position calculée.

VI.1. Définition normalisée des axes numériques [10,11] :

Le système normal de coordonnées est un système cartésien rectangulaire de sens direct et ayant des arêtes parallèles aux glissières principales de la machine. Il est désigné par les lettres X, Y, Z non munies du signe «+».

- **Axe de mouvement Z :** est l'axe du système normal parallèle à l'axe de la broche principale et peut être horizontal ou vertical en fonction en fonction du type de la machine.

- **Axe de mouvement X :** l'axe X est perpendiculaire à l'axe Z.

- **Axe de mouvement Y :** il forme avec les axes **X** et **Z** un trièdre de sens direct. La règle des trois doigts de la main droite permet de retrouver facilement l'orientation des axes X, Y et Z (voir Fig.II.5).

- **Mouvements de rotations A, B, C :** les angles **A**, **B** et **C** définissent les mouvements de rotation respectivement autour des axes **X**, **Y** et **Z.**

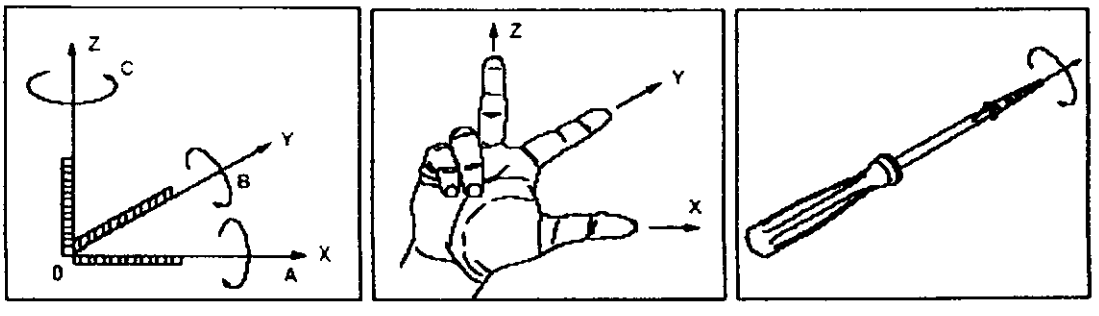


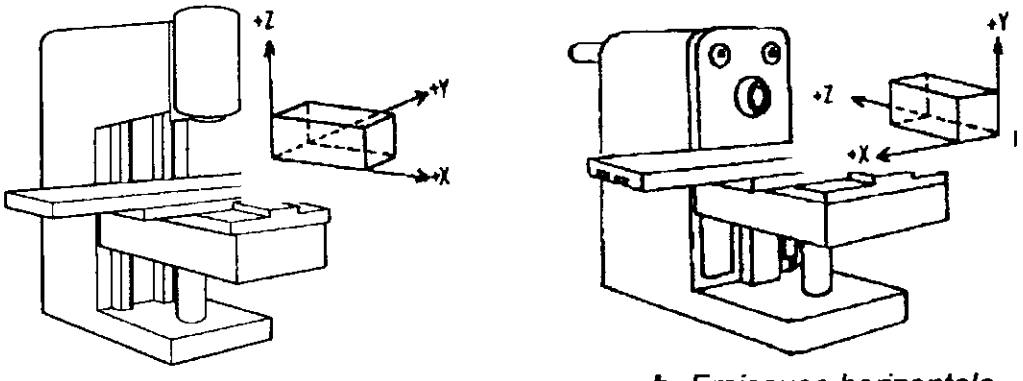
Fig.II.5. Définition et orientation des axes [12].

Les déplacements suivant les axes X, Y et Z définissent 3 degrés de liberté de translation. Par contre, les rotations autour des axes X, Y et Z définissent 3 degrés de liberté de rotation.

VI.2. Types de fraiseuses [9] :

On distingue deux types de fraiseuses selon la position des axes :

- **Fraiseuse verticale :** la broche est verticale (voir Fig.II.6.a).
- **Fraiseuse** horizontale : la broche est horizontale (voir Fig.II.6.b).



a. Fraiseuse verticale. *b. Fraiseuse horizontale.*

Fig.II.6. Types de fraiseuses et axes associés.

VI.3. Différentes architectures d'une fraiseuse [9, 11] :

Il existe plusieurs architectures de fraiseuses :

- Fraiseuse 3 axes broche verticale
- Fraiseuse 3 axes broche horizontale
- Fraiseuse 3 axes avec plateau d'indexation
- Fraiseuse à 4 ou 5 axes

Pour notre travail, nous allons considérer les fraiseuses verticales à 3 axes.

VI.4. Avantages des fraiseuses à commande numérique [9] :

Les principaux avantages des fraiseuses à commande numérique sont les suivants :

- Outil fiable et précis.
- Gestion automatique (chargement/déchargement automatique des outils).
- Travail continu et réduction du temps d'usinage.
- Souplesse, polyvalence et usinage de pièces complexes.

VII. PROGRAMMATION DES MOCN [12] :

Les MOCN sont des machines commandées par un programme écrit dans un langage bien défini et stocké dans une mémoire et exécuté par la machine.

VII.1. Définition d'un programme [11] :

Un programme est une suite d'instructions écrites dans un langage codé propre à la commande numérique décrivant les opérations d'usinage. Le programme d'usinage comporte deux types d'informations :

- des ordres de déplacements,
- des ordres auxiliaires.

On appelle « instruction » le plus petit élément d'un programme susceptible de donner lieu à une modification de [13] :

1. l'état physique de la machine commandée ;
2. son état logique.

VII.2. Modes de préparation des programmes d'usinage [13,9] :

Quelque soit le langage de programmation utilisé pour l'écriture des programmes pièces, le seul langage compréhensible par la machine est le langage ISO et le passage d'un langage de haut niveau au langage ISO est possible en utilisant un logiciel de traduction. La programmation peut être faite de deux manières :

- Manuelle.
- Assistée : pour ce cas, nous pouvons utiliser la méthode conversationnelle ou un logiciel de F.A.O.

VII.3. Programmation en langage ISO [9] :

On appelle le programme interprété et exécuté par la machine « G-Code » employé pour définir les fonctions d'usinage (déplacements, changement d'outil, vitesse d'avance, vitesse de broche, sens de rotation, ...etc.). Ce programme est constitué d'un ensemble de lignes appelées « Bloc ». Ce dernier est un ensemble d'instructions relatives à une séquence d'usinage. Un Bloc est constitué d'un ensemble de « Mots » qui est aussi composé d'un ensemble de caractères et de chiffres constituant une information (voir Fig.II.7).

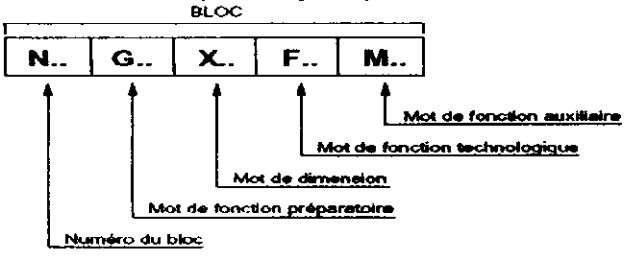


Fig.II.7. Format général des blocs [12].

VII.3.1. Format de mot [10,9] :

Le mot définit une instruction ou donnée à transmettre au système de commande (voir Fig.II.8). Il existe des mots définissant des dimensions et des mots définissant des fonctions.

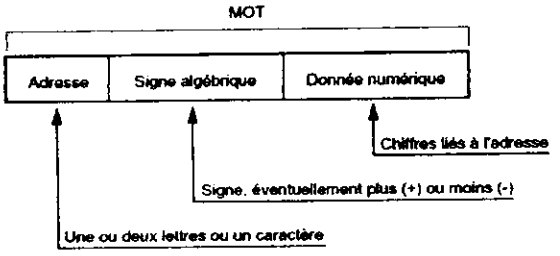


Fig.II.8. Format général des mots [12].

VII.3.2. Structure générale d'un programme [11] :

Un programme ISO comporte les éléments essentiels suivants (voir Fig.II.9) :

- Début de programme : caractère % suivi du numéro de programme et éventuellement d'un commentaire entre parenthèses.
- Ensemble des blocs à exécuter (programme effectif).
- Fin de programme : code M02.
- Fin de chargement de programme : caractère XOFF.

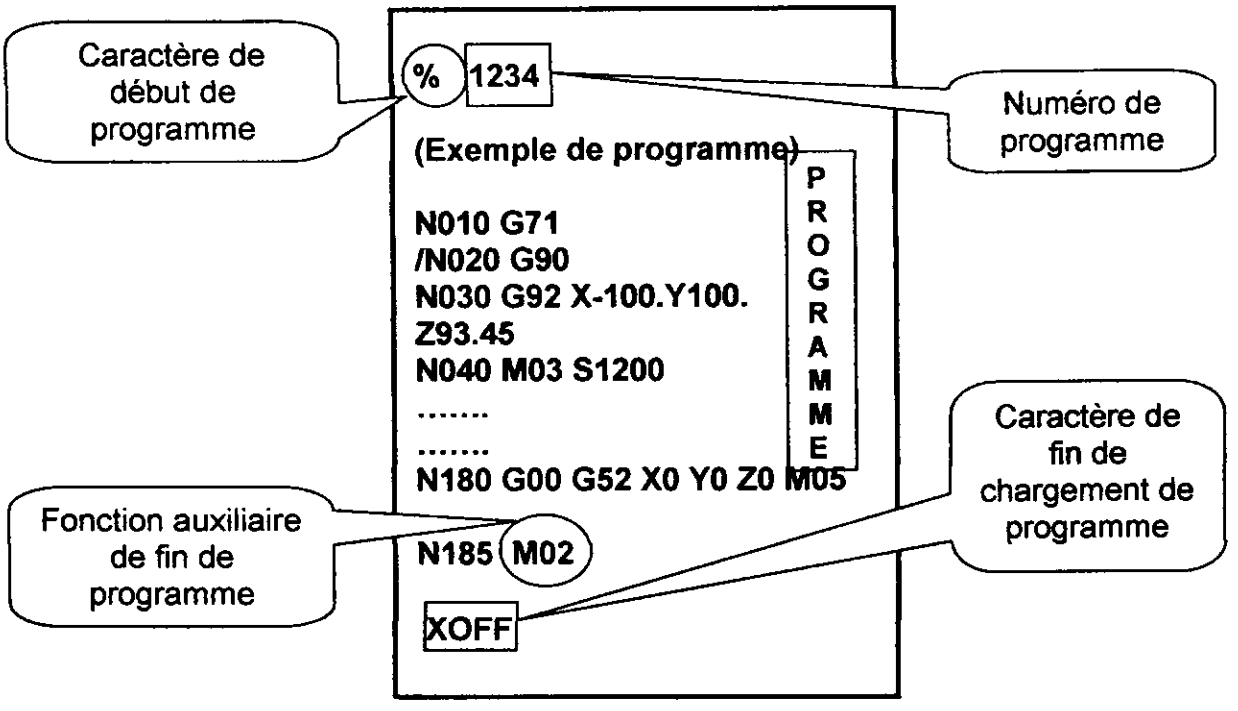


Fig.II.9. Structure d'un programme ISO [12].

VIII. USINAGE DES SURFACES GAUCHES :

Le processus d'usinage des surfaces gauches passe par plusieurs étapes en. Parmi ces étapes, nous avons l'étape de choix des outils et des conditions de coupe ainsi que l'étape de choix des stratégies d'usinage pour les opérations d'ébauchage, de demi finition et de finition.

VIII.1. APPROCHE ET STRATEGIE D'USINAGE DES SURFACES GAUCHES :

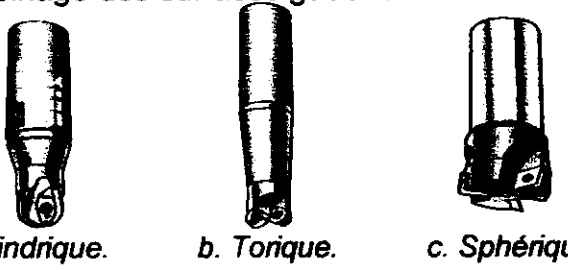
La complexité des surfaces qui existent dans le milieu industriel rend leur usinage très difficile. Les difficultés surgissent souvent quand on essaye de déterminer l'opération d'usinage dans la phase de programmation. Pour cela, la fabrication des pièces passe par plusieurs étapes et ceci en prenant en compte des contraintes géométriques des pièces ainsi que le taux d'enlèvement de la matière dans le processus d'ébauche et le mouvement d'outil de coupe avec la précision requise dans le processus de finition. Le processus d'usinage se traduit par la séquence d'opérations nécessaires pour l'enlèvement de la matière dans chaque plan d'usinage :

- ✓ Dans l'ébauche, utiliser de grandes fraises afin d'enlever le maximum de matière en minimum de temps.
- ✓ En semi finition, utiliser de grandes fraises hémisphériques pour supprimer les épaulements laissés sur la surface après la phase d'ébauche et pour avoir une forme qui se rapproche plus de la forme théorique.
- ✓ En finition, utiliser de petites fraises hémisphériques pour enlever les parties qui restent afin d'avoir la forme voulue.

VIII.2. LES DIFFERENTS PARAMETRES D'USINAGE :

VIII.2.1. Choix de l'outil :

Le choix de l'outil est le résultat d'un compromis entre la rigidité de l'outil, la cinématique de la machine et la forme de la pièce à usiner. Les principaux types d'outils utilisés dans l'usinage des surfaces gauches sont les suivants (voir Fig.II.10) :



a. Cylindrique. *b. Torique.* *c. Sphérique.*

Fig.II.10. Différents types de fraises.

Pour usiner une surface gauche il faut avoir un enchaînement de déplacements et une forme d'outil qui soit toujours tangente à la surface à usiner en chaque point de contact. La forme la plus simple est la sphère et donc l'outil hémisphérique est la solution pour tous les usinages du type 3 axes.

VIII.2.2.Positionnement de l'outil sur la pièce à usiner :

Pour usiner une surface gauche, l'outil utilisé est constamment tangent aux différents points de contact C_C (cutter contact) entre l'outil et la surface à usiner. Lors du calcul du trajet de l'outil on fait toujours référence au point centre outil C_E et c'est ainsi qu'on forme une enveloppe au dessus de la surface qui représente la trajectoire de ce point (voir Fig.II.11). La pointe de la fraise est le point C_L (cutter location).

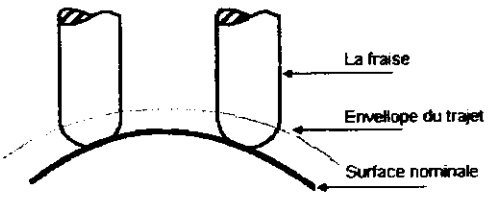


Fig.II.11. Enveloppe de la surface à usiner.

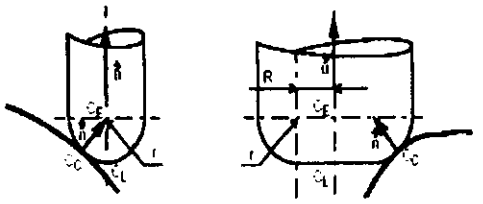


Fig.II.12. Position de la fraise par rapport à la surface.

En définissant « **r** » : rayon de l'outil. « **u** » : axe de rotation de la fraise. « n » : la normale de la surface au point de contact (voir Fig.II.12), les positions des points de l'outil hémisphérique sont données par :

$$\begin{cases} \overrightarrow{OC_E} = \overrightarrow{OC_C} + r\vec{n} \\ \overrightarrow{OC_L} = \overrightarrow{OC_E} - r\vec{u} = \overrightarrow{OC_C} + r\vec{n} - r\vec{u} \end{cases} \tag{2.1}$$

Les positions des points de l'outil torique sont données par :

$$\overrightarrow{K} = \frac{\vec{u} \wedge \vec{n}}{\left\| \vec{u} \wedge \vec{n} \right\|} \tag{2.2}$$

$$\begin{cases} \overrightarrow{OC_E} = \overrightarrow{OC_C} + r\vec{n} + R \cdot \dfrac{\vec{k} \wedge \vec{u}}{\left\| \vec{v} \wedge \vec{u} \right\|} \\ \overrightarrow{OC_L} = \overrightarrow{OC_E} - r\vec{u} = \overrightarrow{OC_C} + r\vec{n} - r\vec{u} + + R \cdot \dfrac{\vec{k} \wedge \vec{u}}{\left\| \vec{v} \wedge \vec{u} \right\|} \end{cases} \tag{2.3}$$

VIII.3. DEFINITON DE L'EBAUCHE :

L'ébauchage (dégrossissage) est l'étape de préparation ou l'en ne tient pas compte de la qualité d'usinage ni de la forme et ce qui nous importe le plus est l'enlèvement du maximum de matière en un temps réduit. L'opération d'ébauchage peut être faite par un outil cylindrique ou un outil hémisphérique (voir Fig.II.13 et Fig.II.14).

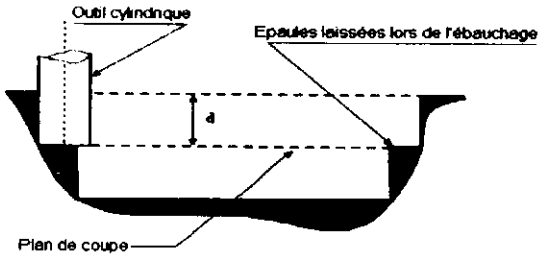
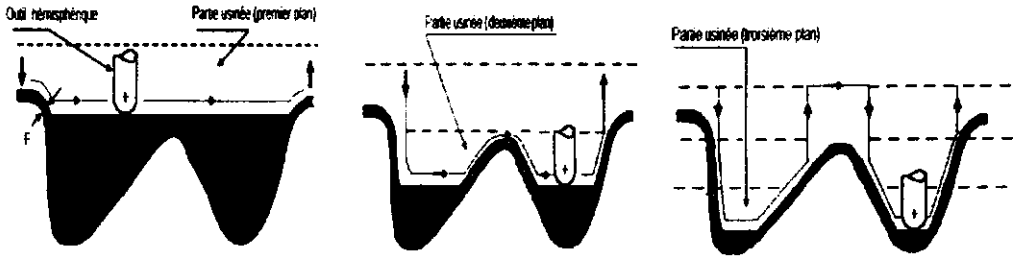


Fig.II.13. Ebauchage avec un outil cylindrique.



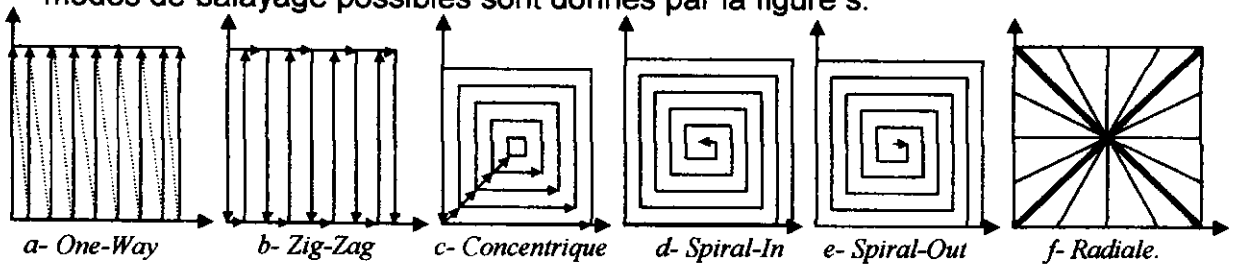
a. Premier plan usiné. b. Deuxième plan usiné. c. Troisième plan usiné.

Fig.II.14. Ebauchage avec un outil hémisphérique.

VIII.4. TECHNIQUES D'USINAGE DES SURFACES GAUCHES EN FINITION :

Pour usiner des surfaces gauches en finitions, plusieurs stratégies sont utilisées :

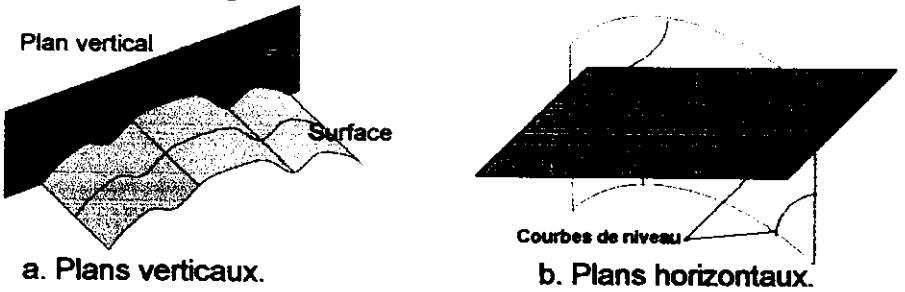
1. ***Stratégie d'usinage en isoparamétriques :*** pour ces stratégies, l'outil suit des courbes isoparamétriques ou des portions de courbes isoparamétriques. Les modes de balayage possibles sont donnés par la figure s.



a- One-Way *b- Zig-Zag* *c- Concentrique* *d- Spiral-In* *e- Spiral-Out* *f- Radiale.*

Fig.II.15. Usinage en isoparamétriques.

2. ***Stratégie d'usinage avec des plans parallèles :*** elle consiste à utiliser des plans parallèles verticaux (voir Fig.II.16.a) ou des plans parallèles horizontaux (voir Fig.II.16.b) pour calculer l'intersection de ces plans avec la surface et par la suite générer le trajet d'usinage.



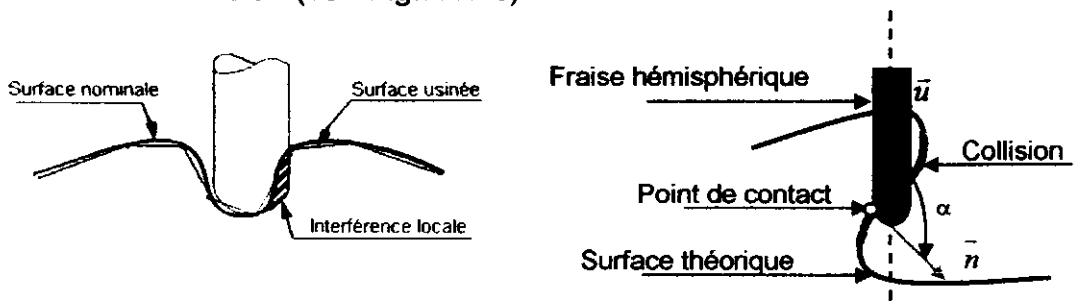
a. Plans verticaux. b. Plans horizontaux.

Fig.II.16. Usinage avec des plans parallèles.

VIII.5. PROBLEME D'USINAGE :

Le phénomène d'interférence entre l'outil et la surface se produit lorsqu'il existe une partie concave dont le rayon de courbure est inférieur à celui de l'outil, c'est l'interférence locale (voir Fig.II.17.a). Tandis que le problème de collision : ce genre de problème est observé dans les parties où l'angle formé par la normale à la surface

au point de contact et l'axe de l'outil est supérieur à 90°. Dans ce cas, l'outil et la surface sont en collision (voir Fig.II.17.b).



a. Interférence locale. **b.** Collision.

Fig.II.17. Interférence locale.

IX. CONCLUSION :

Dans ce chapitre, nous avons présenté la construction des fraiseuses à commande numérique et leurs modes de fonctionnement ainsi que la syntaxe générale des programmes d'usinage décrivant la trajectoire des outils. Dans la suite de ce mémoire, nous allons considérer uniquement les fraiseuses à commande numérique à 3 axes. Nous avons aussi présentés l'usinage des surfaces gauches et les problèmes rencontrés.

Chapitre III

Reverse Engineering
« Reconstruction »

I. INTRODUCTION

Dans les méthodes de conception des formes en *CAO*, on distingue deux voies possibles :

- Les méthodes directes qui consistent à utiliser les caractéristiques qu'offre un système de *CAO* pour la conception des formes. L'utilisateur est appelé dans ce cas à exploiter les logiciels *CAO* et leurs combinaisons pour obtenir la forme qu'il cherche à concevoir.
- Les méthodes indirectes qui consistent à traiter une représentation discrète (un nuage de points) d'un objet existant pour aboutir à sa forme avec des techniques de reconstruction de solides.

En pratique, la forme d'un objet (ou surface) nouveau (à concevoir) est dans l'imagination du concepteur, sa représentation mathématique et ses propriétés géométriques ne sont pas connues à priori. Le concepteur, peut se baser, pour construire cet objet, sur un nuage de points pour tendre vers la forme désirée.

II. METHODES ET MOYENS DE NUMERISATION [16] :

Un grand nombre de méthodes d'acquisition de formes, simples ou complexes, a été développé durant la deuxième moitié du siècle dernier. Celles-ci ne cessent de connaître des améliorations croissantes afin de les rendre de plus en plus performantes au sens précision et rapidité d'acquisition. L'éventail des objets numérisables est très large, il va de la numérisation des microformes jusqu'à la numérisation des reliefs par satellite. Essentiellement, chaque méthode use d'un mécanisme ou d'un phénomène physique afin d'interagir avec la surface de l'objet à numériser. Ces méthodes sont classées principalement en deux types (voir Fig.III.1) :

1. Les méthodes à contact.
2. Les méthodes sans contact.

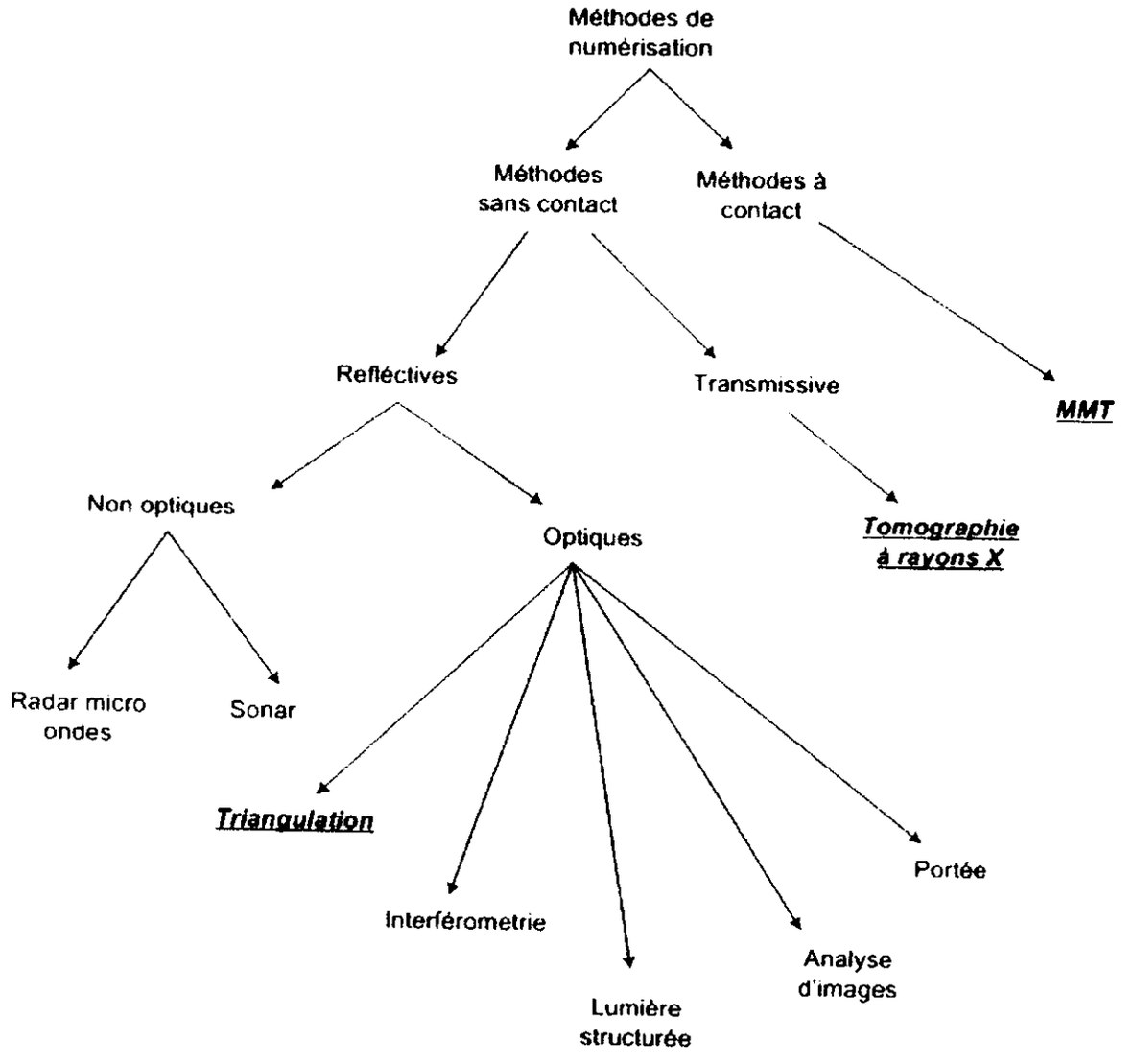


Fig.III.1. Techniques de numérisation.

III. REVERSE ENGINEERING « RECONSTRUCTION »

Le Reverse Engineering est une technique qui permet de donner une représentation numérique (mathématique) à un modèle physique existant, par l'intermédiaire des techniques appropriées. L'objet (ses caractéristiques de formes) est reconstruit à partir de sa représentation discrète, supposée copiée à l'aide d'une machine à mesurer tridimensionnelle « MMT » « un système de scanning ».

III.1. Processus de reverse engineering

Le processus de Reverse Engineering consiste généralement en trois étapes :

III.1.1. Acquisition du nuage des points (MMT) :

De nombreuses méthodes d'acquisition de données existent, elles sont classées en deux catégories :

- les méthodes sans contact (optique, laser...).
- les méthodes directes (palpage) telles que les machines à mesurer tridimensionnelle (MMT). Les méthodes directes procèdent par contact d'un palpeur sur les surfaces de l'objet à mesurer.

Machine à mesurer tridimensionnelle[17] :

La machine à mesurer tridimensionnelle est le moyen de mesure des formes simples et complexes le plus populaire. Avant l'avènement d'autres moyens de capture des surfaces, celle-ci représentait le seul moyen de numérisation connu. La mesure d'une surface par une MMT se fait point par point et un seul point à la fois. Ce qui fait qu'un tel système de mesure est très long et nécessite la programmation du chemin de scanning par un opérateur via le logiciel qu'utilise la machine. La précision de mesure est de l'ordre du micron, ce qui fait de la MMT l'instrument de numérisation le plus précis. Le nombre d'axes d'une telle machine varie de 03 au minimum jusqu'à 05 ou 06 axes. Un nombre important d'axes confère à la machine des possibilités de trajectoires variées mais pose en même temps le problème de leur optimisation. La partie terminale du stylet de mesure est constituée d'une pinule de mesure qui détecte le contact avec l'objet. L'évènement point est alors exécuté par le logiciel machine afin d'enregistrer les coordonnées du dit point.

Composition d'une machine à mesurer

Une machine à mesurer est composée :

- ❖ d'une mécanique constituée d'un marbre, de trois guidages rectilignes et d'une motorisation réalisée par des moteurs à courant continu.
- ❖ de trois règles de mesure constituées chacune d'une règle en verre ou en acier graduée par photogravure et d'un détecteur constitué de photodiodes.
- ❖ d'un coffret électronique permettant la commande numérique des déplacements de la machine, ainsi que le comptage et l'acquisition des valeurs des déplacements des détecteurs par rapport aux règles graduées.
- ❖ d'une tête de palpage dynamique ou statique qui établit une relation entre le contact physique du palpeur sur la pièce et la lecture des trois déplacements.

- d'un calculateur qui réalise trois fonctions : apporter une assistance à la mesure, gérer des déplacements par commande numérique de la machine et corriger par logiciel la géométrie de la machine (voir Fig.III.2).

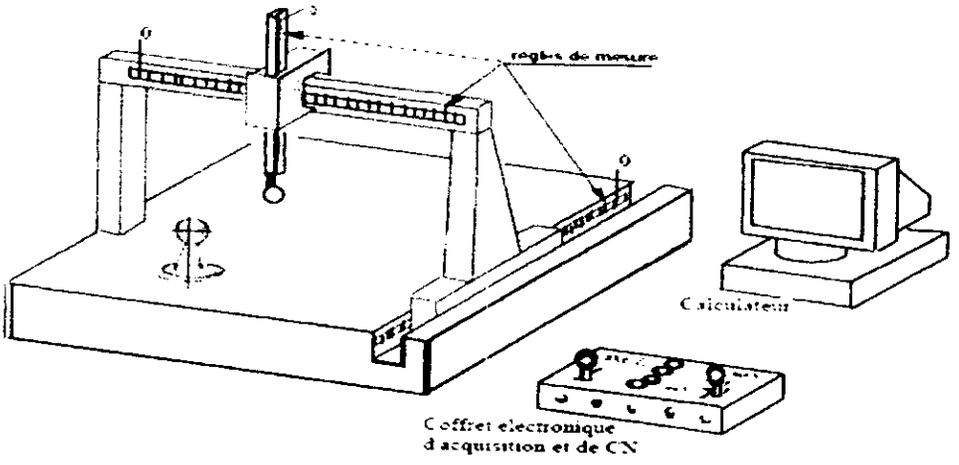


Fig.III.2. Machine à mesurer tridimensionnelle.

Tête de palpage dynamique :

C'est la tête de palpage la plus répandue, elle permet une mesure suivant toutes les directions. Elle est constituée d'une liaison isostatique réalisée entre deux pièces par un triplet de trois liaisons linéaires annulaires (liaison Boys). Un ressort réglable maintien les deux pièces de la liaison en contact. L'information qui déclenche l'acquisition de la mesure est obtenue par rupture du contact électrique établi entre les deux pièces de la liaison isostatique. La touche du palpeur étant escamotable cette tête de palpage ne permet que des mesures point par point (voir Fig.III.3).

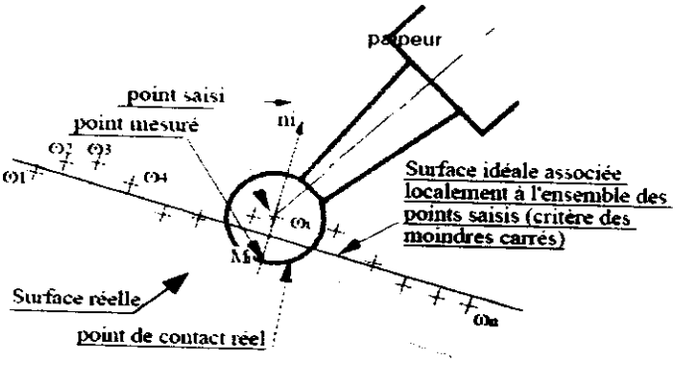


Fig.III.3. Mesure d'un point.

Calcul des points mesurés :

Le point de contact réel entre le palpeur et la surface mesurée étant inconnu, on lui substitue un point de contact estimé ou point mesuré. Ce dernier est calculé à partir des coordonnées du point saisi (centre du palpeur), du sens d'accostage et du rayon du palpeur. Pour cela, on fait l'hypothèse que le point de contact recherché est à l'intersection de la sphère de palpage et de la normale à la surface passant par le point saisi. La nature de la surface nominale étant connue le calcul du point mesuré peut-être le suivant :

- association d'une surface nominale passant au mieux des points saisis (suivant le critère des moindres carrés),
- calcul de la normale ni à la surface nominale, passant par le point saisi ωi et orientée vers l'extérieur de la matière (sens d'accostage),
- calcul des coordonnées du point mesuré Mi (ou point de contact estimé) données par la relation vectorielle :

$$OMi = O\omega i - r\,n \qquad (3.1)$$

Avec r : rayon de la sphère du stylet du palpeur. Dans le cas ou la surface nominale est de nature inconnue, la normale à la surface peut être fixée arbitrairement ou être estimée localement. Dans ce dernier cas, on mesure deux points supplémentaires proches du point saisi considéré, la normale est alors définie par le plan passant par les trois points.

Etapes permettant de mener à bien le contrôle et la mesure tridimensionnelle d'une pièce mécanique

Les opérations de contrôle et de mesure tridimensionnelle peuvent-être regroupés en deux gammes distinctes : une gamme de mesurage permettant de faire l'acquisition des points mesurés, et une gamme de traitement permettant par calcul : d'identifier, d'interpréter et de vérifier les spécifications géométriques. La gamme de mesurage est nécessairement exécutée sur une machine à mesurer tridimensionnelle, elle prend en compte toutes les précautions techniques habituelles qui sont liées à la qualité des mesures désirées. La gamme de métrologie s'appuie sur une base de données acquises lors du mesurage, elle peut donc être exécutée sur un calculateur indépendant de la machine à mesurer.

La méthodologie suivie pour concevoir ces deux gammes peut-être décrite en quatre étapes.

- ***Première étape : inventaire des éléments géométriques réels.***

L'analyse du dessin de définition doit permettre de faire l'inventaire des éléments géométriques concernés par les spécifications et d'en choisir les ensembles de points à mesurer les plus représentatifs. Il existe peu de règles permettant d'effectuer ces choix, l'expérience tient ici un rôle essentiel, on peut cependant énoncer les quelques règles suivantes :

- la nature des éléments géométriques doit être choisie dans une liste disponible dans le logiciel de mesurage (point, droite, cercle, plan, sphère, cylindre, cône).
- la nature des éléments géométriques doit tenir compte de l'étendue de la surface (par exemple choix entre un cylindre et un cercle), et de l'éloignement de la caractéristique géométrique recherchée par rapport à la surface (par exemple intersection de l'axe d'un cylindre de faible hauteur avec un plan éloigné de la surface du cylindre),
- le nombre de points doit être supérieur ou égal au nombre de paramètres de l'élément géométrique,
- la répartition des points doit se faire sur toute l'étendue de la surface et mettre en évidence les défauts de forme dus au mode de fabrication,
- l'algorithme des moindres carrés est sensible à une densité de points localement plus importante,
- le nombre de points doit permettre un compromis entre une bonne représentativité de l'élément, et un temps de mesure minimum.

- ***Deuxième étape : palpage des surfaces des lignes et des points***

Choix des palpeurs s : pour chaque orientation de palpage, le stylet est choisi de tel sorte que sa longueur reste suffisante pour atteindre toutes les surfaces, et que son diamètre de sphère de palpage reste inférieur à la plus petite des cavités de la pièce. Cependant le diamètre de la sphère du stylet doit être suffisamment grand pour éviter toute collision entre la tige du stylet et la surface palpée, en effet le stylet n'est jamais parfaitement aligné avec la direction générale de la surface (voir Fig.III.4).

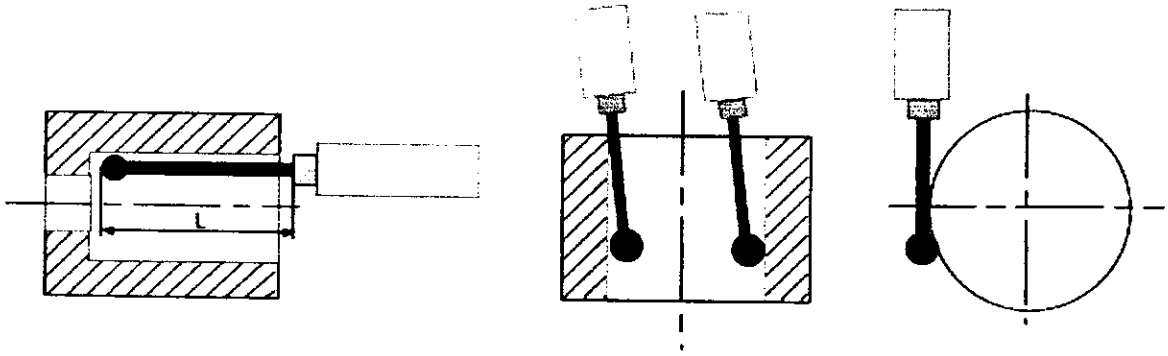


Fig.III.4. Choix des palpeurs.

Étalonnage d'un palpeur : L'opération d'étalonnage d'un palpeur a pour objectif d'estimer les deux caractéristiques géométriques nécessaires au calcul du recalage dans un repère unique des coordonnées des points de contact palpeur/pièce :

- un "rayon étalonné" de la sphère de palpage, qui sera obtenu par mesure de la dimension d'une bague ou d'une cale ou d'une sphère étalonnées,
- un "vecteur étalonné" représentant le décalage d'origine du à la variation de longueur et d'orientation des différents palpeurs. Il est obtenu par les coordonnées prises par le centre de la sphère de palpage lorsque celui-ci est mathématiquement confondu avec le centre d'une sphère de référence matérialisant l'origine des mesures.

Palpage des surfaces : à chaque contact entre le palpeur et la, on relève les coordonnées du centre de la sphère du stylet exprimées dans un repère de mesure.

- ***Troisième étape*** : constitution de la base de données des éléments géométriques réels (associés et mesurés).

L'information géométrique contenue dans la base de données d'une MMT s'appuie sur trois éléments géométriques de base : le point, la droite et le plan. Ils sont définis par les coordonnées d'un point et dans le cas d'une droite ou d'un plan par les composantes d'un vecteur unitaire. A partir des ensembles de points saisis on calcule les points de contacts avec les surfaces réelles ainsi que les paramètres des éléments géométriques associés aux points de contacts.

- ***Quatrième étape*** : Définitions normalisées des spécification , interprétation des spécifications, vérification des spécifications.

Cette étape nécessite une bonne connaissance des normes sur les spécifications ainsi qu'une bonne maîtrise des possibilités de calcul offertes par le logiciel de mesure tridimensionnelle. Par constructions géométriques, les logiciels permettent de définir à partir des éléments contenus dans la base de données de nouveaux éléments de types point, droite et plan ainsi que de construire des repères. A la fin de cette étape, on aura une représentation discrète de l'objet.

III.1.2. Arrangement des données acquises (segmentation) :

C'est l'enregistrement et le classement des informations acquises de façon à distinguer les surfaces de l'objet (plan, partie sphérique, partie cylindrique, ...etc.).

III.1.3. Reconstruction de solide :

C'est la dernière phase, elle consiste en la reconstruction de l'objet à l'aide de l'une des méthodes de reconstruction. A la fin de cette étape, on aura une représentation mathématique de l'objet (passage d'un état discret à un état continu).

III.2. Problématique de reconstruction des modèles CAO :

Dans notre étude, on considère que les données de localisation des points sur la (les) surface(s) de l'objet sont données par un modeleur CAO simulant l'opération de scanning, c'est-à-dire que le système dont on veut construire sa représentation est connu et la phase de mesure est supposée être accomplie dans de bonnes conditions. Donc, chercher à reconstruire l'objet, revient à trouver la représentation mathématique des surfaces qui le compose en respectant les contraintes de continuité au niveau des surfaces contiguës.

Il existe plusieurs méthodes pour la reconstruction des surfaces, mais, les méthodes polynomiales deviennent inefficaces lorsque le nombre des points représentant la forme est grand (>20) à cause des oscillations créées par le polynôme caractéristique et des exigences en temps et en mémoire nécessaires pour sa détermination, par contre, les B-Spline, Bézier et les NURBS ont des propriétés très intéressantes leur permettent de représenter un éventail très large de surfaces. De part les propriétés telles que le libre choix du degré utilisé, le contrôle local et l'enveloppe convexe, les B-Spline et les NURBS sont des outils très puissants en représentation leur permettant de tendre vers n'importe quelle forme voulue.

III.2.1. Interpolation globale des surfaces [5,17] :

Etant donnés un réseau de $(n+1)\times(m+1)$ points $Q_{k,l}$ $(0 \le k \le n$ et $0 \le l \le m)$. L'interpolation globale d'une surface consiste, connaissant les degrés (p,q) selon les deux directions u,v à construire une surface B-Spline définie par $(n+1)\times(m+1)$ points de contrôle inconnus $p_{i,j}$ $(0 \le i \le n, 0 \le j \le m)$ qui passe par les points de donnée $Q_{k,l}$. Les hypothèses précédentes se traduisent par l'équation suivante :

$$Q_{k,l} = S(\bar{u}_k, \bar{v}_l) = \sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(\bar{u}_k) N_{j,q}(\bar{v}_l) P_{i,j} \qquad (3.2)$$

Pour déterminer ces points de contrôle, il faut résoudre un système d'équation linéaires après le choix d'une méthode de calcul des paramètres $(\bar{u}_k)$ et $(\bar{v}_l)$.

III.2.2. Approximation globale des surfaces [5,17] :

Etant donnés $(n+1)\times(m+1)$ points $Q_{k,l}$ $(0 \le k \le n$ et $0 \le l \le m)$. La surface souhaitée étant supposée de degré p, q suivant les directions u et v. Reconstruire cette surface consiste à chercher S(u,v) définie par $(e+1)\times(f+1)$ points de contrôle (un choix du concepteur) $P_{i,j}$ avec $(l \le p \le e)$ et $l \le q \le f$ telle que :

$$Q_{k,l} = S(\bar{u}_k, \bar{v}_l) = \sum_{i=0}^{e}\sum_{j=0}^{f} N_{i,p}(\bar{u}_k) N_{j,q}(\bar{v}_l) P_{i,j} \qquad (3.3)$$

En général $S(u,v)$ ne passe pas par les points de donnée et le nombre des points de contrôle de $S(u,v)$ est inférieur ou égale au nombre des points de donnée (selon les deux directions). Les valeurs de la paramétrisation, les éléments du vecteur nodal et les matrices $N_{i,p}(\bar{u}_k)$ et $N_{j,q}(\bar{v}_l)$ étant supposés précalculés. Avec la sélection des points de contrôle $P_{i,j}$, l'approximation de la surface revient à minimiser la distance, au niveau des points connus entre la surface $S(u,v)$ et les points de donnée. Ceci revient à minimiser la fonction suivante au critère des moindres carrés :

$$\sum_{k=0}^{n}\left|\sum_{l=0}^{m}(S(u,v)-Q_{k,l})\right|^2 \qquad (3.4)$$

Qui s'exprime par :

$$F(P_{ij})=\sum_{k=0}^{n}\sum_{l=0}^{m}\left|\sum_{i=0}^{l}\sum_{j=0}^{f}N_{i,p}(u_k)N_{j,q}(v_l)P_{ij}-Q_{k,l}\right|^2 \qquad (3.5)$$

Ceci implique :

$$\frac{\partial F(P_{i,j})}{\partial P_{i,j}}=0 \qquad (3.6)$$

Remarque : la surface générée doit passer par les quartes coins du nuage de points pour mieux contrôler la forme de la surface. Cette condition est donnée par les relations suivantes :

$$\begin{cases} P_{0,0}=Q_{0,0} \\ P_{e,0}=Q_{n,0} \\ P_{0,f}=Q_{0,m} \\ P_{e,f}=Q_{n,m} \end{cases} \qquad (3.7)$$

IV. CONCLUSION :

Dans ce chapitre, nous avons commencé par la présentation du processus du Reverse Engineering et par la suite nous avons détaillés les différentes étapes nécessaires pour la reconstruction des surfaces à partir d'un nuage de point existant, en utilisant l'interpolation et l'approximation. L'interpolation et l'approximation globale des surfaces nous permettent d'avoir une allure très proche de la réalité des objets qu'on veut concevoir.

Chapitre IV

Triangulation de Delaunay

I. INTRODUCTION :

L'objectif final du processus du Reverse Engineering est la reconstruction du modèle informatique « Modèle CAO ». Cependant, cette étape est très délicate, fastidieuse et consommatrice de temps et dépend des fonctionnalités du logiciel utilisé ainsi que de l'expérience du concepteur. Afin de faciliter cette tâche, il est préférable d'approximer le nuage de points par un ensemble d'éléments géométriques simples (triangle, tétraèdre, ...etc.). C'est l'opération de « triangulation » où de très nombreux algorithmes ou techniques ont ainsi été développés pour construire une triangulation ou en améliorer une, mais la triangulation la plus utilisée est celle de Delaunay qui possède des propriétés géométriques remarquables et qui permet d'optimiser un certain nombre de critères. De plus, elle est l'une des rares triangulations définies de manière unique que l'on sache construire [15].

Dans ce chapitre nous allons présenter la triangulation de Delaunay, ces propriétés et les algorithmes de construction de cette triangulation.

II. TRIANGULATIONS GEOMETRIQUES :

Dans cette section vont être présentées les triangulations qui ne tiennent compte que de la nature géométrique des points donnés, à savoir leurs coordonnées spatiales. La triangulation de Delaunay, qui est à ce titre une référence, va être définie et ses diverses propriétés seront rappelées avec la présentation de quelques types d'algorithmes qui existent pour construire une triangulation. Nous allons étudier plus particulièrement les triangulations à partir du plan. Etant donné un nuage de points, nous présenterons des méthodes pour trouver une triangulation de leur enveloppe convexe avec des triangles les "meilleurs" possibles.

II.1. Triangulations De Delaunay [18] :

Les triangulations et particulièrement la triangulation de Delaunay ont été étudiées depuis longtemps et notamment dans le cadre de la géométrie algorithmique. Parallèlement la nécessité de modéliser sur ordinateur des phénomènes physiques a fait que les techniques de triangulation sont largement utilisées et abordées en analyse numérique, en mécanique et plus généralement dans les sciences de l'ingénieur. D'autres recherches se sont attelées à utiliser la triangulation de Delaunay en robotique, imageries et traitement d'image, CAO et plus récemment dans les techniques de reconstruction d'objet dans différents domaines des sciences.

II.1.1. Diagramme de Voronoï [19] :

Définition : soit S un nuage de points P_i , $i = 1,...,n$, en dimension d, le diagramme de Voronoï est l'ensemble des cellules ou polytopes V_i définis par :

$$V_i = \{P \quad tel que \quad d(P,P_i) \leq d(P,P_j), \quad \forall j \neq i \} \qquad (3.8)$$

Où $d(.,.)$ est la distance entre deux points induite par la métrique Euclidienne.

Une cellule, connue aussi sous le nom de cellule de Wigner-Seitz, est le lieu des points les plus proches de P_i que tout autre point de S. Les cellules V_i sont des polygones fermés convexes et elles recouvrent l'espace sans chevauchement et forment dans le plan la tesselation de Dirichlet et plus généralement le diagramme de Voronoï.

Propriété : le polygone de Voronoï d'un point *s* de S est non borné si et seulement si *s* n'est pas dans l'enveloppe convexe de S.

Définition : les polygones de Voronoï de tous les sites de S forment une partition du plan appelée diagramme de Voronoï de S. Cette partition du plan définit une notion de voisinage topologique :

Définition : deux points *s* et s_0 de S sont voisins dans le diagramme de Voronoï si leurs polygones associés ont une arête commune (voir Fig.IV.1).

Fig.IV.1. Diagramme de Voronoï.

Plusieurs définitions de la triangulation de Delaunay sont possibles; l'une d'elles utilise la notion de diagramme de Voronoï.

Définition : sous l'hypothèse que S est non dégénéré, la triangulation de Delaunay de S est le graphe dual du diagramme de Voronoï associé à S: deux sites s et s_0 seront reliés par une arête si et seulement s'ils sont voisins dans le diagramme de Voronoï (voir Fig.IV.2).

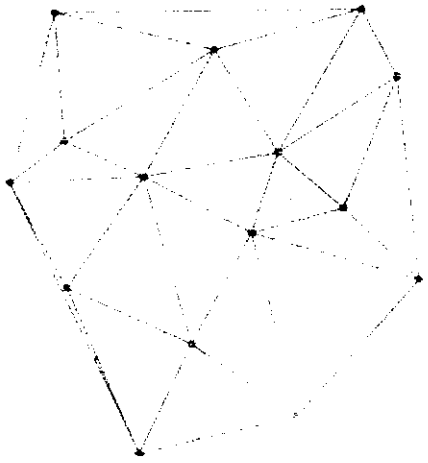


Fig.IV.2. Dualité de la triangulation de Delaunay et du diagramme de Voronoi.

II.1.2. Propriétés de la triangulation de Delaunay :

La triangulation de Delaunay revêt de nombreuses propriétés géométriques très utiles. Parmi les plus importantes on cite :

- ***1. Cercles circonscrits***

Parmi toutes les propriétés, le cercle circonscrit à un triangle jouera un rôle très important dans les algorithmes de triangulation comme nous le verrons plus tard. Ce cercle est défini par son centre et son rayon. Le calcul du centre et du rayon peut se faire essentiellement de deux manières, soit en utilisant directement l'équation du cercle, soit en calculant le point d'intersection de deux médiatrices du triangle.

Propriété : la triangulation de Delaunay de S est l'unique triangulation de S tel que le cercle circonscrit à chaque triangle ne contienne aucun site de S en son intérieur, propriété fondamentale appelée aussi "propriété de Delaunay" (voir Fig.IV.3).

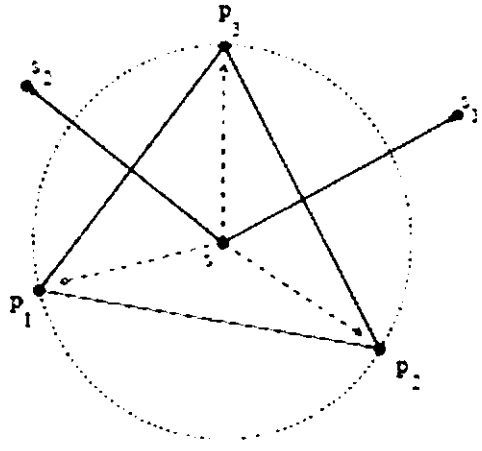


Fig.IV.3. Cercles circonscrits

- ***2. Maximum de l'angle minimum***

Une des propriétés les plus classiques de la triangulation de Delaunay est la maximisation de l'angle minimum de ses triangles (voir Fig.IV.4). L'utilité de cette propriété apparaît clairement dans les maillages pour la méthode des éléments finis où on a besoin de triangles ne comportant pas d'angles trop petits dans certaines applications. Ceci explique en partie la forte utilisation de Delaunay comme triangulation de base pour engendrer des maillages triangulaires.

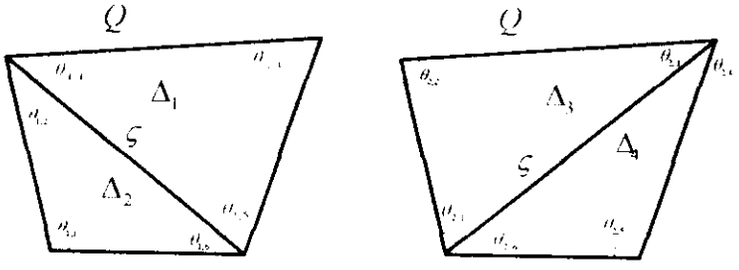


Fig.IV.4. Triangulations possibles d'un quadrilatere Q.

Propriété : soit S un ensemble de n points distincts du plan, dont quatre quelconques ne sont pas cocycliques, et soit T une triangulation de l'enveloppe convexe de S. Alors, toute arête intérieure de T qui satisfait le critère du maximum de l'angle minimum local est de Delaunay, et réciproquement.

- ***3. Équiangularité***

Concernant la triangulation de Delaunay dans le plan, une des propriétés essentielles est qu'elle est globalement équiangulaire. Ceci signifie que de toutes les triangulations possibles, celle de Delaunay donne des triangles les plus équilatéraux possibles. En pratique, ce résultat est essentiel, surtout pour l'utilisation de la triangulation de Delaunay dans la méthode des éléments finis.

- ***4. Enveloppe convexe***

Propriété : les arêtes extérieures de la triangulation de Delaunay d'un ensemble de points S constituent la frontière de l'enveloppe convexe de S.

- ***5. Graphe de Gabriel*** [20] :

Définition : le graphe de Gabriel d'un ensemble S de points $s_1, s_2, \ldots, s_n$ est le graphe dans lequel $\overline{s_i s_j}$ est une arête si et seulement si le cercle ayant $\overline{s_i s_j}$ pour diamètre est un cercle vide (voir Fig.IV.5)

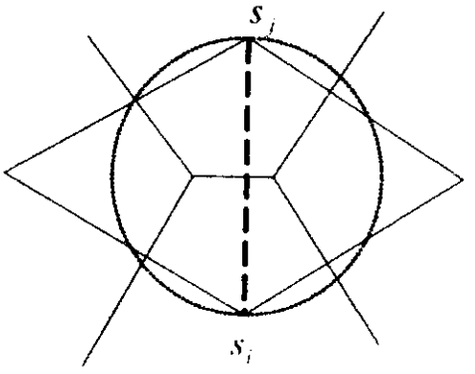


Fig.IV.5. Une arête du graphe de Gabriel

- ***6. Qualité des triangles générés*[21] :**

La qualité utilisée pour les triangles exprime le rapport du rayon de la sphère inscrite au triangle sur le rayon de sa sphère circonscrite par (voir Fig.IV.6):

$$Q_T = \gamma \frac{R_i}{R_c} \tag{3.9}$$

Avec $\gamma = 2$ le coefficient de normalisation par rapport à un triangle équilatéral.

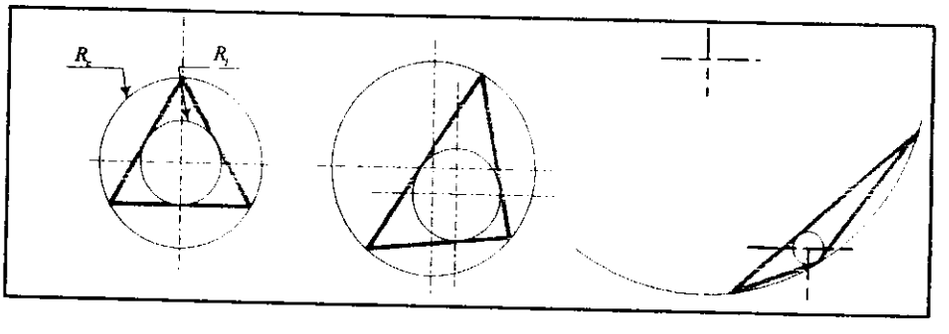


Fig.IV.6. Evolution de la qualité d'un triangle.

Le rayon du cercle inscrit à un triangle est donné par la relation suivante :

$$R_c = \frac{\sqrt{s(s-a)(s-b)(s-c)}}{s} \tag{3.10}$$

Avec $s = \frac{1}{2}(a+b+c)$ est le demi périmètre du triangle.

Le rayon du cercle circonscrit à un triangle est donné par la relation suivante :

$$R_i = \frac{abc}{4\sqrt{s(s-a)(s-b)(s-c)}} \tag{3.11}$$

Par définition, la qualité d'un triangle équilatéral est égale à 1 et son demi périmètre vaut $s = \frac{3a}{2}$, sachant que $a = b = c$.

Le rayon du cercle circonscrit à un triangle sera donné par :

$$R_c = \frac{a}{2\sqrt{3}} \qquad (3.12)$$

Le rayon du cercle inscrit à un triangle sera donné par :

$$R_i = \frac{a}{\sqrt{3}} \qquad (3.13)$$

Ce qui donne finalement :

$$Q_T = \gamma \frac{R_i}{R_c} = \gamma \frac{1}{2} = 1 \Leftrightarrow \gamma = 2 \qquad (3.14)$$

III. ALGORITHMES DE CONSTRUCTION DE LA TRIANGULATION DE DELAUNAY :

III. 1 Algorithmes à insertion incrémentale [21] :

Ce sont les algorithmes les plus simples et ceux pour lesquels la bibliographie est la plus importante. Leur principe est d'ajouter les points les uns après les autres et de mettre à jour la triangulation après chaque ajout. Les deux étapes d'un tel algorithme sont :

1. La localisation qui doit permettre de trouver le triangle contenant le point à insérer en supposant disposer au départ triangle infini qui contient tous les points du plan. Si le point se trouve en dehors de l'enveloppe convexe, il suffit de réactualiser celle-ci. Le problème est un peu différent lorsque le point est à l'intérieur de l'enveloppe convexe. Il faut alors rechercher le triangle qui le contient.

2. La mise à jour, qui peut être réalisée en échangeant les arêtes invalidées par l'insertion du nouveau point. Une arête peut être testée en appliquant le critère du cercle vide; à chaque fois qu'elle est invalidée, elle est échangée avec l'autre diagonale du quadrilatère où elle était située, et de nouvelles arêtes sont ajoutées à la liste de celles qui doivent être testées.

1. Approche de Watson :

Au lieu de procéder par échanges d'arêtes dans la phase de mise à jour, une autre méthode consiste à détruire tous les triangles en conflit avec le nouveau point, ces triangles sont connus par le critère du cercle vide. Les triangles ainsi supprimés forment un polygone étoilé que l'on triangule en reliant le nouveau sommet à chaque arête de sa frontière. Cette méthode est due à Watson et est reprise dans plusieurs articles. La figure Fig.IV.7 illustre les deux cas qui peuvent se produire lors de l'insertion d'un point pour cette approche.

2. Approche randomisée :

Dans ce type d'algorithme, les points sont insérés dans un ordre aléatoire, cette approche a été utilisée par Boissonnat et Teillaud. Ces derniers ont ainsi prouvé que le nombre moyen de swaps (basculement d'arêtes) est linéaire quelle que soit la distribution des points. Pour la phase de localisation, une structure d'arbre est utilisée, à savoir l'arbre de Delaunay, dans laquelle un nœud interne est un triangle qui a été détruit par l'insertion d'un point à un certain moment. Les feuilles de l'arbre contiennent la triangulation courante. Situer un point p revient alors à localiser p dans toutes les triangulations ayant existé successivement.

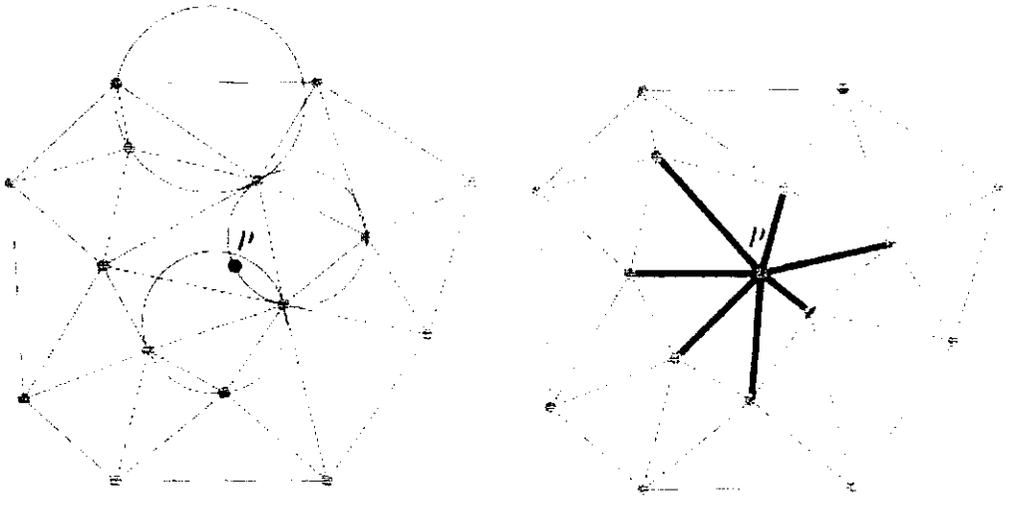


Fig.IV.7. Approche de Watson après insertion d'un nouveau point.

3. Algorithme par bascule de diagonales [23]:

Etant donne une triangulation $T(S)$ des points de S, on peut obtenir une autre triangulation $T'(S)$ en basculant une diagonale. Cette opération de bascule consiste à choisir deux triangules pqr et rqs adjacents dans $T(S)$ tel que le quadrilatère $pqsr$ soit convexe et on enlève alors l'arête qr de la triangulation et on ajoute l'arête ps (voir Fig.IV.8). En utilisant suffisamment d'opérations de bascules, on peut ainsi passer de n'importe quelle triangulation de S à n'importe quelle autre. Etant donné une triangulation quelconque, on peut alors obtenir la triangulation de Delaunay par une suite de flip (bascule). En fait, si pqr et rqs sont deux triangules

on va flipper (basculer) l'arête qr si et seulement si le cercle passant par p, q et r contient s. Si l'on poursuit cette opération de flip jusqu' à ce que plus aucune arête de la triangulation ne corresponde au critère on obtient une triangulation localement de Delaunay et donc la triangulation de Delaunay. Lawson a prouvé que quel que soit l'ordre des échanges considéré, en utilisant le critère du cercle vide, on converge toujours vers la triangulation de Delaunay (voir Fig.IV.8). pour construire la triangulation initiale, on peut choisir par exemple la triangulation de poids minimum ou tout autre triangulation généralement obtenue en insérant les points les uns après les autres et en les reliant aux sommets visibles déjà présents.

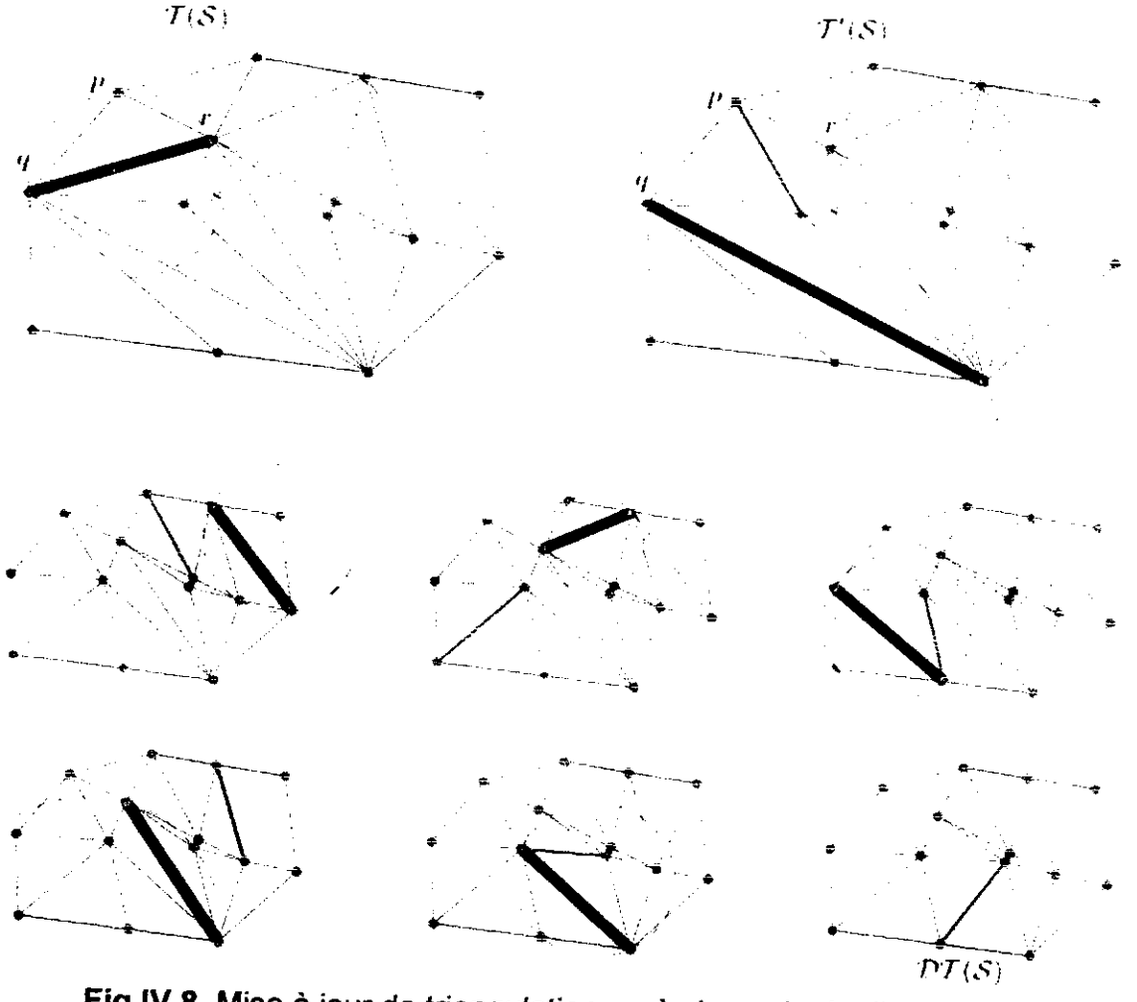


Fig.IV.8. Mise à jour de triangulation après bascule de diagonales.

4. *Algorithme incrémentale avec marche :*

Si les points sont bien reparti dans le plan on peut accélérer la recherche des triangles. On remarque que l'ensemble des triangules à détruire est connexe et que le triangule qui contient le nouveau point en fait partie. On peut donc chercher le triangule qui contient p, puis ne faire des tests de cocircularité que pour des triangles adjacents à des triangles dont le cercle a déjà été reconnu comme contenant p. Si k et le nombre de triangles détruits, le nombre de tests de

cocircularité effectués est alors $(k-1)$ avec réponse positive (le triangle contenant le point n'a pas besoin d'être testé) et $(k+2)$. avec réponse négative, le degré de p après son insertion $(k+2)$. Pour des points répartis aléatoirement, la valeur moyenne de k est 4, on fait donc en moyenne 9 tests de cocircularité. Une «marche» dans la triangulation permet de trouver le triangle contenant p (voir Fig.IV.9). Si on connaît le triangle contenant un point w alors on se propage dans la triangulation en utilisant les liens de voisinage entre les triangles et en visitant tous les triangles traversés par le segment wp. Si les triangles sont bien répartis, on visite ainsi $O(\sqrt{n})$ triangles. Une telle marche n'utilise que des tests d'orientations de trois points. La complexité totale d'un algorithme incrémentale de construction de Delaunay base sur une telle localisation est donc de $O(n\sqrt{n})$ si les points sont répartis aléatoirement dans le plan. Plus précisément on utilisera $O(\sqrt{n})$ tests de cocircularité et $O(n\sqrt{n})$ tests d'orientations de trois points.

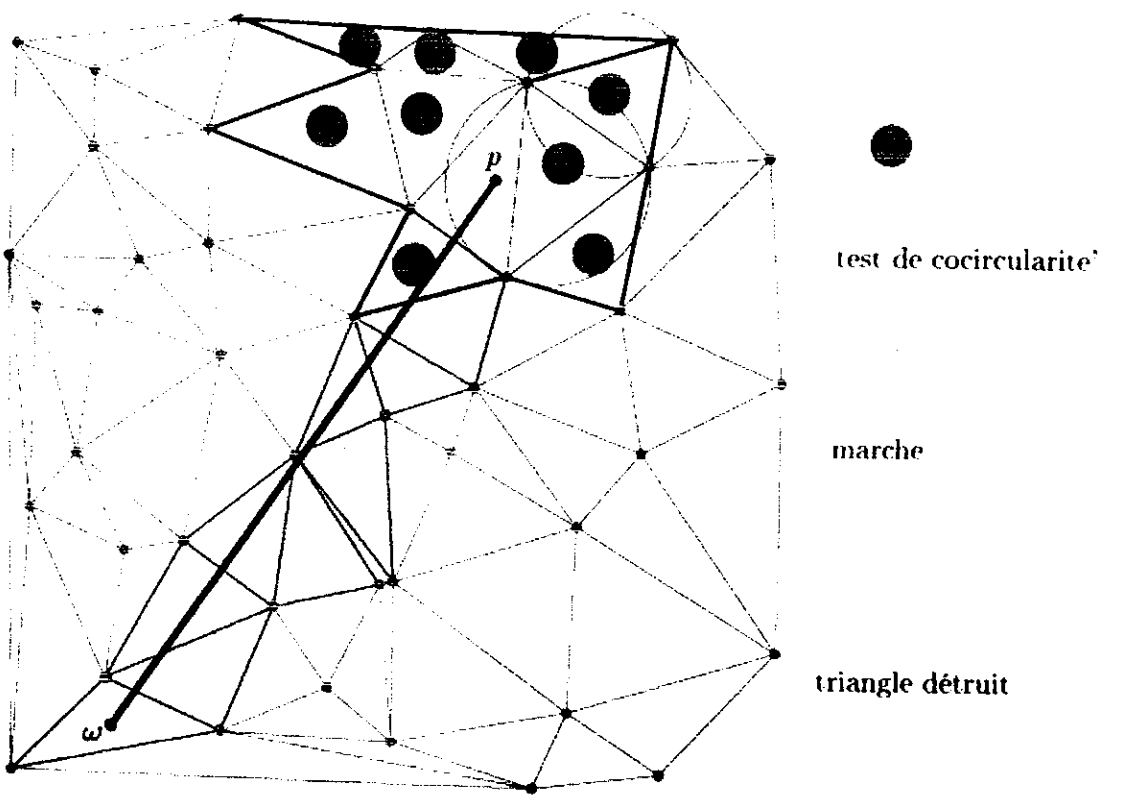


Fig.IV.9. Mise à jour de la triangulation avec marche.

IV. Conclusion :

Dans ce chapitre, nous nous sommes intéressé à la présentation de la triangulation de Delaunay, ces propriétés et les différents algorithmes permettant d'obtenir cette triangulation. Dans notre travail nous allons utiliser l'insertion séquentielle, puis l'insertion aléatoire avec utilisation de bascule de diagonales pour la mise à jour de la triangulation.

Chapitre V

Conception de l'Application

I. INTRODUCTION :

Dans ce chapitre nous allons présenter la conception de notre application avec le langage de modélisation UML en utilisant cinq diagrammes de l'ensemble des diagrammes de ce langage en détaillant le diagramme de classe, diagramme de cas d'utilisation, diagramme de collaboration et en finissant par les diagrammes d'activité et les diagrammes de séquence.

II. METHODE DE MODELISATION [1,2] :

Pour le développement de notre application logicielle, nous avons utilisé le Builder C++ version 6 comme langage de programmation. Nous avons aussi utilisé la bibliothèque graphique OpenGL pour gérer l'affichage et les quelques transformations géométriques. Nous avons utilisé le modèle orienté objet et le langage de modélisation UML (*Unified Modeling Language*) qui est un langage graphique qui permet de représenter, visualiser, construire un système informatique vu qu'il est compatible avec toutes les techniques de développement des applications et indépendant du langage de programmation.

III. REALISATION DE L'APPLICATION :

Pour réaliser notre application, nous devons passer par plusieurs étapes :

III.1. Cahier de charges :

Pour structurer les besoins de notre système, nous allons créer un cahier de charges.

III.1.1. Présentation du projet :

Notre projet s'insère dans le cadre de développement d'outils de conception et de fabrication des surfaces gauches initiés par l'équipe Conception et Fabrication Assistées par Ordinateur (CFAO) au niveau de la Division Productique et Robotique du centre de développement des technologies avancés (CDTA).

Dans ce projet on s'intéressera à l'approximation d'un nuage de points par un ensemble de triangles en utilisant la triangulation de Delaunay, détermination de la forme locale de chaque point, groupement des points en des régions distinctes et en dernier lieu la détermination du rayon d'outil optimum pour chaque région permettant d'éviter les problèmes d'interférence. La finalité de ce travail est le développement d'une application logicielle graphique et interactive sous Windows permettant d'automatiser ce processus.

III.1.2. Problématique :

Les pièces de formes gauches sont utilisées dans la conception et la réalisation des moules, des matrices, des formes esthétiques, ...etc. En raison des formes géométriques très complexes, ces pièces sont usinées sur des fraiseuses à commande numérique à 3, 4 et 5 axes après la génération de la trajectoire d'usinage à partir des modèles CAO des formes des surfaces à usiner. Ces modèles peuvent être obtenus par deux méthodes. La première méthode se base sur l'utilisation d'un logiciel de CAO. Cependant, il n'est pas toujours évident d'obtenir les formes voulues. Dans le cas où les formes des pièces sont très complexes et ne peuvent pas être conçues dans un logiciel de CAO ou le modèle CAO n'est pas disponible, une deuxième méthode est utilisée « Reverse Engineering » qui repose sur l'acquisition d'un nuage de points avec une machine à mesurer tridimensionnelle « MMT », et par suite faire des traitements appropriés pour générer le trajet d'usinage. Avant la génération du trajet d'usinage, une étape très importante qu'il faut considérer c'est l'étape de choix des formes et des dimensions des outils à utiliser pour minimiser les temps d'usinage et pour éviter les problèmes d'interférences et de collisions. Dans le cas où les modèles CAO sont donnés, ce choix est simplifié par la possibilité de calculer les propriétés géométriques des surfaces, mais dans le cas où les modèles CAO ne sont pas disponibles et la surface est représentée par un nuage de points très dense, cette tâche devient fastidieuse, très complexe et prend beaucoup de temps.

III.1.3. Objectifs visés :

L'objectif principal de notre travail est la subdivision d'un nuage de points en des régions distinctes en fonction des formes locales et détermination des outils hémisphériques optimums pour chaque région. Cet objectif peut être subdivisé en plusieurs sous objectifs :

- **Premier objectif :** approximation du nuage de points des surfaces à usiner par des éléments géométriques simples (triangles). Dans notre travail nous avons considéré deux cas. Dans le premier cas, le nuage de points est généré à partir du modèle CAO de l'ensemble des surfaces afin de simuler l'opération de digitalisation. Dans le deuxième cas, nous avons le nuage de points sans le modèle CAO de l'objet.

  Pour le premier cas, les triangles sont générés de la manière suivante :

  - Pour déterminer les points de la surface à usiner nous déterminons d'abord ses points dans le plan paramétrique (u, v) tel que le nombre des points est introduit par l'utilisateur puis nous faisons appel à une fonction qui permet d'associer à chaque point sur le plan paramétrique P(u,v) un et un seul point sur la surface dans l'espace F(x,y,z).

$$P(u, v) \xrightarrow{\text{Fonction de transformation}} F(x, y, z)$$

- ❖ Détermination d'un certain nombre de points avec une distribution uniforme des points dans le plan paramétrique.
- ❖ Détermination d'un certain nombre de points avec une distribution aléatoire des points dans le plan paramétrique.
- ❖ Approximation du nuage de points par un certain nombre de triangles à partir des points déterminés.
- ❖ Détermination de la bonne approximation de la surface après filtrage des triangles.

Pour le deuxième cas, les triangles sont générés de la manière suivante :

- ❖ Lecture des coordonnées des points du nuage de points à partir d'un fichier de donnée.
- ❖ Approximation du nuage de points par un certain nombre de triangles à partir des points introduits.
- ❖ Détermination de la bonne approximation de la surface après filtrage des triangles.

➢ **Deuxième objectif :** détermination de la forme locale des points. Cet objectif est aussi divisé en sous objectifs :

- ❖ Estimation du vecteur normal en chaque point.
- ❖ Détermination de l'équation du plan tangent en chaque point.
- ❖ Détermination des points voisins de chaque point.
- ❖ Détermination de la position des points voisins par rapport au plan tangent.
- ❖ Affectation de la forme locale (concave, convexe, concave développable, convexe développable, selle de cheval, plane) au point considéré.

➢ **Troisième objectif :** subdivision des triangles en différentes régions (concave, convexe, concave développable, convexe développable, selle de cheval, plane). Pour cela nous passons au groupement des triangles de même forme locale dans la même région. Dans notre travail nous avons supposé que si deux sommets sont du même type alors le triangle considéré est de ce type.

- **Quatrième objectif :** choix des outils optimums pour chaque région afin d'éliminer les interférences. Cet objectif est aussi divisé en sous objectifs :
  - ❖ Détection des interférences en chaque point.
  - ❖ Correction du rayon d'outil en chaque point jusqu'à absence d'interférence.
  - ❖ Choix pour chaque région, le plus petit rayon d'outil pour l'ensemble des points appartenant à cette région.

III.1.4. Plateforme exigée :

- Recommandations matérielles (configuration minimale nécessaire) :
  1. Intel Pentium 4 de 3 GHz ou équivalent.
  2. Une RAM de 1Go minimum.
  3. Carte accélératrice 3D de 128 Mo compatible OpenGL ou équivalente.
  4. Lecteur de CD-ROM/DVD-ROM.
  5. Souris PS/2.
  6. Clavier.
  7. Ecran 19 pouces.
- Système d'exploitation : Windows XP
- Logiciels requis :
  1. Builder C++ V.6.
  2. Bibliothèque graphique OpenGL.

III.2. Solution de la problématique :

Pour atteindre les objectifs fixés précédemment, nous allons développer une application logicielle graphique et interactive sous Windows. Pour cela, nous devons passer par les étapes suivantes :

- Développement des algorithmes d'approximation du nuage de points par des triangles en utilisant triangulation de Delaunay.
- Développement des algorithmes d'identification de la forme locale de chaque point.
- Développement des algorithmes de groupement des triangles en des régions distinctes.

- Développement des algorithmes de détection et de correction des interférences.
- Développement des algorithmes de choix des outils optimums.

III.3. Modélisation de l'application en UML :

Pour exprimer le fonctionnement du système, nous allons utiliser dans cette modélisation les cinq diagrammes de l'ensemble des diagrammes du langage UML. Nous allons détailler le diagramme de cas d'utilisation, les diagrammes de séquence, les diagrammes d'activité et le diagramme de classes.

Le cycle de vie d'un logiciel désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition. Pour cela, nous allons utiliser le cycle de vie en cascade en commençant par l'analyse, la conception, l'implémentation, les tests et la validation et enfin la maintenance.

III.3.1 Diagramme cas d'utilisation : est un diagramme qui décrit ce que doit faire le système sans spécifier comment il le fait. Les cas d'utilisation permettent :

- ✓ De connaître le comportement du système sans spécifier comment ce comportement sera réalisé.
- ✓ De définir les limites précises du système.
- ✓ Au développeur de bien comprendre l'attente des utilisateurs et les experts du domaine.

Le diagramme cas d'utilisation est caractérisé par :

- **Acteurs :** un acteur représente une personne ou un périphérique qui joue un rôle (interagit) avec le système. Pour notre cas les acteurs sont des ingénieurs mécaniques.
- **Cas d'utilisation :** représentent une fonctionnalité (un objectif à atteindre) du système à construire. Ils sont en relation avec des acteurs et d'autres cas d'utilisation.
- **Relations :** généralement orientées.

Dans ce qui suit, nous allons présenter les principaux cas d'utilisations. Pour notre cas nous avons deux possibilités : la première est la triangulation d'un nuage des points générés à partir de modèles CAO des surfaces tandis que la deuxième est la triangulation à partir d'un fichier contenant les coordonnées des points.

Pour le premier cas, nous commençant par « l'ouverture du modèle CAO » que nous voulons usiner par une action d'un acteur (utilisateur du système), qui doit intervenir dans le choix des paramètres de génération des points. Pour le deuxième cas, c'est « Ouverture d'un fichier » pour lire les cordonnées géométriques des points.

A partir de ces points, nous passons à la « génération des triangles » avec l'option de filtrage des triangles. Cette étape est suivie par le « calcul des normales » et « affectation de la forme locale pour chaque sommet ». Cette dernière permet de grouper les points en des régions distinctes et par la suite la « création des formes locales des triangles » (concave, convexe, concave développable, convexe développable, selle de cheval, plane). La dernière étape englobe la détection, la correction des rayons d'outils et finalement la détermination du rayon d'outil optimum pour chaque région (forme locale) (voir Fig.V.1.) .

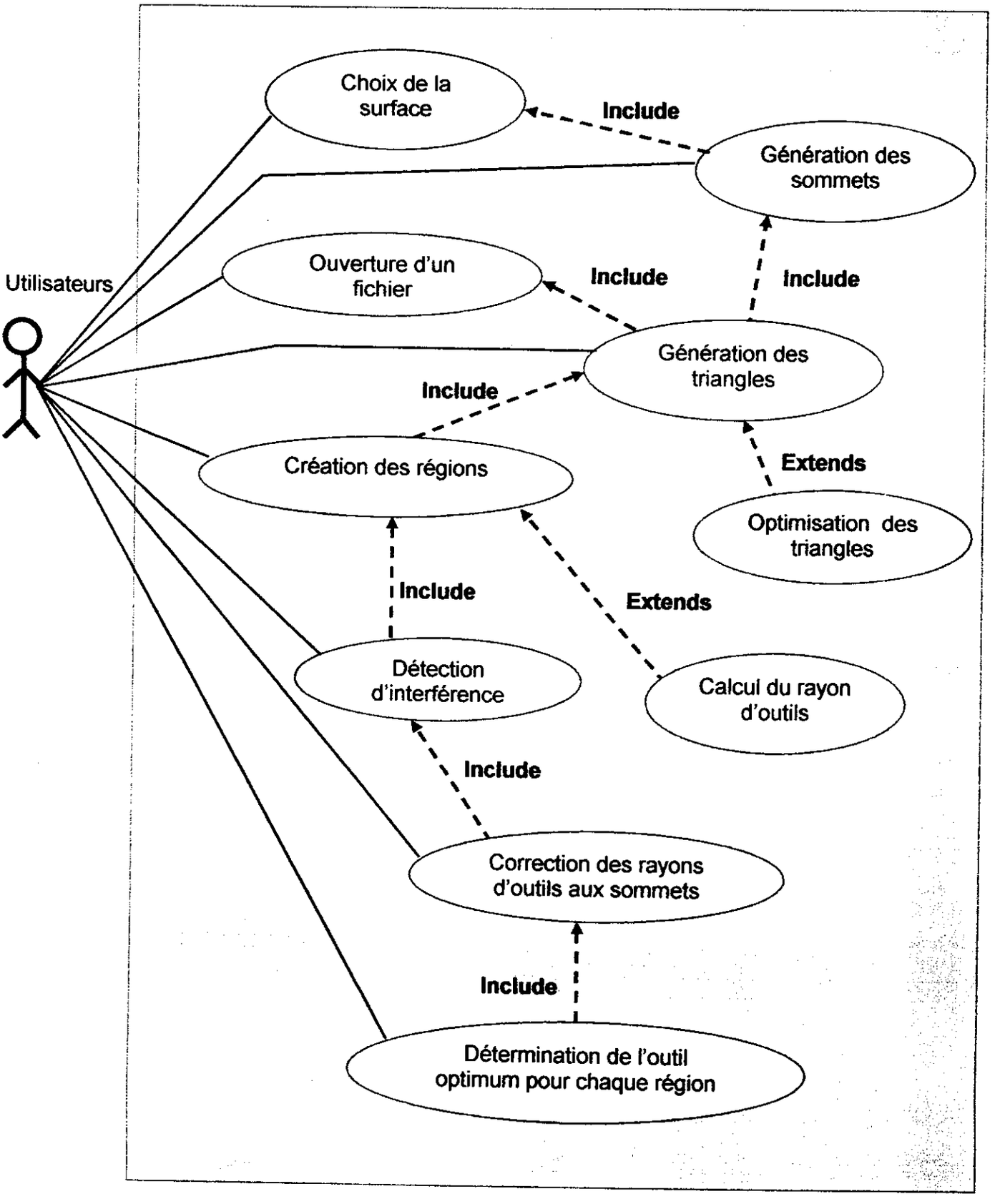


Fig.V.1. Diagramme de cas d'utilisation

III.3.2 Diagramme de séquences : le diagramme de séquence met en évidence l'aspect chronologique de l'envoi des messages. Souvent, le diagramme de séquence permet de compléter le diagramme des cas d'utilisation en mettant en évidence les objets et leur interaction.

La visualisation des modèles CAO passe par les étapes suivantes (voir Fig.V.2) :

- L'utilisateur demande l'ouverture des modèles CAO.
- Le système ouvre et visualise les modèles CAO.

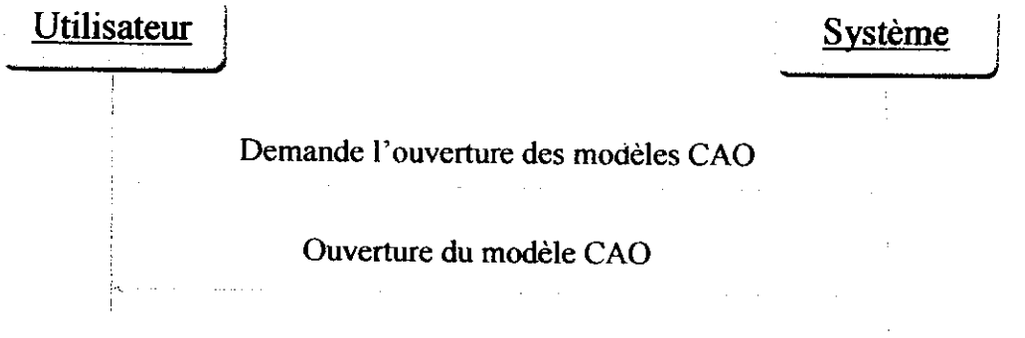


Fig.V.2. Diagramme de séquence d'ouverture du modèle CAO

Le choix de l'outil optimum passe par les étapes suivantes (voir Fig.IV.3) :

- L'utilisateur demande la génération des régions.
- Le système génère les régions.
- L'utilisateur demande la détection et la correction des interférences.
- Le système calcule un rayon optimum qui élimine les interférences.
- Le système affiche le rayon optimum.

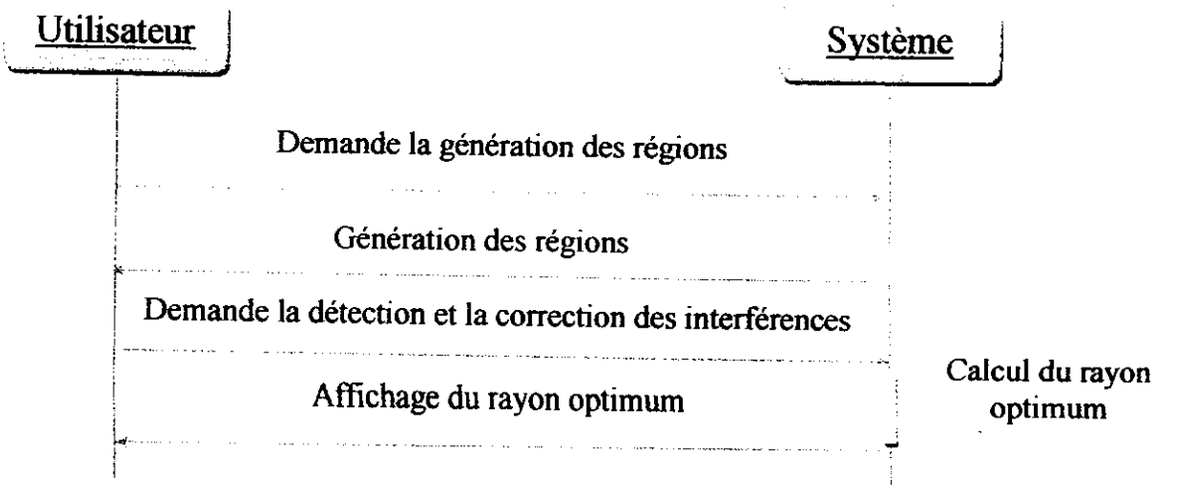


Fig.V.3. Diagramme de séquence de choix d'outil optimum.

L'approximation de la surface à partir d'un nuage de points généré de cette surface passe par les étapes suivantes(voir Fig.V.4) :

- L'utilisateur demande l'ouverture d'une surface.
- Le système ouvre la surface.
- L'utilisateur demande la génération des sommets.
- Le système génère les sommets.
- L'utilisateur introduit le type de la triangulation.
- L'utilisateur demande la triangulation de la surface.
- Le système triangule la surface et vérifie la triangulation.
- L'utilisateur introduit les critères de filtrage des triangles.
- L'utilisateur demande le filtrage des triangles.
- Le système optimise le nombre de triangles.

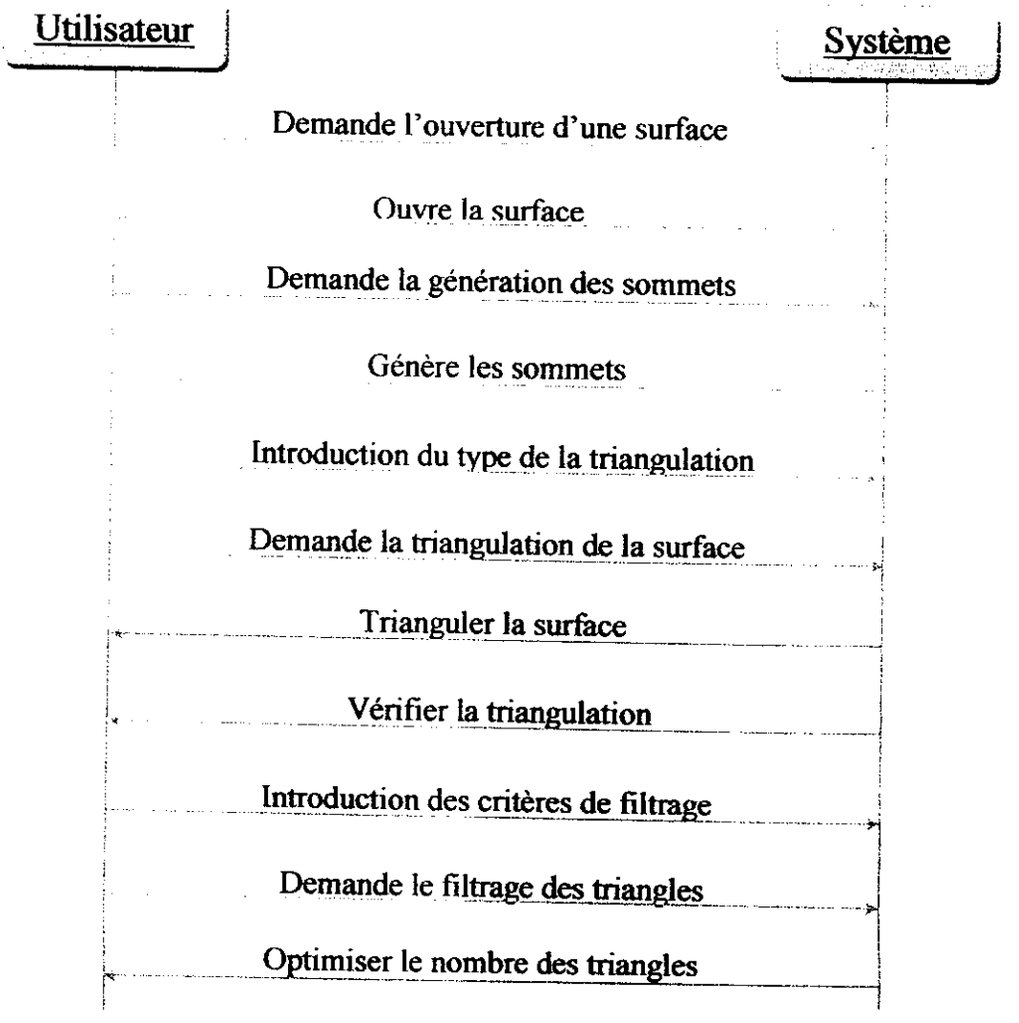


Fig.V.4. Diagramme de séquence d'approximation du modèle

L'approximation de la surface à partir d'un fichier de points passe par les étapes suivantes : (voir FigIV.5)

- L'utilisateur demande l'ouverture d'un fichier.
- Le système ouvre le fichier.
- L'utilisateur demande la visualisation des sommets.
- Le système visualise les sommets.
- L'utilisateur introduit le type de la triangulation.
- L'utilisateur demande la triangulation du nuage de points.
- Le système triangule le nuage de points et vérifie la triangulation.
- L'utilisateur demande l'affichage du type des points.
- Le système affiche le type des points.
- L'utilisateur demande l'affichage du type des triangles.
- Le système affiche le type des triangles.
- L'utilisateur introduit les critères de filtrage des triangles.
- L'utilisateur demande le filtrage des triangles.
- Le système optimise le nombre de triangles.

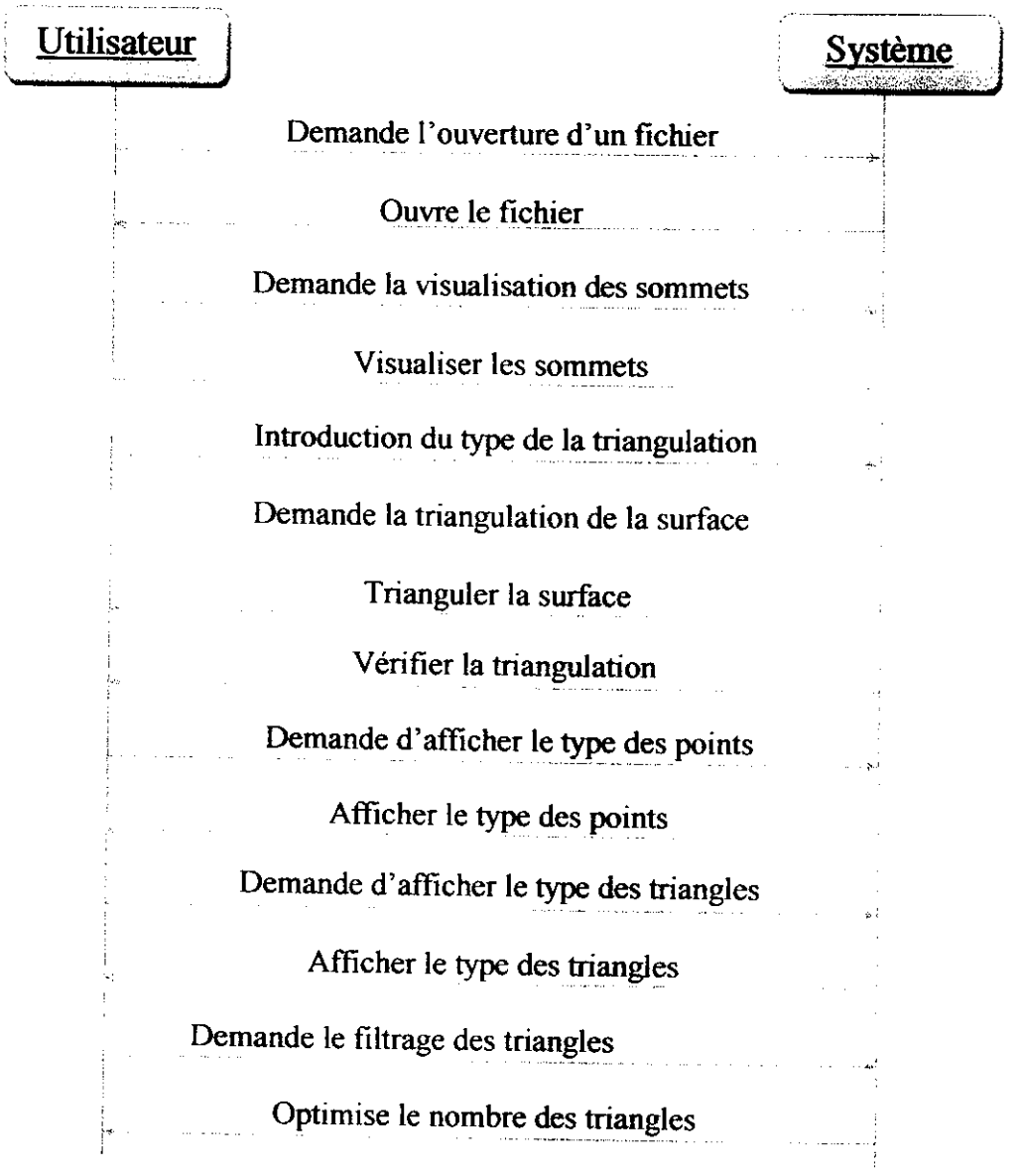


Fig.V.5. Diagramme de séquence d'approximation du modèle.

III.3.3 Diagramme d'activité : le diagramme d'activités est un modèle détaillé, il montre l'activité et le fonctionnement d'une opération d'une classe. Les figures suivantes montrent les diagrammes d'activité « ouverture du modèle CAO », « approximation du modèle CAO », « choix d'outils optimums » et « approximation du modèle à partir d'un fichier ».

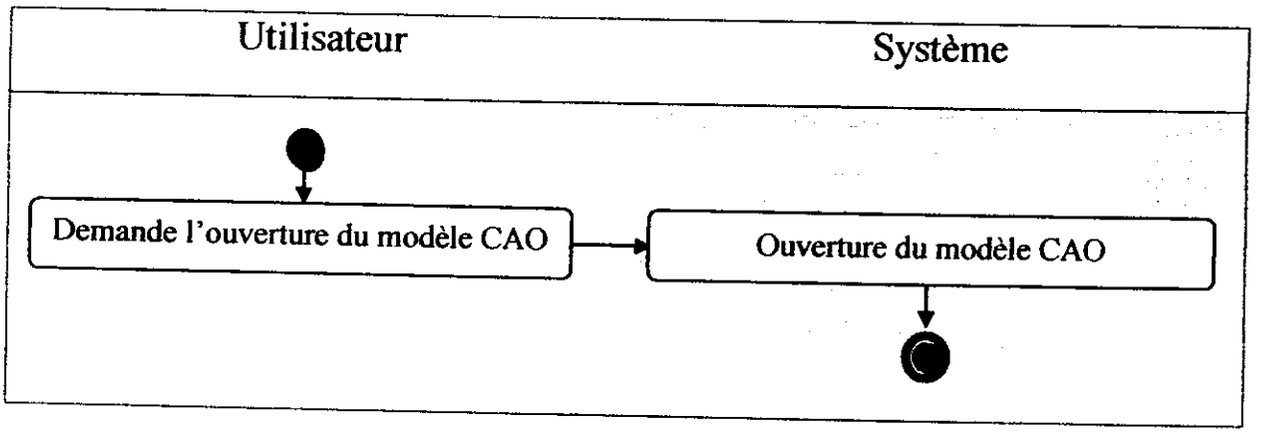


Fig.V.6. Diagramme d'activité d'ouverture du modèle CAO

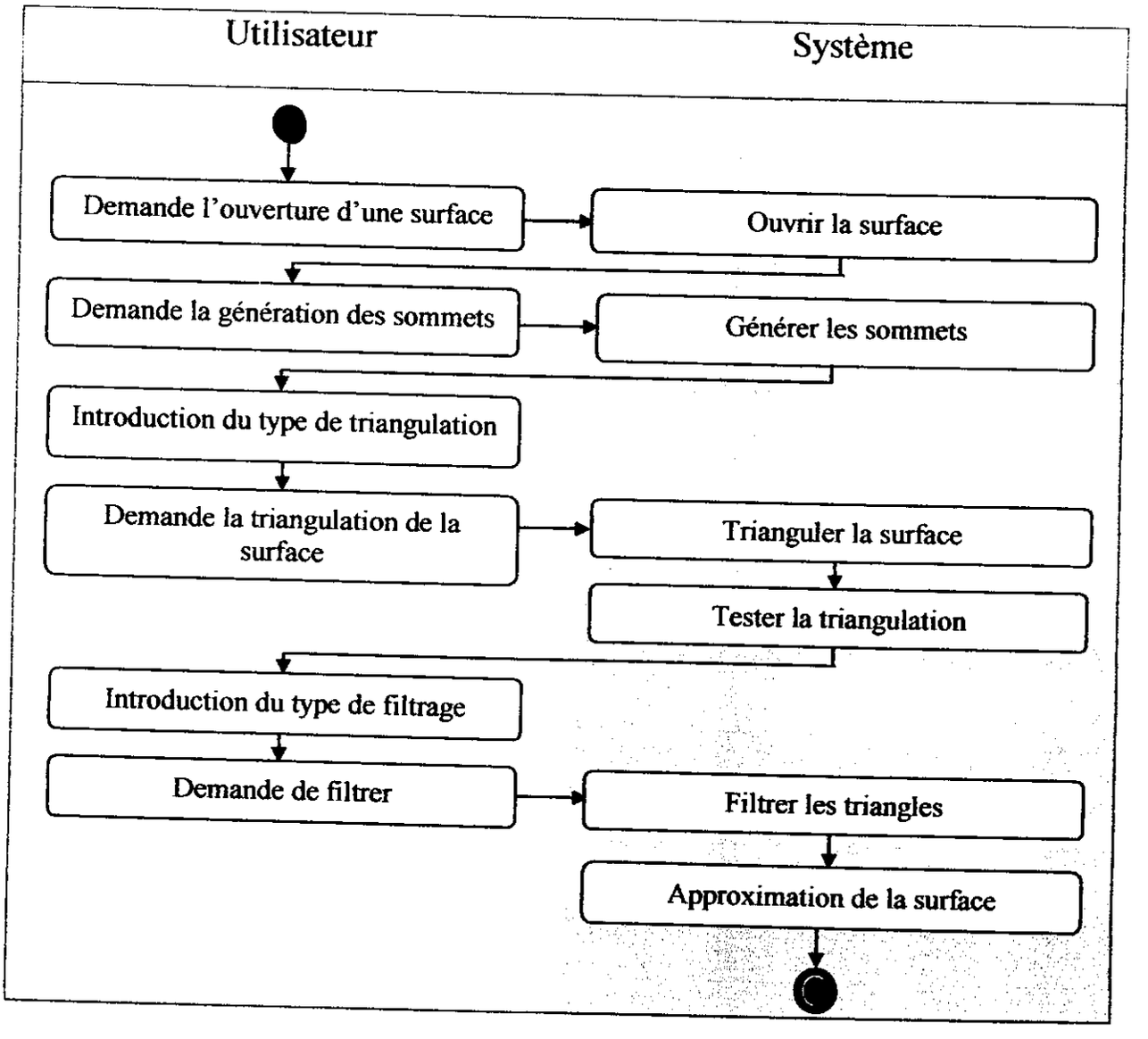


Fig.V.7. Diagramme d'activité d'approximation du modèle CAO.

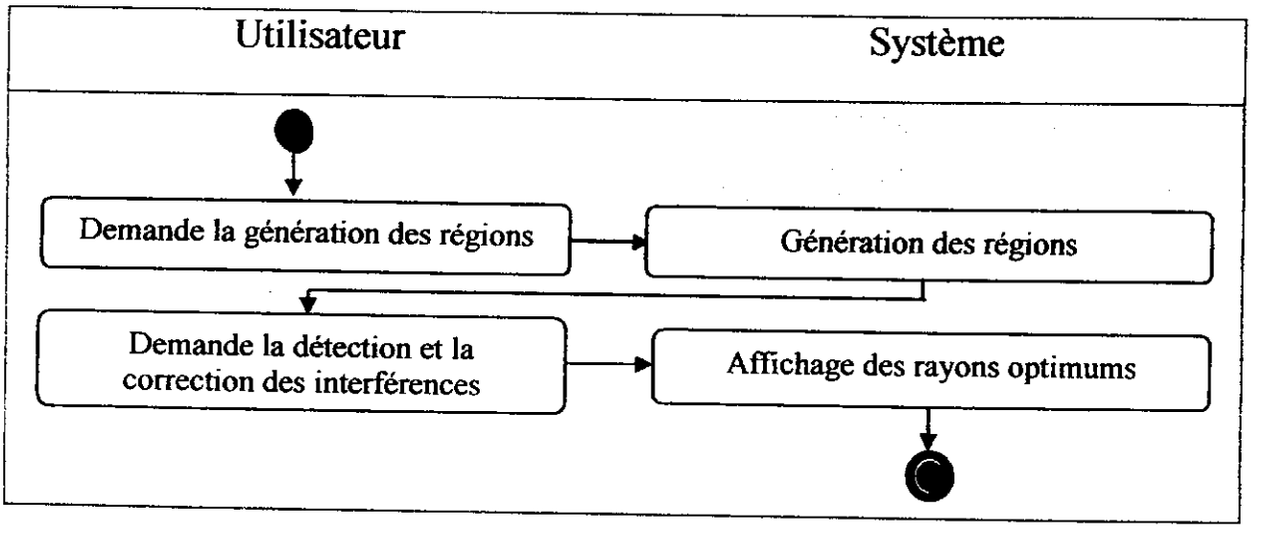


Fig.V.8. Diagramme d'activité de choix d'outil optimum

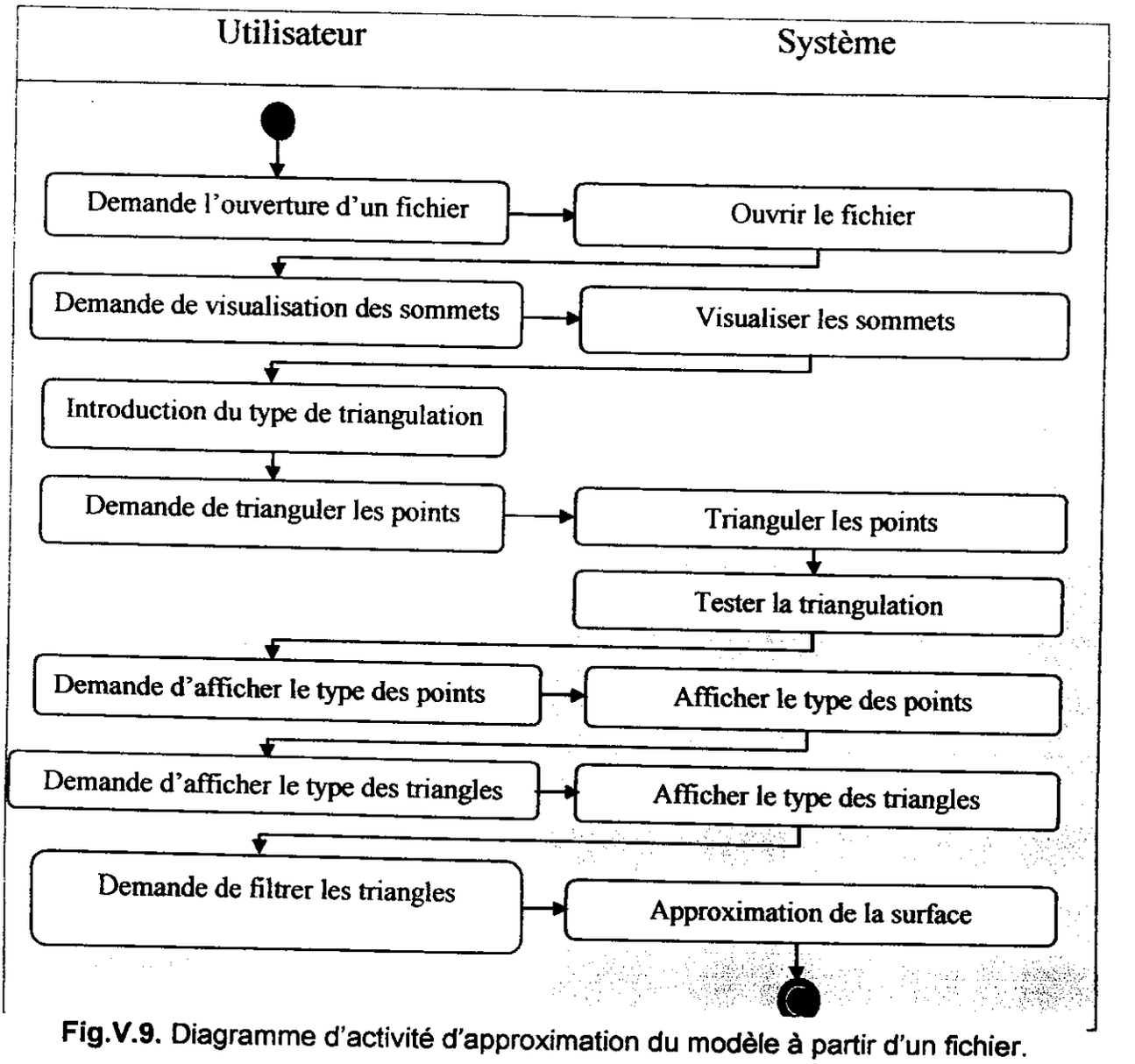


Fig.V.9. Diagramme d'activité d'approximation du modèle à partir d'un fichier.

III.3.4. Diagramme de classe :

Pour représenter la structure de notre système, nous avons représenté le diagramme de classe de notre système, qui comporte toutes les classes que nous avons implémentées avec leurs relations (voir Fig. V.10). Pour réduire la taille du diagramme et donner une vue bien claire, nous avons représenté les classes sans leurs attributs et leurs fonctions.

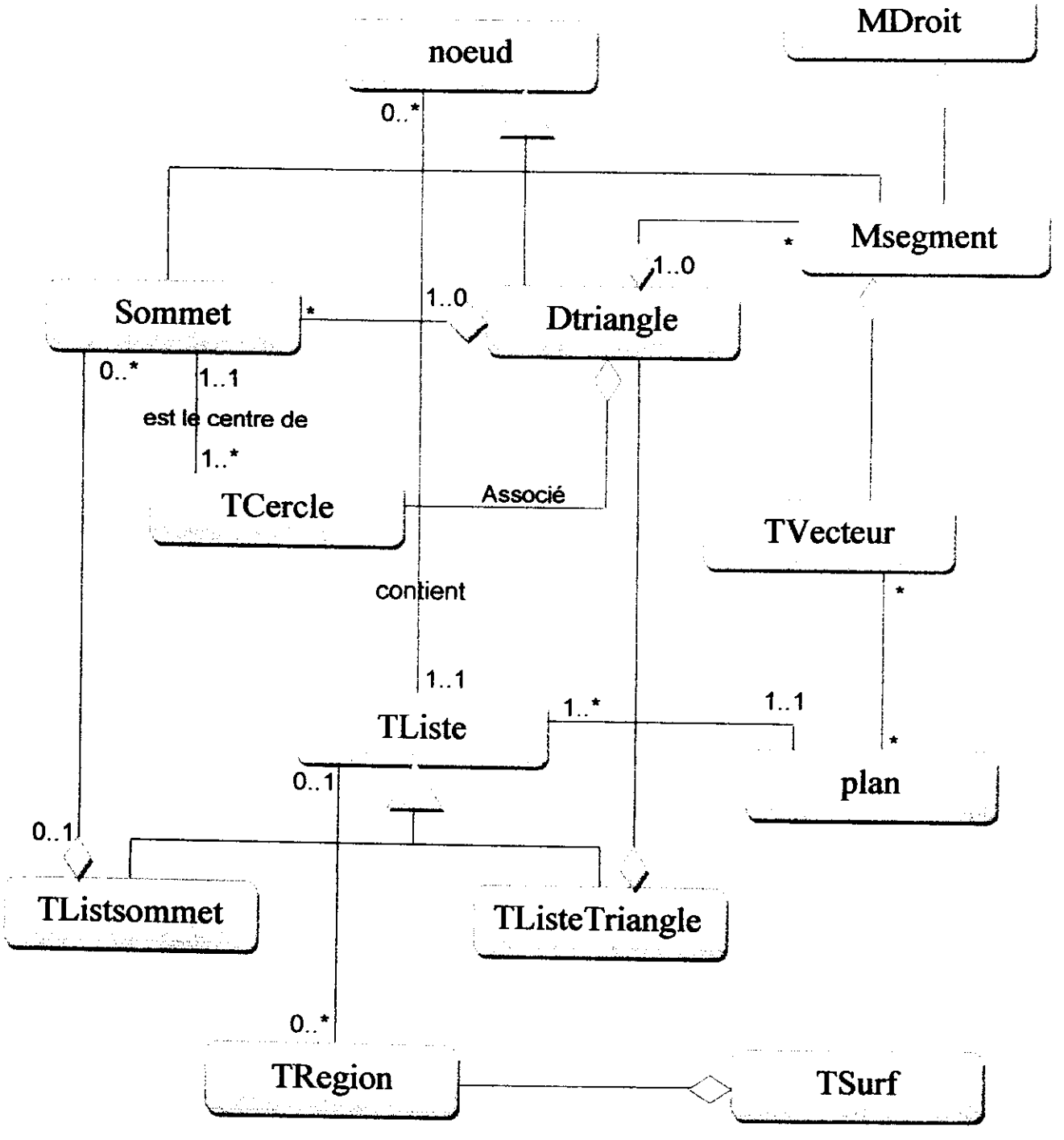


Fig.V.10. Diagramme de classe

Les différentes classes utilisées dans notre conception sont les suivantes :

1. Classe DTriangle : cette classe englobe les paramètres de définition d'un triangle à savoir ses trois sommets, ses voisins, sa normale, ses segments et son cercle circonscrit (voir Fig.V.11)

DTriangle		
-	*s1	sommet
-	*s2	sommet
-	*s3	sommet
-	Cercle	TCercel
-	Cg	sommet
+	parent	DTriangle
+	*t1	DTriangle
+	*t1	DTriangle
+	*t1	DTriangle
+	*seg12	MSegment
+	*seg13	MSegment
+	*seg23	MSegment
+	*normale	TVecteure
+	contient(sommet *s)	bool
+	voisin(DTriangle*,sommet*&,MSegment&)	bool
+	voisin(DTriangle*)	bool
+	operator==(DTriangle* t)	bool
+	operator!=(DTriangle* t)	bool
+	inclutcercle(sommet* s)	bool
+	incluttriangle(sommet*)	bool
+	*eclater(sommet* s)	TListeTriangle
+	operator=(DTriangle*)	DTriangle
+	operator!()	DTriangle
+	dessiner_triangles_voisins()	void
+	dessiner_triangle()	void

Fig.V.11. classe Dtriangle

La classe DTriangle comporte plusieurs fonctions et les plus importantes sont les suivantes :

voisin(DTriangle*,sommet*&, MSegment&p) : retourne le triangle voisin partageant le même segment.

incluttriangle(sommet*) : cette fonction permet de tester l'appartenance d'un point à un triangle donné.

inclutcercle(sommet* s) : cette fonction permet de tester l'appartenance d'un point au cercle circonscrit à un triangle donné.

dessiner_triangle() : cette fonction permet la visualisation d'un triangle.

eclater(sommet* s) : cette fonction permet de subdiviser un triangle en trois triangles avec mise à jour des voisins des triangles crées.

2. Classe sommet : cette classe englobe les paramètres de définition d'un sommet à savoir ses trois coordonnées cartésiennes, ses deux coordonnées paramétriques, ses propriétés géométriques, sa forme locale, les triangles partageant ce sommet et les sommets voisins(voir Fig.V.12) .

Sommet	
- U	double
- V	double
- x	double
- y	double
- z	double
- xtu	double
- ytu	double
- ztu	double
- nx	double
- ny	double
- nz	double
- RCmax	double
- RCmin	double
+ dessiner()	Void
+ operator==(sommet s)	Bool
+ operator!=(sommet *s)	Bool
+ Set_Attributs()	Void
+ image(Plan*p)	double

Fig.V.12. classe sommet

La classe sommet comporte plusieurs fonctions et les plus importantes sont les suivantes :

dessiner() : cette fonction permet de visualiser un point.

image(Plan*p) : cette fonction permet de connaître la position du point par rapport au plan tangent.

Set_Attributs() : cette fonction permet de calculer une estimation du vecteur normal au sommet et de déterminer sa forme locale.

Get_ListeT() : cette fonction permet de déterminer les triangles partageant ce sommet.

3. Classe MDroite : cette classe englobe les paramètres de définition d'une droite à savoir les coefficients de la droite et le type de la droite (verticale, horizontale et inclinée) (voir Fig.V.13) .

MDroite	
- A	Double
- B	double
- X	double
+ operator=(MDroite)	MDroite
+ operator==(MDroite)	bool
+ intersection(MDroite d, sommet & s)	bool
+ memecote(sommet *, sommet *)	bool

Fig.V.13. classe MDroite

La classe MDroite comporte plusieurs fonctions et les plus importantes sont les suivantes :

intersection(MDroite d, sommet & s) :cette fonction permet de vérifier l'appartenance d'un point à cette droite.

memecote(sommet *, sommet *) : cette fonction permet de tester si les deux sommets sont du même coté ou non par rapport à la droite.

4. Classe MSegment : cette classe englobe les paramètres de définition d'un segment à savoir ses deux sommetscoefficients de la droite et le type de la droite (verticale, horizontale et inclinée) (voir Fig.V.14).

MSegment	
- s1	Sommet
- s2	Sommet
+ operator()(bool sens)	TVecteur
+ milieu()	Sommet

Fig.V.14. classe MSegment

Les principales fonctions de la classe MSegment sont les les suivantes :

milieu() : cette fonction permet de calculer le point milieu du segment.

transeversale() : cette fonction permet de déterminer la droite transversale à un segment.

5. Classe TCercle : cette classe englobe les paramètres de définition d'une sphère à savoir ses deux paramètres le centre et le rayon (voir Fig.IV.15).

TCercle		
-	centre	sommet
-	R	double
+	operator==(TCercle c)	bool
+	inclut(sommet *s)	bool

Fig.V.15. classe TCercle

Les principales fonctions de la classe TCercle sont les suivantes :

inclut(sommet *s) : cette fonction permet de tester l'appartenance d'un point à la sphère.

dessiner_sphere() : cette fonction permet de visualiser la sphère.

6. Classe Plan : cette classe englobe les paramètres de définition d'un plan à savoir ses coefficients A, B, C et D (voir Fig.V.16).

Plan		
+	A	Double
+	B	double
+	C	double
+	D	double

Fig.V.16. classe Plan

7. Classe noeud : cette classe englobe les paramètres de définition d'un noeud à savoir le nœud lui-même, son suivant et son précédent (voir Fig.V.17).

noeud		
+	*t	noeud
+	*svt	noeud
+	*prd	noeud

Fig. V.17. classe noeud

8. Classe TListeSommet : cette classe englobe les paramètres de définition d'une liste de sommets à savoir la tête et le nombre de sommets (voir Fig.V.18).

TListeSommet	
- * tetel	noeud<sommet>
- Nbre	int
+ * Get_tetel()	noeud<sommet>
+ longueur()	int
+ inserer(sommet*)	TListeSommet
+ inserer(TListeSommet*)	TListeSommet
+ supprimer(sommet*)	TListeSommet
+ operator=(TListeSommet)	TListeSommet
+ contient(sommet*)	bool

Fig.V.18. classe TListeSommet

Les principales fonctions de la classe TListeSommet sont les suivantes :

longueur() ; cette fonction permet de retourner la langueur de la liste.

inserer(sommet*) : cette fonction permet d'insérer un sommet dans la liste.

supprimer(sommet*) : cette fonction permet de supprimer un sommet de la liste.

contient(sommet*) : cette fonction permet de tester l'appartenance d'un sommet à la liste des sommet.

9. Classe TListeTriangle : cette classe englobe les paramètres de définition d'une liste de triangles à savoir la tête et le nombre de triangles (voir Fig.V.19).

TListeTriangle	
- * tetel	noeud<sommet>
- Nbre	int
- void ListeNormaleDeSommet(sommet* s,DTriangle* t)	
+ longueur()	Int
+ inserer(DTriangle*)	TListeTriangle
+ inserer(TListeTriangle*)	TListeTriangle
+ supprimer(DTriangle*)	TListeTriangle
+ eclater(sommet*)	TListeTriangle
+ operator=(TListeTriangle)	TListeTriangle
+ dessiner_triangles()	void
+ bascule_diagonal(DTriangle,DTriangle,TListeTriangle)	void
+ normale_sommet(sommet*)	void
+ verification()	void
+ contient(DTriangle*)	bool

Fig.V.19. classe TListeTriangle

Les principales fonctions de la classe TListeTriangle sont les suivantes :

longueur() : cette fonction permet de retourner la langueur de la liste.

inserer(sommet*) : cette fonction permet d'insérer un triangle dans la liste.

supprimer(sommet*) : *cette fonction permet de supprimer un triangle de la liste*

contient(sommet*) : cette fonction permet de tester l'appartenance d'un triangle à la liste des triangles.

eclater(sommet*): cette fonction permet de subdiviser un triangle en trois triangles ou en quatre triangles avec mise à jour des voisins des triangles générés.

bascule diagonal (DTriangle,DTriangle,TListeTriangle) : c'est une fonction récursive permettant de basculer la diagonale d'un quadrilatère convexe avec mise à jour des triangles voisins.

10. Classe TVecteur: cette classe englobe les paramètres de définition d'un vecteur à savoir ses trois composantes cartésiennes, son signe (voir Fig.V.20).

TVecteur	
- X	double
- Y	double
- Z	double
- Sign	Signe
+ operator==(TVecteur v)	Bool
+ operator=(TVecteur v)	TVecteur
+ operator!()	TVecteur
+ operator+(TVecteur v)	TVecteur
+ operator*(double alpha)	TVecteur
+ operator*(TVecteur v)	TVecteur
+ produitscalair (TVecteur v)	double
+ normer()	TVecteur
+ diviser (double n)	TVecteur

Fig.V. 20. classe TVecteur

Les principales fonctions de la classe TVecteur sont les suivantes :

Produitscalair (TVecteur v) : cette fonction permet de retourner le produit scalaire de deux vecteurs.

dessiner() : : cette fonction permet de visualiser le vecteur.

normer() : cette fonction permet de calculer la norme du vecteur.

11. Classe TRegion : cette classe englobe les paramètres de définition d'une région à savoir X_min, Y_min, Z_min, X_max, Y_max, Z_max ainsi que le numéro de la ligne et de la colonne de la région (voir Fig.V.21).

TRegion	
- X_min	Double
- X_max	double
- Y_min	double
- Y_max	double
- Z_min	double
- Z_max	double
- ligne_reg	int
- colone_reg	Int
+ interferance(sommet*)	Void
+ contient (sommet* s)	bool
+ operator=(TRegion A)	TRegion
+ *tete_outil	TCercle
+ * sommets()	TListeSommet

Fig.V. 21. classe Tregion

Les principales fonctions de la classe TRegion sont les suivantes :

interferance(sommet*) : cette fonction permet de détecter et corriger les interférences des points de la région.

contient (sommet* s) : cette fonction permet de tester l'appartenance d'un sommet à la région.

Dessine_region : cette fonction permet la visualisation de l'ensemble des points de la région.

Sommets () c'est une fonction qui permet de retourner l'ensemble des points de la région.

12. Classe TSurf: cette classe englobe les paramètres de définition d'une surface à savoir son rayon, X_min, Y_min, Z_min, X_max, Y_max, Z_max, et le nombre de lignes et de colonnes de la surface (voir Fig.IV.22).

Tsurf	
- x_min	Double
- x_max	double
- y_min	double
- y_max	double
- z_min	double
- z_max	double
- R	double
- Ligne	int
- Colone	int
- nbre	int
- num_surf	int
- *tete_outil	TCercle
+ interferance(sommet*)	Void
+ Set_Line(int l)	void
+ Set_Colone(int c)	void
+ creer_regions(int,int)	void

Fig.V.22. classe TSurf

Les principales fonctions de la classe TSurf sont les suivantes :

interferance(sommet*) : cette fonction permet de détecter et de corriger l'interférence des points de la surface.

creer_regions(int,int): cette fonction permet de créer les régions pour l'ensemble des points de la surface.

Dessine_region : permet la visualisation de l'ensemble des régions de la surface.

IV. CONCLUSION :

Dans ce chapitre, nous avons défini notre système en commençant par la problématique et les différents objectifs à atteindre, ensuite les solutions que nous avons proposées avec les différents diagrammes de conception. Dans le chapitre suivant, nous allons présenter l'implémentation de notre application.

Chapitre VI

Implémentation de l'Application

I. INTRODUCTION :

Après l'étude de la conception de notre application ainsi que les objectifs fixés et les solutions proposées dans le chapitre précédent, nous allons maintenant présenter notre application logicielle où son implémentation a été faite en utilisant un langage évolué doté d'une interface qui facilite à l'utilisateur la manipulation et l'exécution des différentes tâches. En plus, nous allons faire une description des principaux algorithmes développés.

II. IMPLEMENTATION :

Pour l'implémentation des solutions proposées, nous avons utilisé le C++ Builder et la plate forme Windows XP. Notre application est un logiciel de FAO complémentaire, qui comporte des fonctionnalités d'usinage et qui utilise des géométries conçues en 3D qui sont totalement associées à la CAO. Ce travail est un ensemble de fenêtres Windows permettant à l'utilisateur d'interagir avec l'application en lui permettant de visualiser tous les objets manipulés via la bibliothèque graphique OpenGL.

II.1. Fenêtre principale :

Cette fenêtre est composée de deux parties, une pour la visualisation des différents objets crées et l'autre pour la manipulation. Cette dernière est constituée d'une barre de menu et d'un ensemble de boutons de manipulation des différents paramètres des modèles CAO des surfaces (création d'une surface, suppression d'une surface, modification des points de contrôle, ...etc.) (voir Fig. V.1).

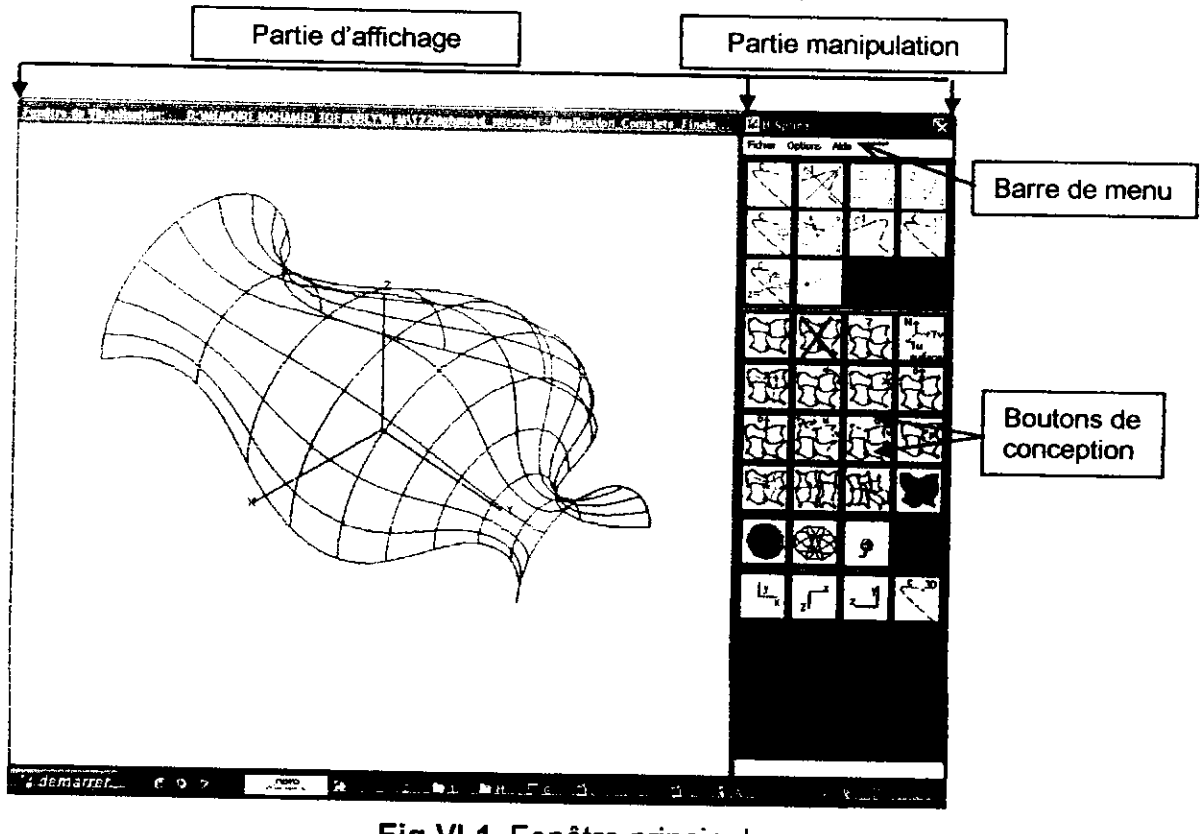


Fig.VI.1. Fenêtre principale.

II.2. Barre du menu principal :

La barre de menu principal est composée de trois rubriques :

- **Rubrique fichier :** qui comporte toutes les fonctionnalités de manipulation des fichiers comme l'ouverture d'un fichier, la création d'un nouveau fichier et la sauvegarde du fichier.
- **Rubrique Option :** permet la modification des différents paramètres des courbes, des surfaces ainsi que des fonctions liées à l'usinage des surfaces gauches en ébauche et en finition.
- **Rubrique Aide :** pour nous aidez pendant l'utilisation de l'application.

II.3. Rubrique triangulation d'un nuage de points :

Pour accéder à la partie que nous avons réalisée et intégrée dans l'application, nous avons deux façons. La première consiste à ouvrir le fichier contenant le modèle CAO des surfaces dans la rubrique ***Fichier*** et ensuite aller à la rubrique menu ***Option*** puis vers la rubrique ***reconstruction de surfaces gauches à partir d'un nuage de points*** et cliquer pour lancer notre application logicielle. La deuxième consiste à aller directement vers notre rubrique sans avoir à ouvrir un modèle CAO (voir Fig. VI.2).

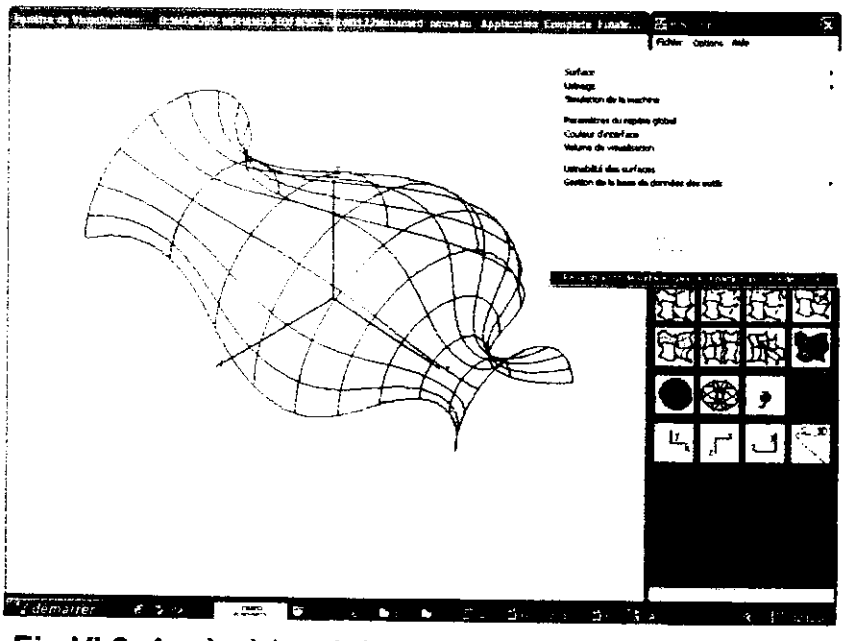


Fig.VI.2. Accès à la rubrique triangulation de Delaunay

III. PRESENTATION DES FENETRES :

Nous allons présenter maintenant toutes les fenêtres créées dans notre travail.

III.1. Fenêtre de triangulation de Delaunay :

C'est la première fenêtre qui apparaît à l'utilisateur (voir Fig. VI.3) qui permet de générer une triangulation à partir d'un nuage de points et par la suite grouper les triangles en des régions distinctes en fonction des formes locales des points.

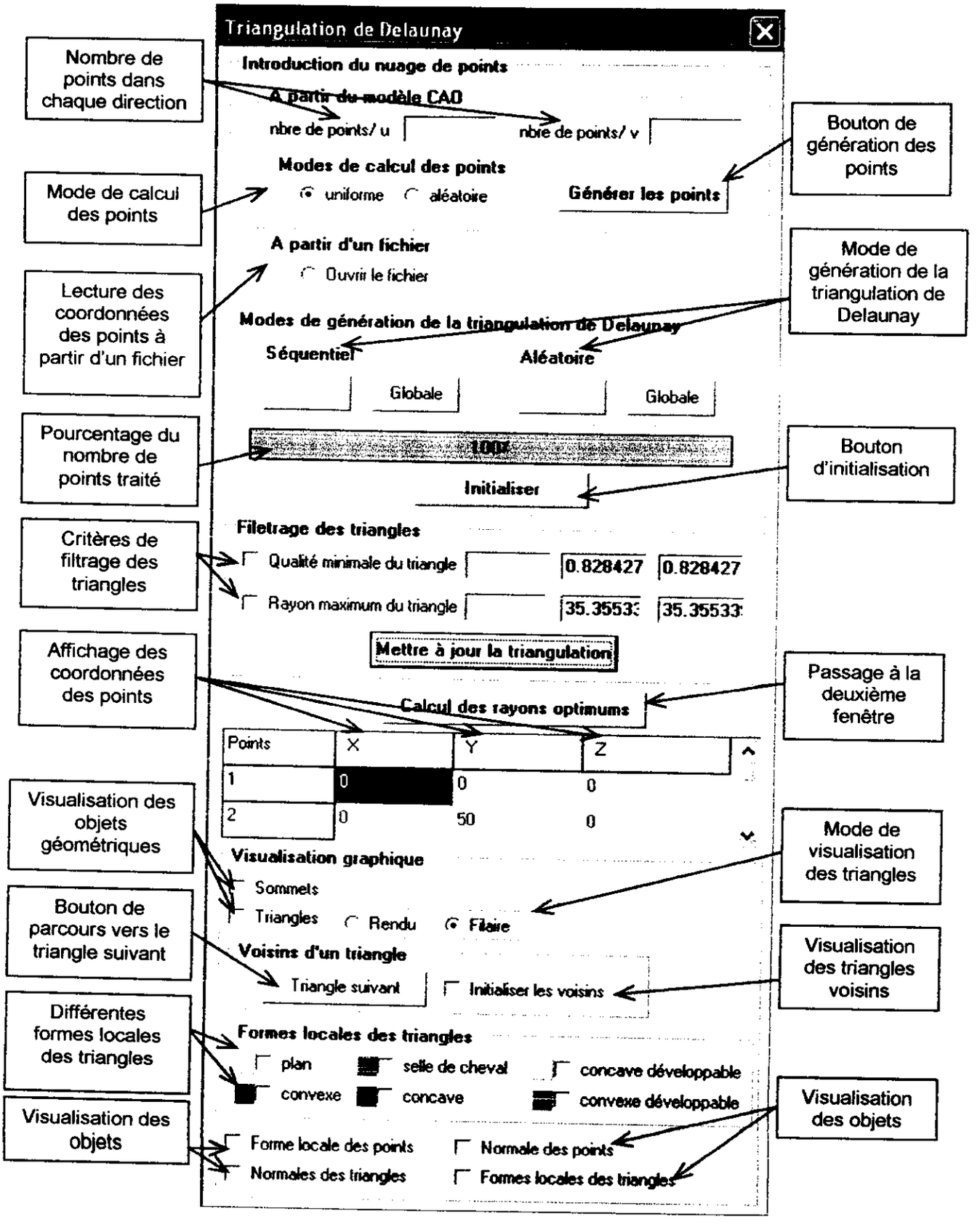


Fig.VI.3. Fenêtre de triangulation

Cette fenêtre apparaît dès qu'on clique sur la rubrique "***reconstruction du surfaces gauches à partir d'un nuage de points***" dans la barre de menu de la fenêtre principale. Elle permet de trianguler la surface en fonction des paramètres de triangulation introduits par l'utilisateur pour donner une bonne approximation de la surface. Dans cette fenêtre nous avons la possibilité soit de générer les points avec une distribution uniforme ou une distribution aléatoire à partir des modèles CAO des surfaces existantes après l'introduction du nombre de points à créer dans chaque direction et en cliquant sur le bouton *générer les sommets* où bien par l'ouverture d'un fichier texte contenant les coordonnées des points en cliquant sur le bouton *ouvrir le fichier*. Pour les deux cas, les coordonnées des points sont affichées dans un tableau. De même, elle permet aussi la visualisation des sommets, des triangles avec leurs formes locales.

Ensuite, l'utilisateur peut générer les triangles après le choix du mode de génération de la triangulation de Delaunay. Ce mode peut être séquentiel ou aléatoire avec une insertion point par point ou global des points tout en ayant la possibilité de visualiser les voisins d'un triangle donné. Une fois que la triangulation est terminée, il y a activation des zones de saisie pour permettre à l'utilisateur de spécifier la qualité minimale du triangle et le rayon maximum du cercle circonscrit au triangle afin de filtrer les triangles et ainsi obtenir les frontières de la surface. En cliquant sur le bouton *Mettre à jour la triangulation*, les triangles sont filtrés en fonction des contraintes spécifiées et en même temps détermination de la forme locale de chaque point et groupement des triangles en des régions distinctes. L'utilisateur a la possibilité d'afficher les sommets, les triangles générés, les formes locales des triangles, les formes locales des points et les normales aux points et aux triangles.

Les principaux algorithmes développés dans cette partie sont les suivants :

III.1.1 Algorithme de création de la liste des sommets uniforme :

- Soit pas_u et pas_v, u et v des réels tels que :
  Pas_u = 1/nbr_colonnes, pas_v = 1/nbr_lignes; u et v initialisés à 0,
- Pour i de 0 à nbr_lignes
  v =pas_v*i,
- Pour j de 0 à nbre_colonnes
  u =pas_u*j,
- Créer un sommet de coordonnées (u, v) (voir Fig. V.4).

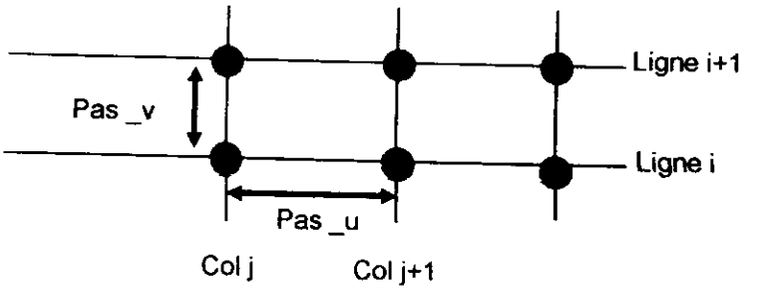


Fig.VI.4. Création des sommets uniforme.

III.1.2. Algorithme de création de la liste des sommets aléatoire :

- Soit pas_u et pas_v, u et v des réels tels que :
  Pas_u = 1/nbr_colonnes, pas_v = 1/nbr_lignes; u et v initialisés à 0,
- Pour i de 0 à nbr_lignes
- Pour j de 0 à nbre_colonnes
  u =pas_u*j,
  v =pas_v*i,
- u = RandomRange (0, nb_pts_u-1)/d1 ;
- v = RandomRange (0, nb_pts_v-1)/d2 ;
- Créer un sommet de coordonnées (u, v).

III.1.3. Algorithme de lecture des coordonnes des points à partir d'un fichier :

- Fichier_Point=new TStringList;
- Liste = new TListeTriangle;
- Fichier_Point = LoadFromFile(Filename);
- nbre= nbre_points + 3;
- nbre_initial = nbre;

III.1.4. Principe de l'algorithme de la triangulation de Delaunay :

Le cœur de l'algorithme réside dans les structures de données utilisées. Ainsi, l'algorithme procède par insertions successives de points P. Le point P à insérer doit être associé à un triangle le contenant de la triangulation en cours. Puis, de nouveaux triangles sont construits ayant pour sommet P et les sommets du triangle associé. Ce dernier triangle doit être invalidé pour la triangulation finale mais conservé dans une structure pour les besoins algorithmiques de recherche du triangle contenant un prochain point à insérer. En effet, cette recherche procède par parcours d'une liste D contenant tous les triangles créés. Ainsi, un nœud de la liste correspond à un triangle créé et ses fils et ses voisins. La tête de cette liste est un triangle virtuel englobant l'ensemble des points à trianguler. Ainsi, à la fin de l'algorithme, la triangulation de Delaunay du nuage de points S est l'ensemble des triangles de la liste D. Parallèlement, on doit conserver pour chaque sommet inséré, l'information permettant de retrouver rapidement les triangles touchant ce sommet. Cette information est stockée dans une autre liste doublement chaînée où chaque élément de cette liste contient deux pointeurs vers les deux triangles précédent et suivant. A chaque création d'un nouveau triangle, on vérifie si ce triangle est valide, c'est-à-dire que pour chaque arête AB, BC et CA on teste si la nouvelle association de triangles PAB-QAB par exemple appartient à la triangulation de Delaunay (voir Fig. VI.5.a). Pour cela, on teste si le cercle circonscrit à PAB ne contient pas le point Q. Si tel est le cas, on bascule l'arête AB pour créer deux nouveaux triangles

dont la diagonale est l'arête PQ (voir Fig. VI.5.b) qui eux appartiendront à la triangulation de Delaunay finale. A la fin de l'algorithme, il ne reste plus qu'à supprimer tous les triangles partageant un sommet avec le premier triangle virtuel englobant correspondant à la tête de la liste globale D.

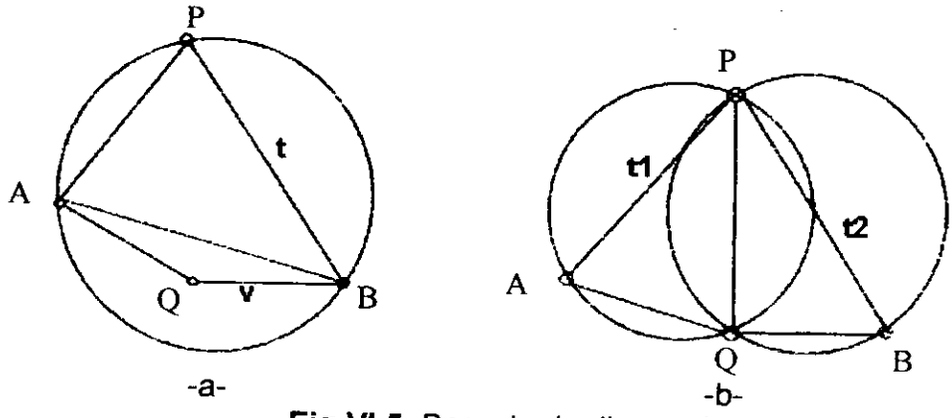


Fig.VI.5. Bascule de diagonale.

Le triangle de sommets A, B et C est liée à ses trois arêtes AB, BC, et CA et à ses triangles si elles existent AB, PBC, PCA. La procédure de validation de l'arête AB lors de la création du nouveau triangle PAB : si le cercle circonscrit au triangle PAB ne contient pas le point Q, l'arête est valide ; sinon, on doit basculer l'arête AB vers l'arête PQ.

III.1.4. Algorithme du mode de génération de la triangulation de Delaunay:

III.1.4.a. Algorithme du mode séquentiel :

L'algorithme procède par insertion successive des points dans l'ordre de leur création ou de leur lecture.

- for(int i = 0;i< nbre-3;i++) eclater(sommets[i]);

III.1.4.b. Algorithme du mode aléatoire :

L'algorithme procède par insertion aléatoire des points.

- for(int j = 0;j< nbre-3;j++)
  do i = random (nbre-3);
- while(sommets[i]. inserer());
  eclater (sommets[i]);
  Sommets[i]. inserer (true);

Pour les deux modes d'insertion des points, la triangulation de Delaunay est faite par l'algorithme suivant (voir Fig. V.6) :

1. ***construire le triangle global*** :

   - *Écrire tinitiale = new DTriangle (s1, s2, s3).* Cette fonction permet de construire un triangle englobant l'ensemble des points en ajoutant les trois points (n), (n+1), (n+2).

2. ***Ajout des points*** :

   - Écrire une fonction *bool incluttriangle (sommet*).* A l'aide de cette fonction, on sait si un point p est contenu dans un triangle t. Le principe de fonctionnement est le suivant :

     - on test si p est du même côté que le troisième sommet par rapport à la droite formée par deux sommets du triangle.

     - Si le test est vérifié pour tous les sommets, alors p appartient à t.

3. ***Ajout des triangles*** :

   - Écrire une fonction *TListeTriangle& eclater(sommet*).* L'algorithme de cette fonction est le suivant :

     - tous d'abord on crée trois nouveaux triangles à partir du nouveau point et des sommets du triangle qui le contient;

     - établissement des liens avec leurs voisins;

     - actualisation des voisins de leurs voisins car leurs nouveaux voisins ne pointent pas encore sur eux et pointent toujours sur le triangle t contenant le point p;

     - insertion des trois nouveaux triangles dans la liste;

     - suppression du triangle t contenant p;

     - on renvoie alors au triangle parant correspondant à l'un des trois nouveaux triangles car il a pour voisins les deux autres. Il est important pour la suite de l'algorithme de ne pas perdre l'adresse de ces trois triangles.

4. ***Cocircularité*** :

   - Écrire une fonction *bool voisin (DTriangle*t, sommet*s, MSegment& st).* Cette fonction qui permet de retourner le voisin du triangle t qui contient le sommet s et le segment st.
   - Écrire une fonction *bool inclutcercle (sommet* s).* Cette fonction permet de vérifier si un point p est dans le cercle circonscrit à un triangle t. On

test le triangle t avec le sommet s. Cette fonction va nous permettre de vérifier le critère principal de l'algorithme.

- Écrire une fonction *void bascule_diagonal (DTriangle*, DTriangle*, TListeTriangle*)*. Cette fonction est récursive qui permet de tester si le cercle circonscrit aux trois nouveaux triangles. On bascule l'arête st pour créer deux nouveaux triangles qui eux appartiendront à la triangulation de Delaunay finale.

5. ***Orientation des triangles*** :

   - Écrire une fonction *void pptes_triangle ()*. Cette fonction permet de déterminer la bonne orientation de tous les triangles de la liste et le calcul du vecteur normal pour chaque triangle.

6. ***Vérification de la triangulation*** :

   - Écrire une fonction *void vérification ()*. Cette fonction permet de parcourir la liste des triangles et de vérifier le critère du cercle vide pour chaque triangle.

7. ***Suppression des triangles*** :

   - Écrire une fonction *TListeTriangle& supprimer (DTriangle*)*. Il s'agit maintenant de supprimer tous les triangles partageant un sommet avec le premier triangle virtuel englobant le nuage de point.

8. ***Filtrage des triangles*** :

   - Il s'agit de supprimer tous les triangles qui ne vérifient pas le critère de la qualité minimale du triangle ou du rayon maximum du cercle circonscrit.

9. ***Détermination de la forme locale d'un point*** :

   - Il s'agit de déterminer la forme locale de chaque point en fonction de la position des points voisins par rapport au plan tangent.

10. ***Détermination de la forme locale d'un triangle*** :

    - Il s'agit de déterminer la forme locale de chaque triangle en fonction de la forme locale de ces sommets.

11. ***Groupement des triangles en des régions distinctes*** :

    - Il s'agit de grouper les triangles de même forme locale dans une même liste.

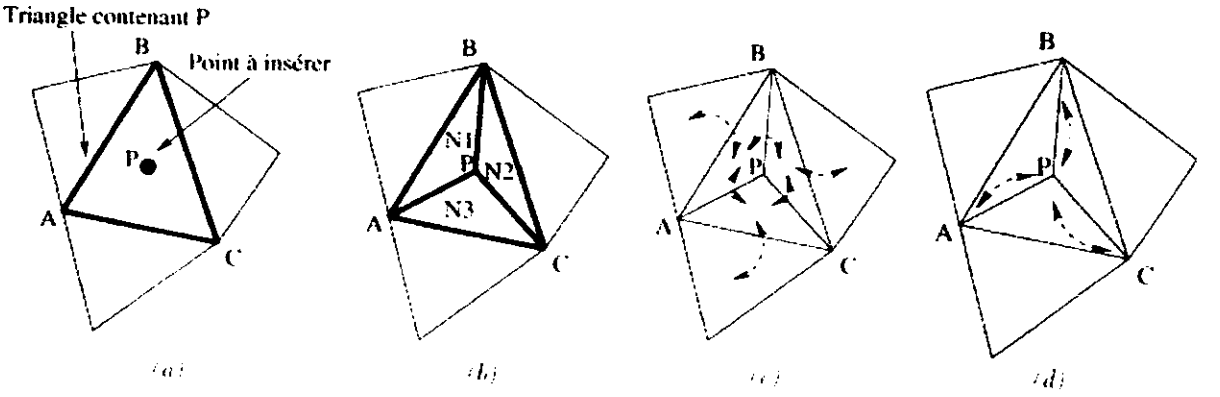


Fig.VI.6. Insertion d'un point à la triangulation

III.2. Fenêtre des rayons optimums des régions :

Cette fenêtre apparaît dès qu'on clique sur le bouton *calcul des rayons optimums* de la fenêtre Triangulation de Delaunay. Elle permet de calculer le rayon d'outil optimum en chaque point ainsi que le rayon d'outil optimum pour chaque région (Fig. VI.7).

Dans un premier temps, l'utilisateur est invité à introduire le nombre de régions à créer par la spécification du nombre de lignes et de colonnes. En cliquant sur le bouton *créer les régions*, les régions seront créées et il a la possibilité de visualiser les limites et les points des régions. Après cette étape, pour lancer le processus de *détection, de vérification des interférences et de détermination du rayon optimum* pour chaque point et pour chaque région, nous n'avons qu'à cliquer sur le bouton *calculer les outils optimums*. Avant le lancement des calculs, l'utilisateur a la possibilité de visualiser pendant le calcul la sphère optimale en chaque point avec son rayon. A la fin des calculs et pour afficher les résultats finaux à savoir le rayon optimum en chaque point et pour chaque région, l'utilisateur n'à qu'à cliquer sur le bouton *Afficher les rayons optimums*. De même, il est possible de visualiser la sphère optimale en chaque point par la sélection d'une ligne du tableau d'affichage.

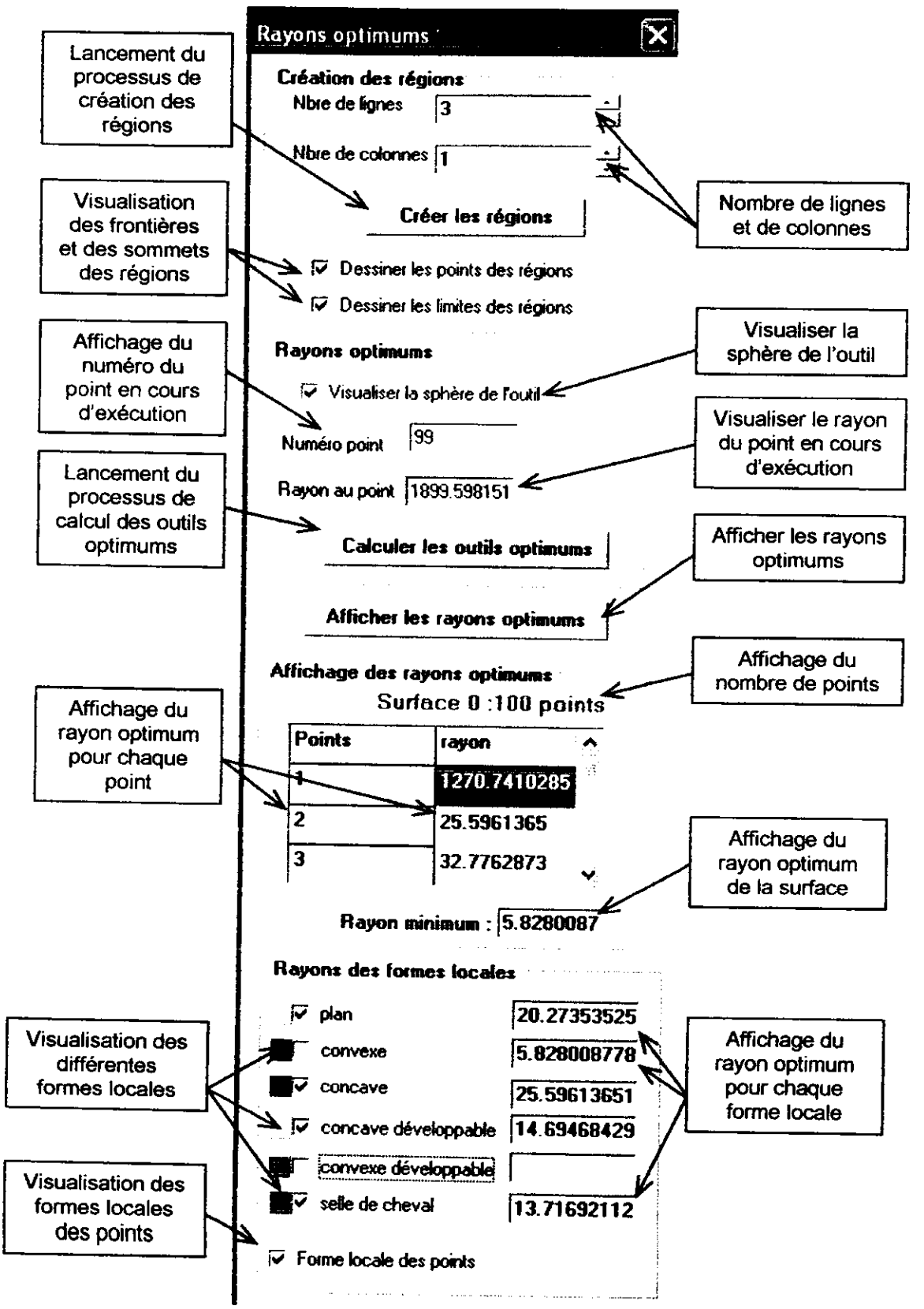


Fig.VI.7. Fenêtre de rayons optimums

III.2.1 Génération des régions :

Afin d'accélérer les calculs de détection et de correction des inférences et par conséquent minimiser les temps de calcul, les points sont groupés dans des régions (voir Fig. VI.8).

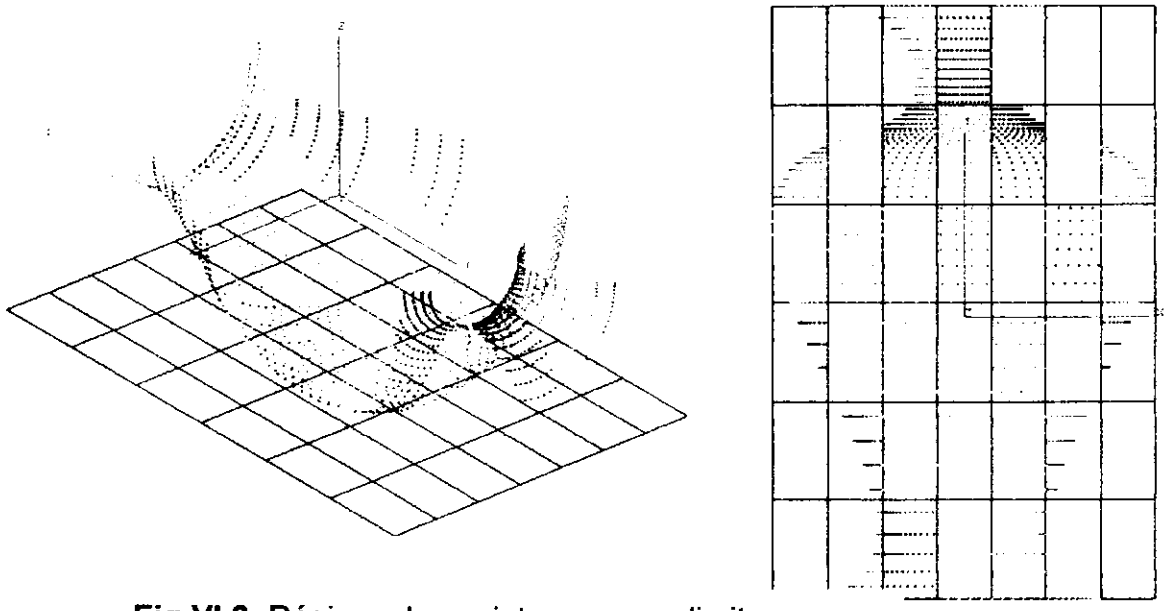


Fig.VI.8. Régions des points avec ces limites.

Nous définissons une région comme étant l'ensemble des points appartenant au même intervalle délimité par xmin, xmax, ymin et ymax. Pour générer les régions, nous avons appliqués l'algorithme suivant :

III.2.2. Algorithme de génération des régions de points :

- La taille de la région suivant $\vec{x}$ est la longueur de la surface suivant $\vec{x}$ divisée par le nombre de colonnes,
- La taille de la région suivant $\vec{y}$ est la longueur de la surface suivant $\vec{y}$ divisée par le nombre de lignes,
- Pour chaque point de coordonnées x et y de la liste des sommets, ces positions i et j sont données par :

  La position j = x / longueur de la région suivant $\vec{x}$

  La position i = y / longueur de la région suivant $\vec{y}$

III.2.3.Algorithme de détection des interférences :

Après la création des régions de points, il faut définir le rayon d'outil pour chaque sommet. Pour éviter les interférences, nous avons appliqué l'algorithme suivant :

1. ***Vérification du l'interférence du surface*** :

   - Écrire une fonction *void interférence ()*. Cette fonction permet de parcourir la liste des sommets et définir ymax, ymin, xmax, xmin, zmax, zmin pour chaque région.

2. ***Vérification du l'interférences du région*** :

   - Écrire une fonction *void interférence (sommet &s)*. Cette fonction permet de parcourir la liste des sommets pour chaque région et d'initialiser les *cordonnées du centre d'outil et le rayon d'outil de chaque sommet s.*

   - Écrire une fonction *void interférence (sommet& p, sommet* q)*. Cette fonction permet de comparer la distance entre le centre d'outil de sommet s et le sommet q avec le rayon d'outil de sommet s. Cette fonction va nous permettre de vérifier le critère principal de l'algorithme de l'interférence. Si q est différent de S et la distance entre les deux sommets S et q est inférieure au rayon il faut appeler la fonction de correction du rayon et passer au sommet suivant dans la région.

III.2.4. Classification des sommets :

En raison d'absence de modèle mathématique continu du nuage de points et donc l'impossibilité de déterminer exactement la forme locale d'un point à partir des rayons de courbures non connus, et comme la triangulation nous donne une relation de voisinage entre les triangles et les points, c'est cette information que nous avons utilisée pour connaître la forme locale d'un point. La première étape consiste à estimer la normale en chaque point en se basant sur les triangles partageant ce sommet (voir Fig. VI.9). La normale est estimée à partir des normales des triangles et des aires de ces triangles.

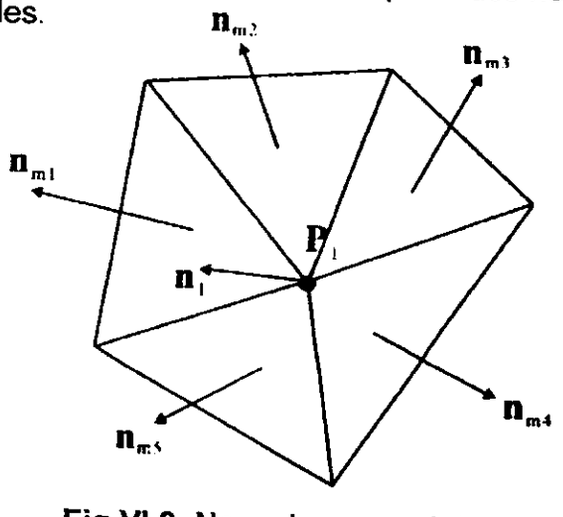


Fig.VI.9. Normal en un point

Une méthode simple permettant de connaître la forme locale d'un point consiste à déterminer la position des points voisins par rapport au plan tangent du point considéré (voir Fig. VI.10.). Nous devons noter que le vecteur normal au plan est le vecteur normal au sommet. Un sommet est classé parmi les cas suivants :

a. Si le sommet *S* est ***convexe,*** tous ses voisins sont au dessous de son plan tangent.

b. Le cas où *S* est ***convexe parabolique*** est un cas limite des sommets convexes lorsque *S* appartient à une arête frontière de deux plans : certains sommets voisins à *S* sont confondus au plan tangent.

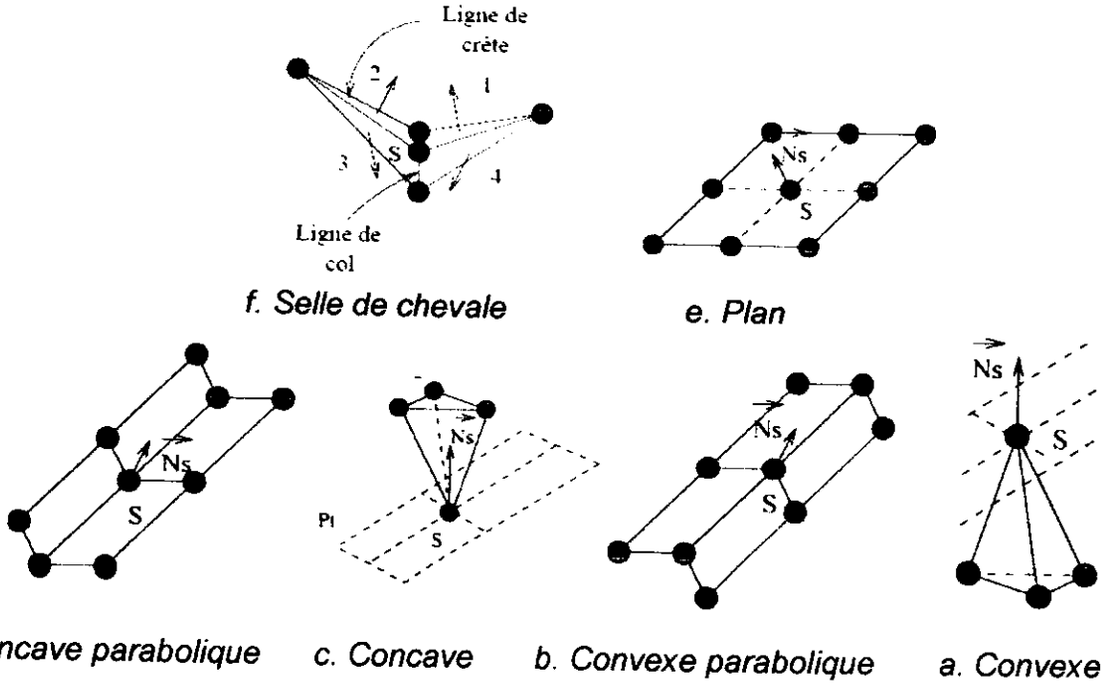
c. Si le sommet *S* est ***concave,*** tous ses voisins sont au dessus de son plan tangent.

d. Le cas où *S* est ***concave parabolique*** est un cas limite des sommets concaves lorsque *S* appartient à une arête frontière de deux plans : certains sommets voisins à *S* sont confondus au plan tangent.

e. Si le sommet *S* est à la fois *convexe* et *concave,* c'est qu'il appartient au plan, on dit alors que *S* est ***plan.***

f. Un sommet dit ***selle de cheval*** a la particularité d'être à l'intérieur de l'enveloppe convexe de ses voisins ; ils sont schématiquement représentés par une ligne de crête et une ligne de col.

Il est important de noter que cette classification nécessite un nuage de points très dense pour donner de bons résultats.



f. Selle de chevale *e. Plan*

c. Concave parabolique *c. Concave* *b. Convexe parabolique* *a. Convexe*

Fig.VI.10. Forme locale d'un point

III.2.5. Visualisation des sommets selon la forme locale :

L'utilisateur a la possibilité de visualiser les sommets selon leurs formes locales s'il a coché la case prise dans la partie de visualisation. La figure suivante montre les différentes couleurs des formes locales.

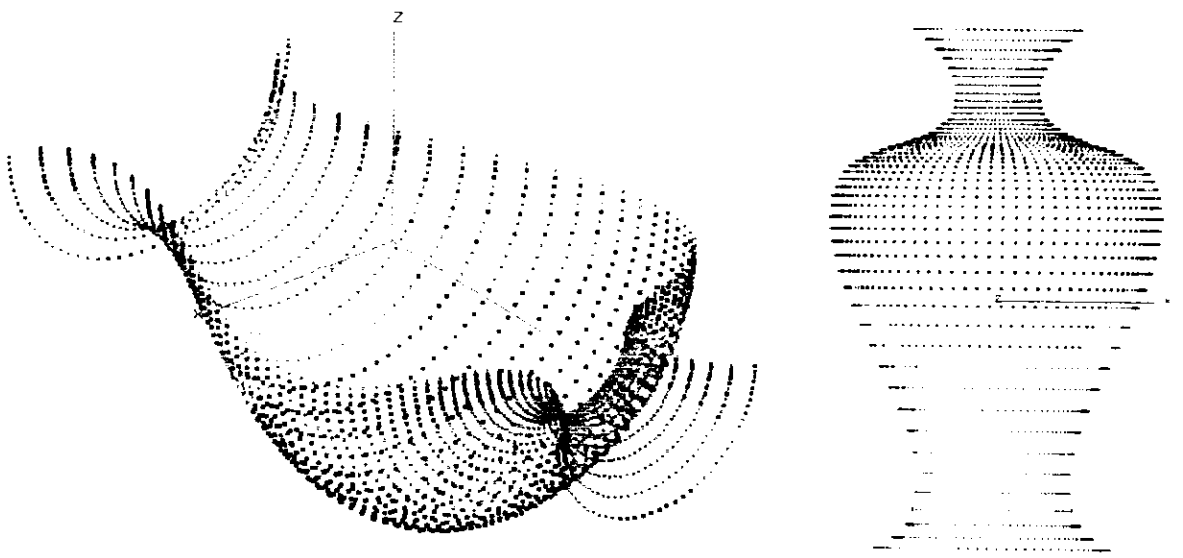


Fig.VI.11. Couleurs des formes locales de la surface.

Nous avons maintenant le nouveau rayon corrigé pour chaque région qui sera affiché en cliquant sur le bouton « affichage des rayons optimums » (voir Fig. VI.12.).

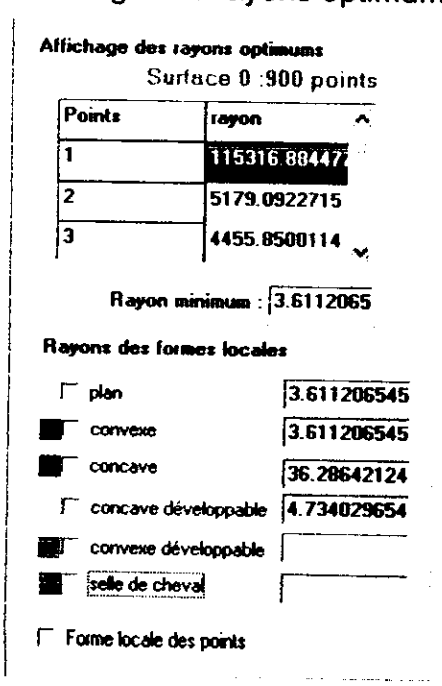


Fig.VI.12. Fenêtre d'affichage des rayons.

IV. CONCLUSION :

Dans ce chapitre, nous avons présentés les formes que nous avons créées et intégrées à une application de CFAO permettant de générer une triangulation à partir d'un nuage de points, de filtrer les triangles, de déterminer les différentes formes locales et de calculer l'outil optimum à associer pour chaque région.

Chapitre VII

Tests et Validations

I. INTRODUCTION :

Après avoir présenté l'implémentation de notre application logicielle, nous passons à la dernière étape qui celle des tests et validations. Cette étape est très importante puisqu'elle permet de valider le logiciel par rapport au cahier des charges et de faire les corrections nécessaires en cas de problèmes qui peuvent apparaître pendant l'exécution.

II. TESTS ET VALIDATIONS :

Cette phase de validation consiste à tester les cas d'utilisation de triangulation. Pour les tests, nous allons considérer deux exemples d'application. Considérons le nuage des points d'une surface gauche « surface ondulée » car elle comporte des parties convexes et des parties concaves et des parties selle de chevale, ce qui permet de localiser les interférences de l'outil dans les différentes régions des surfaces et la deuxième surface « demi_vase_cavité». Nous commençons par l'ouverture d'une surface. Les différentes étapes de génération de la triangulation de Delaunay sont données par le diagramme suivant :

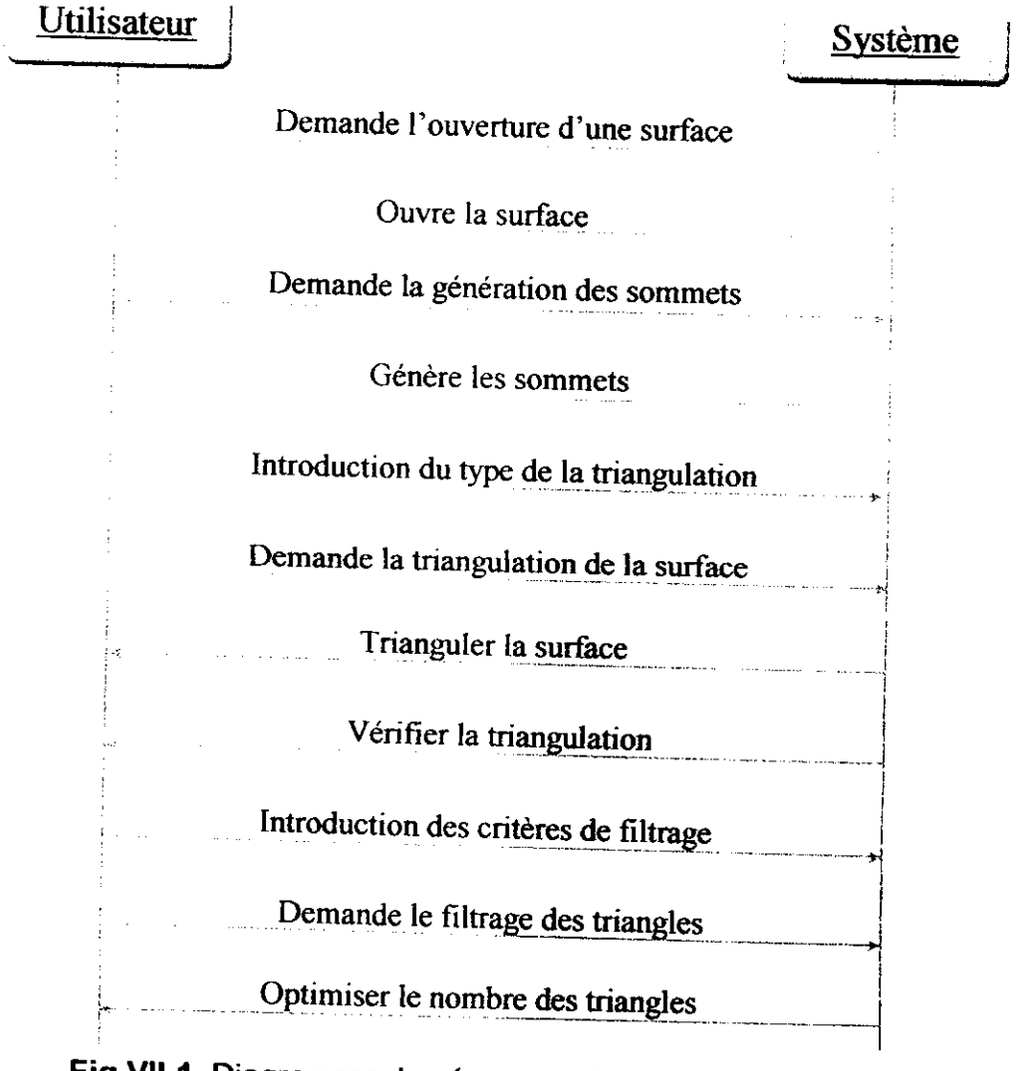


Fig.VII.1. Diagramme de séquence d'approximation du modèle.

Les différentes étapes de la détermination des rayons optimums sont données par le diagramme suivant :

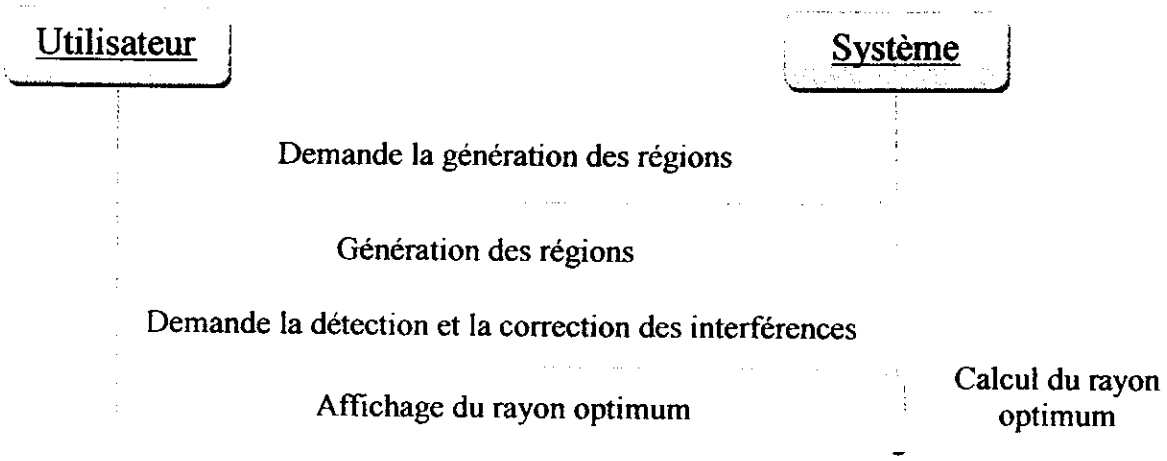


Fig.VII.2. Scénario de choix de l'outil optimum.

II .1. Première surface :

Le modèle CAO de la première surface (demi_vase_cavite) que nous considérons est représenté par la figure suivante :

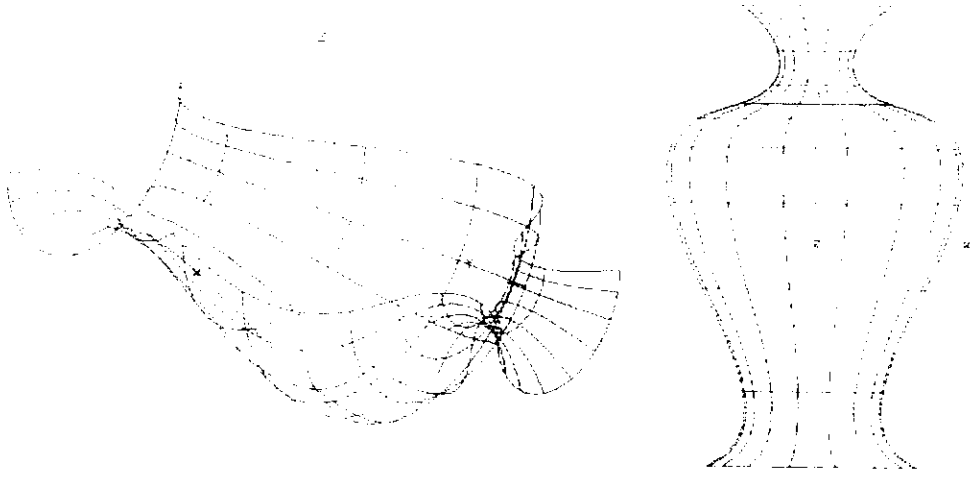


Fig.VII.3. Surface « demi_vase_cavite ».

II.1.1 Triangulation :

Après l'ouverture de cette surface, nous activons la forme de la triangulation et nous avons deux modes pour générer la triangulation de Delaunay. Dans cet exemple nous allons introduire les paramètres suivants afin de simuler l'opération de digitalisation de cette surface sur une MMT:

- 50 points dans la direction u.
- 50 points dans la direction v.

Le nuage de points généré à partir de ces données est représenté par la figure suivante avec une vie dans l'espace et une projection dans le plan XY.

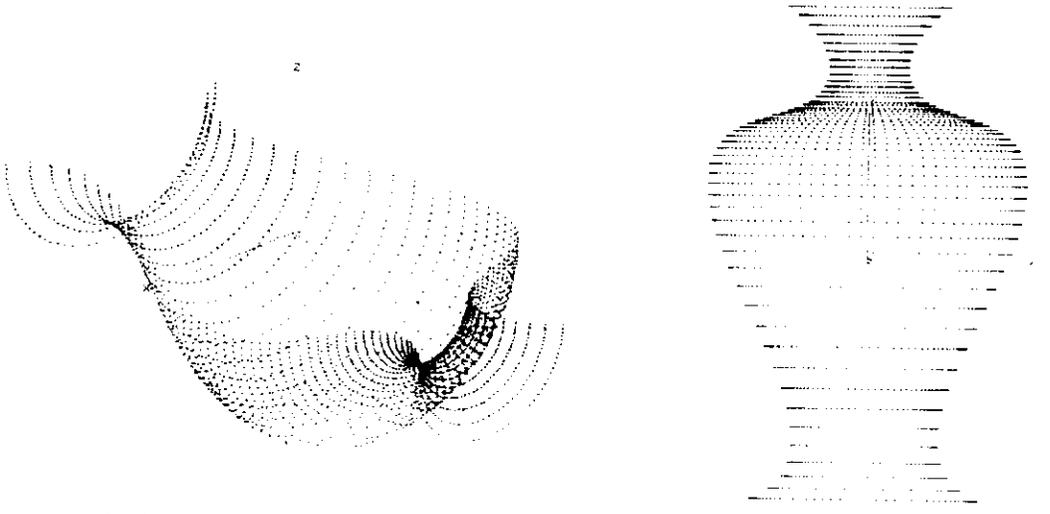


Fig.VII.4. Nuage de points de la surface « demi_vase_cavite ».

Avant de commencer la triangulation de Delaunay, devons en premier lieu déterminer les coordonnées maximales et minimales suivant les deux directions X et Y. Ces informations seront utilisées dans le calcul des coordonnées des sommets d'un triangle qui englobe le nuage de points (triangle départ de la triangulation). Ce triangle est représenté par la figure suivante :

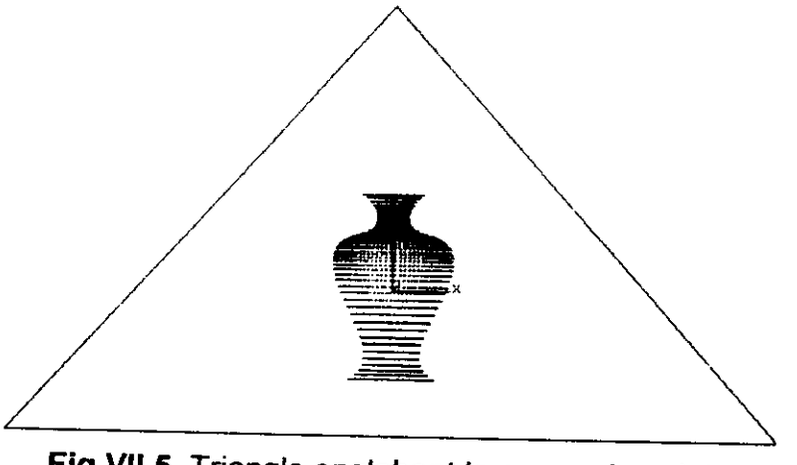


Fig.VII.5. Triangle englobant le nuage de points

Pour la génération de la triangulation, nous avons la possibilité d'utiliser un mode séquentiel d'insertion des points ou un mode aléatoire.

Mode séquentiel :

Dans ce mode, les points sont insérés dans leur ordre de lecture soit point par point (insertion manuelle) soit globalement (insertion automatique). La figure suivante montre ce mode d'insertion.

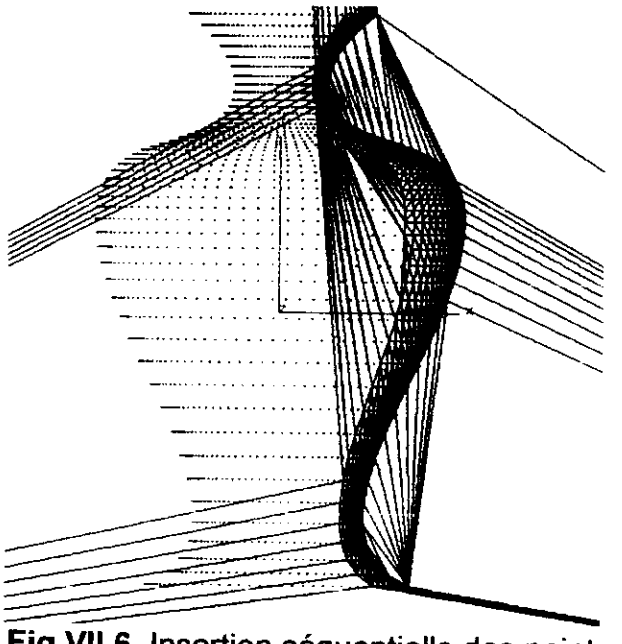


Fig.VII.6. Insertion séquentielle des points.

Mode aléatoire :

Dans ce mode, les points sont insérés dans un ordre aléatoire soit point par point (insertion manuelle) soit globalement (insertion automatique). La figure suivante montre ce mode d'insertion.

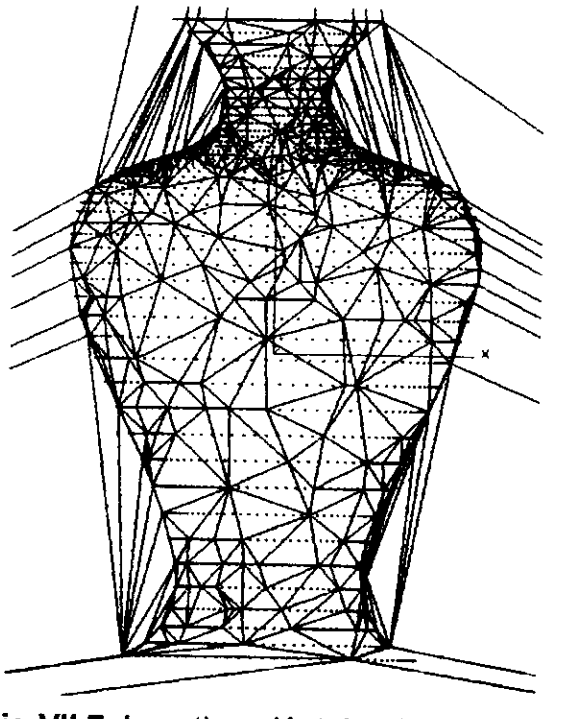


Fig.VII.7. Insertion aléatoire des points.

A la fin de la triangulation, nous obtenons comme résultats un ensemble de points de la surface (sommets) et un ensemble de triangles qui vérifient les contraintes de la

triangulation de Delaunay. Le résultat final de la triangulation pour les deux modes est donné par la figure suivante :

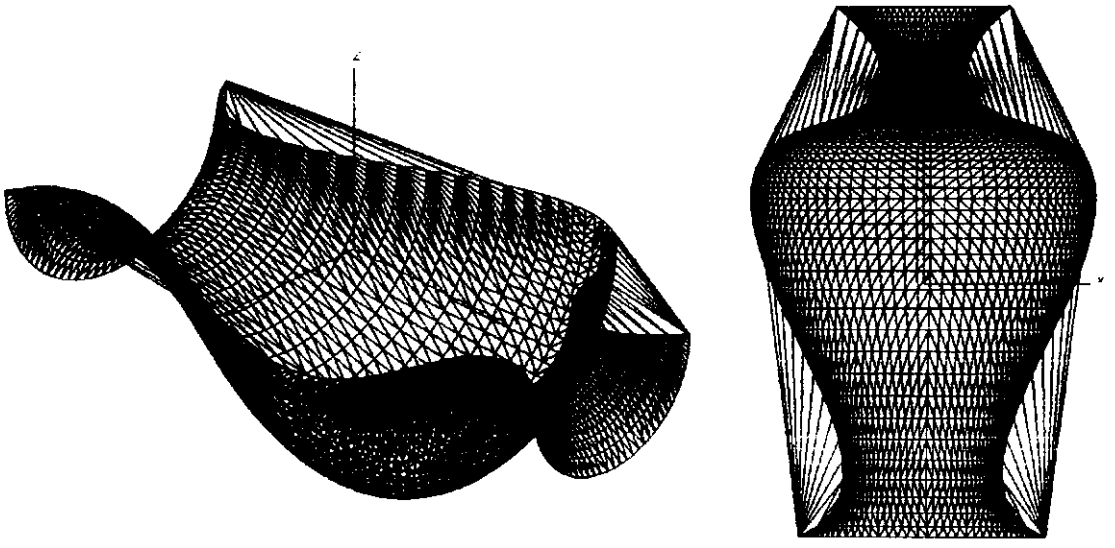


Fig.VII.8. Triangulation finale de Delaunay.

L'ensemble des triangles générés ne donne pas forcément la vraie forme de l'objet. A cet effet, il faut passer à une étape d'élimination d'un certain nombre de triangles afin d'obtenir la forme réelle de l'objet et donc la triangulation finale. Cette étape est appelée « filtrage » et elle est basée dans notre cas sur deux critères à savoir la qualité minimale d'un triangle et le rayon maximum du cercle circonscrit à ce triangle.

Dans l'exemple que nous avons considéré, nous obtenons les indications suivantes pour l'ensemble des triangles :

- Qualité minimale du triangle = 0.00238497222962255.
- Qualité maximale du triangle = 1.66433048720132.
- Rayon minimum du cercle circonscrit = 0.433518028601701 mm.
- Rayon maximum du cercle circonscrit = 163.482062263236 mm.

Afin de montrer le filtrage des triangles, nous avons considéré quatre cas :

1^er^ cas :

- Qualité minimale du triangle = néant.
- Rayon maximum du cercle circonscrit = 3 mm.

Le résultat du filtrage est donné par la figure suivante où nous pouvons voir que nous n'avons pas obtenu la forme réelle de la surface :

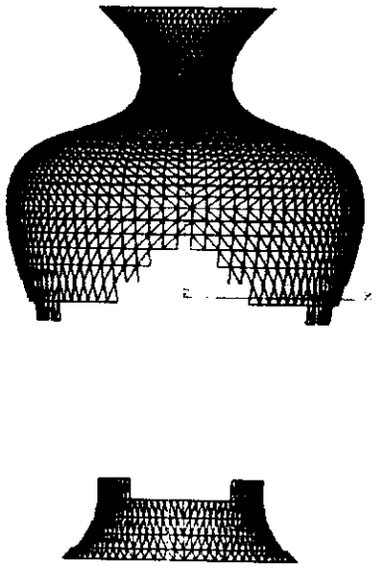


Fig.VII.9. Premier cas de filtrage des triangles.

2ème cas :

- Qualité minimale du triangle = 0.55
- Rayon maximum du cercle circonscrit = néant

Le résultat du filtrage est donné par la figure suivante où nous pouvons voir que nous n'avons pas obtenu la forme réelle de la surface :

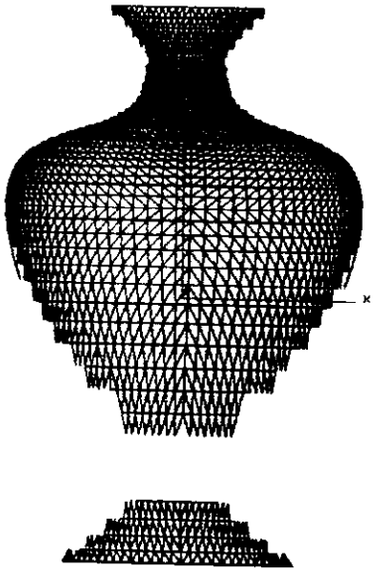
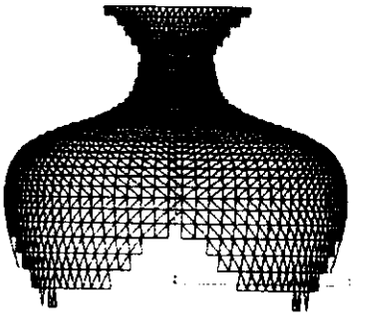


Fig.VII.10. Deuxième cas de filtrage des triangles.

3ème cas :

- Qualité minimale du triangle = 0.55
-
- Rayon maximum du cercle circonscrit = 3 mm

Le résultat du filtrage est donné par la figure suivante où nous pouvons voir que nous n'avons pas obtenu la forme réelle de la surface :



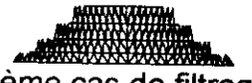


Fig.VII.11. Troisième cas de filtrage des triangles.

4ème cas :

- Qualité minimale du triangle = 0.2
- Rayon maximum du cercle circonscrit = 7 mm

Le résultat du filtrage est donné par la figure suivante où nous pouvons voir que nous avons obtenu la forme réelle de la surface :

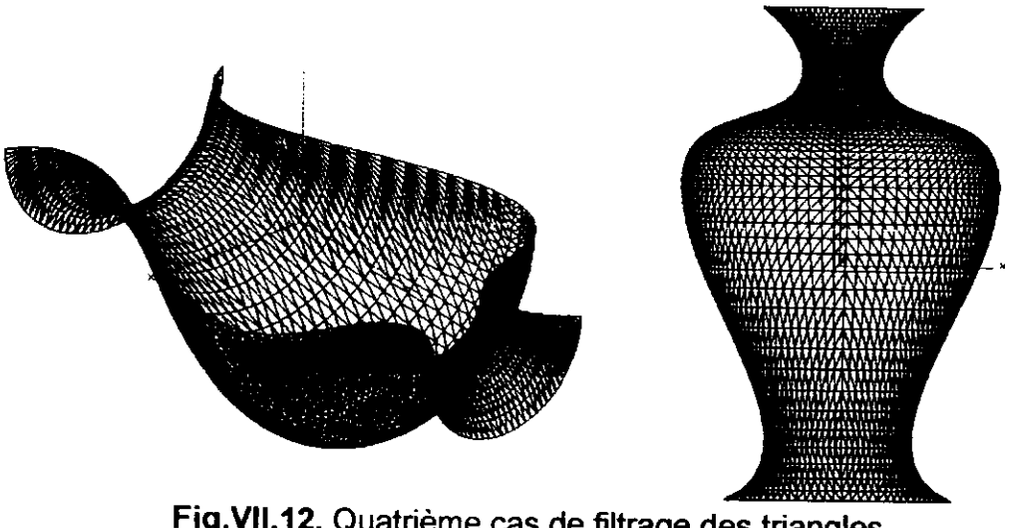


Fig.VII.12. Quatrième cas de filtrage des triangles.

Après l'obtention de la triangulation finale, nous pouvons visualiser les résultats suivants :

- le vecteur normal en chaque sommet (voir Fig.VII.13),
- la forme locale de chaque sommet (voir Fig.VII.14),
- la forme locale des triangles (voir Fig.VII.15),
- la normale pour chaque triangle (voir Fig.VII.16).

Nous pouvons visualiser la forme locale et le vecteur normal de chaque point et la forme locale et le vecteur normal de chaque triangle.

Fig.VII.13. Normale pour chaque sommet.

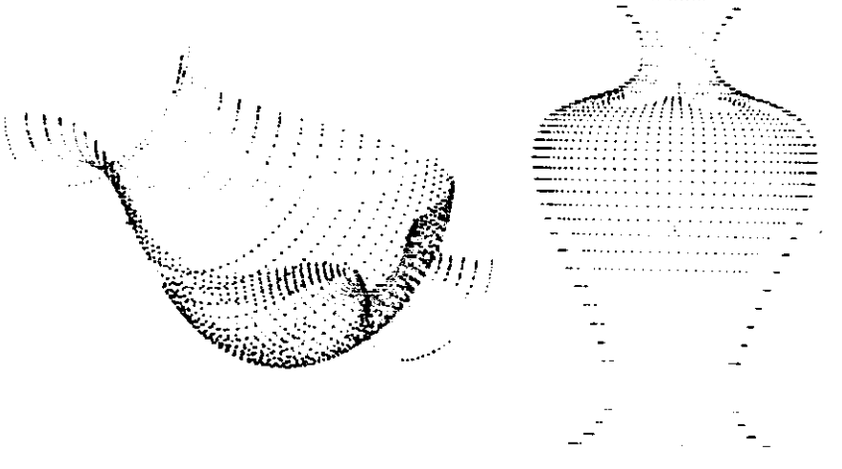


Fig.VII.14. Forme locale de chaque sommet.

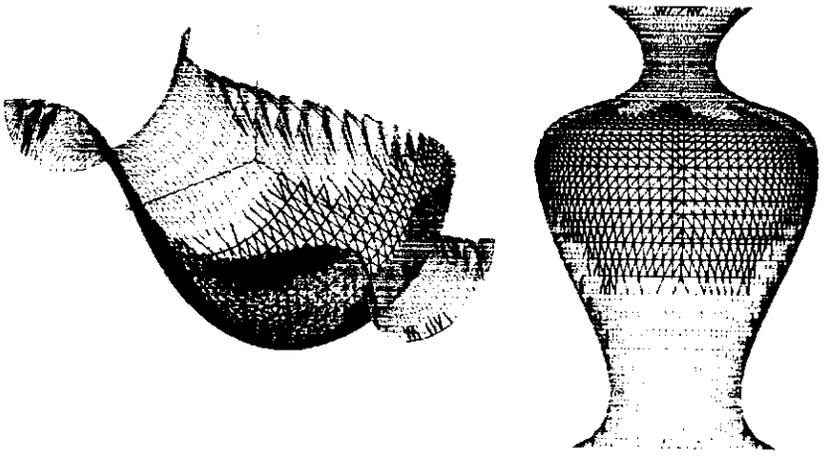


Fig.VII.15. Forme locale de chaque triangle.

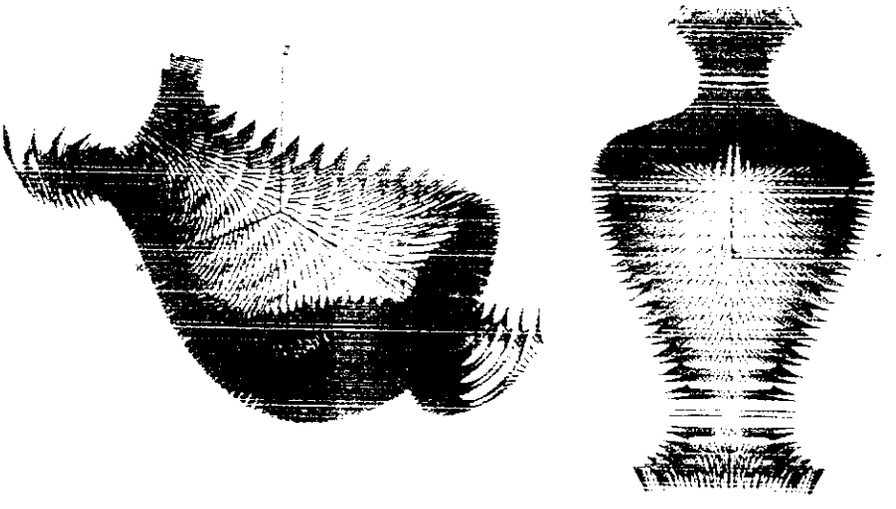


Fig.VII.16. Normale de chaque triangle.

II .1.2. Choix d'outil optimum :

Pour lancer la détection des interférences, nous activons la forme de rayon optimum et nous introduisons le nombre de lignes et le nombre de colonnes pour subdiviser les sommets du nuage de points dans une matrice des zones. Cette matrice est utilisée pour localiser les zones de test de chaque sommet. Pendant le test des interférences, les rayons d'outils théoriques des sommets sont corrigés. Dans notre exemple, nous utilisons les paramètres suivants :

- Le nombre de lignes est égal à 6
-
- Le nombre de colonnes est égal à 7.

La figure suivante montre les différentes régions avec ses points et ses limites.

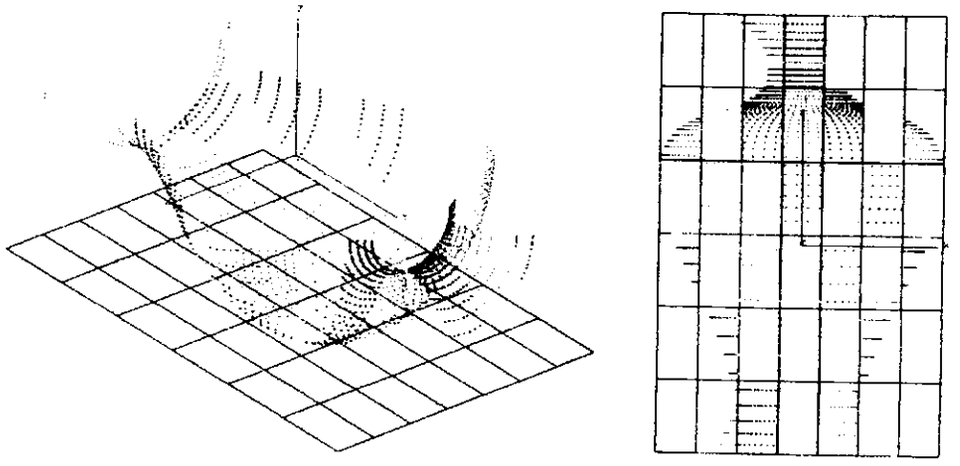


Fig.VII.17. Régions du nuage de points avec ses limites.

Après la subdivision du nuage de points, nous allons passer à l'étape de détection et de correction des interférences pour chaque sommet afin de déterminer le rayon optimum de l'outil hémisphérique. Dans cette étape, nous avons la possibilité de visualiser la sphère de l'outil pendant les calculs. Les figures suivantes montrent les rayons d'outils pour différents points de la triangulation où nous pouvons constater que la dimension de la sphère (son rayon) s'adapte avec les différentes formes locales du nuage de points (voir Fig.VII.18).

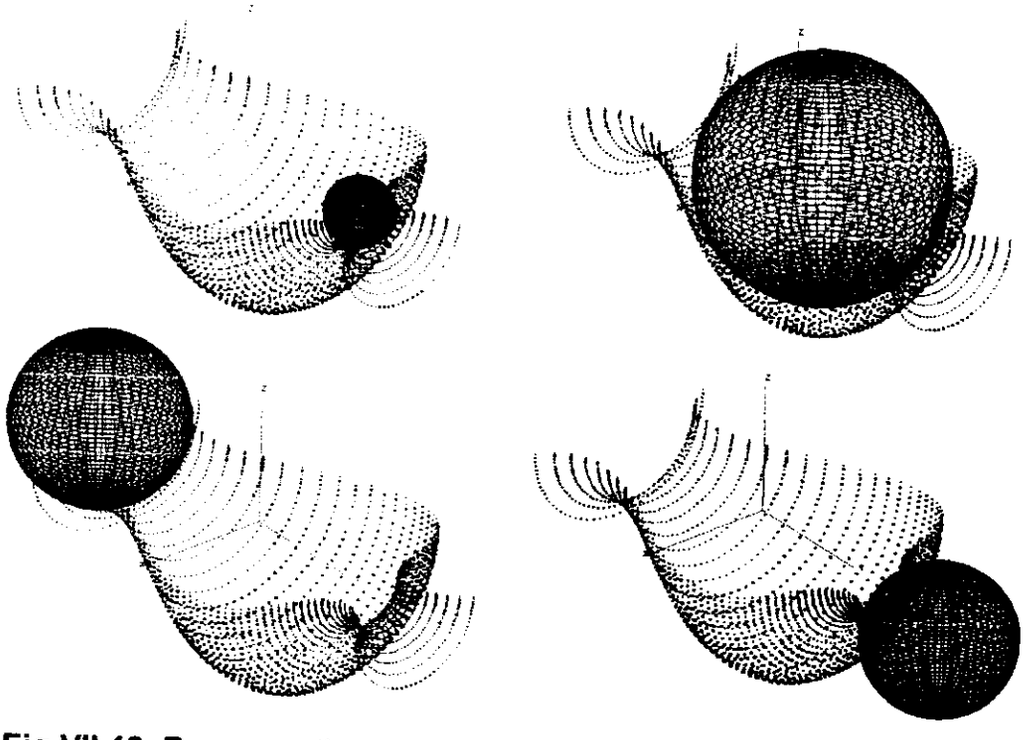


Fig.VII.18. Rayons optimums de la sphère pour différents sommets.

Une fois que le rayon optimum est déterminé pour chaque sommet, il est possible maintenant de déterminer le rayon optimum de l'outil à associer à chaque région. La figue suivante montre les différentes formes locales de points ainsi que les rayons d'outils optimums.

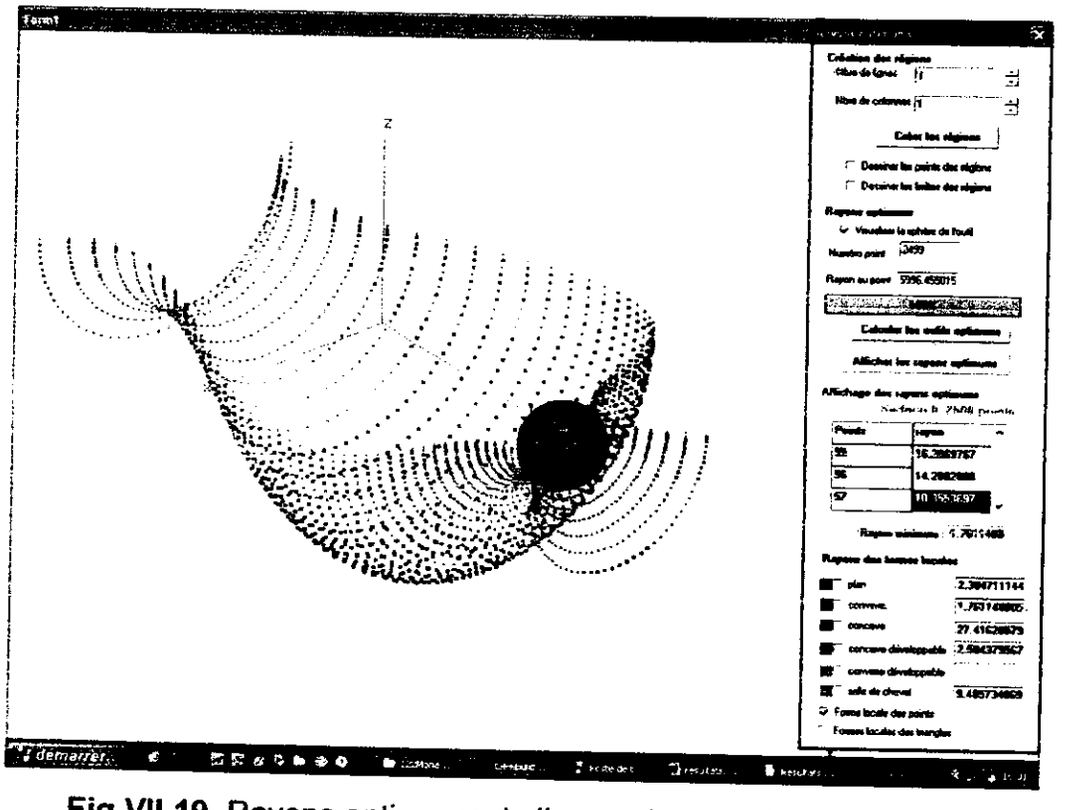


Fig.VII.19. Rayons optimums de l'ensemble des formes locales.

II.2. Deuxième surface :

Le modèle CAO de la deuxième surface (surface ondulée) que nous considérons est représenté par la figure suivante :

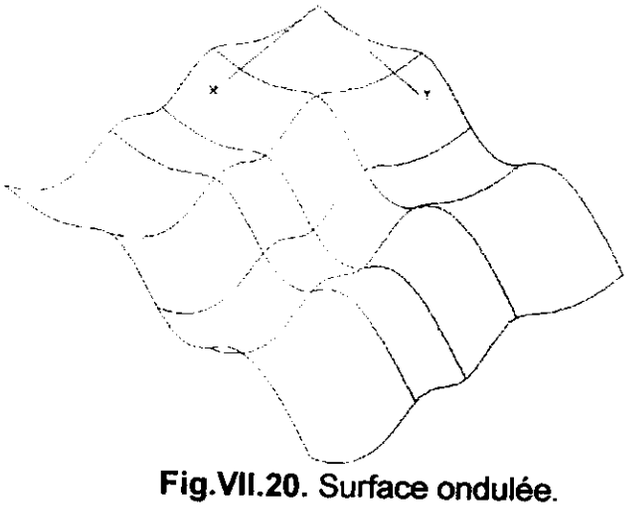


Fig.VII.20. Surface ondulée.

II.2.1. Triangulation :

Après l'ouverture de cette surface, nous activons la forme de la triangulation et nous avons deux modes pour générer la triangulation de Delaunay. Dans cet exemple nous allons introduire les paramètres suivants afin de simuler l'opération de digitalisation de cette surface sur une MMT:

- 50 points dans la direction u.
- 50 points dans la direction v.

Le nuage de points généré est représenté par la figure suivante.

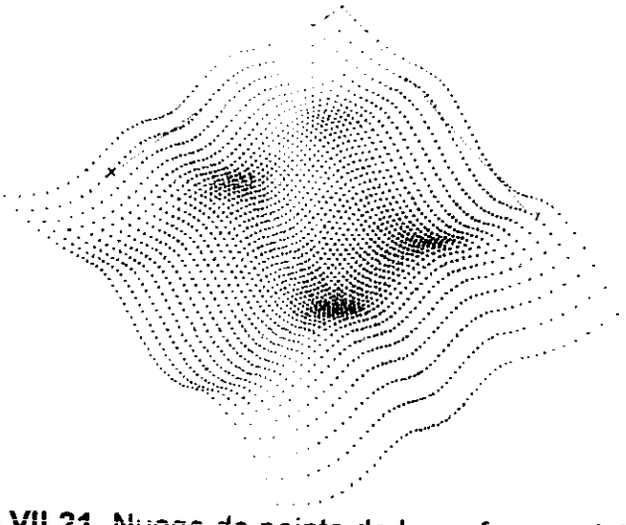


Fig.VII.21. Nuage de points de la surface ondulée.

Le triangle qui englobe le nuage de points (triangle départ de la triangulation) est représenté par la figure suivante :

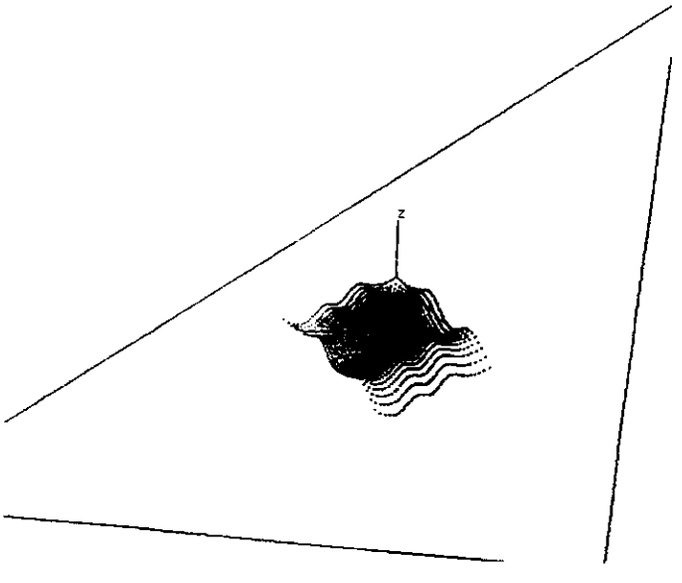


Fig.VII.22. Triangle englobant le nuage de points.

Pour la génération de la triangulation, nous avons utilisé un mode séquentiel et un mode aléatoire. Les deux figures suivantes montrent les deux modes d'insertion :

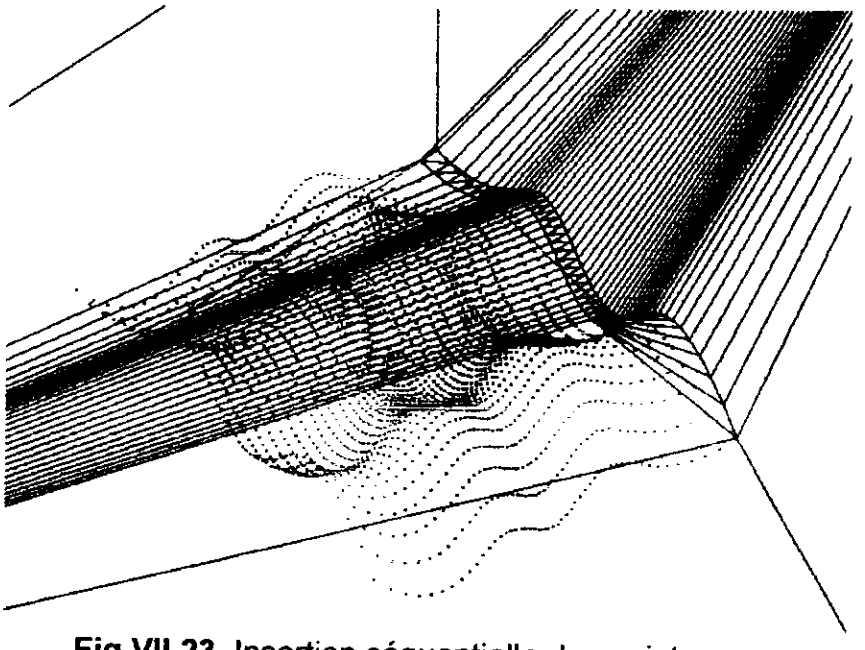


Fig.VII.23. Insertion séquentielle des points.

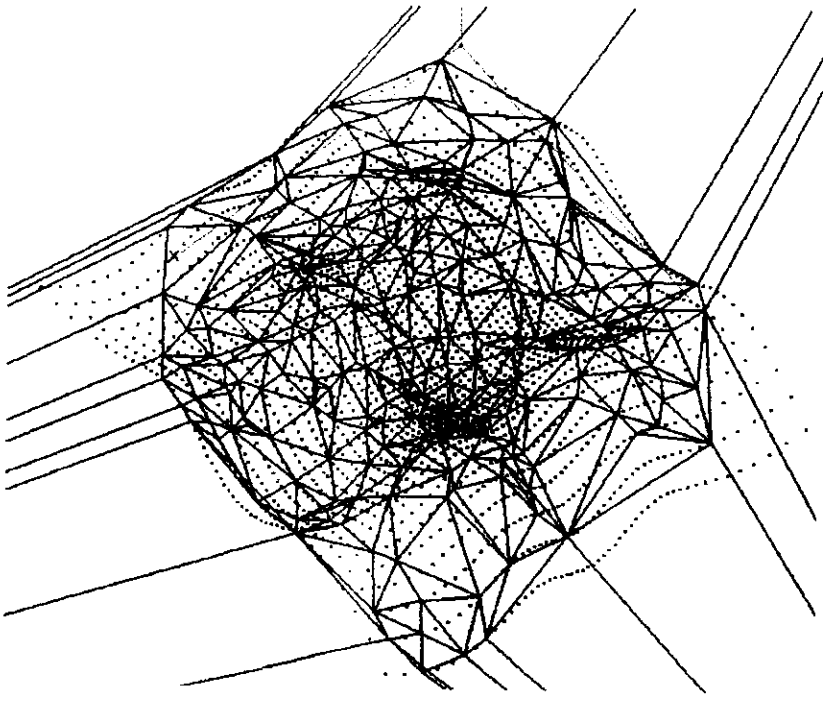


Fig.VII.24. Insertion aléatoire des points.

Le résultat final de la triangulation pour les deux modes est donné par la figure suivante :

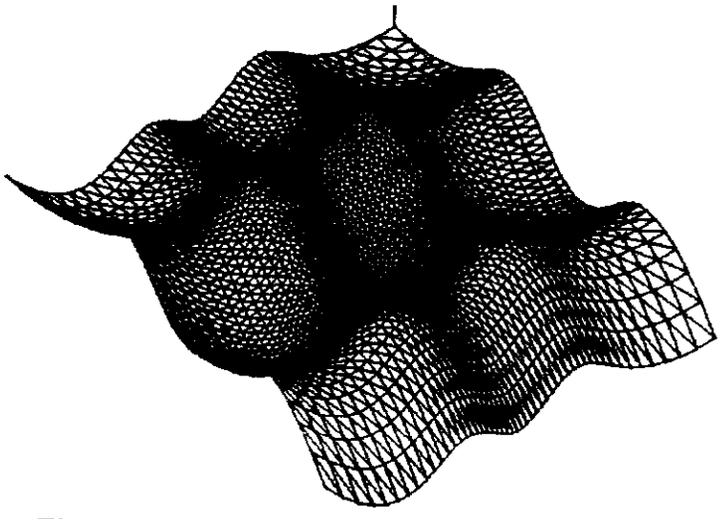


Fig.VII.25. Triangulation finale de Delaunay.

Après l'obtention de la triangulation finale, nous pouvons visualiser les résultats suivants :

- le vecteur normal en chaque sommet (voir Fig.VII.26),
- la forme locale de chaque sommet (voir Fig.VII.27),
- la forme locale des triangles (voir Fig.VII.28),
- la normale pour chaque triangle (voir Fig.VII.29).

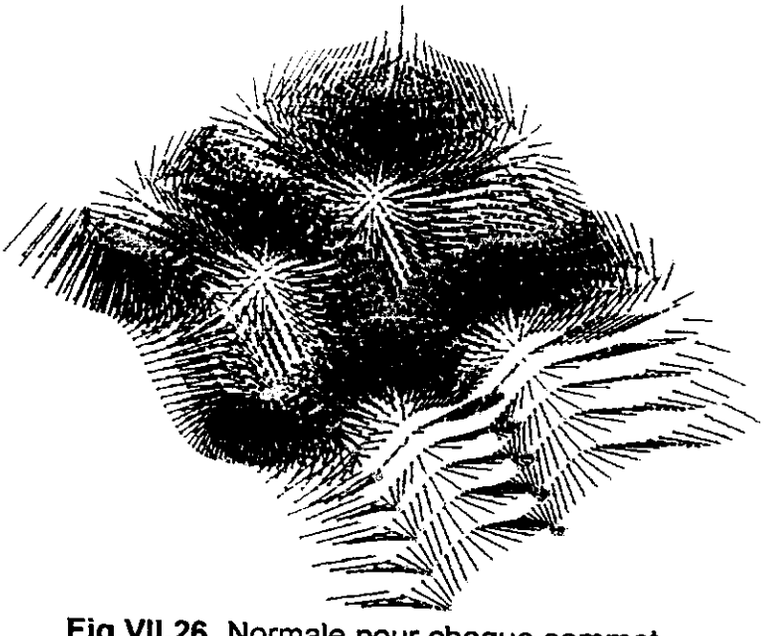


Fig.VII.26. Normale pour chaque sommet.

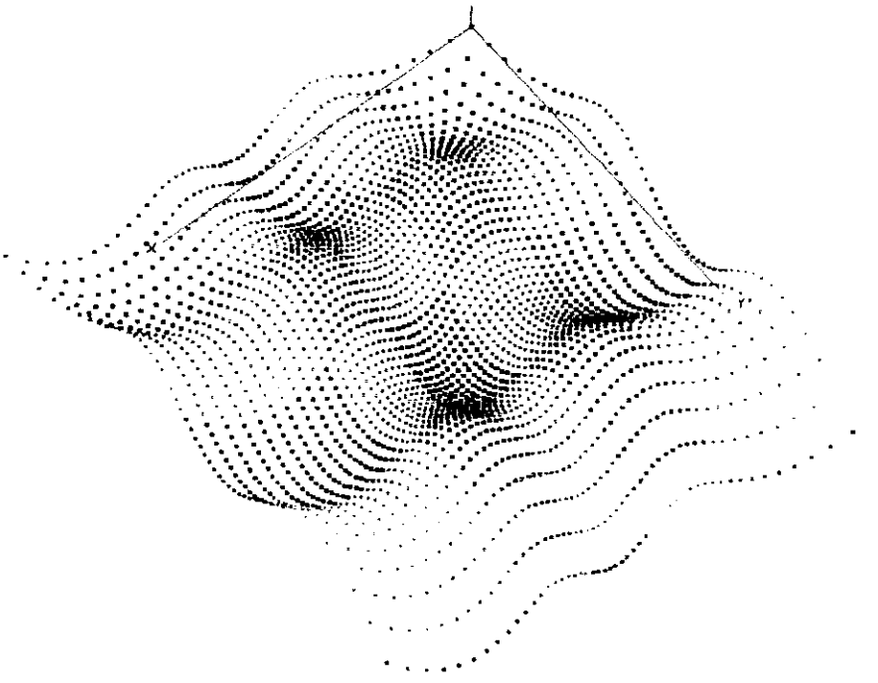


Fig.VII.27. Forme locale de chaque sommet.

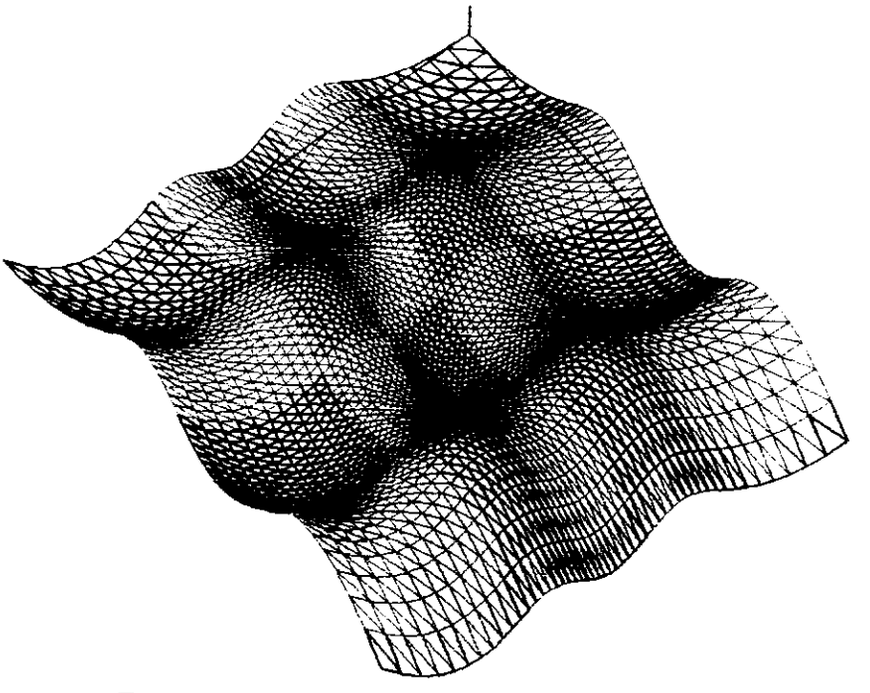


Fig.VII.28. Forme locale de chaque triangle.

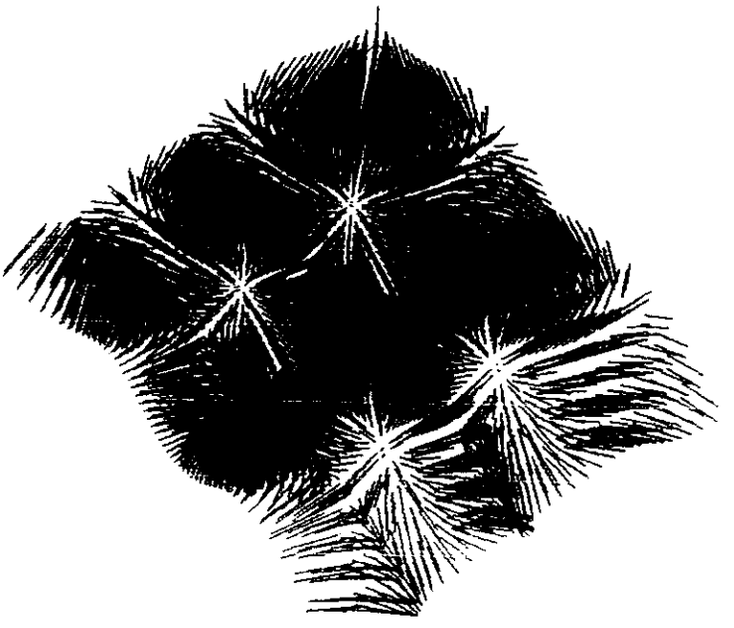


Fig.VII.29. Normale de chaque triangle.

II .2.2. Choix d'outil optimum :

Pour lancer la détection des interférences, nous activons la forme de rayon optimum et nous introduisons le nombre de lignes et le nombre de colonnes pour subdiviser les sommets du nuage de points dans une matrice des zones. Dans notre exemple, nous utilisons les paramètres suivants :

- Le nombre de lignes est égal à 6
- Le nombre de colonnes est égal à 7.

La figure suivante montre les différentes régions avec ses points et ses limites.

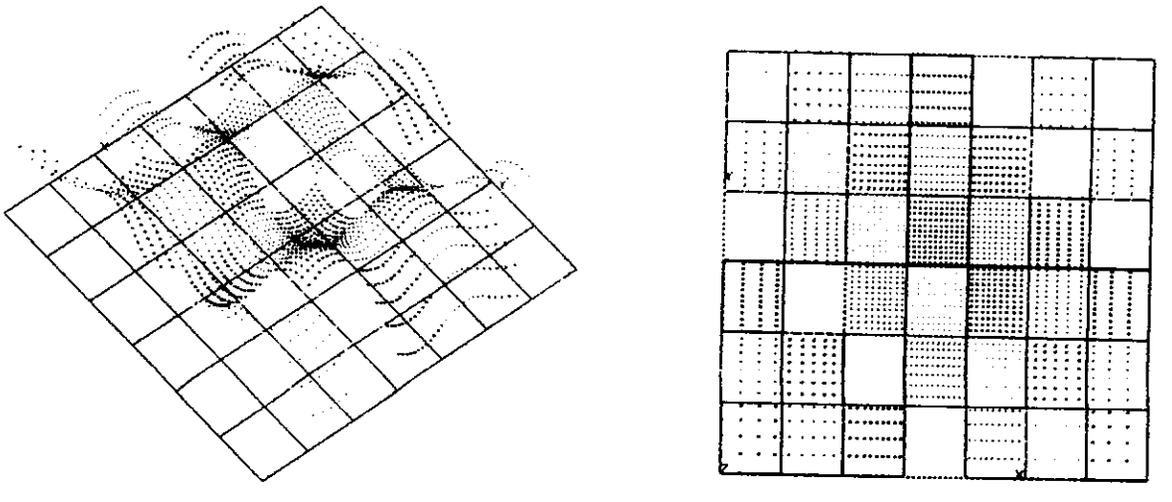


Fig.VII.30. Régions du nuage de points avec ses limites.

Après la subdivision du nuage de points, nous allons passer à l'étape de détection et de correction des interférences pour chaque sommet afin de déterminer le rayon optimum de l'outil hémisphérique. Dans cette étape, nous avons la possibilité de visualiser la sphère de l'outil pendant les calculs. Les figures suivantes montrent les rayons d'outils pour différents points de la triangulation où nous pouvons constater que la dimension de la sphère (son rayon) s'adapte avec les différentes formes locales du nuage de points (voir Fig.VII.31).

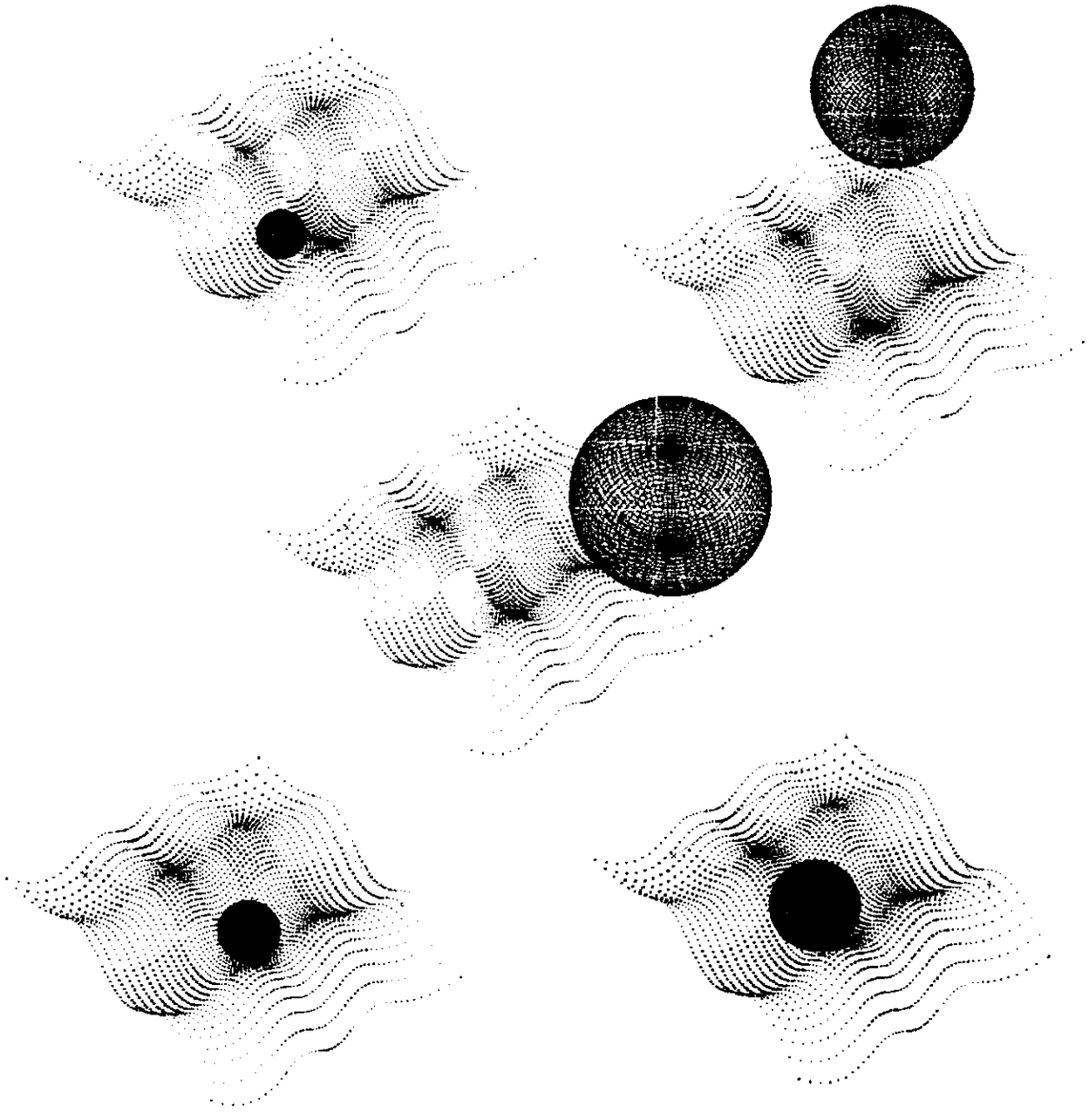


Fig.VII.31. Rayons optimums de la sphère pour différents sommets

Une fois que le rayon optimum est déterminé pour chaque sommet, il est possible maintenant de déterminer le rayon optimum de l'outil à associer à chaque région. La figue suivante montre les différentes formes locales de points ainsi que les rayons d'outils optimums.

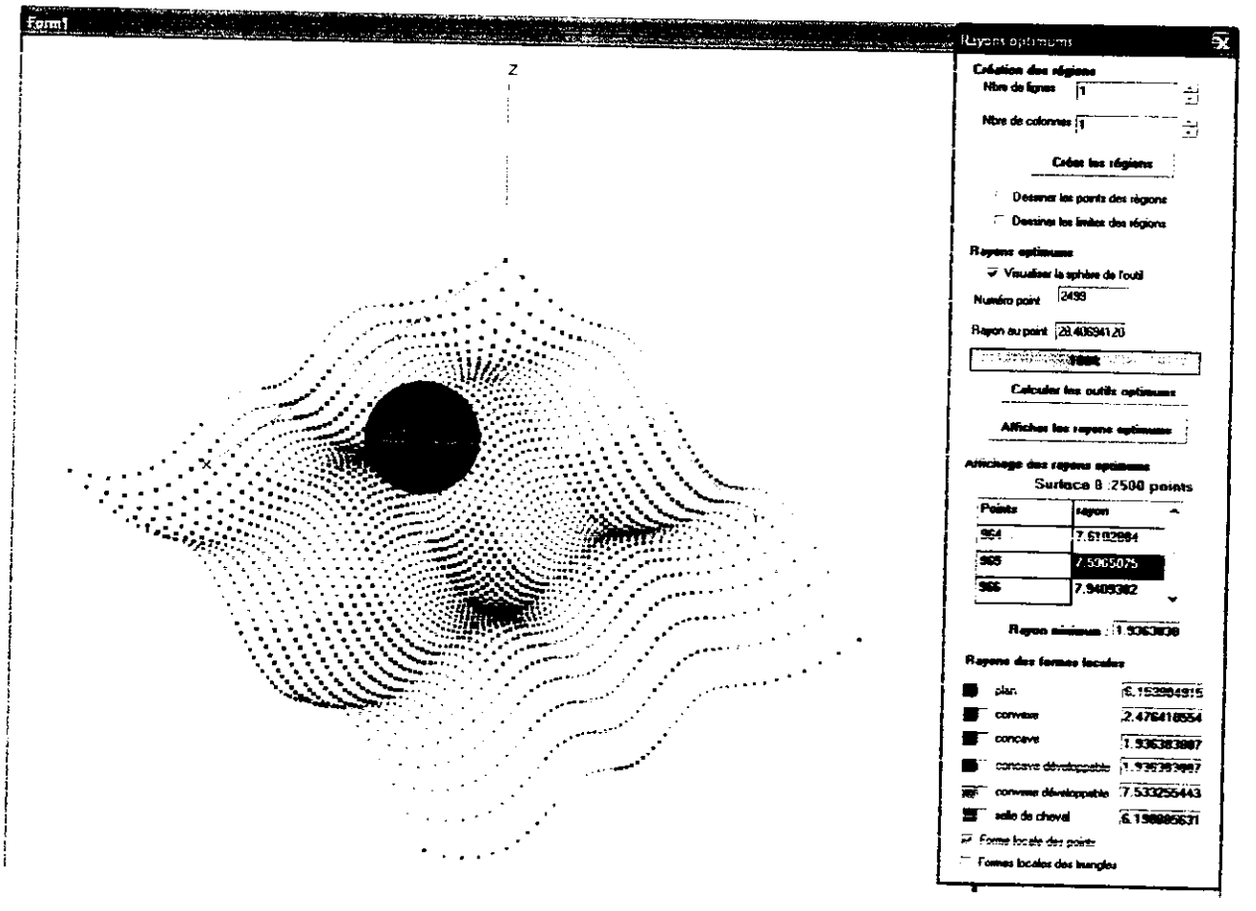


Fig.VII.32. Rayons optimums de l'ensemble des formes locales.

III. CONCLUSION :

Nous avons présenté dans ce chapitre les différents scénarios des fenêtres de notre application logicielle en prenant un exemple de deux surfaces pour le testé et valider toutes les fonctions et les étapes de la triangulation jusqu'à la détermination des rayons optimums des outils.

Conclusion Générale

CONCLUSION GENERALE

Le travail que nous avons présenté dans ce mémoire est relatif à la triangulation d'un nuage de points, au groupement des triangles en des régions distinctes en fonction de la forme locale des points et enfin la détermination du rayon d'outil hémisphérique optimum associé à chaque région pour la finition des surfaces gauches sur des fraiseuses à commande numérique à 03 axes.

Dans ce mémoire, nous avons étudié en premier lieu les méthodes de conception et de représentation des surfaces gauches en appuyant notre étude sur les surfaces B-Spline et NURBS. Ensuite, nous avons présentés les machines utilisées dans l'usinage de ces surfaces et la syntaxe des programmes G-Code. Par la suite, nous avons étudié le processus du Reverse Engineering et la triangulation de Delaunay. A la fin, nous avons montré les différentes fonctions offertes par l'application logicielle développée suivi d'un test de validation.

Le résultat de notre travail est l'intégration de modules à l'application logicielle graphique de conception et d'usinage des surfaces gauches développée par l'équipe « CFAO » de la Division Productique et Robotique du Centre de Développement des Technologies Avancées -CDTA- permettant de :

- ✓ Générer une triangulation qui approxime le mieux un nuage de points en utilisant la triangulation de Delaunay.
- ✓ Filtrer des triangles en fonction de deux critères à savoir la qualité minimale du triangle et le rayon maximum du cercle circonscrit à un triangle.
- ✓ Déterminer la forme locale des points.
- ✓ Grouper les triangles en des régions distinctes en fonction de la forme locale des sommets de chaque triangle.
- ✓ Détection et correction des interférence et calcul du rayon d'outil optimum en chaque point.
- ✓ Détermination du rayon optimum pour chaque région.

En perspective de notre travail, nous recommandons de traiter les points suivants :

- ✓ Usinage des surfaces gauches en ébauche à partir d'un nuage de points régulier et non régulier.
- ✓ Usinage des surfaces gauches en finition à partir d'un nuage de points régulier et non régulier.
- ✓ Adapter les vitesses d'avance lors de l'usinage en finition avec des outils hémisphériques, cylindriques et toriques.
- ✓ Considérer l'usinage en 5 axes.
- ✓ Considérer la triangulation d'un nuage de points représentant un objet de forme quelconque.

Bibliographie

REFERENCES BIBLIOGRAPHIQUES

[1]: A. Doneddu. « *Géométrie différentielle, intégrales multiples* ». Tome 6. Editions Vuibert. 1981.

[2]: Y. Zhou. « *Surface Interpolation and Approximation* ». Rapport de projet de Master. Université de Michigan, Février 1999.

[3]:Y. Zhao. « *Problems in Surface Intersection Representation and Construction* ». Thèse de Master. Université de Michigan, Août 1998.

[4]: W. Ma and P. He. « *B-spline surface local updating with unorganized points* ». Computer Aided Design, 1998, Vol 30, No. 11, pp. 853–862.

[5]: Y. Zhao, Y. Zhou, J.L. Lowther and C.K. Shene. « *Cross-Sectional Design: A Tool for Computer Graphics and Computer-Aided Design Courses* ». 29th ASEE/IEEE Frontiers in Education, 10-13 Novembre, San Juan, Puerto Rico, Vol. II (1999), pp. (12b3-1)-(12b3-6).

[6]: L.A Piegl and W. Tiller. « *Geometry-based triangulation of trimmed NURBS surfaces* ». Computer Aided Design, 1998, Vol 30, No. 1, pp. 11–18.

[7]: M. Peternell, H. Pottmann and B. Ravani. « *On the computational geometry of ruled surfaces* ». Computer-Aided Design, 1999, Vol 31, pp. 17–32.

[8]: H.S Heo, M.S. Kim and G. Elber. « *The intersection of two ruled surfaces* ». Computer-Aided Design, 1999, Vol 31, pp. 33–50.

[9]: Djamel Chabane et Lyes Mouterfi « Automatisation de l'opération d'ébauchage des surfaces sur des fraiseuses à commande numérique à 3 axes » PFE 2005 Université de Blida.

[10]: E.Duc E. Lefur « *Machine outils à commande numérique structure modélisation et réglage* » préparation à l'agrégation de génie mécanique 16 Septembre 1997.

[11]: « *Manuel de programmation volume 1* », 1020/1040/1060.

[12]: Bernard Méry. « *Machines à commande numérique* ».

[13]: STI. GMA. « *Programmation CN* ».

[14]: Roland Maranzana. «*GPA-664 Fabrication Assistée Par Ordinateur*». École de technologie supérieure . Génie de la production automatisée.

[15]: V.T.Rajan. Optimality of the Delaunay triangulation in R^d. Discrete Computational Geometry 12; 189-202; 1994.

[16]: H. Grabovski, V. Schillen, G. Storz, G. Weres. Inspection of micro mechanical parts.
Revue de CFAO et d'informatique graphique . Vol. 15.n° 2-3-4, Décembre 2000, pp: 393-407.

[17]: F. Priéto. Métrologie assistée par ordinateur. Apport des capteurs 3D sans contact.
Thèse de doctorat, INSA de Lyon, Décembre 1999

[18]: M.Bern, D.Eppstein. Mesh generation and optimal triangulation.
In D-Z.Du and F.K.Hwang editors, computing in Euclideen Geometry, Volume 1 of Lecture Note Series on Computing, pp: 23-90. World Scientific, Singapore, 1992.

[19]: S.J. Fortune. A sweep line algorithm for Voronoï diagrams.
Algorithmica ('1987) 2, pp: 153-174.

[20]: P. Desnoguès. Triangulations et qquadriques.
Thèse de doctorat, INRIA Sophia Antipolis, Valbonne, Frace, Décembre 1993.

[21]: P.L.George, H. Bourouchaki. Triangulation de Delaunay et maillage, application aux éléments finis.
Editions Hermes, Paris 1997

[22]: J. D. Boissonnat, M. Teillaud. On the randomized construction of the Delaunay tree.
Rapport de recherche N° 1140, INRIA Sophia Antipolis, Valbonne, France Décembre 1989 révisé en Janbvier 1991.

[23]: C. L. Lawson. Software for C1 surface interpolation.
In J. R. Rice, editor,Math. Software III, pages 161-194, New York, NY, 1977. Academic Press.