

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.



**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 d'aide au choix des outils et
de stratégies d'usinage pour l'usinage en
 finition des surfaces gauches**

Présenté par : BOUKHALFA Karim
KADER Soufyane

Promoteur : MAZOUZI Meriem
Encadreur : BENDIFELLAH Hassen

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

Soutenu 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

Remercement

Avant toute chose, le véritable remerciement revient à grand Dieu tout puissant qui nous a donné la force de réaliser ce travail.

Nous remerciant notre promoteur Melle MAZOUZI Meriem et Mr BENDIFALLEH Hassen pour avoir bien voulu nous proposer ce sujet et pour son patience et son soutien continue durant notre stage. Et nous remercions aussi Mr BEY Mohamed qui a su rester à mon écoute dans les moments où j'avais besoin de lui.

Nous remercions également tous les professeurs de l'USDB qui m'ont accompagnés dans mes études supérieures, ainsi qu'à tous nos enseignants et enseignantes d'étude primaire, moyenne et lycéenne.

Nous remercions tous ceux qui ont contribué à la réalisation de ce travail de prêt ou de loin.

إهداء

أهدي هذا العمل المتواضع إلى:

أعلى وأعز ما أملك في هذا الوجود

قرة عيني و نور بصيرتي أمي و أبي

خطيبي الغالية "كريمة" عسى الله

أن يجمعنا في القريب العاجل

جميع إخوتي و أخواتي "عبد الرزاق، كمال، أعمار،

مديحة، سهيلة، جمال، محمد، فواز و خصوصا

برعمي العائلة و بسمات الحياة مهدي و هدى

كل أساتذتي الذين كان لهم الفضل في تعليمي

جميع الأصدقاء و الزملاء

كل من يكن لي الاحترام و التقدير

سفيان

Dédicace

Je dédie ce modeste mémoire :

A qui est très chère à mon coeur, ma merveilleuse mère pour son amour.

A qui est sacrifier sa vie pour nous, mon cher père.

A mes frères Hakim, Toufiq, Samir, Abdelouahab et Abdellah.

A mes soeurs Raziqa, Houria, Rafiqa, Salima, Raoya et la petite fleur Zahra.

A ma tante Fatma et tous ses fils.

A ma tante Raziqa.

Et à toute ma grande famille sans exception.

A tout les gens qui m'ont aimé, à tous mes amis et collègues sans restriction.

A Mr Bey Med et Bendifallah Hassen.

B. Karim

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 à l'usinage en finition des surfaces de formes libres sur des fraiseuses à commande numérique à 3 axes. Le but de ce travail est le développement d'une application logicielle graphique et interactive sous Windows permettant :

- ✚ La subdivision des surfaces en des régions distinctes selon leurs géométries.
- ✚ Choix des outils optimums pour chaque région permettant d'éviter les interférences.
- ✚ Choix de la stratégie d'usinage adéquate.
- ✚ Génération du code machine.

Abstract

This work fits in the setting of development of tools of conception and left surface manufacture initiated by the team Conception and Fabrication Attended by Computer (CFAO) at the level of the Division Robotics and Production of the Centre of Development of Advanced Technology (CDTA).

In this project we are interested in the machining of the free shape surfaces on the numeric tools machine at 3 axes. The goal of this work is the development of a graphic and interactive software application under Windows which allows:

- ✚ Approximate the surfaces with a set of triangles.
- ✚ Find the radius of optimum tool that avoid the interferences.
- ✚ Find the strategies for using.
- ✚ In the end the generation of the tool path and the machining program

ملخص

إنّ هذا العمل يندرج في إطار تطوير وسيلة, قادرة على تصميم و تصنيع الأسطح ذات الأشكال الحرّة و التي يقوم بتطويرها فرقة البحث الخاصة بالتصميم و التصنيع عن طريق الكمبيوتر على مستوى فرع التصنيع الآلي الخاص بمركز تطوير التكنولوجيا المتقدمة.

في هذا المشروع, نهتم بتصنيع المساحات الحرّة باستعمال آلة النقطيع الرقمية, ذات الثلاث محاور. إنّ الهدف من هذا العمل هو تطوير برنامج آلي تصويري قابل للاستعمال على نظام التشغيل ويندوز و الذي يسمح بما يلي:

- ✚ تقسيم المساحات و تشكيلها بمساحات من النقط متباينة.
- ✚ اختيار نصف قطر القاطعة الملائم و الذي يجنبنا تداخل مساحات النقطيع فيما بينها.
- ✚ اختيار طريقة التصنيع الملائمة (طريقة المحاور الأفقية زادت ثابت).
- ✚ و أخيرا كتابة برنامج التصنيع.

SOMMAIRE

INTRODUCTION GENERALE.....	1
CHAPITRE 1 : METHODE DE MODELISATION DES SURFACES	
I- INTRODUCTION.....	3
II- CONCEPTS DE BASE	3
II.1-Surfaces non paramétriques	3
II.1.1- Forme Explicite	3
II.1.2- Forme Implicite.....	3
II.2- Surfaces paramétriques	3
II.3- propriétés géométriques des surfaces paramétriques	4
II.3.1- Vecteurs tangents et vecteur normal à la surface paramétrique	4
II.3.2- Courbes isoparamétriques	5
II.3.3- Courbures	5
III- MODELISATION MATHEMATIQUE DES SURFACES	7
III.1-Methodes de conception des surfaces	7
III.1.1- Méthodes basées sur les courbes	7
III.1.2- Méthodes basées sur les points.....	7
III.2- Surfaces B-Spline	8
III.2.1- Définition	8
III.2.2- Propriétés importantes des surfaces B-Spline	10
III.2.3- Etude critiques des B-Splines	11
III.3- Surfaces NURBS	11
III.3.1- Définition	11
III.3.2- Propriétés importantes des surfaces NURBS	12
IV- CONCLUSION.....	13

CHAPITRE2 : Opérations de fraisage et MOCN

I. INTRODUCTION.....	14
II. GENERALITE SUR LA MACHINE-OUTIL A COMMANDE NUMERIQUE.....	14
II.1. Définition.....	14
II.2. La Commande Numérique (CN)	15
II.3. Les domaines d'applications de MOCN	15

III. ETUDE ORGANIQUE D'UNE MACHINE-OUTIL A COMMANDE NUMERIQUE	15
III.1. Structure physique d'une MOCN	15
III.2. Les principales machines-outils à commande numérique.....	15
III.3. Présélection des M.O.C.N.....	16
IV- PRESENTATION DES FRAISEUSES	16
IV.1- Fraiseuse.....	16
IV.2. Avantage des fraiseuses	16
IV.3. Construction d'une Fraiseuse à CN.....	16
V. DEFINITION DES ORIGINES DE LA MACHINE	17
VI. LES AXES DE DEPLACEMENT	17
VII. RÉFÉRENTIEL NORMALISÉ DE LA MACHINE	19
VII.1. Les axes	19
VII.2. Situation du trièdre de référence par rapport à la fraiseuse	20
VII.3. Les mouvements de rotation	20
VII.4. Les axes de la machine	20
VII.5. Types de fraiseuse	21
VIII. LA PROGRAMMATION DU MOCN	21
VIII.1. Définition d'un programme Les ordres de déplacements	21
VIII.2. Mode de programmation.....	21
VIII.3. Programmation en langage ISO	22
VIII.3.1. Format de Mot.....	22
VIII.3.2. Format de Bloc.....	23
VIII.4. Structure générale d'un programme.....	24
VIII.5. Classification des fonctions préparatoires G et fonctions auxiliaires M..	24
VIII.5.1. Fonctions préparatoires G.....	24
VIII.5.2. Fonctions auxiliaires M.....	24
IX. NOTIONS SUR LE FRAISAGE	24
IX.1. Définition de fraisage.....	24
IX.2. Modes d'usinage.....	26
X. LA COUPE	26
X.1. Différents mouvements qui réalisent la coupe.....	26
X.1.1. Mouvement de coupe.....	26
X.1.2. Mouvement d'avance.....	27
X.2. Paramètres de coupe.....	28

X.3. Les types de trajectoires.....	28
X.3.1. Positionnement point à point.....	28
X.3.2. Déplacement en paraxia.....	28
X.3.3. Déplacement en continu (trajectoires de contournage)	29
XI. CONCLUSION.....	29

CHAPITRE 3 : Stratégie d'usinage des surfaces gauches

I. INTRODUCTION.....	30
II- PARAMETRES D'USINAGE DES SURFACES GAUCHES.....	30
II.1. Choix de l'outil.....	30
II.2. Les paramètres de guidage.....	31
II.3. Paramètres de passe et Trajet d'outil.....	31
III. POSITIONNEMENT DE L'OUTIL A LA SURFACE.....	32
III.1. Calcul de la position d'outil tangente à la surface.....	32
III.2. Méthodes de calcul des positions d'outils.....	33
III.2.1. Calcul par offset de la forme.....	33
III.2.2. Calcul par la méthode de plongée (le copiage informatique)	34
IV. DEFAUTS D'USINAGE.....	34
IV.1. Erreur de flèche.....	34
IV.2. Erreur de crête.....	35
IV.3. Problème d'interférence.....	36
V. METHODES D'USINAGE DES SURFACES GAUCHES EN FINITION.....	36
V.1. Méthodes isoparamétrique.....	36
V.1.1. Aller simple (One-Way)	36
V.1.2. Aller retour (Zig-Zag)	37
V. 1.3. Concentrique.....	37
V. 1.4. Spiral-In et Spiral-Out.....	37
V. 1.5. Radiale.....	38
V.2. AUTRES METHODES D'USINAGE.....	38
V.2.1. Usinage par la méthode des plans parallèles.....	38
V.2.2. Usinage par la méthode Z-Constant.....	39
VI. CONCLUSION.....	39

CHAPITRE 4: Triangulations des surfaces

I. Introduction	40
II. Intersection d'une surface libre et d'un plan	40
III. Triangulation.....	40
III.1. Triangulation uniforme	41
III.2. Triangulation adaptative.....	41
IV. Intersection d'un triangle et un plan dans l'espace.....	43
tridimensionnelle	
IV.1. La droite dans l'espace.....	45
IV.2. Intersection entre un plan et une droite dans l'espace	47
IV.3. Appartenance d'un point à un segment de droite	48
V. Conclusion	48

CHAPITRE 5 : Conception et Implémentation

I- INTRODUCTION.....	49
II- METHODE DE MODELISATION	49
III- REALISATION DE L'APPLICATION	49
III.1- Etablissement du cahier de charge	49
III.1.1- Présentation du projet.....	49
III.1.2- Problématique.....	49
III.1.3- Objectifs visées.....	50
III.1.4- Plateforme exigé.....	50
III.1.5- Solutions proposés.....	51
IV- MODELISATION DE L'APPLICATION EN UML	51
IV.1- diagramme de cas d'utilisation	52
IV.2- Diagramme de classe.....	54
IV.3. Diagramme de collaboration.....	62
IV.4. Diagramme d'activité	63
IV.5. Diagramme de séquence.....	67
V. CONCLUSION.....	70

CHAPITRE 6 : Présentation de l'application

I. INTRODUCTION.....	71
II. PRESENTATION DU TRAVAIL.....	71
III. ENVIRENMENT DE TRAVAIL.....	71
III.1. Fenêtre principale.....	71
III.2. Barre du menu principal.....	72
III.3 Rubrique de l'usinage par régions.....	72
IV. PRESENTATION DU PROJET.....	73
IV.1. Fenêtre "Triangulation des surfaces".....	73
IV.2. Fenêtre "Résultats".....	80
IV.3. Fenêtres "Localisation globales" et "Localisation détaille des régions".....	81
IV.4. Fenêtre "Les outils adéquats pour l'usinage".....	82
IV.5. Fenêtre "Division des surfaces".....	83
IV.6. Fenêtre "visualisation l'angles des triangles".....	84
IV.7. Fenêtre "Stratégie d'usinage de chaque région".....	85
IV.8. Fenêtre "choix des outils".....	86
IV.9. Fenêtre modes de choix des outils.....	88
IV.10. Fenêtre "génération du trajet d'usinage".....	89
V. CONCLUSION.....	90

CHAPITRE 7 : Test et validation

I. INTRODUCTION.....	91
II. TESTS ET VALIDATIONS.....	91
II.1. Détermination de la triangulation.....	91
II.2- Détection des interférence et correction d'outils.....	96
II.3- Stratégies d'usinage.....	98
II.4- Choix des outils.....	101
II.5- Génération de trajet d'usinage.....	102
II.6- Génération de programme d'usinage par « G-CODE ».....	107
III. Conclusion.....	108
CONCLUSION GENERALE.....	109

ANNEXE A	110
ANNEXE B	118
LISTE DES FIGURES	122
REFERENCES BIBLIOGRAPHIQUES	125

INTRODUCTION GENERALE

I. SITUATION DU PROBLEME :

Ces dernières années, l'utilisation de l'outil informatique est devenue un maillon incontournable et indispensable pour la résolution des problèmes de nature complexe, car il offre l'efficacité, la fiabilité et la rapidité dans le traitement.

Comme tout autre domaine, celui de l'industrie mécanique n'échappe pas à cette règle, étant en concurrence permanente, elle cherche continuellement à réduire le temps et le coût de fabrication tout en améliorant la qualité et la quantité du produit.

Cette industrie s'appuie essentiellement sur la puissance des chaînes de C.F.A.O. (Conception et Fabrication Assistées par Ordinateur), qui lui même s'appuie sur le concept de la flexibilité et de la souplesse dans la conception et la fabrication des pièces de complexités diverses tels que l'usinage des pièces de formes gauches. Le processus de réalisation de ces formes a subi de grandes évolutions ces dernières décennies par l'introduction de concept des chaînes de CFAO au niveau de la préparation à la fabrication et par l'introduction de la commande numérique au niveau de la fabrication.

Les pièces dites de formes gauches ou libres sont très utilisées dans le domaine de fabrication des moules, des matrices, des formes aérodynamiques, des carrosseries de voitures et dans les formes esthétiques, ...etc. Ces pièces sont devenues des pièces courantes de notre vie quotidienne. Comme toute pièce utilisée en mécanique, les pièces de formes gauches sont conçues dans le but d'assurer des fonctions inscrites dans le cahier de charges.

Vu la complexité géométrique de ces formes d'une part, et la nature hétérogène des informations à manipuler lors de la phase de fabrication d'autre part, ces pièces ne peuvent être usinées que sur des machines outils à commande numérique, en particulier les fraiseuses à 03 ou à 05 axes. L'usinage de ces surfaces passe généralement par trois étapes :

- phase d'ébauche : permet d'enlever le maximum de matière.
- phase de demi finition : où on s'approche de la forme finale.
- phase de finition : où on obtient la forme voulue.

Toutes ces étapes nécessitent la génération d'un ensemble d'instructions écrites dans un langage propre à la machine appelé programme « G-Code ».

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 déjà réalisés par l'équipe CFAO et traitant de:

- La modélisation et la conception des courbes et des surfaces B-Spline et NURBS;
- La reconstruction des courbes et des surfaces B-Spline et NURBS par interpolation et par approximation à partir d'un nuage de points;
- L'usinage des surfaces gauches par les courbes isoparamétriques en utilisant les six stratégies d'usinage (One-Way, Zig-Zag, Concentrique, Spiral-In, Spiral-Out et Radiale).
- L'usinage des surfaces gauches par la méthode des plans parallèles.
- L'ébauchage des surfaces gauches.

On peut synthétiser l'objectif global de notre travail de la façon suivante :

« Conception et développement d'une application d'aide au choix des outils et de stratégies d'usinage pour l'usinage en finition des surfaces gauches »

II. OBJECTIF DU TRAVAIL :

L'objectif de notre projet est de développer une application logicielle graphique et interactive sous Windows permettant d'usiner les surfaces gauches en finition sur des fraiseuses à 03 axes, selon une approche basée sur la discrétisation de la surface en des régions. Nous focaliserons notre travail sur les étapes suivantes :

- La subdivision des surfaces en des régions distinctes selon leurs géométries.
- Le choix des outils optimums pour chaque région permettant d'éviter les interférences.
- Le choix de la stratégie d'usinage adéquate.
- La génération du code machine.

III. DESCRIPTION DU TRAVAIL :

Le présent mémoire est composé de sept chapitres :

- Le premier chapitre sera consacré à l'étude des différentes méthodes de modélisation des surfaces et en particulier les surfaces NURBS et B-Spline.
- Dans le deuxième chapitre, nous décrirons les machines outil à commande numérique (MOCN) dédiées au fraisage et en particulier les fraiseuses.
- Dans le troisième chapitre, nous présenterons les différentes stratégies d'usinage des surfaces gauches.
- Le quatrième chapitre, sera consacré à l'étude des différentes méthodes d'approximation des surfaces gauches (triangulation).
- La conception de notre application logicielle sera présentée dans le cinquième chapitre.
- Dans le sixième chapitre, nous présenterons l'implémentation de notre système.
- Le dernier chapitre sera réservé à l'étape de test et de validation du travail réalisé.



Chapitre **1**

Méthodes de modélisation des surfaces

I. INTRODUCTION :

Les pièces de forme gauche sont devenues par l'évolution du style et des techniques des Pièces courantes dans notre vie quotidienne. Les surfaces de cette forme sont très utilisées dans le domaine industriel pour la conception et la fabrication des moules, les coques des navires et des formes aérodynamiques très complexes ; Pour cela, les modèles des courbes et des surfaces doivent avoir des interprétations géométriques et permettent de connaître intuitivement la forme finale de la courbe ou de la surface.

II. CONCEPTS DE BASE :

Deux méthodes sont généralement utilisées pour la modélisation des surfaces :

II.1. Surfaces non paramétriques [1,2]:

On distingue deux formes des surfaces non paramétriques :

II.1.1. Forme Explicite :

Dans ce cas, la surface est donnée par l'équation algébrique suivante :

$$Z = f(x, y). \quad (1.1)$$

Où à chaque couple (x, y) correspond une et une seule valeur de Z . Cette surface ne peut pas être fermée puisque des valeurs multiples ne sont pas permises.

À cause de la complexité de la modification interactive et de la modélisation ; Ainsi que l'impossibilité de représenter toutes les surfaces, Cette forme n'est pas utilisable en CAO.

II.1.2. Forme Implicite :

Les surfaces implicites sont définies par un polynôme de trois variables :

$$f(x, y, z) = 0 \quad (1.2)$$

Cette forme n'a pas la limitation de la représentation explicite, mais la manipulation interactive des surfaces et la représentation d'une surface en 3D sous une forme implicite peut être difficile.

II.2. Surfaces paramétriques [1,2]:

Les surfaces paramétrées sont très utilisées dans les systèmes de représentation de surfaces. Une surface paramétrique est définie par un ensemble de trois fonctions, une pour chaque coordonnée, et chacune ayant deux paramètres u et v , cette surface est donnée par :

$$f(u, v) = (x(u, v), y(u, v), z(u, v)). \text{ Avec } u, v \in [0, 1]. \quad (1.3)$$

Ainsi (u, v) est un point dans le carré défini par les sommets $(0,0)$, $(0,1)$, $(1,0)$ et $(1,1)$.

Les surfaces paramétriques ne sont pas employées individuellement, mais beaucoup de morceaux des surfaces paramétriques sont joints ensemble côte à côte pour former une forme plus compliquée.

II.3. Propriétés géométriques des surfaces paramétriques :

Les principales propriétés géométriques qui sont utilisées dans la phase d'usinage des surfaces gauches sont :

II.3.1. Vecteurs tangents et vecteur normal à la surface paramétrique [2,3]:

Les vecteurs tangents T_u et T_v à la surface au point $f(u, v)$ dans les deux directions u et v (respectivement), sont exprimé par les dérivées partielles de f comme suivantes :

$$T_u = \frac{\partial f}{\partial u} = \left(\frac{\partial x}{\partial u}, \frac{\partial y}{\partial u}, \frac{\partial z}{\partial u} \right). \quad (1.4)$$

$$T_v = \frac{\partial f}{\partial v} = \left(\frac{\partial x}{\partial v}, \frac{\partial y}{\partial v}, \frac{\partial z}{\partial v} \right). \quad (1.5)$$

Ces deux vecteurs définissent le plan tangent (ou plan rectifiant) à la surface au point $f(u, v)$.

Le vecteur normal à la surface à $f(u, v)$, $n(u, v)$, est le résultat de produit vectoriel des deux vecteurs tangents et est donné par la formule suivante :

$$n = \frac{\frac{\partial f}{\partial u} \times \frac{\partial f}{\partial v}}{\left| \frac{\partial f}{\partial u} \times \frac{\partial f}{\partial v} \right|}. \quad (1.6)$$

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

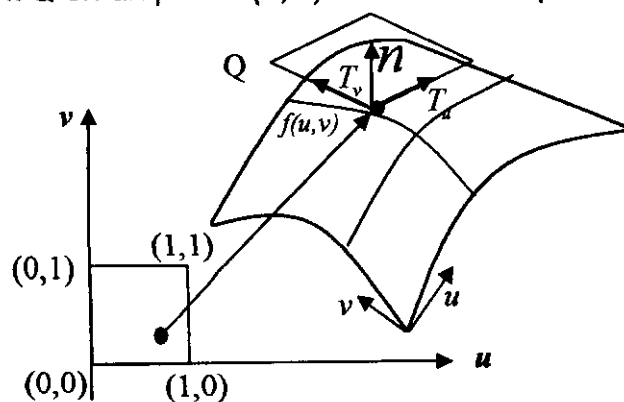


Figure 1: Vecteur tangent et vecteur normal en un point sur une surface paramétrique.

II.3.2. Courbes isoparamétriques [1,2]:

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 $f(u, v)$ 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. (Voire figure 2)

La figure 2 montre deux courbes isoparamétriques.

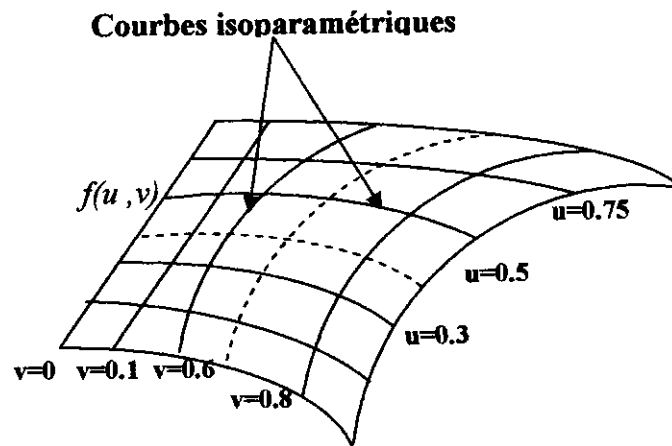


Figure 2 : Courbes isoparamétriques.

II.3.3. Courbures d'une surface [2]:

La courbure en un point de courbe est l'inverse du rayon du cercle qui approxime le mieux la courbe que tout autre cercle dans ce point (cercle osculateur). Cette courbure est donnée par l'équation suivante :

$$K = \frac{1}{R} \quad (1.7)$$

Avec : R est le rayon de cercle de courbure.

K est la courbure.

t est le vecteur tangent et n est le vecteur normal au point considéré.

La figure 3 montre le cercle osculateur et la courbure d'une courbe.

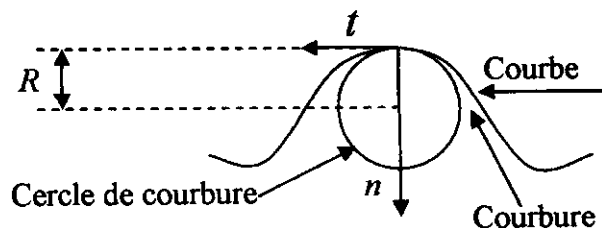


Figure 3 : Courbure d'une courbe

En n'importe quel point de la surface paramétrique, il existe un nombre infini de courbes qui passe par ce point et appartenant a cette surface paramétrique, et pour chaque courbe on peut avoir une courbure ayant une valeur différente des autres courbures en ce point. Parmi ces valeurs de courbures, il y a deux valeurs dont la première est la valeur minimale K_1 et la deuxième représente la valeur maximale K_2 . Ces valeurs extrêmes correspondants aux courbures extrêmes sont appelées les courbures principales.

Les rayons de courbures principaux d'une surface sont les solutions de l'équation de second degré suivante :

$$(LN - M^2)K^2 + (2MF - GL - EN)K + EG - F^2 = 0 \quad (1.8)$$

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 :

$$E = \frac{\partial P}{\partial u} \times \frac{\partial P}{\partial u} \quad (1.9)$$

$$F = \frac{\partial P}{\partial u} \times \frac{\partial P}{\partial v} \quad (1.10)$$

$$G = \frac{\partial P}{\partial v} \times \frac{\partial P}{\partial v} \quad (1.11)$$

$$L = n \times \frac{\partial^2 P}{\partial u^2} \quad (1.12)$$

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

$$N = n \times \frac{\partial^2 P}{\partial v^2} \quad (1.14)$$

La courbure moyenne H d'une surface au point $f(u, v)$, est défini comme suit :

$$H = \frac{K_1 + K_2}{2} \quad (1.15)$$

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

$$K = K_1 \times K_2 \quad (1.16)$$

Suivant les valeurs de ces deux courbures (H , K) en un point x sur la surface, celle-ci aura les propriétés suivantes :

- Si $K > 0$ alors x est un point elliptique ;
- Si $K < 0$ alors x est un point hyperbolique ;
- Si $K = 0$ alors x est point parabolique ;
- Si $K < 0$ et $H = 0$ alors x est un point plat.

La courbure absolue est calculée par la formule suivante :

$$A = |K_1| + |K_2| \quad (1.17)$$

III. MODELISATION MATHEMATIQUE DES SURFACES :

III.1. Méthodes de conception des surfaces [2, 3,4]:

La conception et la modélisation des surfaces sont très importantes dans le domaine de la conception assistée par ordinateur (CAO). Les concepteurs de surfaces ont besoin d'un système de modélisation, qui leur permet de concevoir et de manipuler les surfaces d'une manière interactive. La forme de la surface est déterminée après avoir donné sa représentation mathématique. Mais pratiquement, dans le plus part des situations de conception de nouvelles surfaces, le concepteur n'a en tête que la forme de la surface qu'il veut obtenir sans connaître ni sa représentation mathématique ni ses propriétés géométrique. Pour résoudre ce problème, on trouve dans le domaine de la conception et la modélisation des surfaces deux classes de méthodes qui peuvent aider le concepteur dans sa conception :

- Méthodes basées sur l'utilisation des courbes,
- Méthodes basées sur l'utilisation des points.

Chaque classe englobe un certain type des surfaces.

III.1.1. Méthodes basées sur les courbes :

Dans ces méthodes la surface est déterminée à partir des courbes. Cette classe regroupe les surfaces suivantes :

- surface de révolution.
- surface lissée (Skinned surface).
- surface de coons.
- surface de gordon.
- surface oscillante (Swing surface).
- surface réglée.
- surface extrudée.
- surface balayée.

III.1.2. Méthodes basées sur les points :

Dans ces méthodes l'information élémentaire est le point. Les types des surfaces de cette classe sont :

- interpolation et approximation d'un nuage de point.
- surfaces de Bézier.
- surfaces de Bézier rationnelle.
- surfaces B-Spline.
- surfaces NURBS.

Il existe plusieurs critères pour choisir la méthode de représentation :

- La présentation de tout les surfaces de la même façon.
- La représentation des surfaces fermées.
- La modification de la surface en temps réel d'une manière intuitive.
- La moins coûteuse en source informatique et la plus fidèle à la représentation de l'objet.
- La translation, la rotation, le changement d'échelle, le changement de repère et toutes autres transformation géométrique linéaires n'influent pas sur la méthode de représentation de la surface.
- peut satisfaire au maximum les contraintes de design et certaines contraintes de continuités.
- La conception ou la modification d'une surface existante est intuitive et flexible.

III.2. Surfaces B-Spline [1, 2, 3,4]:

II.2.1. Définition : une surface B-spline est définie par les informations suivantes :

1. un réseau de contrôle $(m+1) \times (n+1)$ points de contrôle $P_{i,j}$, où $0 \leq i \leq m$ et $0 \leq j \leq n$.
2. un vecteur de $h+1$ nœuds dans la direction u , $U = \{u_0 = 0, u_1, \dots, u_h = 1\}$.
3. un vecteur de $k+1$ nœuds dans la direction v , $V = \{v_0 = 0, v_1, \dots, v_k = 1\}$.
4. le degré p dans la direction u .
5. le degré q dans la direction v .

La surface B-spline définie par ces informations est la suivante :

$$S(u, v) = \sum_{i=0}^m \sum_{j=0}^n N_{i,p}(u) \times N_{j,q}(v) \times P_{i,j} \quad \text{Avec } 0 \leq u \leq 1 \text{ et } 0 \leq v \leq 1 \quad (1.18)$$

Où l'ensemble des $\{P_{i,j}\}$ sont les points de contrôle ; et les $\{N_{i,p}(u)\}$ et $\{N_{j,q}(v)\}$ sont les fonctions de base B-spline de degré p et q respectivement données par récursivité :

$$N_{i,0}(u) = \begin{cases} 1 & \text{si } u_i \leq u \leq u_{i+1} \\ 0 & \text{sinon.} \end{cases} \quad (1.19)$$

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

$$N_{j,0}(v) = \begin{cases} 1 & \text{si } v_j \leq v \leq v_{j+1} \\ 0 & \text{sinon} \end{cases} \quad (1.21)$$

$$N_{j,q}(v) = \frac{v - v_j}{v_{j+q} - v_j} \times N_{j,q-1}(v) + \frac{v_{j+q+1} - v}{v_{j+q+1} - v_{j+1}} \times N_{j+1,q-1}(v). \quad (1.22)$$

Ces fonctions base de surface B-spline de degré (p, q) sont les coefficients des points de contrôle. La fonction base du point de contrôle $p_{i,j}$ est le produit de deux fonctions base B-Spline unidimensionnelles $N_{i,p}(u)$ dans la direction u et $N_{j,q}(v)$ dans la direction v . Tous ces produits sont des fonctions bidimensionnelles de B-Spline.

Puisque les fonctions base B-Splines sont en général non nulles sur quelques intervalles consécutives (schéma de modification local), alors les fonctions base B-Splines bidimensionnelles sont non nulles sur le produit des deux intervalles sur lesquels au moins une fonction base unidimensionnelle est non nulle.

Les identités fondamentales, une pour chaque direction doivent être satisfaites :

$$\begin{cases} h = m + p + 1 \\ k = n + q + 1 \end{cases} \quad (1.23)$$

La figure 4 représente une surface B-Spline.

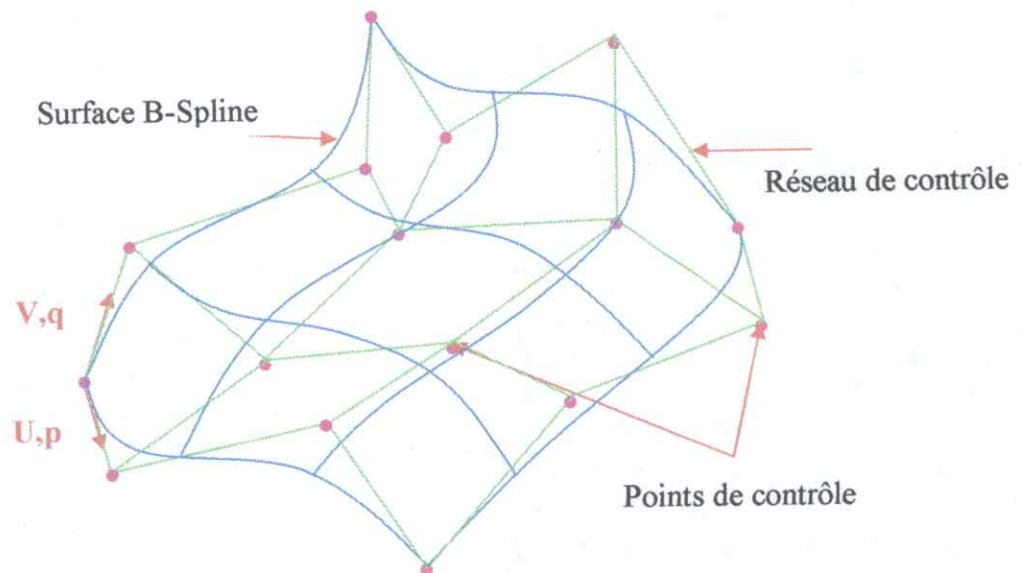


Figure 4 : Surface B-Spline.

Par conséquent, une surface B-Spline est un autre exemple des surfaces de produit tensorielle. L'ensemble des points de contrôle habituellement mentionné est le réseau de contrôle avec u et v dans l'intervalle $[0, 1]$.

Une surface B-spline peut se présenter sous trois formes possibles :

- ❖ **Surface pincée** : si une surface B-Spline est pincée dans les deux directions u et v , alors la surface passe par les points de contrôle $P_{0,0}, P_{m,0}, P_{0,n}, P_{m,n}$ et elle est tangente aux huit segments du réseau de contrôle à ces quatre points de contrôle.
- ❖ **Surface fermée** : si une surface de B-Spline est fermée dans une direction, alors toutes les courbes isoparamétriques dans cette direction sont des courbes fermées et la surface devient un tube.
- ❖ **Surface ouverte** : si une surface B-Spline est ouverte dans les deux directions, alors la surface ne passe pas par les points de contrôle $P_{0,0}, P_{m,0}, P_{0,n}, P_{m,n}$.

III.2.2. Propriétés importantes des surfaces B-Spline :

En utilisant les propriétés des fonctions de base B-Spline, on peut montrer les propriétés suivantes des surfaces B-Spline :

- Si $m = p$, $n = q$, et $U = \{0, 0, \dots, 0, 1, 1, \dots, 1\}$, alors une surface B-Spline devient une surface de Bézier.
- **Invariance affine** : on peut appliquer une transformation affine à la surface B-Spline en appliquant la transformation au réseau de contrôle.
- **Propriétés de l'enveloppe convexe** : La surface est contenue dans l'enveloppe convexe de ses points de contrôle. En fait, si (u, v) est dans $[u_i, u_{i+1}) \times [v_j, v_{j+1})$, alors $S(u, v)$ est contenue dans l'enveloppe convexe définie par les points de contrôle $P_{h,k}$, où $i-p \leq h \leq i$ et $j-q \leq k \leq j$.
- **Propriété de modification locale** : $N_{i,p}(u) N_{j,q}(v)$ est nulle si (u, v) est en dehors de l'intervalle $[u_i, u_{i+p+1}) \times [v_j, v_{j+q+1})$.
- **Non négativité** : la fonction $N_{i,p}(u) N_{j,q}(v)$ est non négative pour tout p, q, i, j et u et v dans l'intervalle $[0, 1]$.
- **Partition de l'unité** : la somme de toutes les fonctions $N_{i,p}(u) N_{j,q}(v)$ est égale à 1 pour tout u et v dans l'intervalle $[0, 1]$. Pour toute paire de u et v , nous avons :

$$\sum_{i=0}^m \sum_{j=0}^n N_{i,p}(u) N_{j,q}(v) = 1. \quad (1.24)$$

- $S(u, v)$ a une continuité C^{p-s} (respectivement C^{q-t}) dans la direction u (respectivement v direction) si u (resp. v) est un nœud de multiplicité s (respectivement t).
- Toute courbe isoparamétrique sur la surface B-Spline est une courbe B-Spline définie par un ensemble de points de contrôle.
- La surface B-Spline est une surface polynomiale de degré (p, q) car $N_{i,p}(u)$ et $N_{j,q}(v)$ sont des fonctions polynomiales de degré p et de degré q .
- Les courbes isoparamétriques $S(0, v)$, $S(1, v)$, $S(u, 0)$ et $S(u, 1)$ sont les courbes de frontières. Les courbes limites correspondantes à $u=0$ et $u=1$ sont des courbes B-Splines définies par la ligne 0 et la ligne m des points de contrôle. Les courbes limites correspondantes à $v=0$ et $v=1$ sont des courbes B-Splines définies par la colonne 0 et la colonne n des points de contrôle.

III.2.3. Etude critiques des B-Spline :

Il existe un certain nombre de surfaces qui ne peuvent pas être représentées précisément à l'aide de polynômes, par exemples :

- Les sphères.
- Les surfaces de révolutions.
- Et les autres coniques.

Pour résoudre ce problème, nous devons généraliser les surfaces B-Spline en utilisant des coordonnées homogènes ; on introduit le concept des surfaces **NURBS** (Non Uniforme Rational B-Spline).

III.3. Surfaces NURBS [1, 2, 3,4]:

III.3.1. Définition : une surface NURBS est définie à partir des mêmes informations utilisées pour la définition de la surface B-Spline, en plus, on associe à chaque point de contrôle un poids.

1. Un réseau de contrôle $(m+1) \times (n+1)$ points de contrôle $P_{i,j}$ où $0 \leq i \leq m$ et $0 \leq j \leq n$.
2. Un vecteur de $h+1$ nœuds dans la direction u ,
 $U = \{u_0 = 0, u_1, \dots, u_h = 1\}$.
3. Un vecteur de $k+1$ nœuds dans la direction v ,
 $V = \{v_0 = 0, v_1, \dots, v_k = 1\}$.
4. A chaque point de contrôle est associé un poids $w_{i,j} \geq 0$.
5. Le degré p dans la direction u .
6. Le degré q dans la direction v .

La surface définie par ces informations est donnée par

$$S(u, v) = \frac{\sum_{i=0}^m \sum_{j=0}^n N_{i,p}(u) N_{j,q}(v) w_{i,j} P_{i,j}}{\sum_{i=0}^m \sum_{j=0}^n N_{i,p}(u) N_{j,q}(v) w_{i,j}} \quad (1.25)$$

Où l'ensemble des $\{P_{i,j}\}$ sont les points de contrôle ; et les $\{N_{i,p}(u)\}$ et $\{N_{j,q}(v)\}$ sont les fonctions de base B-Splines de degré p et q respectivement.

Les identités fondamentales, une pour chaque direction doivent être satisfaites :

$$\begin{cases} h = m + p + 1 \\ k = n + q + 1 \end{cases} \quad (1.26)$$

La figure 5 représente une surface NURBS

Points de contrôles avec leur poids

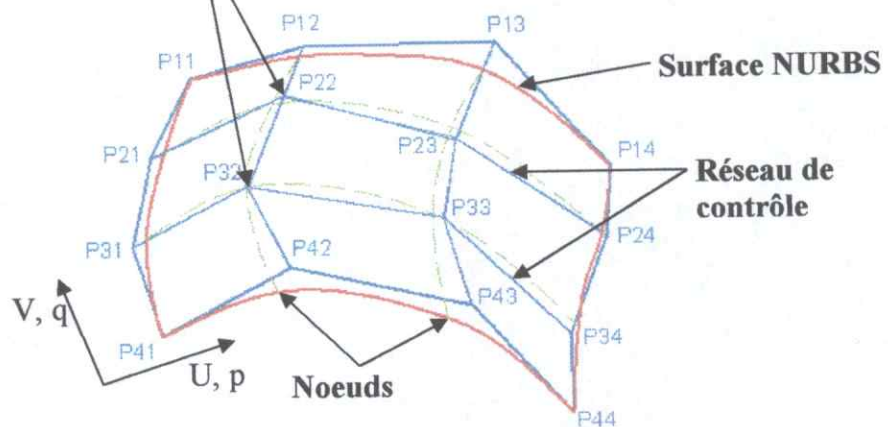


Figure 5 : Surface NURBS.

Par conséquent, une surface NURBS est un autre exemple des surfaces de produit tensorielle. L'ensemble des points de contrôle habituellement mentionné est le réseau de contrôle, avec u et v sont dans l'intervalle $[0, 1]$.

III.3.2. Propriétés importantes des surfaces NURBS :

Les surfaces NURBS possède les propriétés suivantes :

- Les surfaces NURBS sont des généralisations des surfaces B-Spline et des surfaces de Bézier. Si tous les poids sont égaux, la surface NURBS devient une surface B-Spline. Si de plus $m = p$ et $n = q$, $U = \{0, 0, \dots, 0, 1, 1, \dots, 1\}$ et

$V = \{0, 0, \dots, 0, 1, 1, \dots, 1\}$, alors une surface NURBS devient une surface de Bézier.

- Les courbes isoparamétriques $S(0, v)$, $S(1, v)$, $S(u, 0)$ et $S(u, 1)$ sont les courbes de frontières. Les courbes limites correspondantes à $u=0$ et $u=1$ sont des courbes NURBS définies par la ligne 0 et la ligne m des points de contrôle. Les courbes limites correspondantes à $v=0$ et $v=1$ sont des courbes NURBS définies par la colonne 0 et la colonne n des points de contrôle.
- A chaque fois que la valeur du poids $W_{i,j}$ du point de contrôle $P_{i,j}$ augmente (resp. diminue) une portion de la surface est approchée (resp. éloignée) de ce point.
- Toute courbe isoparamétrique sur la surface NURBS est une courbe NURBS définie par un ensemble de points de contrôle.
- Le déplacement d'un point de contrôle cause le déplacement d'une portion de la surface dans la même direction (schéma de modification locale).
- Une surface NURBS est une surface rationnelle, donc, elle permet de représenter toutes les coniques.

IV. CONCLUSION :

Dans ce chapitre, nous avons vu en premier lieu les concepts de base, à savoir les types des surfaces. En suite, nous avons présenté les différentes méthodes de conception et modélisation des surfaces gauches ; et à la fin, nous avons étudié en particulier les surfaces les plus utilisées dans les logiciels de CFAO à savoir B-Spline et les surfaces NURBS, ainsi que leurs principales propriétés. Dans la suite, nous allons étudier les différentes méthodes d'usinages des surfaces gauches que nous allons utiliser pour usiner ces surfaces.



Chapitre **2**

**Opérations de fraisage et
MOCN**

I. INTRODUCTION :

Après l'étape de conception et d'analyse des surfaces gauches conçues; il faut choisir les machines à utiliser ainsi que l'outillage nécessaire; qui sont en général des fraiseuses. Mais en raison de la complexité géométrique de ces surfaces, les fraiseuses doivent être commandées numériquement. D'où la nécessité de comprendre le mode de fonctionnement de ces machines.

Pour cela nous allons présenter dans ce chapitre une structure physique général des MOCN et ses différent axes de déplacements possibles ainsi que la structure générale des programmes d'usinages et des notions sur le fraisage et la coupe.

II. GENERALITE SUR LA MACHINE-OUTIL A COMMANDE NUMERIQUE :

II.1. Définition [9,13] :

Une Machine-outil à Commande Numérique (M.O.C.N.) est une machine d'usinage à cycle automatique programmable.

Cette machine est commandée par des "consignes" numériques fournies par un calculateur. En d'autres termes, on peut dire que les organes mobiles de la machine sont motorisés et qu'un automatisme assure la commande et dans la plupart des cas le contrôle de la position et/ou de la vitesse. Ce type de machine se compose ainsi de deux parties complémentaires (Figure 1) :

- la partie opérative (c'est la machine-outil : elle agit directement sur le produit à réaliser);

- la partie commande (c'est la commande numérique : elle permet d'élaborer des ordres en fonction des consignes et des comptes-rendus).

Ainsi la M.O.C.N. commande et contrôle ses propres mouvements, mesure ses déplacements avec une précision constante. Un exemple de non-qualité serait une mauvaise surveillance (pannes ou bris d'outils) ou le non-contrôle de l'usure des outils.

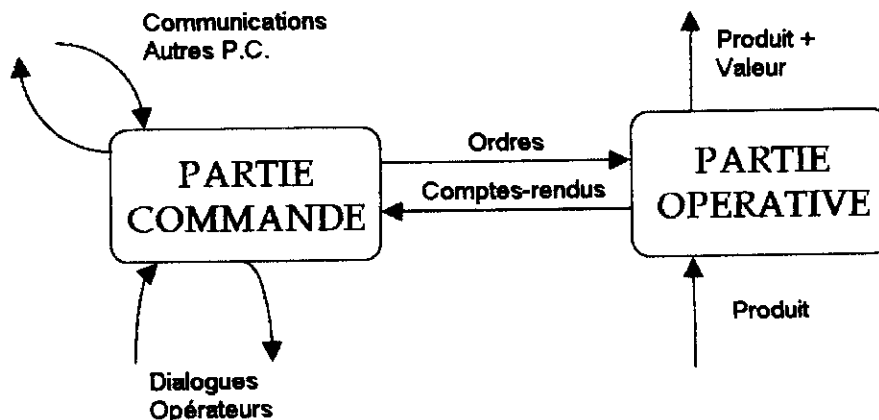


Figure 1 : structure d'une MOCN

II.2. La Commande Numérique (CN) [14] :

Le principe : La commande numérique avec calculateur va gérer et déterminer tous les déplacements. Il suffit de lui préciser les coordonnées des points :

- position de départ (origine) ;
- position à atteindre (arrivée) ;
- changement de direction (horaire et anti-horaire) ;
- dégagement (point de sortie) ;
- la vitesse.

Le système utilise un langage spécifique, c'est le langage machine ISO, il traduit, et transpose les données que nous lui avons imposées.

Remarque : les machines-outils à commande numérique sont essentiellement utilisées pour la production des pièces en moyennes et grandes séries, car leur coût de mise en œuvre et de programmation est élevé.

II.3. Les domaines d'applications de MOCN [14] :

Ce sont essentiellement la mécanique, l'électronique et le domaine de la gravure.

III. ETUDE ORGANIQUE D'UNE MACHINE-OUTIL A COMMANDE NUMERIQUE :

III.1. Structure physique d'une MOCN [9,13] :

Si on ne s'intéresse qu'aux méthodes d'enlèvement de matière par mouvement de rotation (cas classiques du tournage, perçage, fraisage), la machine doit avoir la structure suivante :

- des systèmes, autant que nécessaire, assurant la mise en position de l'outil par rapport à la pièce et les mouvements d'avance. **Ce sont les axes de la machine.**
- un système qui réalise le mouvement de coupe par mise en rotation des outils ou de la pièce : **la broche.**
- un système de contrôle - commande, qui permet le suivi automatique du programme de commande de la machine ;
- un élément mécanique qui assure le lien entre ces systèmes : **le bâti.**

A cela, il faut ajouter des éléments d'interfaces spécifiques à la production permettant la mise en position des outils et des pièces sur la machine.

III.2. Les principales machines-outils à commande numérique [14] :

- **La fraiseuse** : sur laquelle on peut réaliser des opérations de fraisage, de perçage et de gravage.

- **Le tour** : sur lequel on peut réaliser des opérations de tournage mais aussi de perçage.

- **Le centre d'usinage** : c'est une machine polyvalente pouvant réaliser une vingtaine d'opérations d'usinage sur une même pièce sans démontage de celle-ci grâce au magasin d'outils qu'il contient.

III.3. Présélection des M.O.C.N [9,13] :

Les M.O.C.N. permettent l'usinage de pièces de formes diversifiées. Un classement par famille de pièces permet d'effectuer une première sélection du type de machine.

On recherche à faire un maximum d'usinage sans démontage de la pièce pour éviter les dispersions dues à la mise en position des pièces sur les montages d'usinage.

La sélection finale de la machine s'effectue en tenant compte des dimensions des pièces à usiner, de la puissance nécessaire de la broche, des capacités de la machine.

IV- PRESENTATION DES FRAISEUSES :

IV.1- Fraiseuse [10,13,16] : Une fraiseuse est une machine-outil utilisée pour usiner tous types de pièces par enlèvement de matière à l'aide d'un outil nommé fraise. La fraise munie de dents et mise en rotation se déplace, et taille la matière, et elle est souvent montée sur une tête à trois axes (on parle alors de fraiseuse trois axes).

IV.2. Avantage des fraiseuses [3]:

Les principaux avantages des fraiseuses à commande numérique sont :

- Outil fiable et précis,
- Gestion automatique (chargement/déchargement d'outil/pièce à usiner d'une manière automatique),
- Travail continu,
- Réduction du temps d'usinage par rapport au coût de fabrication des pièces en raison de la réponse rapide,
- Souplesse et la polyvalence.

IV.3. Construction d'une Fraiseuse à CN : [3]:

La fraiseuse à CN est composée de plusieurs parties:

- **Bâti** : est la plateforme de la machine qui permet de supporter la chaleur et d'assurer une précision accrue lors de l'usinage.

- **Broche** : elle porte l'outil et transmet ainsi le mouvement de rotation (mouvement de coupe) nécessaire à l'opération de fraisage.

- **Port outil** : assure la liaison entre l'outil et la broche.

- **Outil d'usinage** : utilisé dans le fraisage est appelé fraise.

- **Directeur de Commande Numérique (DCN)** : comme toutes les machines à commande numérique, les fraiseuses disposent d'un ordinateur équipé d'une mémoire intégrée qui exécute les programmes chargés en mémoire d'une façon autonome.

- **Pupitre de commande** : son rôle principal est de dialoguer avec le DCN par le biais des messages envoyés appelés commandes et il possède un écran et un clavier.

- **Armoire électronique** : interposée entre le DCN et la machine (coté mécanique), elle englobe tous les composants électroniques (transistor, condensateur, câblage,...) et des cartes de gestion.

V. DEFINITION DES ORIGINES DE LA MACHINE [3]:

Le processeur CN calcule tous les déplacements par rapport au point d'origine mesure de la machine. Les différentes origines à considérer sont les suivantes :

- **Origine mesure (OM)** : cette origine dépend de la machine et définit l'origine de système physique de mesure de la machine.
- **Origine machine (Om)** : la prise d'origine se fait sur une position physique précise, c'est l'origine machine (Om) qui est déterminée à partir de l'origine mesure.
- **Origine programme (OP)** : Placée par le programmeur pour faciliter l'écriture du programme.
- **Origine pièce (Op)** : Liée à la mise en position de la pièce sur le porte-pièce.

VI. LES AXES DE DEPLACEMENT [3] :

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.

Le guidage précis des éléments est assuré par des glissières, qui sont de plus en plus réalisées au moyen de rouleaux pré-contraints, afin de limiter les frottements, et de supporter des charges plus élevées. Le système de mesure transmet la position de l'élément à la commande numérique.

- **Axes commandés** : dans une structure mécanique, on appelle axe commandé une liaison équipée d'une motorisation qui permet à un instant donné de fournir une valeur à la coordonnée articulaire.

- **Asservissement d'un axe numérique** : la commande d'axe permet d'asservir en position et en vitesse le déplacement des mobiles. Le schéma général de l'asservissement est donné par la (figure 2). Il comporte deux boucles d'asservissement imbriquées :

- Une boucle de régulation de vitesse.
- Une boucle de régulation de position.

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.

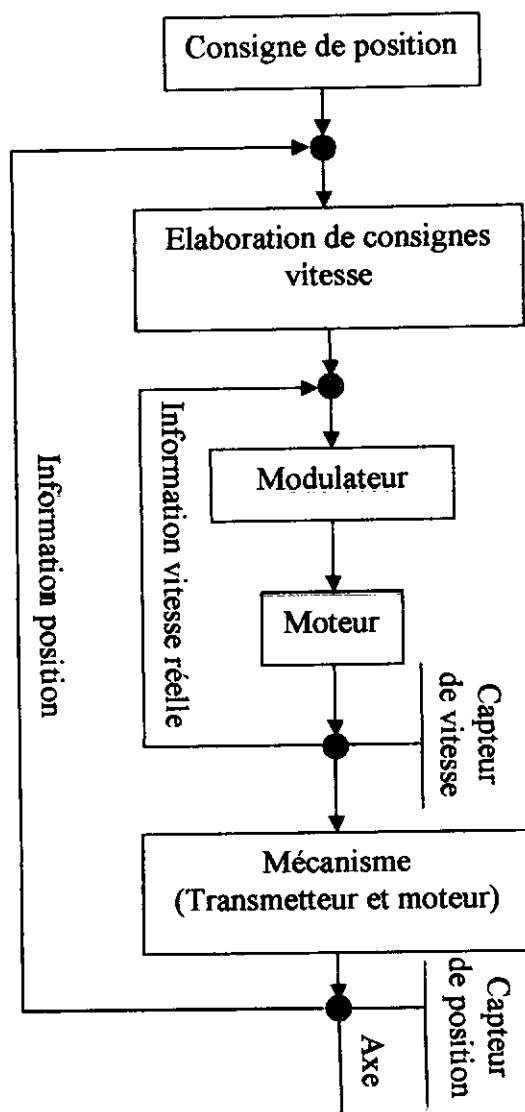


Figure 2 : Schéma général de l'asservissement d'un axe numérique.

VII. RÉFÉRENTIEL NORMALISÉ DE LA MACHINE:

VII.1. Les axes [9,11] :

A pour objet de définir une nomenclature des axes et des mouvements pour machines à commande numérique en vue de faciliter l'interchangeabilité des données de programmation.

Définitions [9,11] :

1- **Axe** : Direction suivant laquelle le mouvement est commandé numériquement d'une manière continue en vitesse et en position.

2- Trièdre de référence :

Le système de coordonnées (X, Y, Z) est un système cartésien de sens direct lié à une pièce placée sur la machine. On peut le définir par la règle des trois doigts (Figure 3).

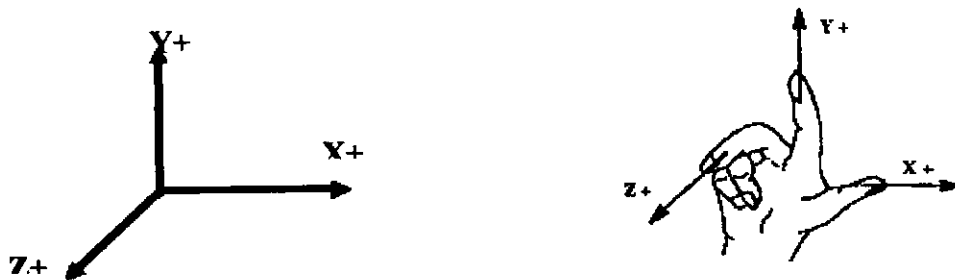


Figure 3 : Règle des trois doigts

VII.2. Situation du trièdre de la référence par rapport à la fraiseuse [6] :

Afin de décrire la trajectoire suivie par l'outil pour usiner la pièce, un système d'axe est normalisé. Ces axes seront notamment utilisés pour écrire des programmes de commande numérique.

On doit toujours savoir reconnaître la broche d'une machine afin de placer correctement les axes. En fraisage, l'axe de broche correspond à l'axe de rotation de l'outil.

L'axe Z correspond à l'axe de broche. C'est l'axe de rotation de la fraise pour l'usinage.

L'axe X correspond à l'axe perpendiculaire à Z, qui permet le plus grand déplacement de la table de la fraiseuse.

L'axe Y correspond à l'axe perpendiculaire à Z et X.

Le sens positif est donné suivant cette règle : la pièce étant la référence, l'outil s'éloignant de la pièce en mouvement suivant le sens positif des axes.

Les axes Z, X et Y définissent une base en 3 dimensions.

VII.3. Les mouvements de rotation [9,11] :

Les symboles A, B et C désignent les mouvements de rotation effectués respectivement autour d'axes parallèles à X, Y et Z.

Les sens positifs de A, B et C sont inversés par rapport au sens trigonométrique. L'observation étant faite en direction du sens positif de l'axe linéaire correspondant.

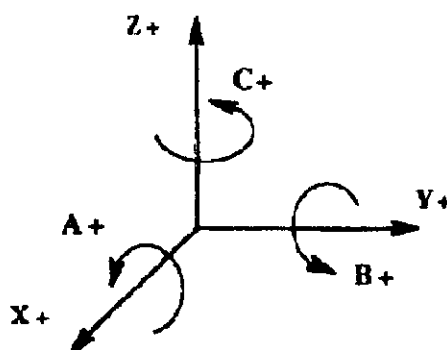


Figure 4 : axes de rotation

VII.4. Les axes de la machine [3]:

Il y a deux types d'axes :

1. **Demi-axe numérique** : axe de déplacement pour lequel un ensemble fini de positions peut être atteint - OU - axe de déplacement asservi en position ou en vitesse.

2. **Axe numérique** : axe de déplacement pour lequel une infinité de positions peut être atteinte à la résolution de positionnement près - OU - axe de déplacement asservi en position et en vitesse.

VII.5. Types de fraiseuse [16] :

Il existe trois types de fraiseuse :

1- **Fraiseuse horizontale** : l'axe de la broche est parallèle à la table.

2- **Fraiseuse verticale** : l'axe de la broche est perpendiculaire à la table.

3- **Fraiseuse universelle** : l'axe de la broche est réglable :

- Tête bi-rotative, avec 2 coulisses circulaires (perpendiculaires l'une par rapport à l'autre).
- Tête oblique, avec 2 coulisses circulaires (inclinaison à 45°).
- Tête articulée.

VIII. LA PROGRAMMATION DU MOCN :

VIII.1. Définition d'un programme [9,11,13] : Un programme est une suite d'instructions écrites dans un langage codé propre à la commande numérique (le plus utilisé est le code ISO : International Organization for Standardization). La commande numérique interprète le programme pour commander un usinage sur la machine outil.

La programmation des MOCN repose sur des conventions, à savoir les langages de programmation normalisés [11].

Le programme est la description structurée de l'opération d'usinage. Il comporte deux types d'informations [11] : des ordres de déplacements et des ordres auxiliaires.

- **Les ordres de déplacements** ; un ordre de déplacement, est spécifier par un mode d'interpolation, un but (point de destination) et une vitesse de déplacement. Les types d'interpolation utilisés sont :
 - interpolation linéaire : trajectoire décrite par un segment.
 - interpolation circulaire : trajectoire décrite par un arc de cercle.
 - interpolation hélicoïdale : trajectoire décrite par une hélice.

On assiste au développement actuel de l'interpolation polynomiale, où la trajectoire est décrite par une courbe polynomiale, ce qui permet d'assurer une meilleure dynamique de la trajectoire.

- **Les ordres auxiliaires** ; sont des ordres traités de manière séquentiels qui permettent l'opération d'usinage.

VIII.2. Mode de programmation [3,13]:

Quelque soit le langage de programmation utilisé pour le développement des programmes pièces, le seul langage compréhensible par la machine est le langage ISO. Le passage d'un langage de haut niveau au langage ISO est possible en utilisant un logiciel de traduction.

VIII.3. Programmation en langage ISO [3,13] :

On appelle le programme interprété et exécuté par le DCN le « G-Code » employé pour définir les fonctions d'usinage (déplacement, changement d'outil, vitesse d'avance, vitesse de broche, sens de rotation, ... etc.).

Un programme « G-Code » est constitué d'un ensemble de lignes appelées «Bloc». Ce dernier est un ensemble d'instructions relatives à une séquence d'usinage. Et le Bloc lui-même, est constitué d'un ensemble de mots.

VIII.3.1. Format de Mot [9,11,13]:

Le mot définit une instruction ou donnée composé d'un ensemble de caractère (voir figure 5) à transmettre au système de commande. Comme un mot représente soit une fonction soit un déplacement suivant l'un des axes.

Le format de mot définit les caractéristiques particulières de chaque mot codé employé en programmation.

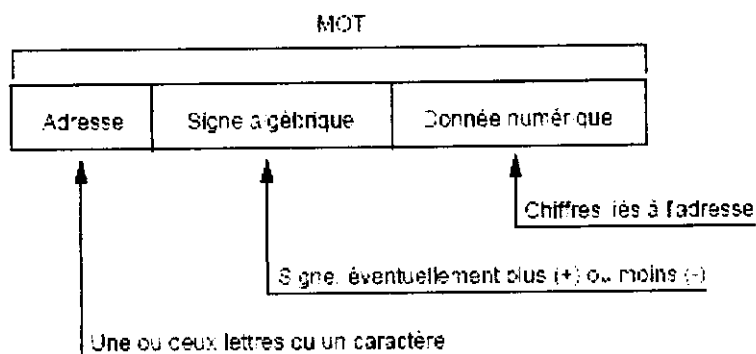


Figure 5 : Format général des Mots [12].

Les fonctions usuelles peuvent être représentées comme suit [13,3] :

- G.. : Fonctions préparatoires.
- D.. : Fonctions de correcteurs.
- F.. : Fonctions technologiques (Fonctions de vitesse d'avance).
- S.. : Fonctions de vitesse de rotation (Fonctions de vitesse de broche).
- T.. : Fonctions d'outils.
- M.. : Fonctions auxiliaires.
- X.. : Fonctions de dimension (Déplacement suivant X).
- Y.. : Déplacement suivant Y.
- P.. : Fonction paramètre.
- Z.. : Mouvement suivant l'axe Z.

VIII.3.2. Format de Bloc [9,11,13] :

Un bloc définit une ligne d'instructions composée de mots (figure 6) codés à transmettre au système de commande.

Le format de bloc définit la syntaxe des mots de fonction et de dimension (déplacement) composant chaque bloc de programmation.

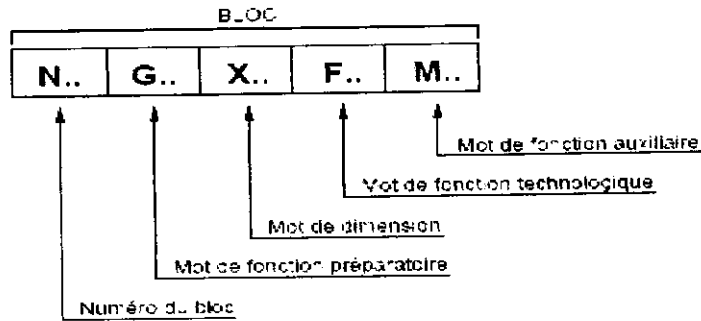


Figure 6 : Format général des Blocs [12].

VIII.4. Structure générale d'un programme [9,11,13,3]:

Un programme d'usinage est comme tout autre programme informatique, caractérisé par (voir figure 7) :

- Des caractères obligatoires de début et de fin de programme.
- Leur exécution se fait dans l'ordre d'écriture des blocs situés entre les caractères de début et de fin de programme.
- La numérotation des blocs n'intervient pas dans le même ordre de déroulement du programme.
- La numérotation des blocs se fait dans l'ordre d'écriture.

Un programme ISO est structuré par les éléments suivants :

- Début de programme : caractère % suivi du numéro de programme et éventuellement d'un commentaire entre parenthèses (voir figure 7).
- Ensemble des blocs à exécuter (programme effectif).
- Fin de programme : code M02.
- Fin de chargement de programme : caractère XOFF.

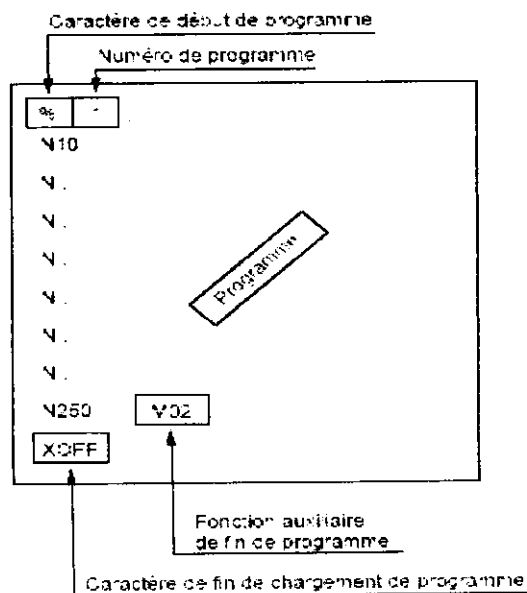


Figure 7: Structure d'un programme ISO [12].

VIII.5. Classification des fonctions préparatoires G et fonctions auxiliaires M :

VIII.5.1. Fonctions préparatoires G [9,11,13]:

Il existe deux types de fonctions préparatoires G :

- **Fonctions G modales** : Fonctions appartenant à une famille de fonctions G se révoquant mutuellement. La validité de ces fonctions est maintenue jusqu'à ce qu'une fonction de même famille révoque leur validité.
- **Fonctions G non modales** : Fonctions uniquement valides dans le bloc où elles sont programmées (révoquée en fin de bloc).

Il existe certaines fonctions G programmées avec leurs arguments associés, et certaines d'autres sont incompatibles avec l'état du programme (fonctions dans la programmation est autorisée ou non selon l'état du programme en cours).

VIII.5.2. Fonctions auxiliaires M [9,11,3]:

Les fonctions auxiliaires M permettent de déterminer les mouvements, la sélection de vitesse, ... etc. Les fonctions M sont comme les fonctions G et peuvent être soit des fonctions M modales ou des fonctions M non modales. De plus, elles peuvent être des fonctions avant ou après :

- **Fonctions M avant** : fonctions exécutées avant le déplacement sur les axes programmés dans le bloc.
- **Fonction M après** : fonctions exécutées après le déplacement sur les axes programmés dans le bloc.

IX. NOTIONS SUR LE FRAISAGE [5,3] :

IX.1. Définition de fraisage : Le fraisage, comme son nom l'indique, regroupe les opérations d'usinage pouvant être effectuées sur une fraiseuse. Ces opérations aboutissent à l'obtention d'une géométrie quelconque (généralement une forme prismatique).

IX.2. Modes d'usinage [5,3]:

On peut distinguer deux types d'usinage en fraisage :

- **Les opérations axiales** : l'outil se déplace uniquement le long de son axe, ces opérations regroupent perçage, alésage, lamage, ... etc.
- **Les opérations de fraisages** : l'outil se déplace dans l'espace, essentiellement dans le plan normal à l'axe de la broche, ces opérations peuvent être :

- **Opération en bout** : La surface usinée est sous la fraise. Cette catégorie regroupe le surfaçage et l'usinage 3D à la fraise boule ou torique. (voir figure 8)



Figure 8 : usinage en bout [5].

- **Opération en roulant** : La surface usinée est tangente à la génératrice de la fraise. C'est donc le profil de la fraise qui va déterminer la forme. (voir figure 9)



Figure 9 : usinage en roulant [5].

De même, on peut distinguer deux modes de fraisage dans le fraisage en roulant :

Fraisage en opposition [5] : Dans ce mode, les dents de la fraise en prise avec la matière avancent dans le sens de l'usinage (Voire figure 10.a). Elles attaquent donc la matière par une épaisseur de copeau nulle et terminent leur travail en quittant le copeau à son épaisseur maxi. L'effort généré est dirigé en sens inverse du sens d'usinage et vers la matière, c'est un effort résistant et qui colle l'outil vers la matière à usiner. On utilise cette technique pour annuler les jeux fonctionnels de la machine car les efforts poussent les écrous des vis en sens inverse du mouvement.

On notera que cette technique recycle le copeau qui fait un tour gratuit avec la fraise s'il lui prend l'envie de rester collé sur la dent. C'est assez mauvais car s'il se recolte sur la matière il ne sera pas dégagé par la suite. Les états de surface générés sont moins bons qu'en travail "en avalant" mais la machine souffre moins car la variation d'effort est moins brutale. Les conditions de coupe ne sont pas très bonnes car il est mauvais d'attaquer la matière tangentiellement (la matière refuse de se faire couper).

Fraisage en avalant [5] : Dans ce mode, les dents de la fraise en prise avec la matière avancent en sens inverse du sens d'usinage (Voire figure 10.b). Elles attaquent donc la matière par une épaisseur de copeau maxi et terminent leur travail en quittant le copeau à épaisseur nulle. L'effort généré est dirigé dans le sens d'usinage et s'éloigne de la matière, c'est un effort qui aide la machine à avancer mais qui décolle l'outil de la matière. Si la machine a un jeu fonctionnel on peut avoir des problèmes car l'effort peut décoller les écrous des vis pendant de brefs instants (jusqu'à ce que l'avance rattrape le jeu). La machine est donc sérieusement secouée.

En revanche les conditions de coupe sont excellentes car la dent de la fraise n'attaque pas tangentiellement. On n'a donc pas de refus de coupe et les états de surface sont meilleurs.



a. Fraisage en opposition.

b. Fraisage en avalant.

Figure 10 : Type de fraisage.**X. LA COUPE [6]:**

Définition : On appelle la coupe la couche de matière qui est enlevée par une action unique de la partie active et transformée en copeaux.

Avant d'entamer les paramètres de coupe, on décrira les différents mouvements entre l'outil et la pièce qui réalisent la coupe.

X.1. Différents mouvements qui réalisent la coupe [6]:

Pour enlever de la matière en cours d'usinage, il faut avoir deux mouvements nécessaires entre l'outil et la pièce :

- Mouvement de coupe.
- Mouvement d'avance.

D'une manière générale, les mouvements de coupe peuvent être donné soit par la pièce soit par l'outil.

X.1.1. Mouvement de coupe [6]:

Le mouvement de coupe est un mouvement relatif principal entre l'outil et la pièce. Il est caractérisé par la vitesse de coupe v_c .

X.1.2. Mouvement d'avance [6]:

Au mouvement de coupe, vient s'ajouter un autre mouvement relatif entre l'outil et la pièce, le mouvement d'avance, nécessaire à la génération de la surface de la pièce. Le mouvement d'avance est caractérisé par la vitesse d'avance F qui est une vitesse instantanée du mouvement d'avance du point considéré de l'arête de coupe par rapport à la pièce.

Le mouvement de coupe et le mouvement d'avance combinés constituent le mouvement résultant de coupe v_e .

Ces mouvements sont assurés par les éléments constitutifs de la machine outil.

X.2. Paramètres de coupe [6]:

Les paramètres de coupe sont des valeurs qui caractérisent les mouvements de l'outil et de la pièce à usiner ainsi que les valeurs de la surépaisseur d'usinage et de la section de copeau (dimension de copeau).

Lors d'un usinage par enlèvement de matière, on se retrouve dans la majorité des cas dans la configuration suivante :

- Une lame d'outil pénètre dans la matière pour enlever une partie de la pièce (copeau). (voir figure 11)
- L'outil suit une trajectoire par rapport à la pièce à usiner.

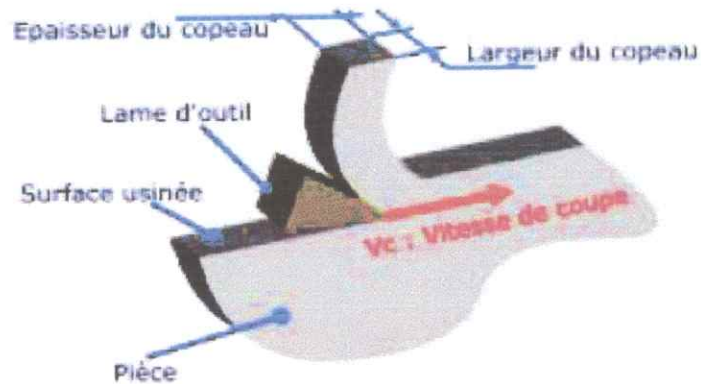


Figure 11 : vue générale sur l'usinage.

Pour obtenir une pièce usinée dans de bonnes conditions (bon état de surface, rapidité de l'usinage, usure modéré de l'outil, ...etc.), Il est nécessaire de régler certains paramètres de la coupe :

- Vitesse de coupe V_c .
- Vitesse d'avance F .
- Profondeur de passe a .

Ces paramètres sont définies avec d'autres critères de choix (paramètres de choix des conditions de coupe) : puissance de la machine, matière de la pièce, matière d'outil, forme d'outil, ... etc.

- **Vitesse de coupe** : est une vitesse instantanée du point considéré de l'arête (le plus éloigné) par rapport à la pièce. Elle est donnée par :

$$V_c = \frac{\pi \cdot D \cdot N}{1000}$$

Avec :

D : diamètre de la fraise en mm.

N : vitesse de rotation de la broche en tr/min.

V_c : vitesse de coupe en mm/min.

- **Vitesse d'avance** : c'est la vitesse de déplacement de l'outil sur la trajectoire d'usinage. pour le fraisage, cette vitesse est donner par :

$$F = Z \cdot f_z \cdot N.$$

Avec :

Z : nombre de dents de la fraise.

F_z : la distance que la dent va parcourir à chaque tour de la fraise mm/(tr.dent).

N : vitesse de rotation de la broche tr/min.

F : vitesse d'avance en mm/min.

- **Profondeur de passe** : c'est la pénétration axiale de l'outil dans la pièce pour enlever la matière, cette profondeur dépend de l'outil et de la matière de la pièce.

En cours d'usinage la machine peut entamer différentes trajectoires dans leurs passages.

X.3. Les types de trajectoires [3,7,13]:

Nous distinguons différents types de trajectoire :

X.3.1. Positionnement point à point : le passage d'un point à un autre s'effectue en programmant la position finale et le trajet parcouru pour atteindre cette position n'est pas contrôlée par le directeur de commande numérique (voir figure 12).

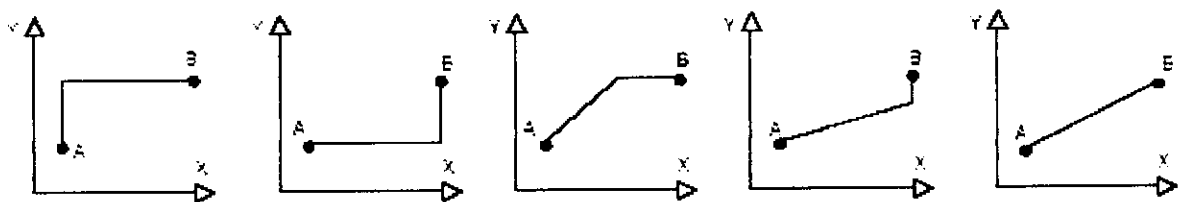


Figure 12 : Trajectoires en positionnement point à point [9].

X.3.2. Déplacement en paraxial : les trajectoires sont parallèles aux axes de déplacement de la machine (voir figure 13), et la vitesse de déplacement (programmable) est contrôlée.

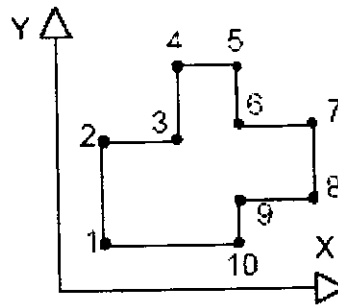


Figure 13 : Trajectoire en déplacement paraxial [9].

X.3.3. Déplacement en continu (trajectoires de contournage) : dans ce cas, les différents axes exécutants la trajectoire sont contrôlés en vitesse et en position pour assurer une synchronisation permanente des mouvements soit dans le plan (contournage en 2D)(voir figure 14)soit dans l'espace (contournage en 3D).

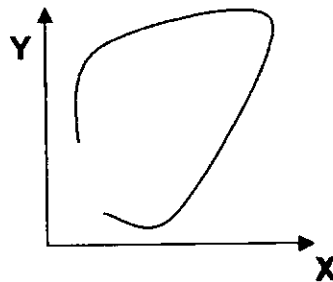


Figure 14 : Trajectoires en continu (contournage en 2D).

XI. Conclusion :

Dans ce chapitre nous avons présentés l'architecture générale des machines outils à commande numérique et en particulier les fraiseuses, et leurs fonctionnements. Ensuite nous avons données la syntaxe générale des programmes d'usinages de ces fraiseuses. Enfin nous avons présentés quelque notion sur le fraisage et la coupe.

A decorative border with a repeating floral motif runs along the top, bottom, and right edges of the page. The left edge features a vertical strip of black rectangular marks.

Chapitre 3

Stratégie d'usinage des surfaces gauches

I. INTRODUCTION :

L'usinage des surfaces gauches peut être réalisé par la succession de trois opérations; l'ébauchage, demi finition et la finition;. Cette dernière est l'opération la plus importante, à cause de son influence directe sur la qualité de la surface finale.

Cet usinage est caractérisé par plusieurs paramètres et stratégies qui permettent d'améliorer la qualité en fonction des contraintes imposées par le bureau d'étude, et de diminuer le temps d'usinage pour réduire les coûts de fabrication.

II- PARAMETRES D'USINAGE DES SURFACES GAUCHES [5] :

Lors de l'usinage d'une pièce, il est nécessaire de tenir compte des paramètres suivants :

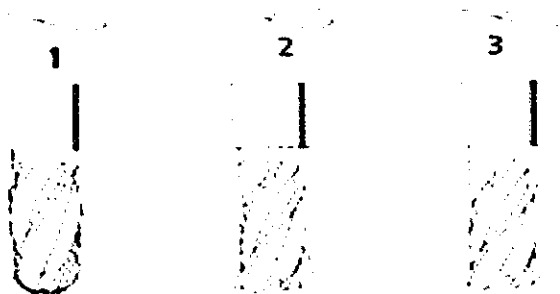
- Choix des outils (Types et dimensions) et modes d'usinage (voir paragraphe II-2 chapitre précédent).
- Détermination de la trajectoire de l'outil.
- Positionnement outil/pièce.
- Compatibilité de l'outil et la pièce.

II-1. Choix de l'outil [5,15,17,18,19,20,21,22,23] :

Pour l'usinage d'une surface de forme libre, il n'existe pas une seule forme d'outil qui permette d'obtenir exactement cette forme en un seul déplacement élémentaire. 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. Le choix d'un 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.

Trois types de fraises sont utilisés pour l'usinage des surfaces gauches :

- Fraise torique (rayonnée) : la partie active prend la forme d'un tore.
- Fraise cylindrique : la partie active est un cylindre.
- Fraise hémisphérique : la partie active prend la forme d'une sphère.



1. Hémisphérique. 2. Cylindrique. 3. Torique.
Figure 1 : les différents types de fraises.

Ces fraises enlèvent la matière par leur partie active qui est en contact avec la pièce à usiner.

Dans le cas de l'usinage sur des fraiseuses à 3 axes, les fraises hémisphériques sont les plus utilisées, pour garantir l'existence de la tangence et donc d'un unique point de contact sur des zones concaves et pour éviter les problèmes d'interférences. Il faut que la valeur de son rayon soit plus petite que le plus petit des rayons de courbures principaux de la forme. Sinon, il existe plusieurs points de contact et une zone non usinée. Comme il est nécessaire de conserver des petits rayons pour permettre d'usiner toute la pièce, ce qui diminue la rigidité de l'outil et augmente le nombre de passes nécessaires au respect d'une hauteur entre crêtes données. Pour cette forme d'outil, la vitesse de coupe est très faible lorsque la zone de contact est très proche de l'axe de rotation de l'outil ce qui entraîne un mauvais comportement de la coupe et une usure importante.

II.2. Les paramètres de guidage :

Il existe deux manières différentes pour le guidage de l'outil le long de la courbe :

- guidage du point extrémité de l'outil.
- guidage du point de contact (outil/matière).

En effet, pour l'usinage d'une surface quelconque, le point extrémité outil et le point de contact n'ont pas la même trajectoire, ainsi que les algorithmes de calcul de trajets si l'on guide l'extrémité de l'outil ou le point de contact.

Le mode de balayage de la surface permet d'usiner dans le sens «aller» ou dans les deux sens «aller et retour», couramment appelés «one-way» et «zig-zag» dans les logiciels. Pour les opérations de finition, on utilise le mode «one-way» pour usiner toujours en «avalant» afin d'améliorer l'état de surface. Pour usiner en avalant, il faut fixer le sens de parcours en fonction du sens de rotation de la broche.

II.3. Paramètres de passe et Trajet d'outil:

L'usinage des surfaces gauches est généralement un usinage par balayage. La génération du trajet d'usinage est une étape importante puisqu'elle sert à obtenir des chemins bien calculés dans le but de construire une surface de haute qualité.

Un trajet est représenté par une succession de couples formés par la position de l'extrémité de l'outil et la direction de l'axe outil. Générer des trajets outil, c'est trouver les positions de l'outil successives qui permettent d'usiner une surface selon des trajectoires définies par le programmeur machine outil. Ces trajectoires sont en général des courbes 3D.

Les déplacements de l'outil étant **rectilignes**, les courbes que doit suivre l'outil sont approchées par **interpolation linéaire**. Un trajet outil est donc **une ligne brisée** qui suit au mieux les surfaces selon un pas longitudinal (voir figure 2).

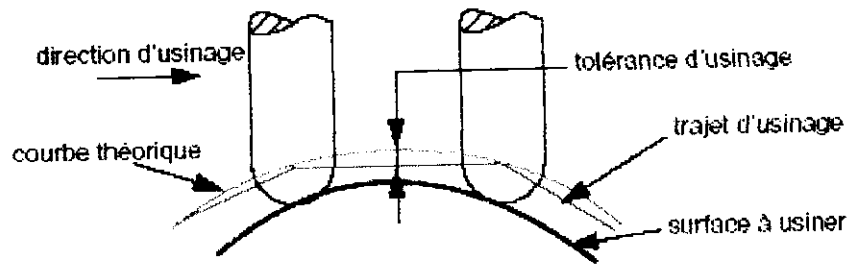


Figure 2: Trajet réel de l'outil pendant l'usinage.

III. POSITIONNEMENT DE L'OUTIL A LA SURFACE :

III.1. Calcul de la position d'outil tangente à la surface

L'outil (fraise) se positionne en un point précis par rapport à la surface à usiner. Pour usiner une surface gauche. Il faut que l'outil utilisé soit constamment tangent aux différents points de contact entre l'outil et la surface. Le repère de l'outil est le point centre (C_E). Lors du calcul du trajet de l'outil on fait toujours référence à ce point et c'est ainsi qu'on forme une enveloppe au dessus de la surface qui représente la trajectoire de ce point (voir la Figure 3). Le point de contact de la fraise avec la surface est appelé C_C (cutter contact) qui se trouve à la périphérie de l'outil à une distance r représentant le rayon de la pointe de l'outil pour une fraise hémisphérique et le petit rayon pour une fraise torique. La pointe de la fraise est le C_L (cutter location).

Le rayon de l'outil est le R . \vec{u} représente l'axe de rotation de la fraise, \vec{n} représente la normale de la surface au point de contact (voir la Figure 4).

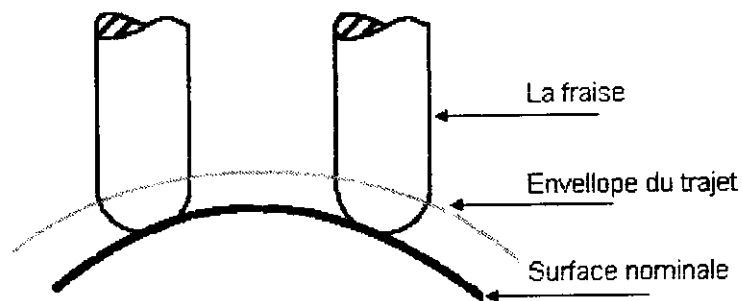


Figure 3 : Enveloppe de la surface a usinée.

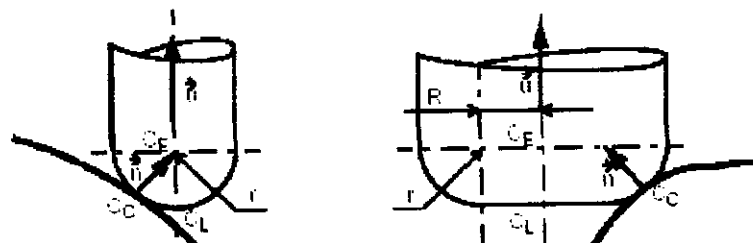


Figure 4 : Position de la fraise sur la surface.

Les positions des points de l'outil hémisphérique sont données par les équations suivantes :

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

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

$$\vec{\kappa} = \frac{\vec{u} \wedge \vec{n}}{\|\vec{u} \wedge \vec{n}\|}$$

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

A partir du point de contact, on peut calculer les coordonnées du centre de l'outil. Le posage de l'outil par rapport à la surface à usiner peut être fait de deux manières différentes :

- Posage par le point de contact de l'outil ; on appelle posage par le point de contact, une opération de calcul de la position du centre de l'outil à partir d'une position de contact donnée.
- Posage par le centre de l'outil : on appelle posage par le centre de l'outil, une opération de calcul direct d'une position du centre de l'outil sans la donnée initiale de la position du point de contact.

III.2. Méthodes de calcul des positions d'outils :

Plusieurs méthodes sont utilisées pour le calcul des positions d'outils :

III.2.1. Calcul par offset de la forme :

Cette méthode est basée sur le calcul a priori de la surface offset à la surface à usiner d'une valeur égale à celle du rayon de l'outil. Dans le cas de l'usinage à trois axes avec un outil hémisphérique, la position de l'outil tangente à la surface à usiner appartient à la surface parallèle (ou offset) de la valeur du rayon de l'outil. Le calcul de la position est alors basé sur l'adaptation de la stratégie d'usinage à la surface offset. L'avantage de cette méthode est qu'elle permet d'éviter les interférences locales, par contre son temps de calcul est plus long et les problèmes se posent aux raccordements entre les surfaces offset obtenues. L'offset d'une surface donnée avec une valeur de r est donnée par :

$$\vec{Q}^{off}(u, v) = \vec{Q}(u, v) + r\vec{n}$$

III.2.2. Calcul par la méthode de plongée (le copiage informatique) :

Le copiage informatique est une méthode non référencée, directement adaptée des méthodes de copiage traditionnelles. C'est une méthode de posage par le centre, implicite, directe ou indirecte, qui élimine les risques d'interférence locale.

Lors d'une opération de copiage classique, l'outil suit la surface. Physiquement, de la valeur de son rayon, il est toujours tangent à la surface. En fait, l'outil tombe sur la surface jusqu'à ce qu'il la touche. Le copiage informatique suit cette stratégie, l'outil glisse le long d'une droite jusqu'à ce qu'il touche la surface (voir figure 5).

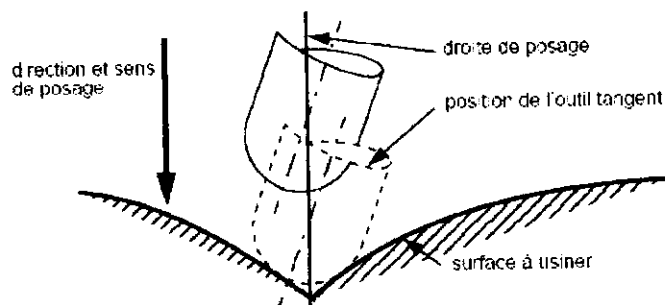


Figure 5 : Copiage informatique, guidage de l'outil tangent à la surface.

Les droites de posage sont parallèles à la direction de posage, qui n'est pas forcément l'axe de l'outil. Le calcul du trajet est partagé en deux activités, la première fait suivre à l'outil une courbe plane caractérisée par la stratégie requise, et la seconde positionne verticalement l'outil de façon à le rendre tangent à la surface.

IV. DEFANTS D'USINAGE :

IV.1. Erreur de flèche :

Un pas longitudinal d'usinage (un pas le long d'une passe d'usinage dans le sens de la longueur) est calculé en tenant compte de la tolérance d'usinage qui exprime la valeur maximale de l'erreur de flèche entre la courbe théorique et chacun des segments de la ligne brisée qui approximent cette courbe. La (Figure 6) représente ce type de défaut.

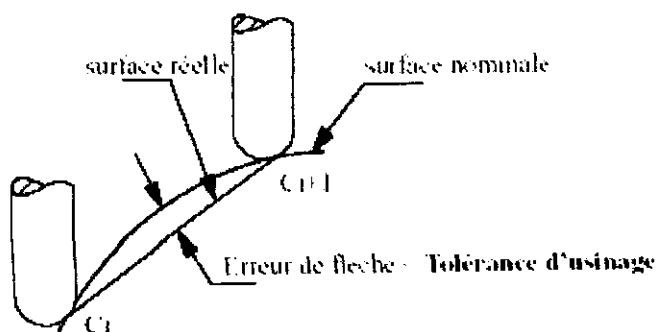


Figure 6 : Erreur de flèche sur les trajets

IV.2. Erreur de crête

Afin d'usiner la totalité de la surface, les passes sont juxtaposées avec une erreur donnée. Le pas d'usinage transversal peut être calculé en respectant soit un critère de distance maximale entre deux passes, soit un critère de hauteur de crête.

Ces deux paramètres sont équivalents et liés par le rayon de l'outil. De plus, la distance maximale entre passes peut être établie entre les surfaces de guidage ou sur la surface nominale.

Pour un outil hémisphérique, si R est le rayon de l'outil, et d la distance entre passes sur la surface, alors la hauteur de crête h_c est donnée par (voir figure 7):

$$h_c = R - \sqrt{R^2 - \left(\frac{d}{2}\right)^2} \quad (3.1)$$

Si $R \gg d$ alors :

$$h_c = \frac{d^2}{8 \times R} \quad (3.2)$$

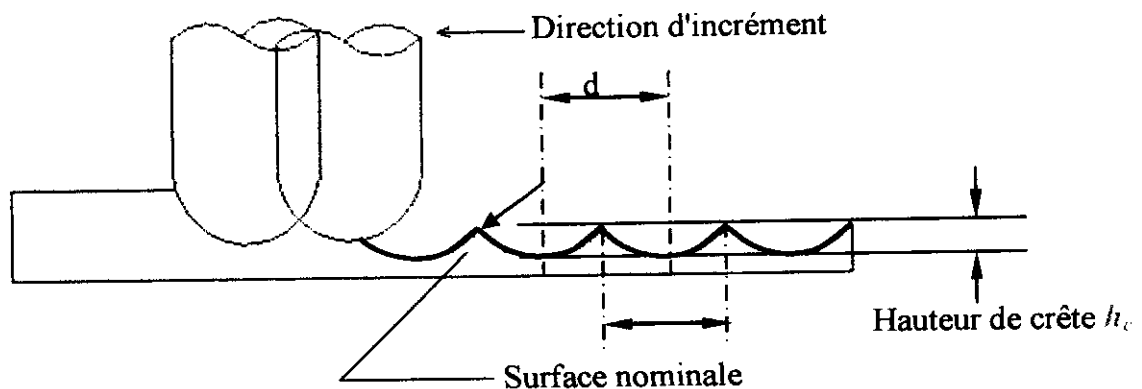


Figure 7 : Hauteur de crête et distance maximale entre passes.

IV.3. Problème d'interférence :

Parmi les problèmes qui se posent lors de la génération des trajets d'usinage des surfaces gauches sont les interférences et les collisions.

Les positions d'outil en interférence sont des portions du trajet d'usinage pour lesquelles l'outil n'est plus tangent localement à la surface qu'il usine et perce la surface de la pièce (voir figure 8.a). Ces interférences se produisent au niveau des zones concaves des surfaces. Par contre, les collisions sont des contacts inopportuns entre l'outil et les surfaces d'arrêt ou avec d'autres surfaces à usiner (voir figure 8.b).

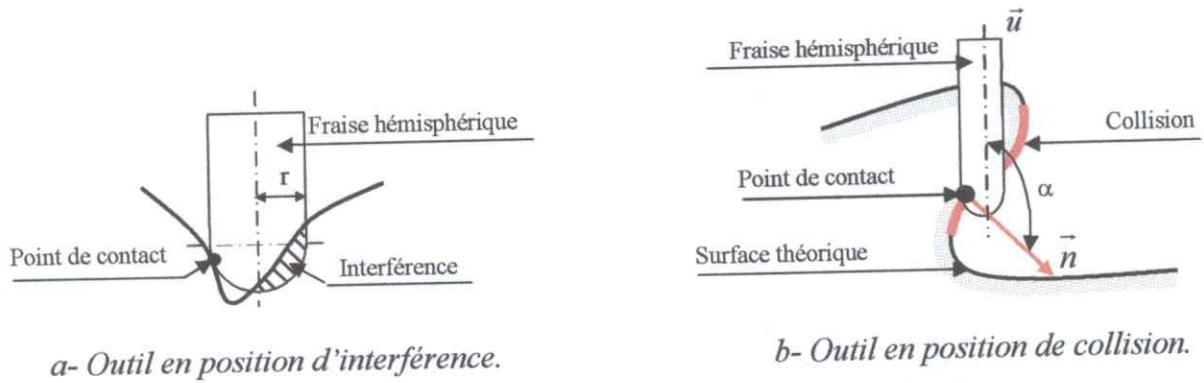


Figure 8 : Outil en position d'interférence et en collision.

V. METHODES D'USINAGE DES SURFACES GAUCHES EN FINITION :

Plusieurs techniques peuvent être utilisées dans l'usinage des surfaces gauches:

V.1. Méthodes isoparamétrique :

V.1.1. Aller simple (One-Way):

L'usinage des surfaces avec la méthode One-Way se fait selon l'isoparamétrique u ou v jusqu'à sa fin, ensuite déplacé l'outil en quittant la surface vers l'autre isoparamétrique, tel que expliquer dans l'algorithme suivant :

Répéter

- Usiner la courbe isoparamétrique (u ou v) jusqu'à sa fin,
- Faire déplacer l'outil en quittant la surface,
- Passer à la courbe suivante,

Jusqu'à fin surface

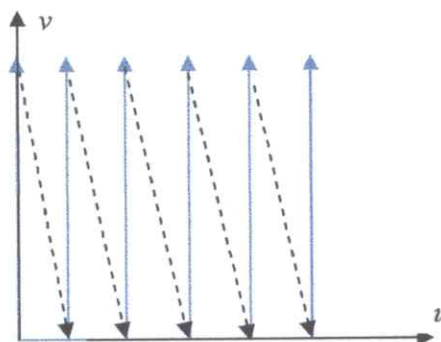


Figure 9 : Usinage en aller simple.

V.1.2. Aller retour (Zig-Zag) :

L'usinage des surfaces avec la méthode de Zig-Zag se fait selon l'isoparamétrique u ou v jusqu'à sa fin (voir figure 10), ensuite on déplace l'outil sans quitter la surface vers l'autre isoparamétrique dans le sens opposé tel que expliqué dans l'algorithme suivant :

Répéter

Usiner la courbe isoparamétrique (u ou v) jusqu'à sa fin,

Faire déplacer l'outil sans quitter la surface,

Passer à la courbe suivante dans l'autre sens,

Jusqu'à fin surface

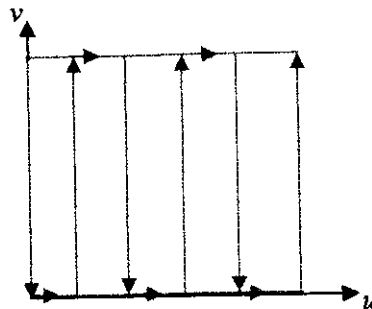


Figure 10 : Usinage en aller retour.

V. 1.3. Concentrique :

L'usinage des surfaces avec la méthode concentrique se fait selon des courbes parallèles aux frontières du domaine de définition de la surface jusqu'à la fin de la surface (voir figure 11).

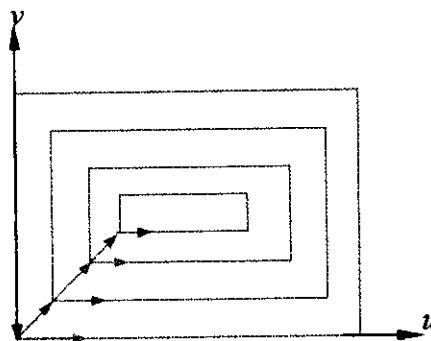


Figure 11 : Usinages par la méthode concentrique.

V. 1.4. Spiral-In et Spiral-Out:

Lors de l'usinage des surfaces avec ces deux méthodes l'outil suit une trajectoire en forme de spirale jusqu'à l'usinage de toute la surface, mais la différence réside dans le point de départ (sens de parcours).

Dans la première méthode (Spiral In); l'outil démarre de l'extérieur de la surface et suit la trajectoire jusqu'à atteindre l'intérieur (voire la Figure 12), tandis que la

seconde méthode c'est le contraire et l'outil démarre de l'intérieur pour atteindre l'extérieur (voir la Figure 13). Les flèches montrent le sens de parcours dans l'usinage en Spiral-In et Spiral-Out.

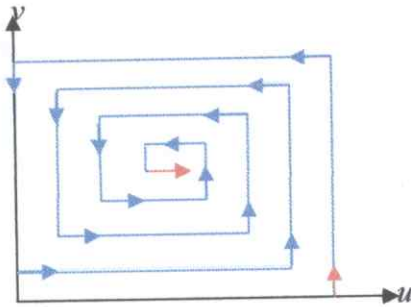


Figure 12 : Usinage en Spiral in.

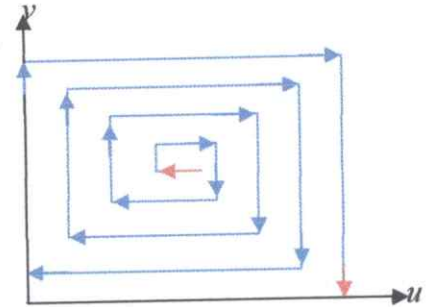


Figure 13 : Usinage Spiral out.

V. 1.5. Radiale :

Cette méthode d'usinage consiste à usiner la surface en commençant par son centre jusqu'à sa limite tout en effectuant un déplacement suivant une trajectoire radiale. Ce processus est répété jusqu'à l'usinage de toute la surface en déplaçant chaque fois le point de l'extrémité. La (Figure 14) montre le principe de l'usinage Radiale et les lignes radiales sont les lignes suivies par l'outil en cours d'usinage.

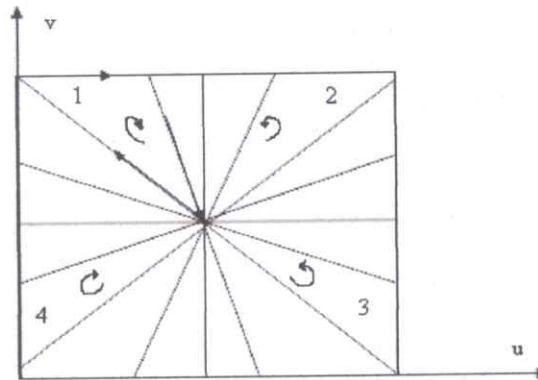


Figure 14 : Usinage avec la méthode Radiale

V.2. Autres méthodes d'usinage [5, 14, 9,11]:

Autres méthodes d'usinage peuvent être utilisées à part les méthodes basées sur les courbes isoparamétriques :

V.2.1. Usinage par la méthode des plans parallèles :

Le trajet d'usinage dans cette méthode est obtenu par la construction des courbes qui sont le résultat de l'intersection de la surface à usiner avec des plans verticaux parallèles au plan (XZ) (voir la Figure 15). Les plans verticaux peuvent avoir n'importe quelle inclinaison.

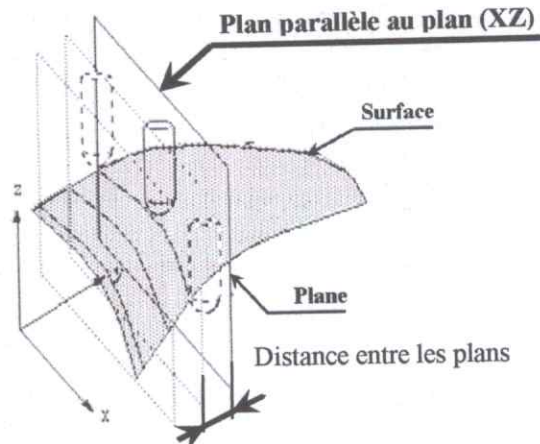


Figure 15 : Méthode des plans parallèles.

V.2.2. Usinage par la méthode Z-Constant :

Le trajet d'usinage dans cette méthode est obtenu par l'intersection de la surface à usiner avec des plans horizontaux qui prennent différentes valeurs de Z. Les courbes obtenues sont appelés « courbes de niveau » (voir la Figure 16).

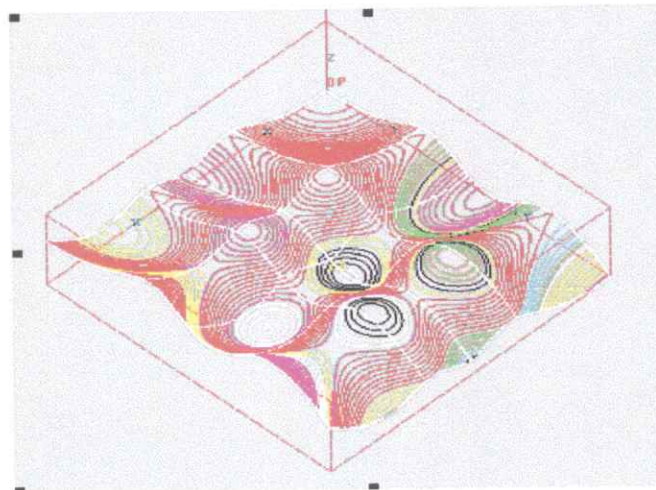


Figure 16 : Méthode de Z-Constant.

VI. CONCLUSION :

Dans ce chapitre nous avons présentés les paramètres d'usinage des surfaces gauches (choix d'outils, les modes d'usinage, positionnement de l'outil, etc....).

Ensuite nous avons vu comment positionner l'outil par rapport à la surface et les défauts d'usinage, ainsi que les différentes stratégies d'usinage à utiliser dans la finition.

L'étude de cette partie nous permet d'entamer notre travail effectif et nous commencerons par l'étude des différents paramètres d'approximation d'une surface conduisant à minimiser le temps d'usinage.



Chapitre 4

Triangulation des surfaces

I. Introduction :

La simulation d'usinage est un moyen très important et très utilisé pour examiner et vérifier les trajets d'usinage avant d'aller vers l'usinage réel. Sans simulation, les chemins d'outil sont examinés à plusieurs reprises sur des pièces modèles et les corrections sont effectuées au fur et à mesure. L'essai sur les machines réelles exige de travailler sur des machines chères où les risques d'erreurs sont très importants ainsi que les temps d'usinage.

Dans ce chapitre on va présenter une méthode d'approximation basée essentiellement sur la triangulation. Elle consiste à diviser la surface en des triangles tout en respectant certaines conditions.

II. Intersection d'une surface libre et d'un plan [15,24,25,26] :

Le problème d'intersection plan considéré comme classe spéciale de problème d'intersection surface-surface (SSI) a été largement étudié en CAO. Et peut être résolu de la même manière que (SSI). Généralement, deux méthodes sont utilisées pour calculer l'intersection surface-surface :

Subdivision récursive (triangulation).
Méthodes analytiques.

Quand une surface libre intersecte un plan infini, quatre situations sont possibles:

- 1/ Ensemble vide.
- 2/ Une collection de points.
- 3/ Une collection de courbes lisses.
- 4/ Toutes les combinaisons 2/ et 3/.

III. Triangulation [15,24,25,26]:

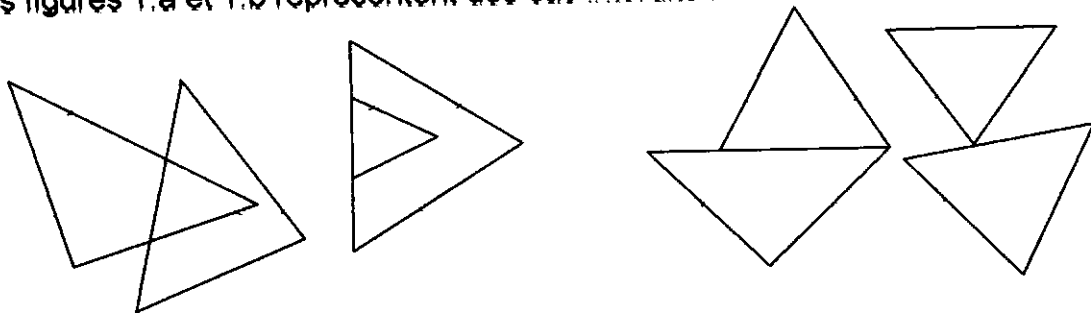
L'approximation du surface libre avec des triangles permet de simplifier les calculs des intersections entre cette surface et un plan infini et permet d'avoir un système d'équation linéaire. Cette étape d'approximation est appelée "Triangulation".

Cette dernière est un problème de la géométrie informatique.

Un ensemble de triangles constituent une triangulation de la surface si :

1. l'intersection de deux triangles est vide à l'intérieur quelque soit les deux triangles.
2. l'intersection de deux triangles est soit vide, soit un coté, soit un sommet.

Les figures 1.a et 1.b représentent des cas interdits :



b : Condition 1 (non vérifiée).

a : Condition 2 (non vérifiée).

Figure 1 : intersection de deux triangles.

La triangulation peut être faite de deux manières différentes :

III.1. Triangulation uniforme [26] :

Dans la triangulation uniforme le nombre de triangles, pour chaque surface, est fixés suivant la direction u et la direction v dans le plan paramétrique (u,v) , ensuite cette information est utilisée pour faire la triangulation dans l'espace 3D. Avec cette méthode de subdivision, la triangulation est rapide, mais il faut choisir un nombre important de triangles dans chaque direction pour avoir une bonne approximation, ce qui augmente l'espace mémoire nécessaire au stockage des différentes informations de chaque triangle et augmente le temps de calcul des intersections.

La figure (2) montre une surface nominale et sa triangulation avec un nombre de triangles égale à 10 et à 5 suivant les directions u et v respectivement.



a. Surface nominale.

b. Paramètres de subdivision

c. Surface triangulée.

Figure 2 : Triangulation uniforme d'une surface nominale.

III.2. Triangulation adaptative [26] :

La triangulation adaptative est la plus utilisée pour approximer une surface de forme libre, puisqu'elle permet de réduire le nombre de triangles générés et permet de réduire le temps de calcul des intersections. De plus, elle s'adapte mieux aux variations géométriques de la forme de la surface. Elle nécessite la spécification des critères d'approximation de la surface afin que l'ensemble des triangles donne une

bonne représentation de la surface théorique. Dans cette triangulation les conditions à respecter pour chaque triangle sont les suivantes :

- La longueur maximale de chaque coté du triangle doit être inférieure à une tolérance « d » spécifiée par l'utilisateur. (figure 3.a).
- La distance entre le milieu de chaque coté du triangle et la surface nominale doit être inférieure à une tolérance « e » spécifiée par l'utilisateur. (figure 3.b).
- La distance entre le centre de gravité du triangle et la surface nominale doit être inférieure à une tolérance « e » spécifiée par l'utilisateur. (figure 3.c).

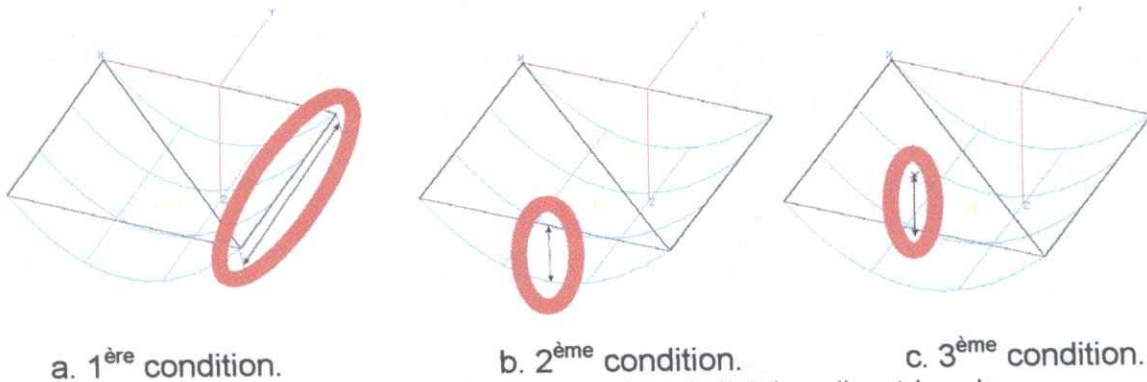


Figure 3 : Conditions de subdivision d'un triangle.

Pour chaque triangle nous calculons ensuite « e » et « d » réels pour voir si le triangle garantira les conditions précédentes. Sinon, le triangle est récursivement subdivisé. Quand un côté est trop long, le triangle est subdivisé (figure 4).

Chacune des arêtes du triangle est testée, et si la subdivision ne converge pas pour chacun des trois côtés, le triangle est subdivisée en quatre triangles en utilisant la subdivision barycentrique. Si la subdivision ne converge pas pour un ou deux côtés, alors le triangle est subdivisée en deux ou trois triangles, respectivement (voir figure 4). Le même procédé est récursivement appliqué à tous les nouveaux triangles résultant de la subdivision jusqu'à ce que chaque triangle vérifie toutes les conditions.

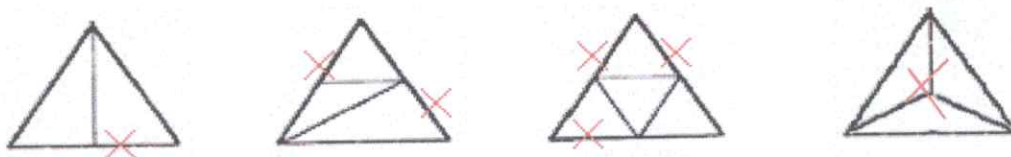
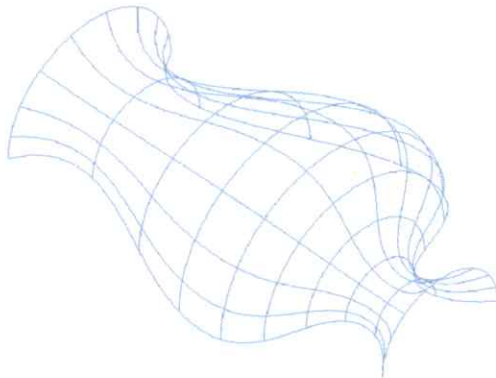


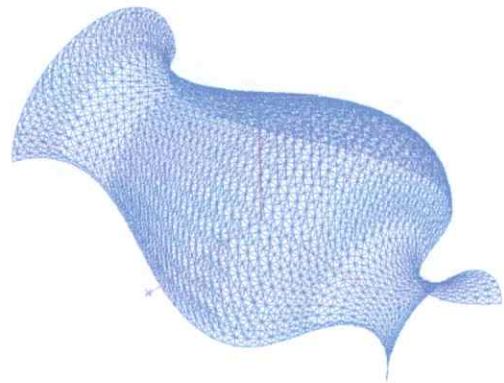
Figure 4 : Différents cas de subdivision d'un triangle.

A la fin de la triangulation, nous obtenons un ensemble de triangle qui donne une bonne approximation de la surface où tous les sommets des triangles appartiennent à la surface nominale. Les triangles générés seront utilisés dans l'étape de calcul de l'intersection.

La figure (5) montre une surface nominale et sa triangulation adaptative pour le cas : $d = e = 5\text{mm}$,



a. surface nominale.



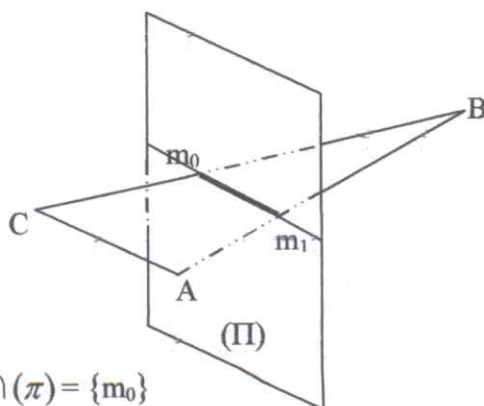
b. surface triangulé avec

Figure 5 : Triangulation adaptative d'une surface nominale.

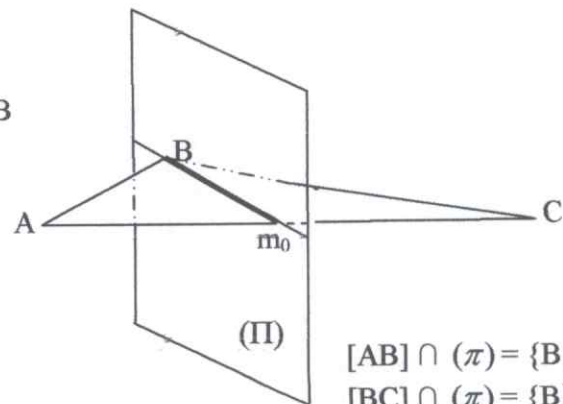
IV. Intersection d'un triangle et un plan dans l'espace tridimensionnelle [15,24,25,26]:

Pour qu'un plan (π) intersecte un triangle, il faut et il suffit qu'il intersecte au moins un segment du triangle (figure 6).

Pour que l'intersection entre un plan et un triangle soit un ensemble vide, il faut qu'il existe deux segments de triangle tels que l'intersection entre chacun d'eux et le plan soit un ensemble vide (figure 6).



$$\begin{aligned}
 [CB] \cap (\pi) &= \{m_0\} \\
 [BA] \cap (\pi) &= \{m_1\} \\
 [CA] \cap (\pi) &= \emptyset \\
 (ABC) \cap (\pi) &= [m_0 m_1]
 \end{aligned}$$



$$\begin{aligned}
 [AB] \cap (\pi) &= \{B\} \\
 [BC] \cap (\pi) &= \{B\} \\
 [AC] \cap (\pi) &= \{m_0\} \\
 (ABC) \cap (\pi) &= [B m_0]
 \end{aligned}$$

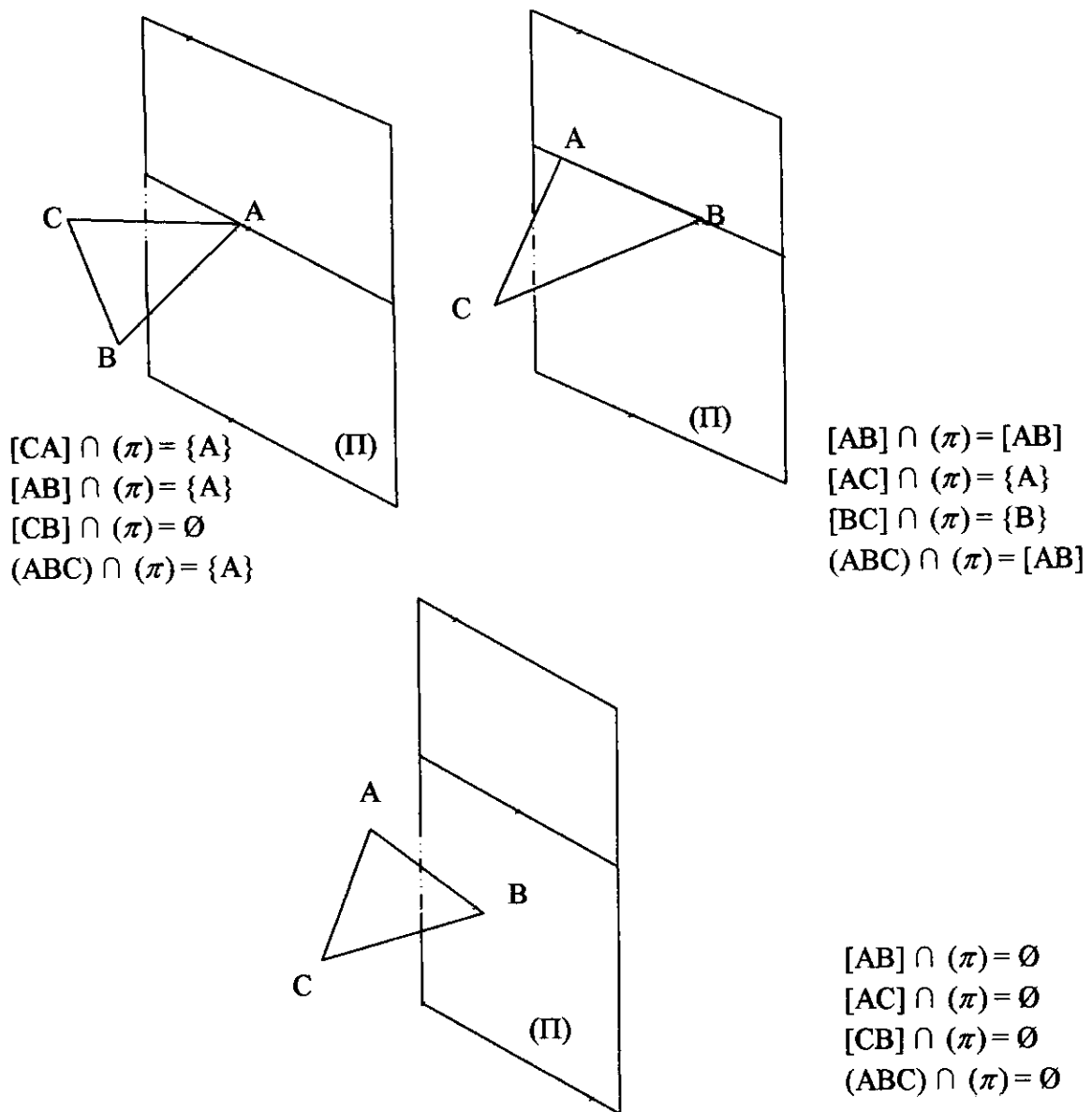


Figure 6 : Intersection entre un triangle et un plan.

Soit $P_0P_1P_2$ un triangle telle que $P_0(x_0, y_0, z_0)$, $P_1(x_1, y_1, z_1)$ et $P_2(x_2, y_2, z_2)$, (π) un plan dont l'équation est: $Ax+By+Cz+d=0$.

- Chercher l'intersection entre $P_0P_1P_2$ et (π) revient à trouver l'intersection entre $[P_0P_1]$ et (π) , $[P_0P_2]$ et (π) , $[P_1P_2]$ et (π) ;
- pour calculer l'intersection entre $[P_0P_1]$ et (π) :

Il faut trouver le système d'équations qui définit la droite (P_0P_1) . Supposons que le système d'équations est:

$$\begin{cases} A'x + B'y + C'z + d' = 0 \\ A''x + B''y + C''z + d'' = 0 \end{cases}$$

- trouver l'intersection entre (P_0P) avec (P) point sur la droite (P_0P_1) et (π) , passe par la résolution du système

$$\begin{cases} Ax + By + Cz + d = 0 \\ A'x + B'y + C'z + d' = 0 \\ A''x + B''y + C''z + d'' = 0 \end{cases}$$

- si le système n'admet pas de solution donc $(P_0P) \cap (\pi) = \emptyset$ d'où $[P_0P_1] \cap (\pi) = \emptyset$
- si le système admet une infinité de solution donc $(P_0P) \cap (\pi) = (P_0P)$ d'où $[P_0P_1] \cap (\pi) = [P_0P_1]$
- si le system admet une solution unique alors $(P_0P) \cap (\pi) = \{m\}$, on vérifie alors si $\{m\} \in [P_0P_1]$.
- si $m \in [P_0P_1]$ alors $[P_0P_1] \cap (\pi) = \{m\}$ sinon $[P_0P_1] \cap (\pi) = \emptyset$

IV.1. La droite dans l'espace:

Par deux points P_0, P_1 dans l'espace on ne peut faire passer qu'une seule droite. Soit P_0, P_1 deux points de l'espace \mathbb{R}^3 . La droite qui passe par P_0 et P_1 est définie comme l'ensemble des points M qui vérifient la relation $\overline{P_0M} \parallel \overline{P_0P_1}$ (figure 7).

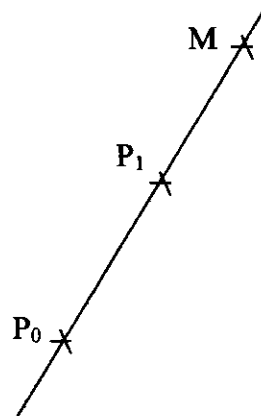


Figure 7 : Droite dans l'espace.

Soient $P_0(x_0, y_0, z_0), P_1(x_1, y_1, z_1)$ deux points de l'espace, soit $M(x, y, z)$ au point de la droite (Δ_M) qui passe par les deux point P_0, P_1 .

$$M \in (\Delta_M) \Leftrightarrow \overline{P_0M} \parallel \overline{P_0P_1}.$$

$$\Leftrightarrow \exists \alpha \in \mathbb{R} / \overline{P_0M} = \alpha \cdot \overline{P_0P_1}.$$

$$\Leftrightarrow \exists \alpha \in \mathbb{R} / \begin{vmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{vmatrix} = \alpha \cdot \begin{vmatrix} x_1 - x_0 \\ y_1 - y_0 \\ z_1 - z_0 \end{vmatrix} \quad (5.4)$$

$$\text{Donc } \begin{cases} x - x_0 = \alpha(x_1 - x_0) \\ y - y_0 = \alpha(y_1 - y_0) \\ z - z_0 = \alpha(z_1 - z_0) \end{cases} \Leftrightarrow \begin{cases} \alpha = \frac{x - x_0}{x_1 - x_0} & \text{si } x_1 \neq x_0 \\ \alpha = \frac{y - y_0}{y_1 - y_0} & \text{si } y_1 \neq y_0 \\ \alpha = \frac{z - z_0}{z_1 - z_0} & \text{si } z_1 \neq z_0 \end{cases}$$

$$\text{D'où ; } \frac{x - x_0}{x_1 - x_0} = \frac{y - y_0}{y_1 - y_0} = \frac{z - z_0}{z_1 - z_0} \quad (5.5)$$

De la relation (5.5) on peut définir un système d'équations composé de trois équations

$$(I) \begin{cases} (x - x_0)(y_1 - y_0) = (x_1 - x_0)(y - y_0) \dots (1') \\ (y - y_0)(z_1 - z_0) = (y_1 - y_0)(z - z_0) \dots (2') \\ (x - x_0)(z_1 - z_0) = (x_1 - x_0)(z - z_0) \dots (3') \end{cases}$$

Dans ce système d'équations on remarque que chaque équation peut être obtenue par les deux autres équations. Donc on peut éliminer une des équations, cette élimination se fait selon les cas suivants :

1^{er} cas: si $x_1 \neq x_0$, $y_1 \neq y_0$, $z_1 \neq z_0$.

Dans ce cas on peut éliminer n'importe quelle équation et en éliminant l'équation (2') on obtient le système.

$$\begin{cases} (y_1 - y_0)x - (x_1 - x_0)y - (y_1 - y_0)x_0 + (x_1 - x_0)y_0 = 0 \\ (z_1 - z_0)x - (x_1 - x_0)z - (z_1 - z_0)x_0 + (x_1 - x_0)z_0 = 0 \end{cases}$$

2^{eme} cas: si $x_1 \neq x_0$, $y_1 \neq y_0$, $z_1 = z_0$.

Remplaçons dans le système (I) $z_1 - z_0$ par 0 on obtient:

$$\begin{cases} (x - x_0)(y_1 - y_0) = (x_1 - x_0)(y - y_0) \\ (y - y_0).0 = (y_1 - y_0).(z - z_0) \\ (x - x_0).0 = (x_1 - x_0).(z - z_0) \end{cases} \Leftrightarrow \begin{cases} (x - x_0).(y_1 - y_0) = (x_1 - x_0).(y - y_0) \\ (y_1 - y_0).(z - z_0) = 0 \\ (x_1 - x_0).(z - z_0) = 0 \end{cases}$$

On sait que $x_1 - x_0 \neq 0$ et $y_1 - y_0 \neq 0$ d'où le système devient

$$\begin{cases} (x - x_0)(y_1 - y_0) = (x_1 - x_0)(y - y_0) \\ z - z_0 = 0 \\ z - z_0 = 0 \end{cases}$$

On remarque que la troisième équation n'est rien d'autre que la deuxième équation, donc on élimine la troisième équation et le système équivalent est donné par

$$\begin{cases} x(y_1 - y_0) - y(x_1 - x_0) - x_0(y_1 - y_0) + y_0(x_1 - x_0) = 0 \\ z - z_0 = 0 \end{cases}$$

3^{ème} cas: si $x_1 \neq x_0$, $z_1 \neq z_0$, $y_1 = y_0$.

Par les mêmes étapes on peut trouver le système suivant:

$$\begin{cases} x(y_1 - y_0) - y(x_1 - x_0) - x_0(y_1 - y_0) + y_0(x_1 - x_0) = 0 \\ z - z_0 = 0 \end{cases}$$

4^{ème} cas: si $x_1 \neq x_0$, $y_1 = y_0$, $z_1 = z_0$.

$$\begin{cases} y - y_0 = 0 \\ z - z_0 = 0 \end{cases} \text{ Cette droite est parallèle à l'axe (ox).}$$

5^{ème} cas: si $x_1 = x_0$, $y_1 \neq y_0$, $z_1 \neq z_0$.

$$\begin{cases} y(z_1 - z_0) - z(y_1 - y_0) - y_0(z_1 - z_0) + z_0(y_1 - y_0) = 0 \\ x - x_0 = 0 \end{cases}$$

6^{ème} cas: si $x_1 = x_0$, $y_1 \neq y_0$, $z_1 = z_0$.

$$\begin{cases} x - x_0 = 0 \\ z - z_0 = 0 \end{cases} \text{ Cette droite est parallèle à l'axe (oy).}$$

7^{ème} cas: si $x_1 = x_0$, $y_1 = y_0$, $z_1 \neq z_0$.

$$\begin{cases} x - x_0 = 0 \\ y - y_0 = 0 \end{cases} \text{ Cette droite est parallèle à l'axe (oz).}$$

IV.2. Intersection entre un plan et une droite dans l'espace:

L'intersection entre un plan et une droite dans l'espace ne peut être que:

- Ensemble vide.
- La droite même.
- Un point.

Pour déterminer l'ensemble d'intersection entre un plan et une droite il suffit de résoudre le système d'équations dont la forme générale est

$$(II) \begin{cases} Ax + By + Cz + d = 0 \\ A'x + B'y + C'z + d' = 0 \\ A''x + B''y + C''z + d'' = 0 \end{cases}$$

Telle que $Ax + By + Cz + d = 0$ est l'équation de plan et la droite est définie par le système suivant:

$$\begin{cases} A'x + B'y + C'z + d' = 0 \\ A''x + B''y + C''z + d'' = 0 \end{cases}$$

On remarque que le système (II) est un système linéaire à trois inconnues x, y, z et à trois équations, qui peut être résolue par la méthode Gauss ou toute autre méthode de résolution.

IV.3. Appartenance d'un point à un segment de droite:

Soit $[AB]$ un segment de droite dans l'espace on dit que

$$M \in [AB] \Leftrightarrow \exists \alpha \mid 0 \leq \alpha \leq 1 \quad \overrightarrow{AM} = \alpha \cdot \overrightarrow{AB}$$

$$\begin{cases} M_1 \in [AB] & \text{et } 0 \leq \alpha \leq 1 \\ M_2 \notin [AB] & \text{car } \alpha \geq 1 \\ M_3 \notin [AB] & \text{car } \alpha < 0 \end{cases}$$

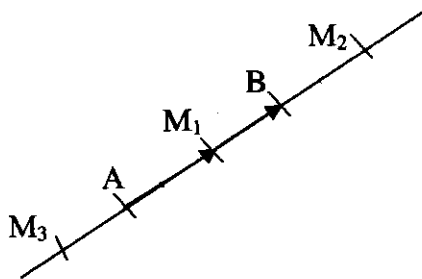


Figure 8 : Position des différents points par rapport à AB.

De la relation précédente on peut déduire les relations suivantes:

$$\overrightarrow{AM} = \alpha \cdot \overrightarrow{AB} \quad 0 \leq \alpha \leq 1$$

On pose $A(x_A, y_A, z_A)$, $B(x_B, y_B, z_B)$ et $M(x, y, z)$ Donc on a:

$$\begin{pmatrix} x - x_A \\ y - y_A \\ z - z_A \end{pmatrix} = \alpha \cdot \begin{pmatrix} x_B - x_A \\ y_B - y_A \\ z_B - z_A \end{pmatrix} \quad \text{et } 0 \leq \alpha \leq 1$$

Donc $x - x_A / x_B - x_A = y - y_A / y_B - y_A = z - z_A / z_B - z_A = \alpha$.

Donc il suffit que: $0 \leq x - x_A / x_B - x_A \leq 1$ ou $0 \leq y - y_A / y_B - y_A \leq 1$ ou $0 \leq z - z_A / z_B - z_A \leq 1$.

V. Conclusion:

Dans ce chapitre nous avons présenté les différentes méthodes de triangulation des surfaces (triangulation uniforme et triangulation adaptative) et de calcul des intersections entre un triangle et un plan. Le prochain chapitre sera consacré à la conception et à l'implémentation de notre application.



Chapitre 5

Conception et implémentation

I- INTRODUCTION :

Nous avons présenté dans les chapitres précédents les notions fondamentales de l'usinage des surfaces gauches, ainsi que les aspects mécaniques liés à l'opération d'usinage de ces surfaces et en particulier le fraisage à 3 axes. La suite du travail sera consacrée à la conception de l'application avec le langage de modélisation UML.

II- METHODE DE MODELISATION :

Dans notre application, nous allons faire la modélisation en utilisant le langage de modélisation UML qu'est un langage visuel et adapté à toutes les phases du développement et compatible avec toutes les techniques de réalisation de l'application et indépendant du langage de programmation utilisé et qui permet différentes vues pour représenter notre système.

III- REALISATION DE L'APPLICATION :

III.1- Etablissement du cahier de charge :

III.1.1- Présentation du projet :

Ce travail s'insère dans le cadre du développement d'outils de conception et de fabrication des surfaces gauches initié par l'équipe Conception et Fabrication Assistées par Ordinateurs (CFAO), <<Automatisation du processus de fabrication des surfaces gauches composées sur des fraiseuses à commande numérique par la prise en compte de l'aspect discret des surfaces et des conditions de coupe>>.

L'objectif de ce travail est le développement d'une application logiciel sous Windows pour l'usinage des surfaces gauches sur des fraiseuses CNC trois axes permettant de minimiser les temps d'usinage par l'optimisation du choix des dimensions et du nombre des outils.

III.1.2- Problématique :

Les surfaces de formes libres (moules, formes esthétiques, formes aérodynamiques, matrices, ...etc.) sont très courantes dans notre vie quotidienne. Ces surfaces doivent répondre aux différents critères esthétiques et fonctionnels.

L'usinage de ces surfaces sur des fraiseuses à commande numérique (CNC) cumulent les différentes difficultés de réalisation (les formes ne sont pas facilement réalisables, le respect de la forme et des contraintes fonctionnelles imposées, problèmes de collision et d'interférences, conditions d'usinage ... etc.) ce qui impose une attention particulière dans leur mise en production. Le temps passé dans les opérations d'usinages des surfaces gauches est très important et sa réduction est très primordiale pour les applications industrielles. Le choix optimum des formes et des dimensions des outils ainsi que les stratégies d'usinage conduisent à réduire d'une façon considérable ce temps.



III.1.3- Objectifs visés :

Notre objectif principal dans l'application est l'aide au choix des outils et de stratégies d'usinage pour l'usinage en finition des surfaces gauches. Cet objectif peut être subdivisé en plusieurs objectifs :

Premier objectif :

Détection des collisions et d'interférence, ce même objectif peut être subdivisé en plusieurs sous objectifs :

- Création des sommets sur la surface nominale que nous souhaitons usinées.
- Approximation de la surface par des triangles à partir des sommets créés.
- Optimisation de nombre de triangles de surface en fonction des paramètres introduits par l'utilisateur (erreur maximum et longueur maximum).
- Regroupement des sommets et des triangles en des régions.

Deuxième objectif :

L'objectif principal de notre travail est la minimisation du temps d'usinage tout en assurant la qualité de surface imposé par le cahier des charges, qui lui même peut être divisé en sous objectifs :

- Choix des outils adéquats.
- Choix de la stratégie d'usinage.
- Continuité d'usinage.

Troisième objectif :

Optimisation du trajet d'usinage pour la stratégie d'usinage Z-constant. Cet objectif peut être aussi divisé en sous objectifs :

- Localisation des points d'intersections.
- Construction des contours par plan.
- Génération du trajet final.
- Simulation d'usinage.
- Génération du programme d'usinage G-Code.

III.1.4- Plateforme exigée :

1. Recommandations matérielles (configuration minimum nécessaire) :

- ◆ Intel Pentium 4 de 2 GHz ou équivalent.

- ◆ 512 Mo de mémoire RAM.
- ◆ Carte accélératrice 3D de 64 Mo compatible OpenGL ou équivalente.
- ◆ Lecteur de CD-ROM/DVD-ROM.
- ◆ Souris PS/2.
- ◆ Clavier.
- ◆ Ecran de 19 pouces.

2. Systèmes d'exploitation nécessaires :

- ◆ Windows 98, Windows ME, Windows XP et Windows 2000
- ◆ Notez que Windows 95, Windows NT 4.0 ne sont pas compatibles.

3. Logiciels requis :

- ◆ Builder C++ V.6.
- ◆ Bibliothèque graphique OpenGL.

III.1.5- Solutions proposées :

Pour résoudre les problèmes cités précédemment, nous allons développer une application logicielle graphique et interactive sous un environnement windows. Pour atteindre notre objectif, nous allons procéder comme suit :

- Développement des algorithmes permettant la subdivision (triangulation) des surfaces (NURBS) et le groupement des points en des régions distinctes (concaves, convexes, ... etc.).
- Développement des algorithmes de détection des interférences et collisions (visibilité) durant l'usinage.
- Développement des algorithmes permettant de calculer le nombre et les dimensions optimales des outils, à utiliser pour l'usinage des différentes régions, pour l'opération de finition sous contraintes (temps minimum d'usinage, qualité de surface, ...etc.) tout en évitant les problèmes d'interférences et de collisions.
- Etude des problèmes de continuité d'usinage.

IV- MODELISATION DE L'APPLICATION EN UML :

Comme cité auparavant ; nous allons présenter la conception de l'application avec le langage de modélisation UML ; en utilisant cinq diagrammes parmi l'ensemble des diagrammes qui existe (9 diagrammes) :

- Diagramme de cas d'utilisation.
- Diagramme de classe.

- Diagramme de collaboration.
- Diagramme d'activité.
- Et enfin diagramme de séquence.

IV.1- diagramme de cas d'utilisation :

Le diagramme de cas d'utilisation est le moyen le plus efficace pour représenter les besoins en UML. Il décrit la succession des opérations réalisées par un acteur et assure la relation entre l'utilisateur et les objets que le système met en œuvre.

Modélisation débutée par le diagramme de cas d'utilisation, qui représente les principaux cas d'utilisation de notre système. (Figure 1)

Le premier cas d'utilisation est le « choix de la surface » que nous voulons usinée en action à la réaction d'un acteur qu'est l'utilisateur ; qui lui-même intervient au cas d'utilisation « création des sommets » qui servent à la « création des triangles ».

A partir de ces triangles, l'utilisateur peut « optimiser la triangulation » en fonction des paramètres introduits au système et par conséquent l'optimisation du nombre des sommets qui approximent la surface.

En fonction de type de chaque sommet (concaves, convexes, ...etc.) et « la méthode des triangles voisines » l'utilisateur peut « créer des régions », tel que tout les sommets de la région sont de même type. Le rayon d'outil de chaque région est utilisé pour « tester les collisions et les interférences ». Dans le cas d'interférences, le test permet de faire une « correction de rayon d'outil ».

Le cas d'utilisation « choix stratégie d'usinage » est adéquat avec la forme de la surface.

Le cas d'utilisation « calcul des points d'intersections » permet de créer des contours dédiés au cas d'utilisation « génération du trajet d'usinage ».

L'avant dernier cas d'utilisation est « simulation de l'usinage » qui est déclenchée par l'action de l'utilisateur en utilisant la trajectoire générée.

Le dernier cas d'utilisation est « génération du programme d'usinage », pour générer un programme d'usinage G-Code qui peut être utilisé par la machine.

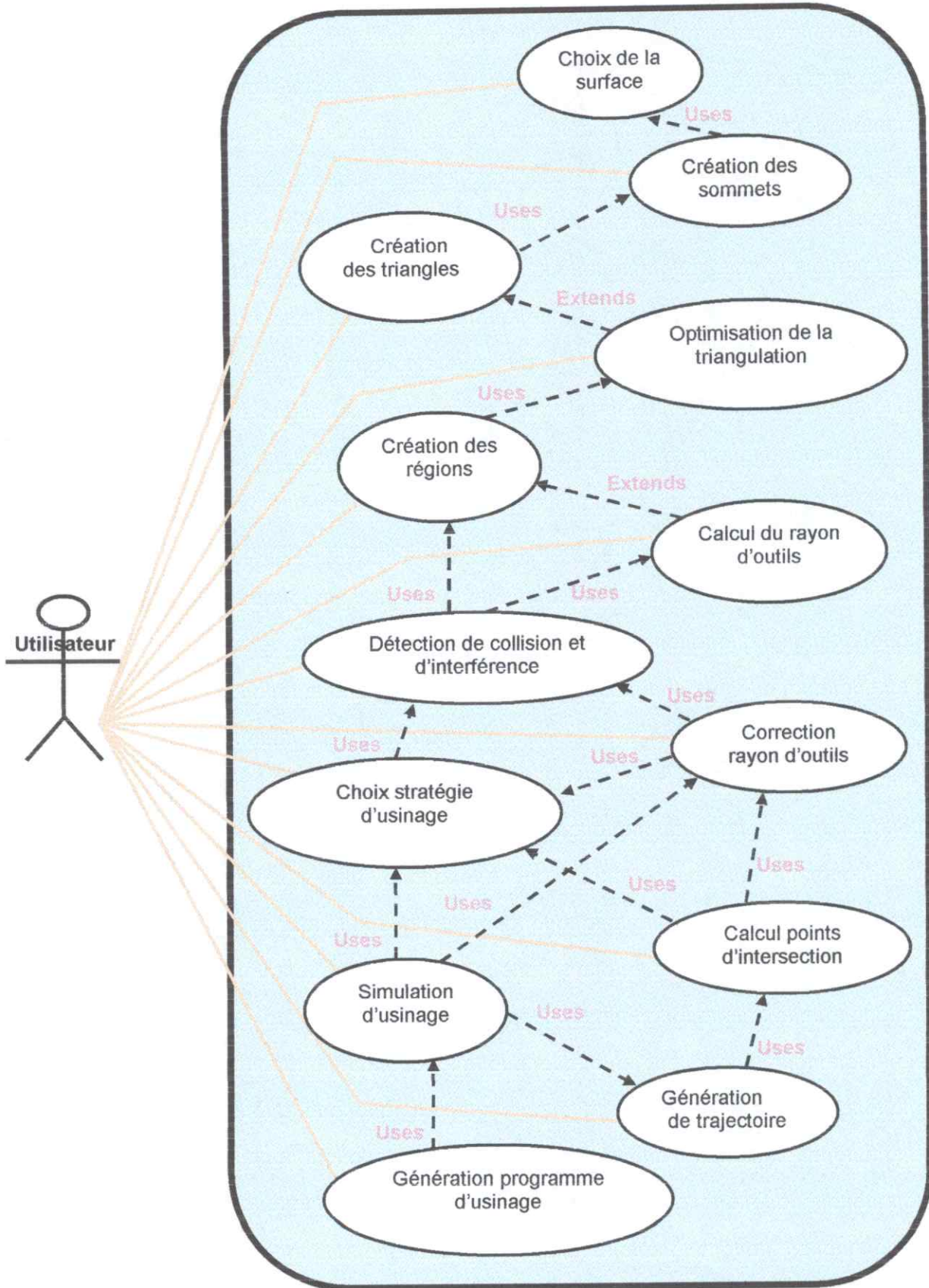


Figure 1 : Diagramme de cas d'utilisation.

IV.2- Diagramme de classe :

Pour représenter la structure de notre système, nous devons le modéliser par le diagramme de classe qui permet une description architecturale générale de notre système par la définition de tous les objets du système, leurs attributs, opérations intégrées et les relations entre ses objets avec leurs multiplicités et leurs types. Les classes sont une description abstraite des ensembles d'objets que nous avons utilisés dans notre système et les relations sont des associations entre les classes d'objets qui elles même peuvent être une classe dite classe d'association (figure 2).

Dans cette figure nous avons représenté le diagramme de classe de notre système, qui comporte toutes les classes que nous avons implémentées avec leurs relations. Et pour réduire la taille de diagramme et donner une vue bien claire, nous avons représenté les classes sans leurs attributs et leurs fonctions.

La représentation détaillée (attributs et les fonctions du classe) de chaque classe est décrite ci-après :

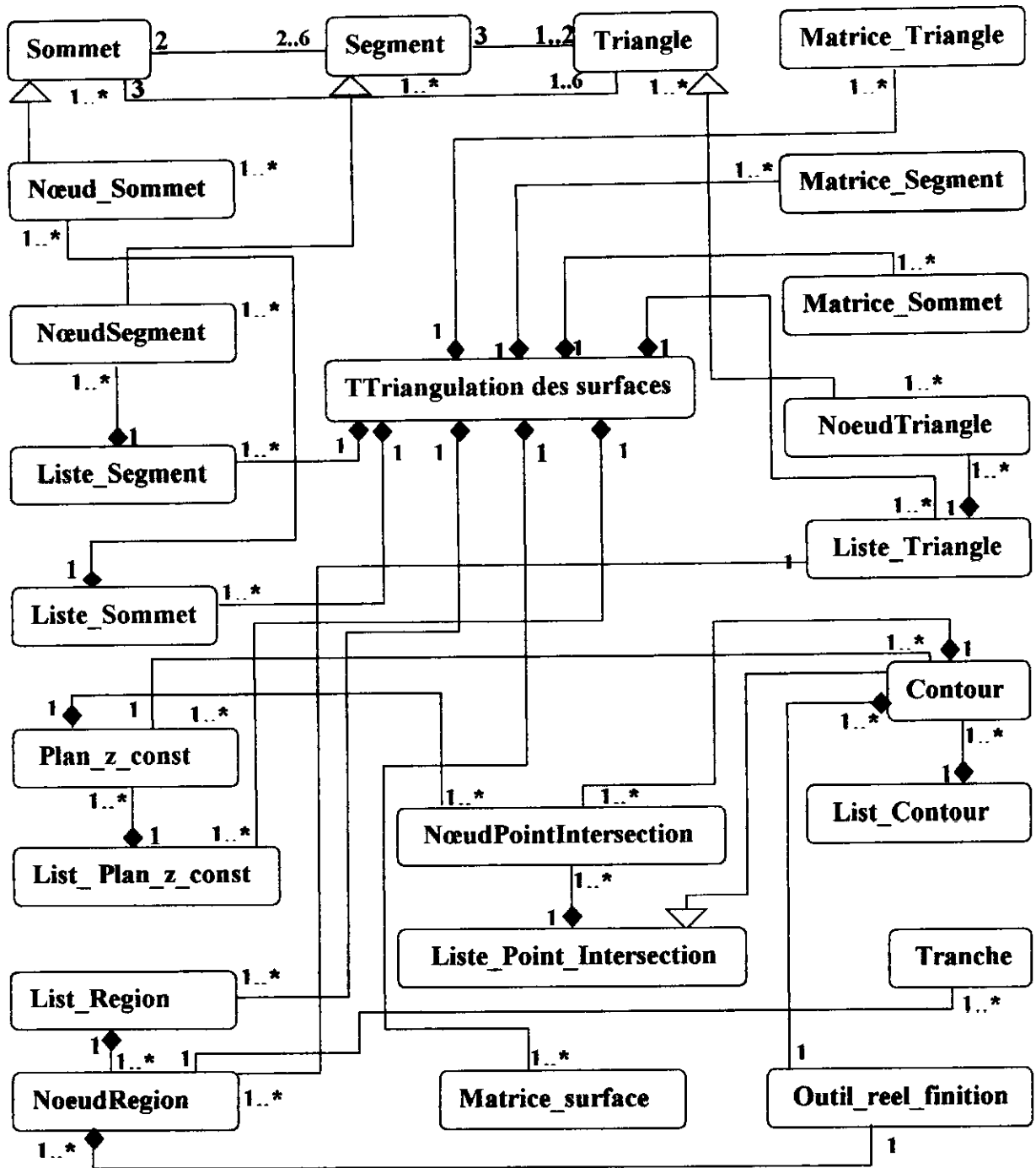


Figure 2 : Diagramme de classe.

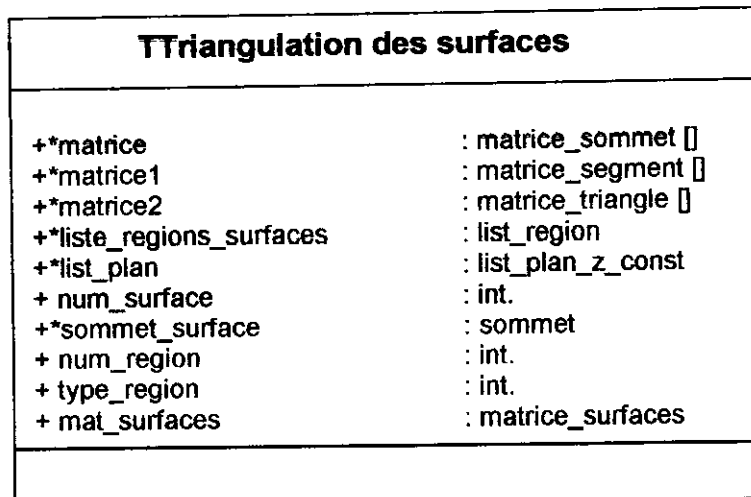


Figure 3 : classe TTriangulation des surfaces.

- **La classe de TTriangulation des surfaces :**

Cette classe comporte les relations avec toutes les autres classes et regroupe les méthodes de génération des sommets et des triangles, elle est considérée comme une classe mère qui réalise toutes les fonctions principales du système. Ces dernières sont destinées à la création des régions, le choix des rayons d'outils optimums et le choix de la stratégie d'usinage.

Elle comporte plusieurs attributs. (Voir figure 3)

Matrice [] : est un vecteur des matrices des sommets de type `matrice_sommet`.
chaque `matrice_sommet` stocke les sommets d'une seule surface.

Matrice1 [] : est un vecteur des matrices des segments de type `matrice_segment`.
chaque `matrice_segment` permet de stocker tous les segments que nous avons générés pour une surface.

Matrice2 [] : est un vecteur des matrices des triangles de type `matrice_triangles`,
permet de stocker tous les triangles que nous avons générés pour toutes les surfaces.

Liste_regions_surfaces : est un vecteur de type `list_region`, contient toutes les listes des régions de chaque surface.

List_plan : est un vecteur de type `list_plan_z_const`, contient toutes les listes des plans Z_Constant.

Num_surface : est variable de type `int` (entier), utilisé pour stocker le numéro de la surface actuelle.

Num_region : de type int (entier), utilisé pour stocker le numéro de la région traitée.

Type_region : de type int (entier), utilisé pour stocker le type de la région traitée.

Sommet_surface : de type sommet qui permet de stocker le sommet dont lequel n'existe pas d'interférence.

Mat_surfaces : de type matrice_surfaces, pour stocker la liste d'interférence.

- Les différentes classes principales de diagramme de classe :

Sommet	
+num_surf	:int
+u	:double
+v	:double
+x	:double
+y	:double
+z	:double
+rmin	:double
+rmax	:double
+r_theorique	:double
+tux	:double
+tuy	:double
+tuz	:double
+tvx	:double
+tvy	:double
+tvz	:double
+nx	:double
+ny	:double
+nz	:double
+ calculer_parametres_sommet	
+ calculer_pptes_geometriques	

Triangle	
++*p1	:sommet
++*p2	:sommet
++*p3	:sommet
++*v1	:triangle
++*v2	:triangle
++*v3	:triangle
+cgx	:double
+cgy	:double
+cgz	:double
+cgu	:double
+cgv	:double
+fleche	:double
+ntx	:double
+nty	:double
+ntz	:double
+type_forme_locale	:int
+calcul_erreur_fleche()	
+Calcul_erreur_centre_gravite()	

Matrice_sommet	
+num_surf	:int
+m	:int
+n	:int
+du ₀	:double
+dv ₀	:double
++*vertices	:sommet
+créer_matrice_sommet()	

Segment	
+p1	:sommet
+p2	:sommet
+long_segment	:double
+erreur_fleche_segment	:double
+z_max_segment	:double
+z_min_segment	:double
+type_segment	:int
+verif_long	:bool
+verif_fleche	:bool
+insere_liste_segment	:bool
+calcul_long_segment()	
+calcul_erreur_fleche()	

Matrice_triangle		Matrice_segment	
+num_surf	:int	+num_surf	:int
+m	:int	+m	:int
+n	:int	+n	:int
+erreur_fleche_triangle_min	:double	***segment_horiz	:segment
+erreur_fleche_triangle_max	:double	***segment_vertic	:segment
***matria	:triangle	***segment_diagon	:segment
+ calcul_matria () + diviser_centre_triangle () + calcule_type_triangle () + calcul_matria_segment ()		+longueur_max :double + longueur_min :double +fleche_max :double +fleche_min :double +créer_matrice_seg_horiz() +créer_matrice_seg_vert() +créer_matrice_seg_diagon() +calcul_max_min_chaque_segment()	

Figure 4 : classes principales de la triangulation.

Les fonctions de la classe "sommets" :

Void calculer_parametres_sommets (int num_surf, double u, double v) : permet de calculer les paramètres de chaque sommet et de les stocker dans la matrice des sommets.

Void calculer_ptes_geometriques (int num_surf, double u, double v) : permet de calculer les propriétés géométriques pour chaque sommet et de les sauvegarder.

Les fonctions de la classe "matrice_sommets" :

Void calcul_m () : cette fonction est utilisée pour trouver du_0 initiale (permettant de calculer le nombre de sommets suivants U) en fonction des paramètres introduits par l'utilisateur avec un pourcentage d'erreur de 85%.

Void calcul_n () : cette fonction est utilisée pour trouver dv_0 initiale (permettant de calculer le nombre de sommets suivants V) en fonction des paramètres introduits par l'utilisateur et avec le même pourcentage d'erreur.

Void calcul_m_n () : permet d'optimiser du_0 et dv_0 suivant la direction U et V respectivement.

Void calcul_vertices () : c'est la fonction qui permet de créer et de calculer les paramètres de tous les sommets d'une surface et de les stocker dans une matrice des sommets.

Les fonctions de la classe "matrice_segment" :

Void calcul_matseg () : c'est la fonction qui permet de générer et de stocker les segments dans les matrices segments.

Void `diviser_seg_horiz ()` : cette fonction réalise l'optimisation par le test de la longueur et l'erreur de flèche des segments horizontaux selon des critères introduits par l'utilisateur et fait la division si nécessaire.

Void `diviser_seg_vertic ()` : réalise le même travail que la fonction précédente pour les segments verticaux.

Void `diviser_seg_diagon ()` : réalise le même travail que les fonctions précédentes pour les segments verticaux.

Les fonctions de la classe "matrice_triangle" :

Void `calcul_matria ()` : cette fonction permet de générer les triangles à partir de la matrice des sommets et les mettent dans la matrice des triangles.

Void `calcul_matria_segment ()` : cette fonction permet de générer les triangles à partir du matrice des segments et les mettent dans la matrice des triangles.

Void `diviser_centre_triangle ()` : permet de diviser le triangle si le centre de gravite du triangle ne satisfait pas les critères introduits par l'utilisateur.

Void `calcule_type_triangle ()` : permet de calculer le type de chaque triangle selon le type des sommets de cette triangle.

- **Les classes de la localisation des régions :**

List_sommet	
+xmin	:double
+ymax	:double
+xmin	:double
+ymax	:double
+tete_liste_sommet	:Nœud_sommet
+queue_liste_sommet	:Nœud_sommet

List_segment	
+tete_segment	:Nœud_segment
+queue_segment	:Nœud_segment

List_triangle	
+tete_triangle	:Nœud_triangle
+queue_triangle	:Nœud_triangle
+nbr_triangle	:int
+inserer_triangle()	:&Liste_triangle

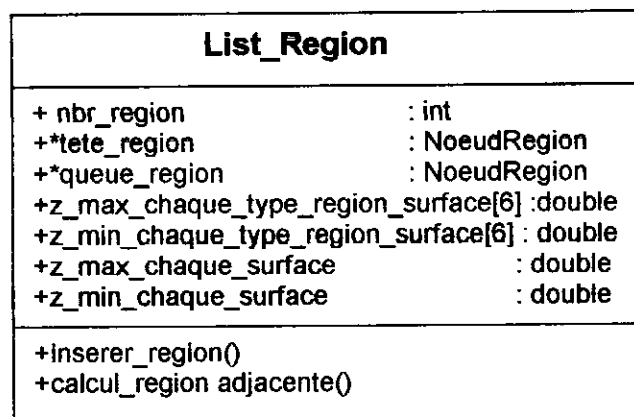
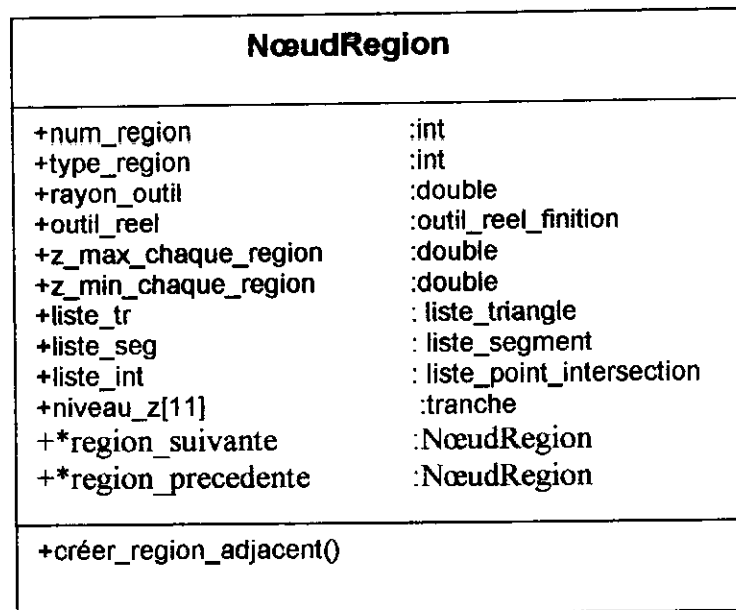


Figure 5 : différentes classes de la localisation des régions.

Les fonctions de la classe "liste_region" :

Void cree_liste_region () : permet de créer des listes qui sont stockées dans les régions selon leur type.

Void calcul_rayon_outil () : permet de calculer le rayon adéquat pour les différentes régions de la surface pour optimiser le temps d'usinage.

Void creer_tranche () : permet de diviser chaque région en plusieurs tranches pour optimiser le temps de test qui implique le temps d'usinage.

Matrice_surface	
+ nbr_ligne	: int
+ nbre_colon	: int
+ xmax	: int
+ xmin	: int
+ ymax	: int
+ ymin	: int
+ zmin	: int
+ x_outit	: int
+ y_outil	: int
+ z_outil	: int
+list_reg	: List_region
+**list_s	: List_sommet
+*list_interf	: List_sommet
+créer_matrice_surface()	
+inserer_interf()	:& List_sommet

Figure 6 : classe Matrice_surface.

- La classe "matrice_surface" :

Cette classe permet de diviser les sommets de l'ensemble des surfaces dans une matrice des zones pour localiser la vérification des interférences.

- Les classes de la génération de trajet outil :

List_contour	
+tete_contour	:contour
+queue_contour	:contour
+inserer contour() :&List contour	

List_plan_z_const	
+tete_plant	:plan_z_const
+queue_plan	:plan_z_const
+inserer_plan() :&Liste_plan_z_const	
+calcul_contour()	

List_point_intersection	
+tete_liste_intesection	:NœudPointIntersection
+queue_liste_intesection	:NœudPointIntersection
+inserer_point_intersection() :&List_point_intersection	

Tranche	
+*list_seg	:liste_segment
+z_min	:double
+z_min	double

Outil_reel_finition	
+rayon	:double
+longueur	:double
+vitesse_avance	:double
+vitesse_engagement	:double
+vitesse_degagement	:double
+outil_existe	:bool
+insérer contour() :&List contour	

Figure 7 : classes de la génération de trajet d'usinage.

IV.3. Diagramme de collaboration :

Nous avons utilisé le diagramme de collaboration (Figure 8) pour décrire les différentes opérations existantes entre les objets de notre système. Ce diagramme est une extension du diagramme d'objets et une vue dynamique du système de telle façon qu'il présente la réalisation des opérations par la description des interactions entre les objets du système par envoi et réception des messages.

Dans notre système, l'utilisateur fait appel à l'objet « **Triangulation des surface** » qui crée des sommets, des triangles et des régions, ensuite il envoie les résultats à l'objet « **Résultats** » pour l'affichage et envoie des messages aux objets « **Visualisation** », « **Localisation des régions** » et « **Les outils adéquats** » pour visualiser les résultats, localiser les régions, et donner le rayon d'outil théorique initial pour chaque région respective. Ces deux derniers objets envoient des messages à l'objet principal d'affichage « **Visualisation** » pour afficher les régions correspondantes.

Puis, l'objet principale de calcul « **Triangulation des surface** » envoie des messages à l'objet « **Division des surfaces** » qui détecte les points d'interférences, ensuite fait appel à l'objet « **Les outils adéquats** » et apporte une correction des rayon d'outils par la récupération des données à partir de l'objet précédent, ainsi que l'envoi des messages à l'objet « **stratégies d'usinages** » qui permet de choisir la stratégie d'usinage correspondante. Ce dernier envoie des messages à l'objet « **Choix des outils** » qui permet de récupérer l'outil optimal existant dans la base de donnée.

Finalement l'objet « **génération du trajet d'usinage** » reçoit les messages de l'objet « **choix des outils** » et après la récupération des données à partir de l'objet principal de calcul fait la visualisation des points d'intersection, du contour et du trajet d'usinage par l'appel de l'objet « **Visualisation** ».

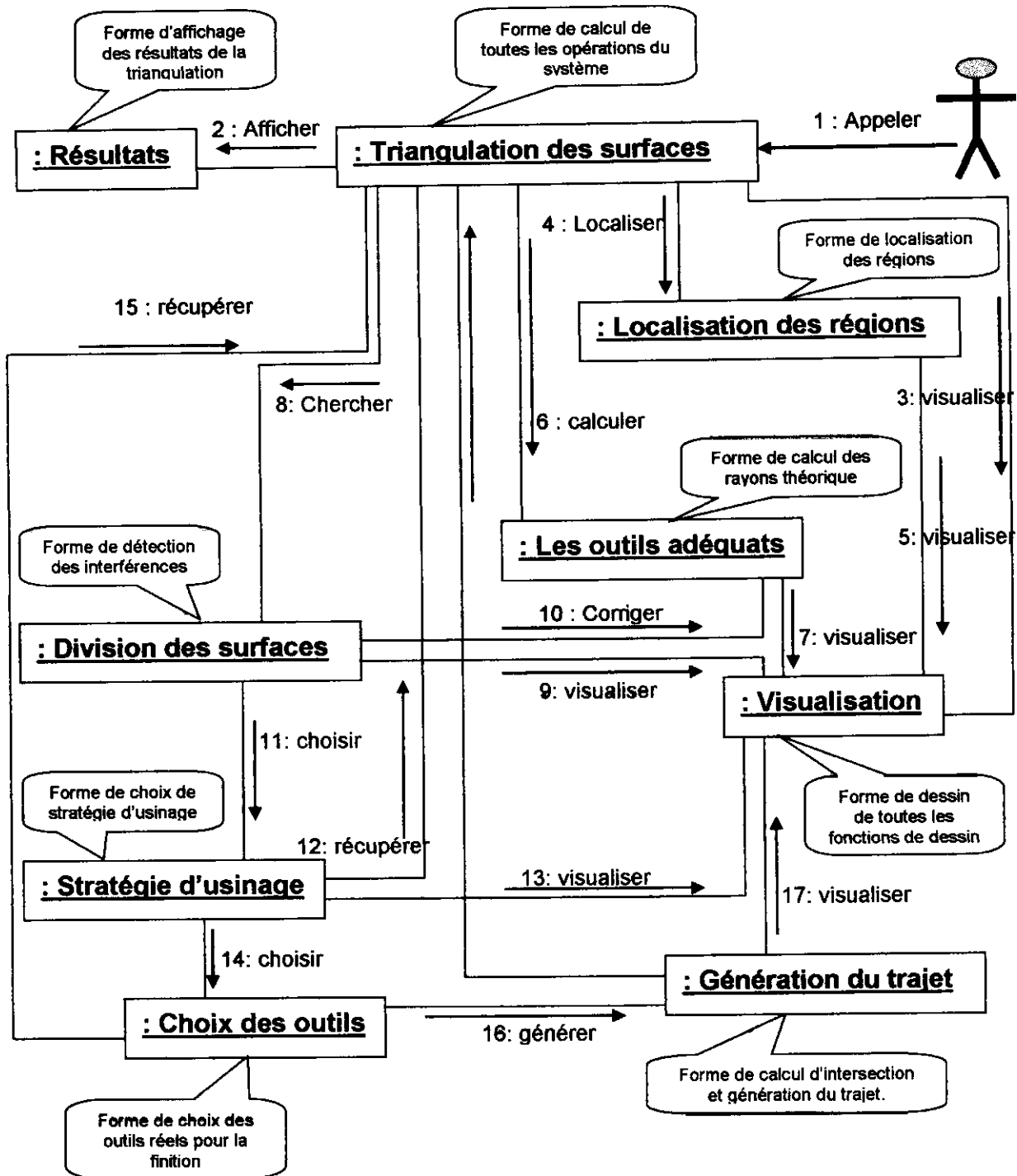


Figure 8 : diagramme de collaboration.

IV.4. Diagramme d'activité :

Nous avons utilisé le diagramme d'activité qui permet de représenter graphiquement le comportement d'une méthode ou le déroulement d'un cas d'utilisation. Il met l'accent sur les activités, leurs relations et leurs impacts sur les objets.

La figure 9 représente le diagramme d'activité de la triangulation des surfaces et localisation des régions qui est un enchaînement logique des différentes activités nécessaire pour la réalisation de la triangulation.

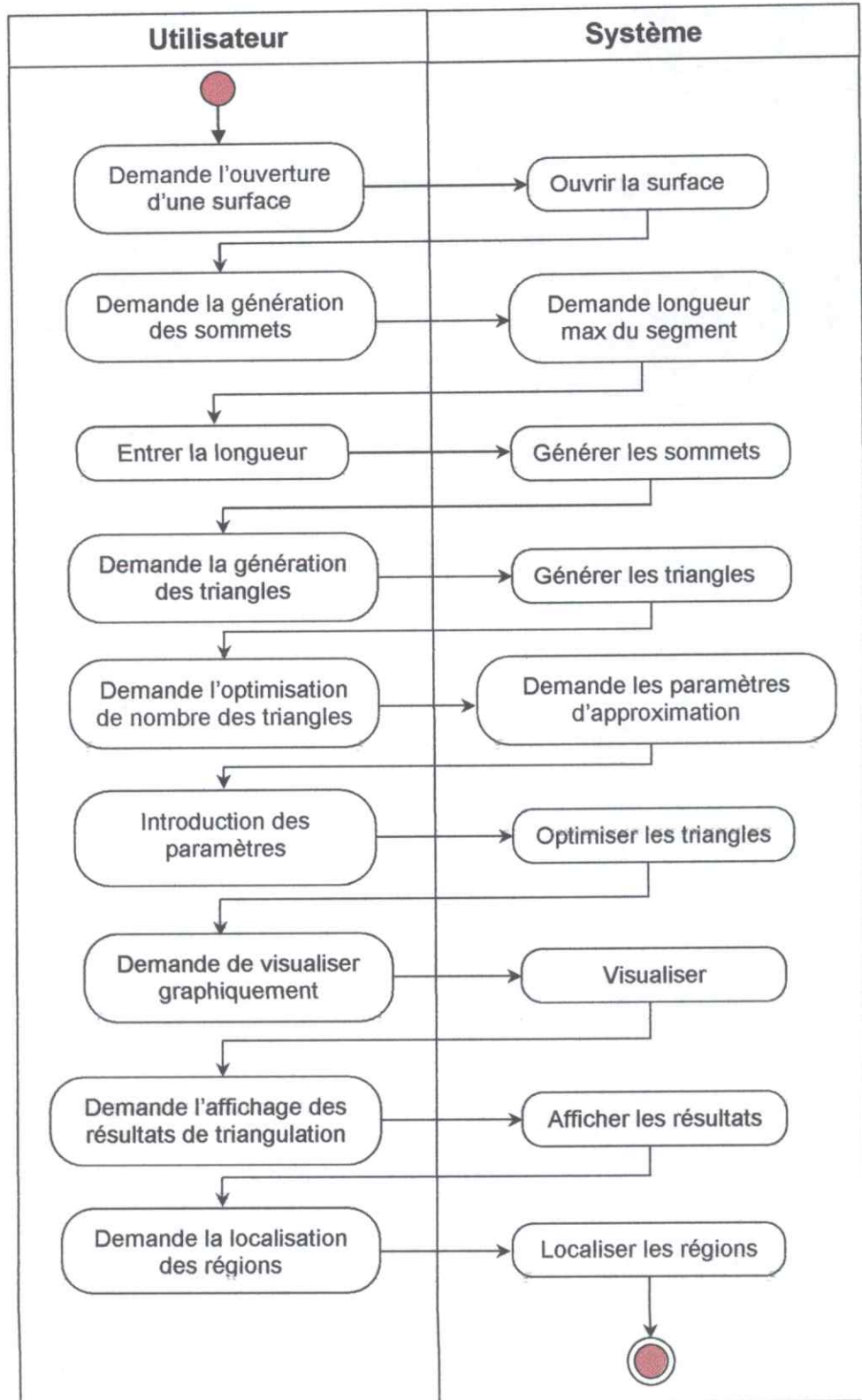


Figure 9 : Diagramme d'activité pour la triangulation des surfaces et la localisation des régions.

La figure 10 représente le diagramme d'activité de processus du choix des outils d'usinage en fonction de type de la surface.

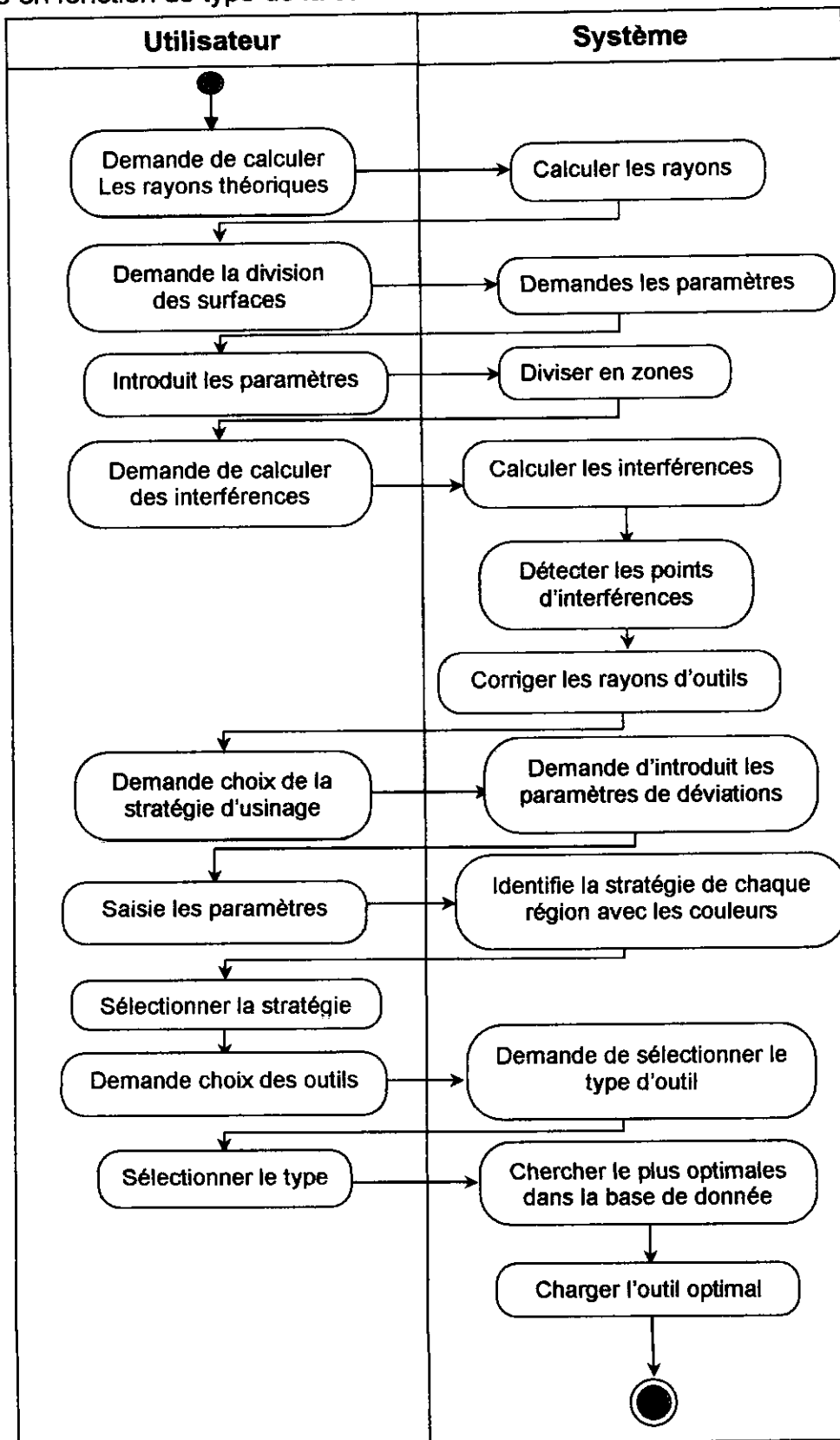


Figure 10 : Diagramme d'activité du choix des outils.

Dans la figure 11 nous avons représenté le diagramme d'activité de processus de génération du trajet d'usinage, qui doit être parcourir par l'outil pour l'usinage en finition. Ce diagramme est un enchaînement logique des différentes activités nécessaire pour la génération du trajet.

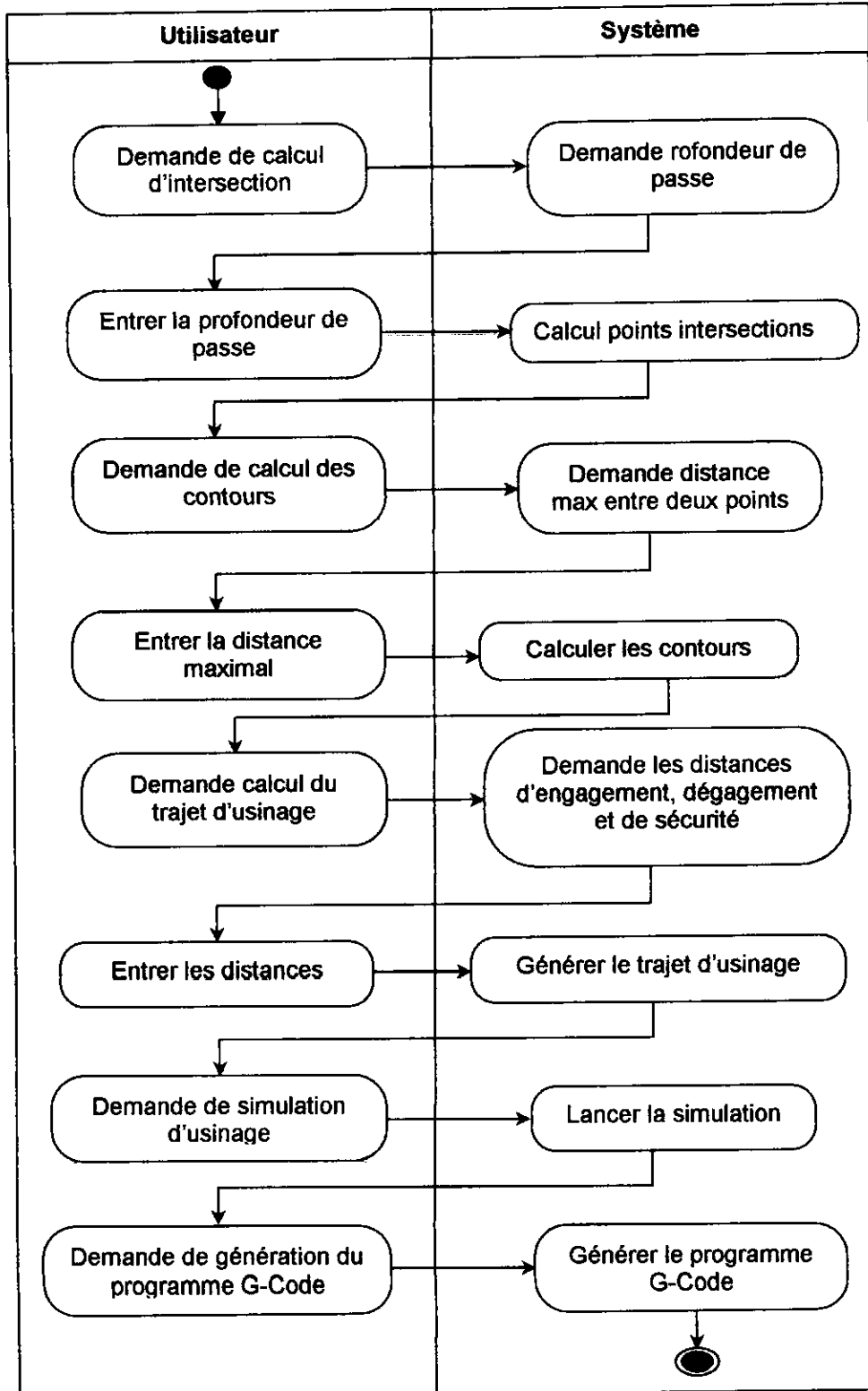


Figure 11 : Diagramme d'activité de génération du trajet.

IV.5. Diagramme de séquence :

Le diagramme de séquence permet de décrire les échanges particuliers entre un ou plusieurs acteurs et le système, et décrire la manière d'utilisation de notre système et exprime la logique des scénarios des différents cas d'utilisation de notre système. La triangulation et la division de la surface passent par les étapes suivantes :

- L'utilisateur demande l'ouverture d'une surface.
- Le système ouvre la surface.
- L'utilisateur demande la triangulation.
- Le système ouvre la fenêtre de triangulation.
- L'utilisateur introduit les paramètres et lance la 1^{ère} triangulation.
- Le système triangule la surface.
- L'utilisateur demande d'optimiser le nombre de triangles.
- Le système demande les paramètres d'approximation.
- L'utilisateur entre les paramètres d'approximation et lance la 2^{ème} triangulation.
- Le système optimiser le nombre de triangles.

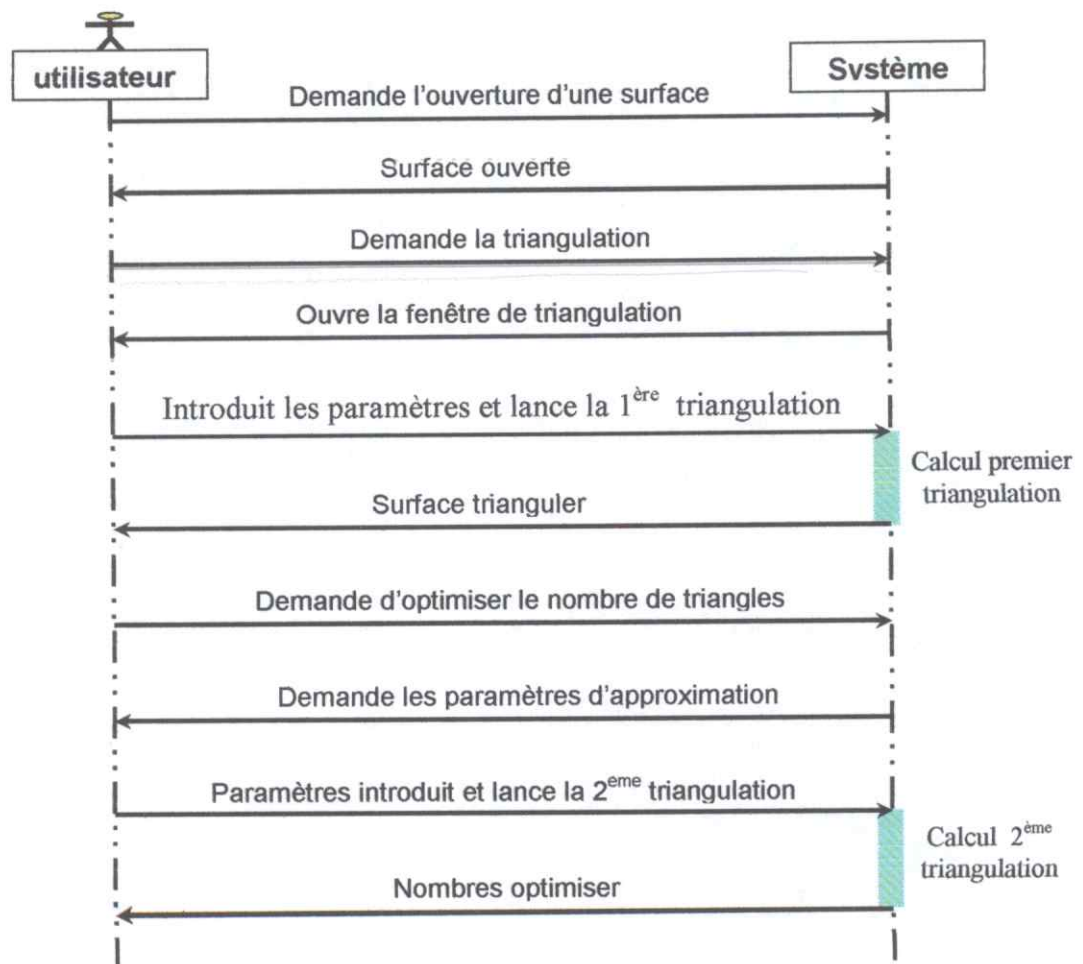


Figure 12 : diagramme de séquence de la triangulation

La figure 13 représente le diagramme de séquence de détection d'interférences qui passe par les étapes suivantes.

- L'utilisateur demande l'ouverture de la forme de détection d'interférence.
- Le système ouvre la forme.
- L'utilisateur demande la division de surface.
- Le système demande d'introduire le nombre de ligne et le nombre de colonne.
- L'utilisateur entre les données.
- Le système lance la division.
- L'utilisateur demande de calculer les interférences.
- Le système lance le calcul.

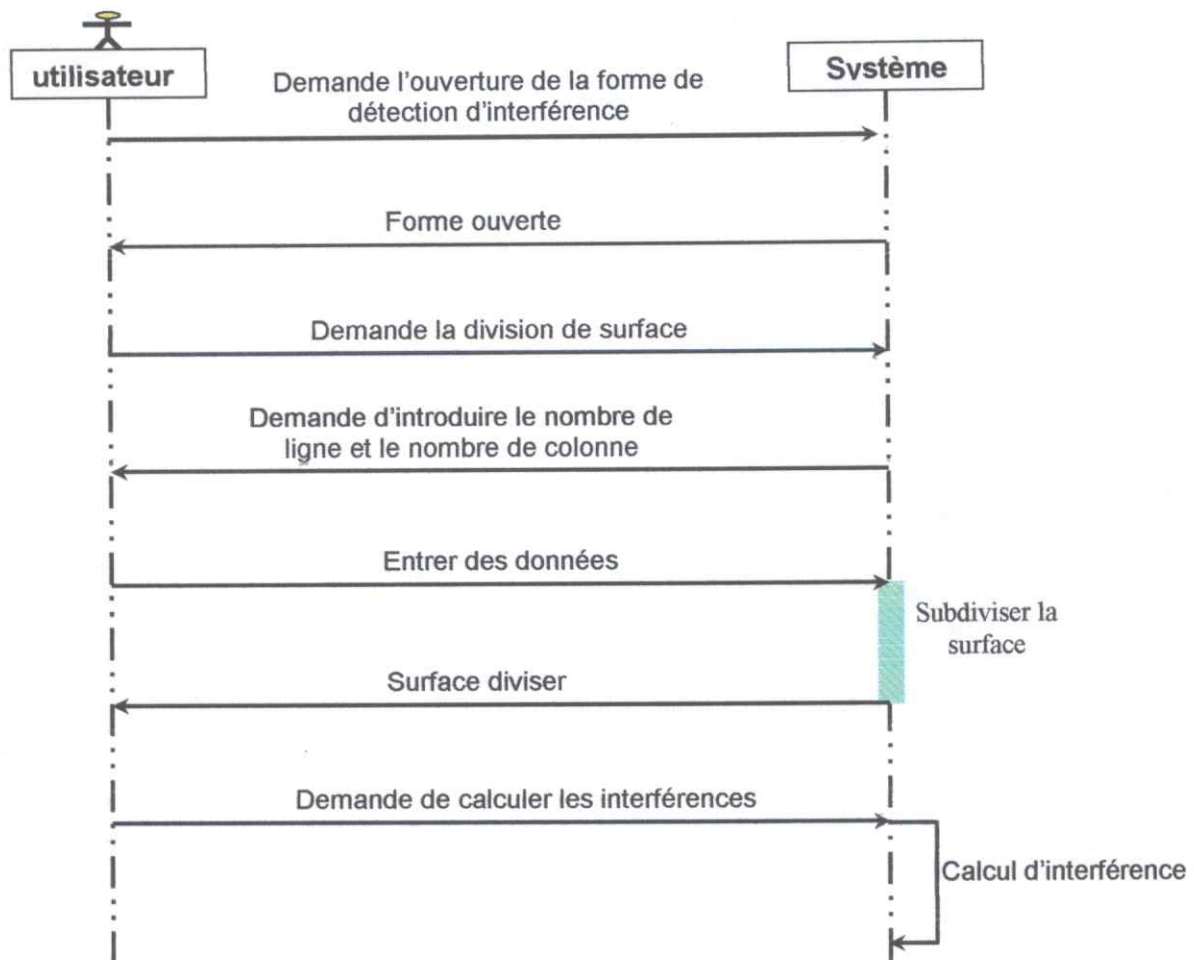


Figure 13 : diagramme de séquence de détection d'interférences.

Le choix de la stratégie d'usinage passe par les étapes suivantes :

- L'utilisateur demande l'ouverture de la forme de choix des outils.
- Le système ouvre la forme.
- Le système demande de choisir la stratégie d'usinage.
- L'utilisateur choisit la stratégie d'usinage.
- Le système demande de choisir le type d'outils.

- L'utilisateur choisit le type.
- Le système cherche le plus optimal dans la base de donnée.

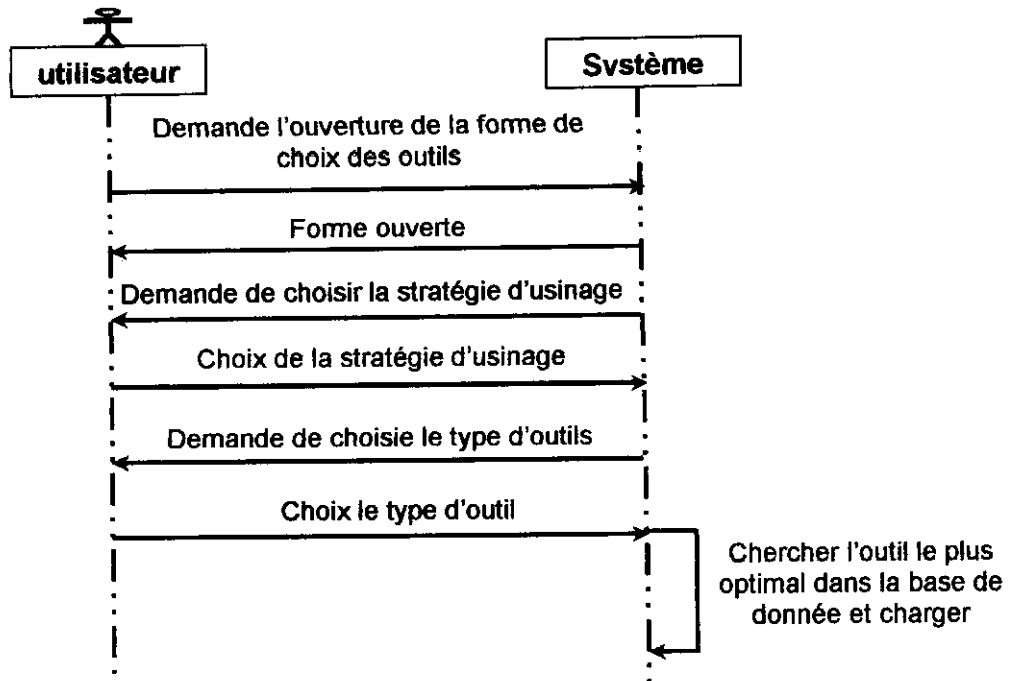


Figure 14 : diagramme de séquence de choix des outils.

La figure 15 représente le diagramme de séquence de génération du trajet d'usinage qui passe par les étapes suivantes.

- L'utilisateur demande le calcul d'intersection.
- Le système demande d'entrer la profondeur de passe.
- L'utilisateur entre la profondeur.
- Le système calcule les points d'intersections.
- L'utilisateur demande le calcul des contours.
- Le système demande la distance max entre deux points.
- L'utilisateur entre la distance.
- Le système calcule les contours.
- L'utilisateur demande le calcul du trajet d'usinage.
- Le système demande les paramètres de déplacement de l'outil.
- L'utilisateur entre les paramètres.
- Le système génère le trajet d'usinage.

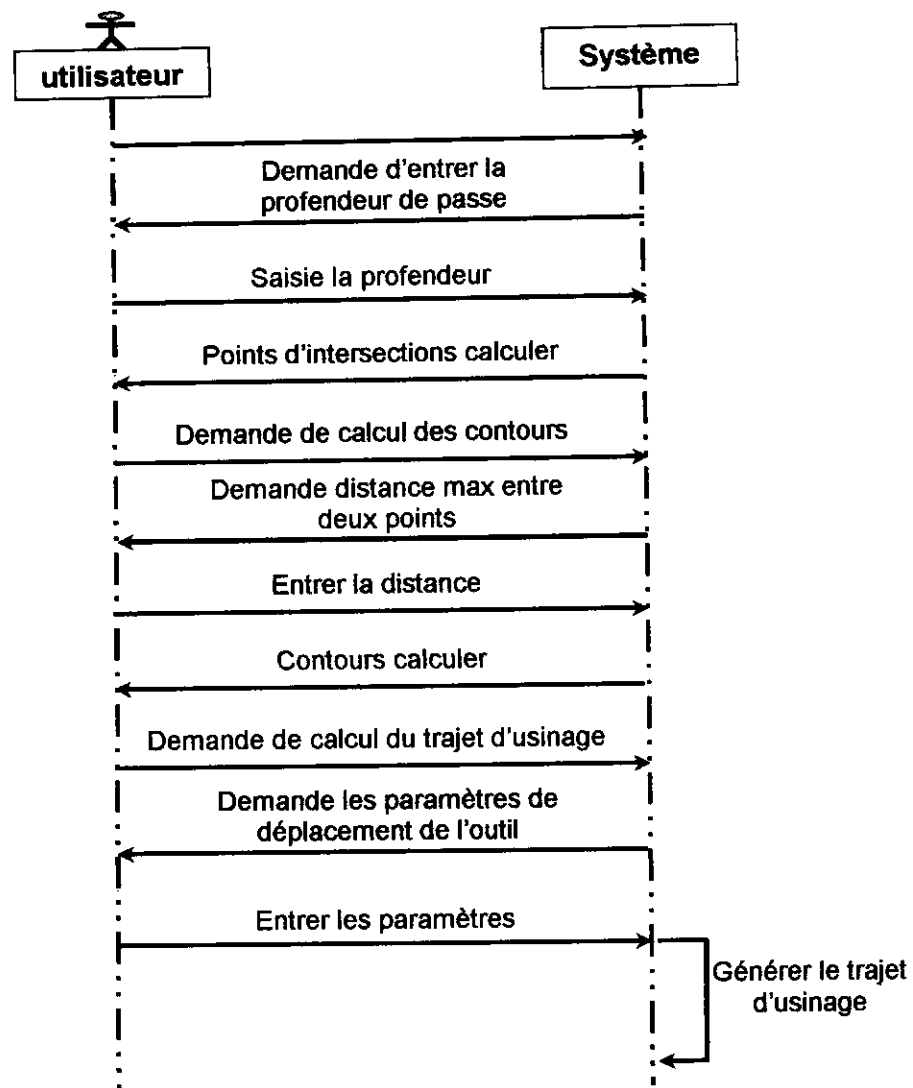


Figure 15 : diagramme de séquence de génération du trajet.

V. 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 6

Présentation de l'application

I. INTRODUCTION :

Dans le chapitre précédent nous avons réalisé la conception de notre système, on mettant en évidence les différents objectifs de l'application ainsi que les solutions proposées, ce qui nous a donné une idée claire sur notre l'application logicielle. Dans ce chapitre nous allons présenté le travail réaliser et l'environnement de travail ainsi que les principaux algorithmes adoptés.

II. PRESENTATION DU TRAVAIL :

Notre application est un logiciel de CFAO graphique et interactive. Elle comporte des fonctionnalités d'usinage et utilise des géométries conçues en 3D et visualiser grâce à la bibliothèque graphique **OpenGL** qui regroupe un ensemble de primitives spécialisées dans la création des formes géométriques. Ce travail est un ensemble de fenêtres permettent la triangulation, la détection des interférences, le choix de stratégie d'usinage, le choix des outils adéquats avec leurs mode d'usinage et la génération de trajet d'usinage.

III. ENVIRENMENT DE TRAVAIL :

III.1. Fenêtre principale :

La fenêtre principale est composée de deux sous fenêtres ; une destinée à la visualisation des différents aspects géométrique en 3D, et l'autre pour la manipulation. Elle est composée d'un ensemble de boutons et d'une barre des menus pour la création et la modification de la forme géométrique, la manipulation CAO, et le lancement des différentes fonctions de fabrication. (Figure 1)

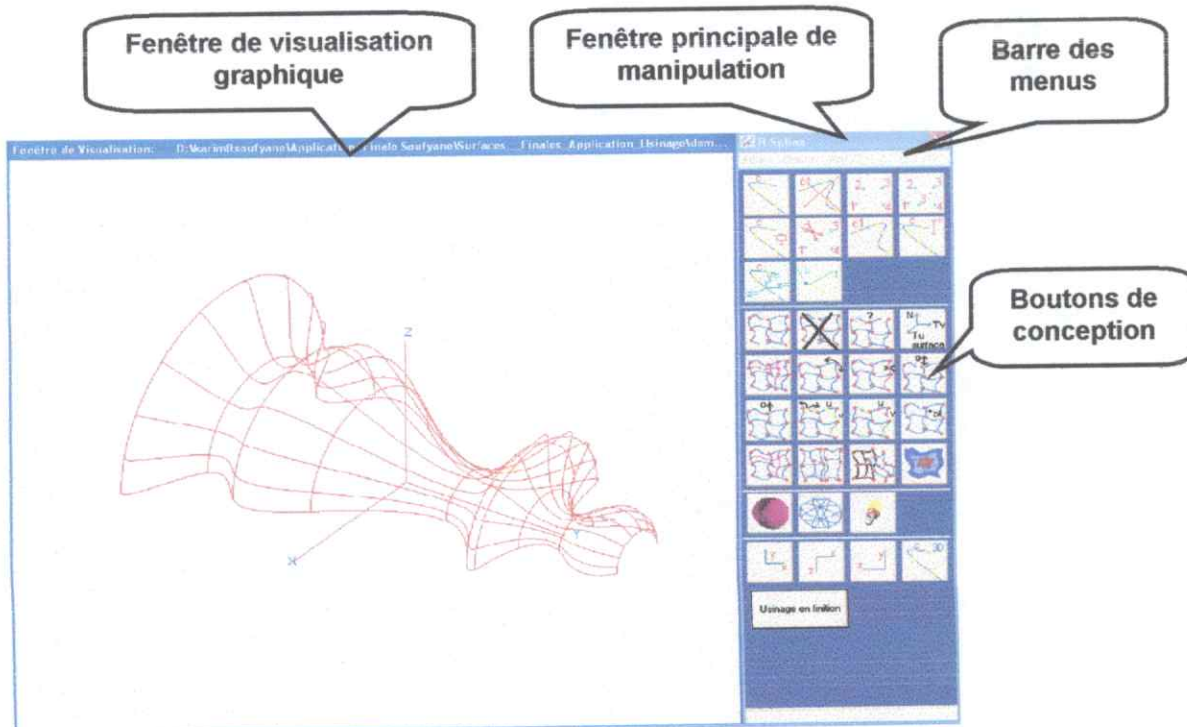


Figure 1 : Fenêtre principale.

III.2. Barre du menu principal :

La barre de menu principale 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 et les paramètres d'usinage et permet aussi l'usinage par régions.
- **Rubrique Aide** : pour nous aidez pendant l'utilisation de l'application.

III.3 Rubrique de l'usinage par régions:

Pour accéder à la partie que nous avons réalisés et intégré dans l'application (cette application est le produit de plusieurs années de développement), il faut d'abord lancer l'application et ouvrir une surface dans la rubrique **Fichier** qui contient les modèles CAO des surfaces à usiner, ensuite aller à la rubrique menu **Option** puis vers la rubrique **Usinage** ensuite vers la sous rubrique **Usinage par régions (outils et stratégies)** et cliquer pour exécuter notre application logiciel. (Figure 2)

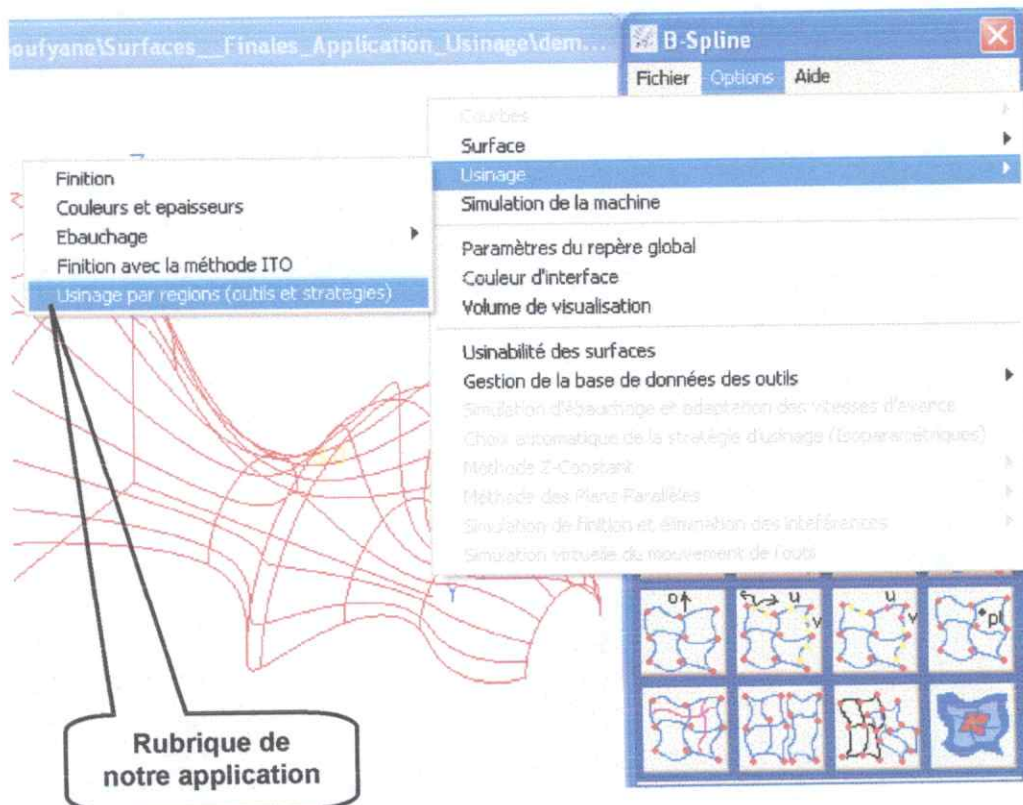


Figure 2 : Rubrique usinage par régions.

IV. PRESENTATION DU PROJET :

Dans ce qui suit, nous allons présenter toutes les fenêtres créées dans notre projet.

IV.1. Fenêtre "Triangulation des surfaces" :

Si on clique sur la rubrique "Usinage par régions (outil et stratégies)" dans la barre de menu de la fenêtre principale, la première fenêtre qui s'affiche, est la triangulation des surfaces (figure 3), qui permet de trianguler la surface en fonction des paramètres de triangulation, introduit par l'utilisateur pour donner une bonne approximation de la surface, et permet de donner le pas initial dans les deux directions U (resp. V) pour la première triangulation. De même, elle permet aussi la visualisation des sommets, de tous les segments (horizontaux, verticaux et diagonaux) et les triangles en filaire et en rendu.

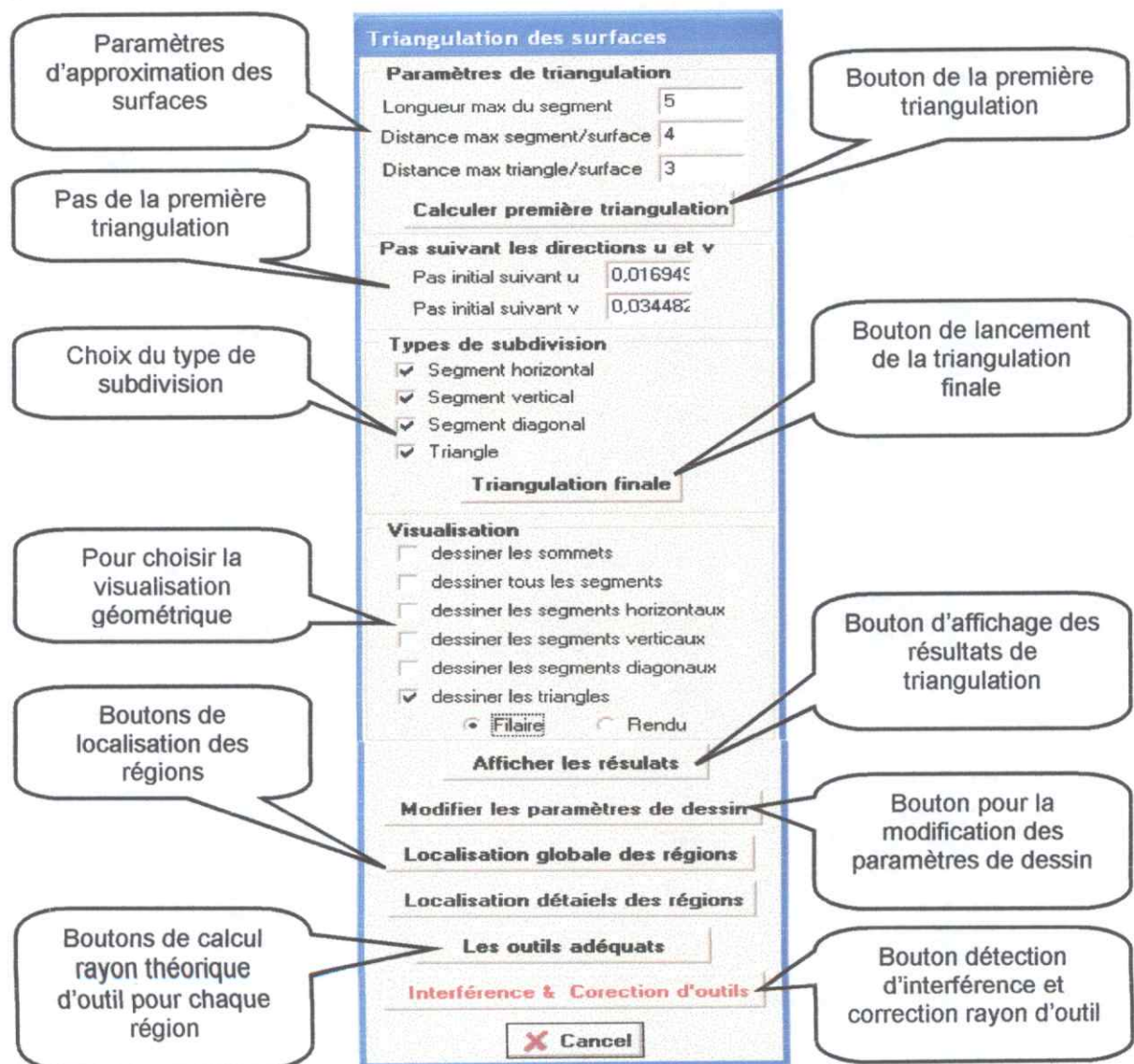


Figure 3 : fenêtre de triangulation des surfaces

Les paramètres de triangulations introduits par l'utilisateur avant le lancement de la triangulation sont :

- ❖ Longueur max du segment. (figure 4.a)
- ❖ Distance max segment surface. (figure 4.b)
- ❖ Distance max triangle/surface. (figure 4.c)

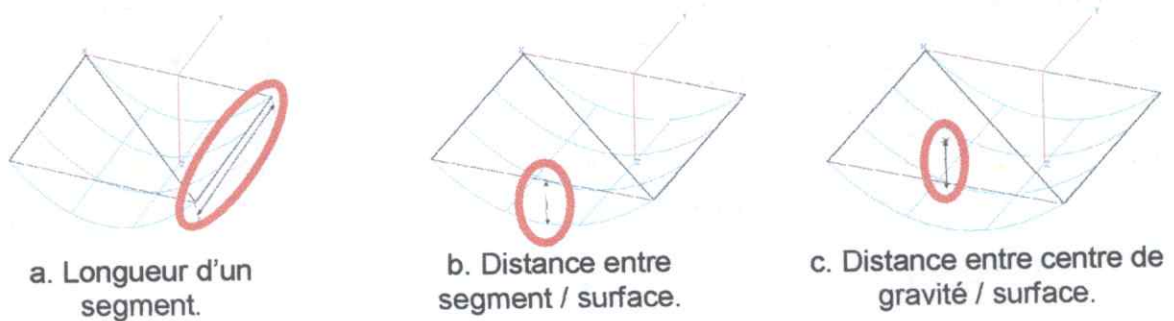


Figure 4 : Paramètres de triangulations

Après saisie de ces paramètres, l'utilisateur peut lancer le calcul de la première triangulation à partir du bouton « calculer première triangulation ». Qui permet d'afficher le pas initial dans la direction U (du_0) et dans direction V (dv_0). La première triangulation suit l'enchaînement de l'algorithme suivant:

Début :

- a. lire les paramètres de triangulations.
- b. calcul le nombre de sommets m (resp. n) dans la direction U (resp. V).
- c. calcul m et n optimal.
- d. création des sommets.
- e. création des segments.
- f. création des triangles.

Fin algorithme.

b. L'algorithme de calcul m :

Le nombre de m se calcule après le calcul de du_0 . du_0 est la longueur du plus petit segment dans l'espace paramétrique qui correspond à la longueur de segment dans l'espace cartésien vérifiant les paramètres de l'utilisateur. (Figure 5)

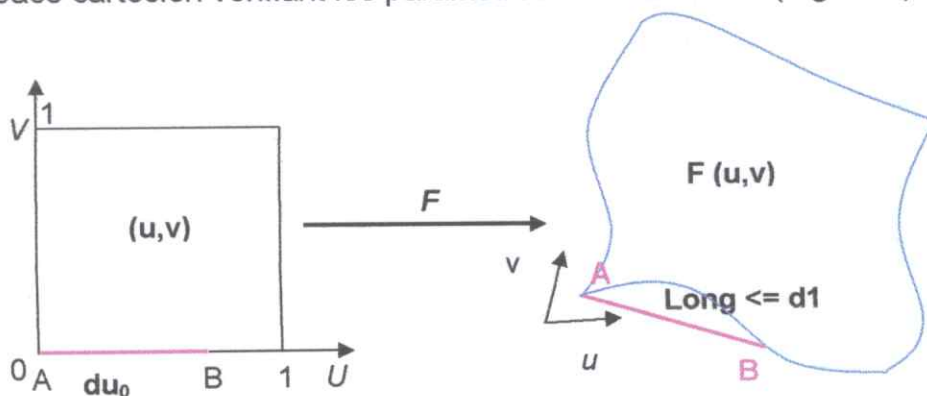


Figure 5 : schéma de calcul de m

Début algorithme :

- a. Lire le paramètre d_1 .
- b. Prend un segment $[A, B]$ tel que $\text{LongMin}(A, B) = 1$.
- c. Calculer l'image de A et B dans l'espace.
- d. Calculer la longueur $\text{long}(A, B)$ dans l'espace.
- e. **Si** $\text{long}(A, B) \leq d_1$ aller à 6.

Sinon déplacer le point B dans le milieu de segment et calculer le nouvel $\text{LongMin}(A, B)$ et aller à 3.

- f. **Si** $\text{long}(A, B) / d_1 \geq 0.85$ garder le minimum des $\text{LongMin}(A, B)$ et aller à 7.

Sinon le point B prend les coordonnées $(\text{LongMin}(A, B) + \text{LongMin}(A, B) / 2, 0)$; calculer le nouvel $\text{LongMin}(A, B)$ et aller à 3.

- g. **Si** les coordonnées de B est $(1, 0)$ alors $m = \lfloor 1 / \text{LongMin}(A, B) + 1 \rfloor$ aller à 8.

Sinon prendre le segment $[A(B, 0), B(1, 0)]$, calculer le nouvel $\text{LongMin}(A, B)$ et aller à 3.

- h. Return m et $du_0 = \text{LongMin}(A, B)$.

Fin algorithme.

Le calcul de n suit le même algorithme que celui de m mais dans la direction V.

c. L'algorithme de calcul m et n optimal:

Après le calcul de du_0 et dv_0 initial on passe à l'optimisation, on appliquant le test de la diagonale :

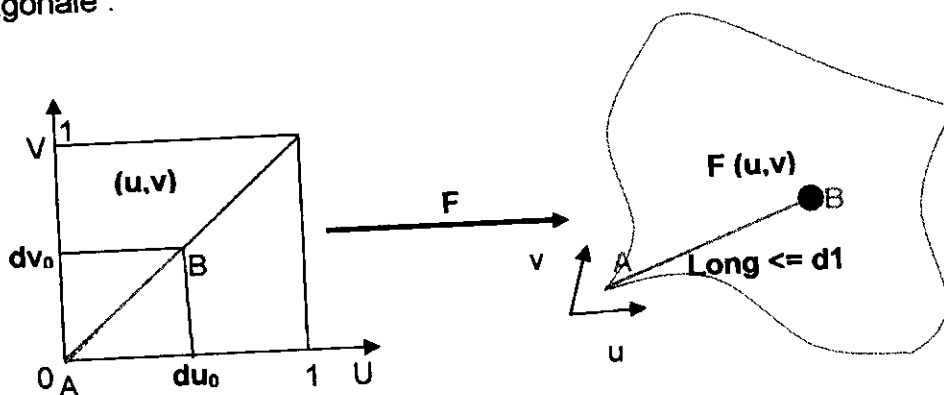


Figure 6 : Schéma d'optimisation de m et n.

Ce schéma est expliqué par l'algorithme suivant :

Début algorithme :

1. Lire le paramètre de longueur d_1 et récupérer du_0 et dv_0 initiale.
2. Trouver l'image de $A(x=0, y=0)$ et $B(z=du_0, w=dv_0)$ dans l'espace.
3. Calculer la longueur $Long(A, B)$ dans l'espace.
4. **Si** $Long(A, B) \leq d_1$ garder du_0 et dv_0 initiale et aller à 8.

Sinon aller à 5.

5. Calculer $z = (z-x)/2$ et $w = (w-y)/2$.
Calculer l'image de $B(z, w)$.
Calculer la longueur $Long(A, B)$.

6. **Si** $Long(A, B) \leq 0.85*d_1$

Alors

- $x=z, y=w, z=du_0, w=dv_0$.
- calculer l'image de $A(x, y)$ et $B(z, w)$.
- calculer $Long(A, B)$.

- **Si** $Long(A, B) \leq 0.85*d_1$ alors $du_0=z-x$ et $dv_0=w-y$ et aller à 8.

Sinon aller à 6.

Sinon aller à 7.

7. **Si** $Long(A, B) \leq d_1$ alors $du_0=z-x$ et $dv_0=w-y$ et aller à 8.

Sinon aller à 5.

8. Return $m = \lfloor 1/du_0 + 1 \rfloor, n = \lfloor 1/dv_0 + 1 \rfloor, du_0$ et dv_0 .

Fin algorithme.**d. Algorithme de création des sommets :**

La création de la matrice des sommets se déroule comme suite :

Début algorithme :

Récupérer le m (nombre de ligne) et n (nombre de colonne).

Pour i varie de 0 à m

Pour j varie de 0 à n

$U_0 = i/m$.

$V_0 = j/n$.

Créer le sommet de coordonnées U_0 et V_0

Fin Pour.

Fin Pour.

Fin algorithme.

Cet algorithme est illustré dans la figure 7.

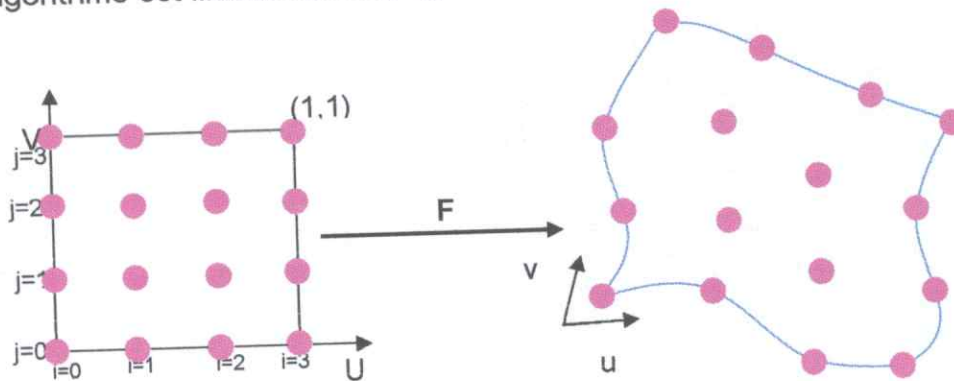


Figure 7 : Schéma de création des sommets

e. Algorithme de création des segments :

Après la création de la matrice des sommets on passe à l'algorithme de création de la matrice des segments.

Début algorithme :

Récupérer le n (nombre de ligne) et m (nombre de colonne).

1. Création de la matrice des segments horizontaux

Pour i varie de 0 à n

Pour j varie de 0 à $m-1$

Segmenthoriz $[i, j] = \text{MatriceSommet}[i, j]$.

Segmenthoriz $[i, j] = \text{MatriceSommet}[i, j+1]$.

Fin Pour.

Fin Pour.

2. Création de la matrice des segments verticaux

Pour i varie de 0 à $n-1$

Pour j varie de 0 à m

Segmentvertic $[i, j] = \text{MatriceSommet}[i, j]$.

Segmentvertic $[i, j] = \text{MatriceSommet}[i+1, j]$.

Fin Pour.

Fin Pour.

3. Création de la matrice des segments diagonaux

Pour i varie de 0 à $n-1$

Pour j varie de 0 à $m-1$

Segmentdiagon $[i, j] = \text{MatriceSommet}[i, j]$.

Segmentdiagon $[i, j] = \text{MatriceSommet}[i+1, j+1]$.

Fin Pour.

Fin Pour.

Fin algorithme.

La figure 8 représente le déroulement de cet algorithme

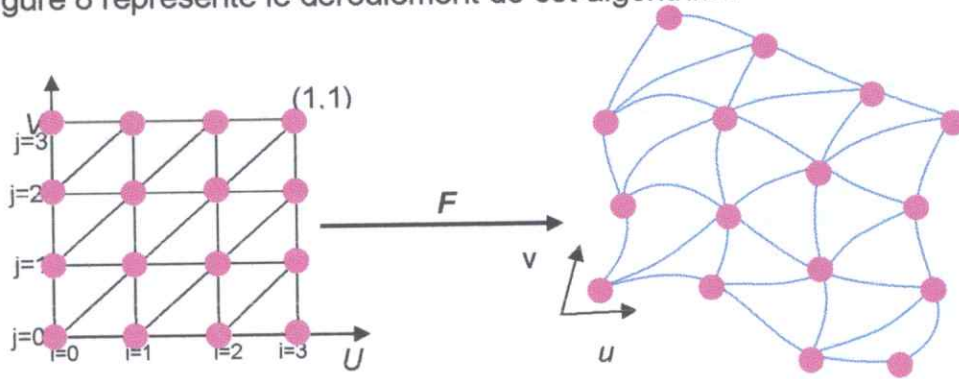


Figure 8 : Schéma de création des segments

f. Algorithme de création des triangles:

L'algorithme suivant illustre en détaille la démarche de création de la matrice des triangles à partir de la matrice des sommets :

Début algorithme :

Récupérer le n (nombre de ligne) et m (nombre de colonne).

Pour i varie de 0 à n

Pour j varie de 0 à 2*m avec un pas de 2

Pour la première triangle

MatriceTriangle [i, j]=MatriceSommet [i, j/2].

MatriceTriangle [i, j]=MatriceSommet [i, j/2+1].

MatriceTriangle [i, j]=MatriceSommet [i+1, j/2+1].

Pour la deuxième triangle

MatriceTriangle [i, j+1]=MatriceSommet [i, j/2].

MatriceTriangle [i, j+1]=MatriceSommet [i+1, j/2].

MatriceTriangle [i, j+1]=MatriceSommet [i+1, j/2+1].

Fin Pour.

Fin Pour.

Fin algorithme.

La figure 9 illustre le principe de calcul de cet algorithme.

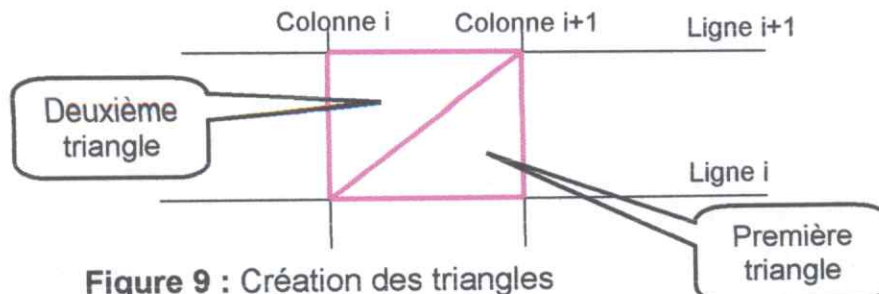


Figure 9 : Création des triangles

Ensuite l'utilisateur lance la triangulation finale, à cet effet, il doit cocher le type de subdivision puis cliquer sur le bouton « Triangulation finale », la subdivision se fait avec la triangulation uniforme (ou semi adaptative). cette subdivision se fait par l'algorithme suivant (algorithme de triangulation uniforme):

Début algorithme :

- Parcourir la matrice des triangles et vérifier s'ils respectent les paramètres introduits par l'utilisateur.
- S'il existe un triangle qui ne vérifie pas l'une des conditions suivantes :

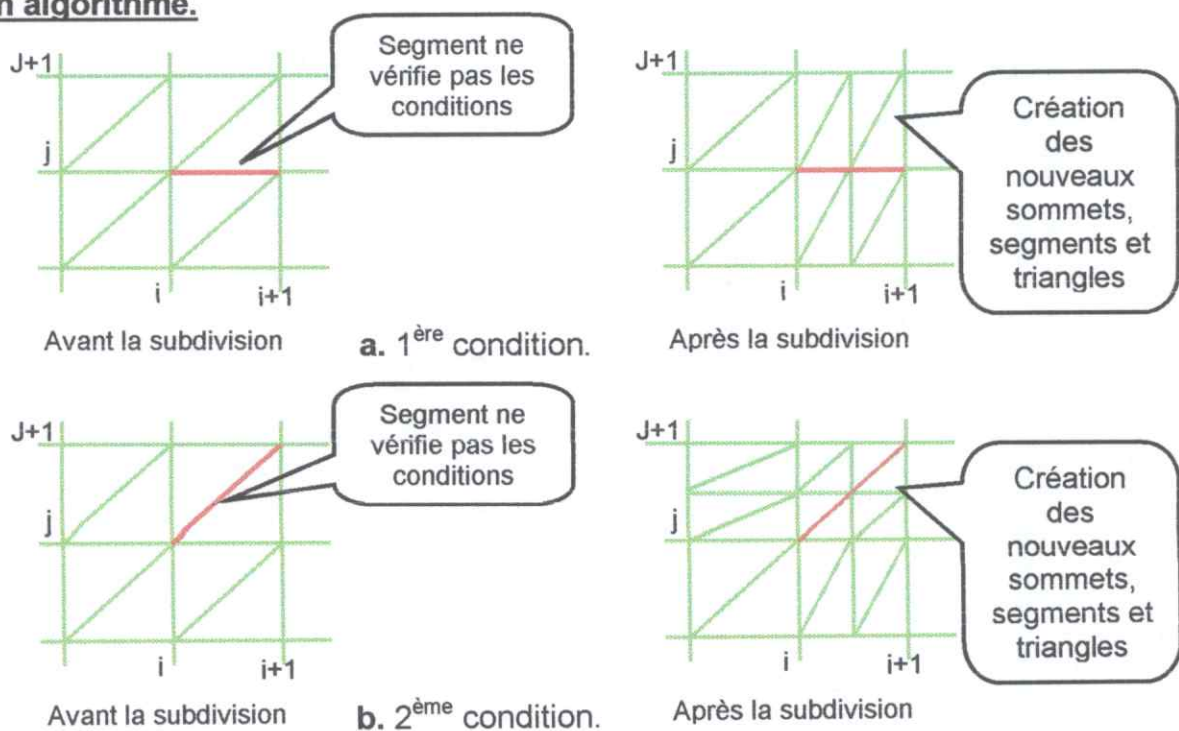
1^{ère} condition : Si le segment horizontal $[(i, j), (i, j+1)]$ (resp. le segment vertical $[(i, j), (i+1, j)]$) ne vérifie pas une des conditions de longueur ou de flèche, tous les segments horizontaux de coordonnées $[(i, j), (i, j+1)]$ quelque soit i (resp. les segments verticaux $[(i, j), (i+1, j)]$ quelque soit j) ainsi que les segments diagonaux de cette colonne (resp. de cette ligne) sont subdivisés en deux segments. (Figure 10.a)

2^{ème} condition : Si le segment diagonal $[(i, j), (i+1, j+1)]$ ne vérifie pas une des conditions de longueur ou de flèche, les segments horizontaux, verticaux et diagonaux de la ligne $[(i, j), (i, j+1)]$ et la colonne $[(i, j), (i+1, j)]$ sont subdivisés en deux segments. (Figure 10.b)

3^{ème} condition : Si le centre de gravité du triangle ne vérifie pas une des conditions de longueur ou de flèche, les segments horizontaux, verticaux et diagonaux de la ligne et la colonne sont subdivisés. (Figure 10.c)

- Pour chaque subdivision, créer des nouvelles matrices des sommets, des segments et des triangles et refaire la vérification avec optimisation jusqu'à ce que toutes les conditions soient vérifiées.

Fin algorithme.



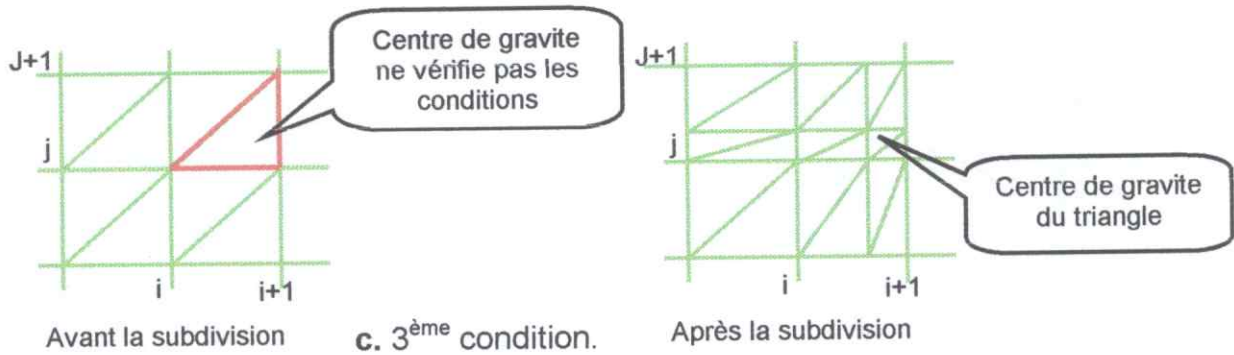


Figure 10 : subdivision avec la triangulation uniforme

Une fois que l'utilisateur clique sur le bouton "calculer la première triangulation", il y a une activation du bouton "Modifier les paramètres de dessin" qui permet de modifier l'épaisseur des objets géométriques (figure 11), et offre la possibilité à l'utilisateur pour afficher et visualiser les dessins de subdivision initial, ainsi que l'activation du choix de type de subdivision.

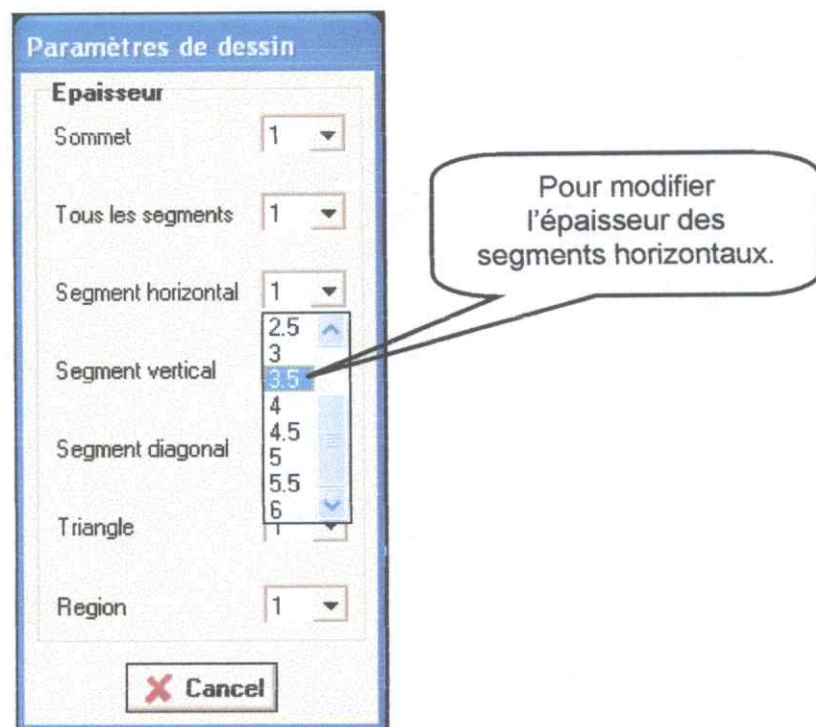


Figure 11 : fenêtre de modification les paramètres de dessin.

Ensuite, et après le clic sur le bouton "Triangulation finale" tous les boutons sont activés.

IV.2. Fenêtre "Résultats" :

Après la subdivision finale et le clic sur le bouton "Afficher les résultats" la fenêtre "Résultats" s'affiche (Figure 12). Cette fenêtre est destinée seulement à

l'affichage des résultats de subdivision initiale et finale (longueur max et min des segments avant et après la subdivision finale, nombre de sommets, nombre de segments, ... etc.).

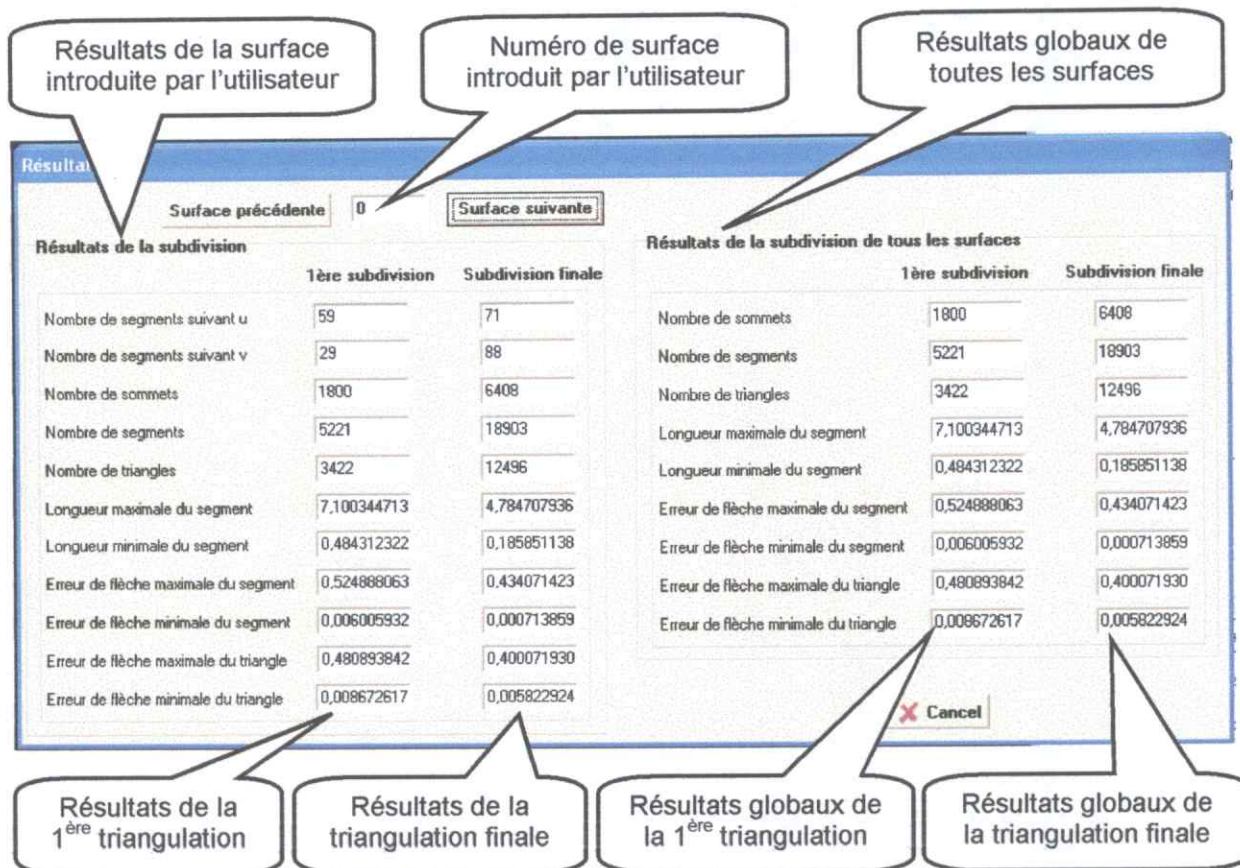


Figure 12 : fenêtre d'affichage des résultats.

IV.3. Fenêtres "Localisation globales" et "Localisation détaille des régions":

Ces deux fenêtres (Figure 13) sont destinées à la visualisation des régions de chaque surface. La génération des régions se fait en fonction des rayons de courbure de chaque sommet (Rmax et Rmin), une région est définie comme étant l'ensemble des sommets (ou des triangles) ayant le même type de forme locale (convexe, concave, selle de cheval, plane, convexe développable et concave développable).

Le regroupement des sommets en des régions permet d'accélérer les calculs de détection des interférences et par conséquent la minimisation de temps de calcul.

La fenêtre de localisation globale des régions, permet à l'utilisateur de visualiser les voisins de chaque triangle après la saisie du numéro de surface et le numéro de triangles ou basculer dans l'ordre avec les boutons suivant et précédent.

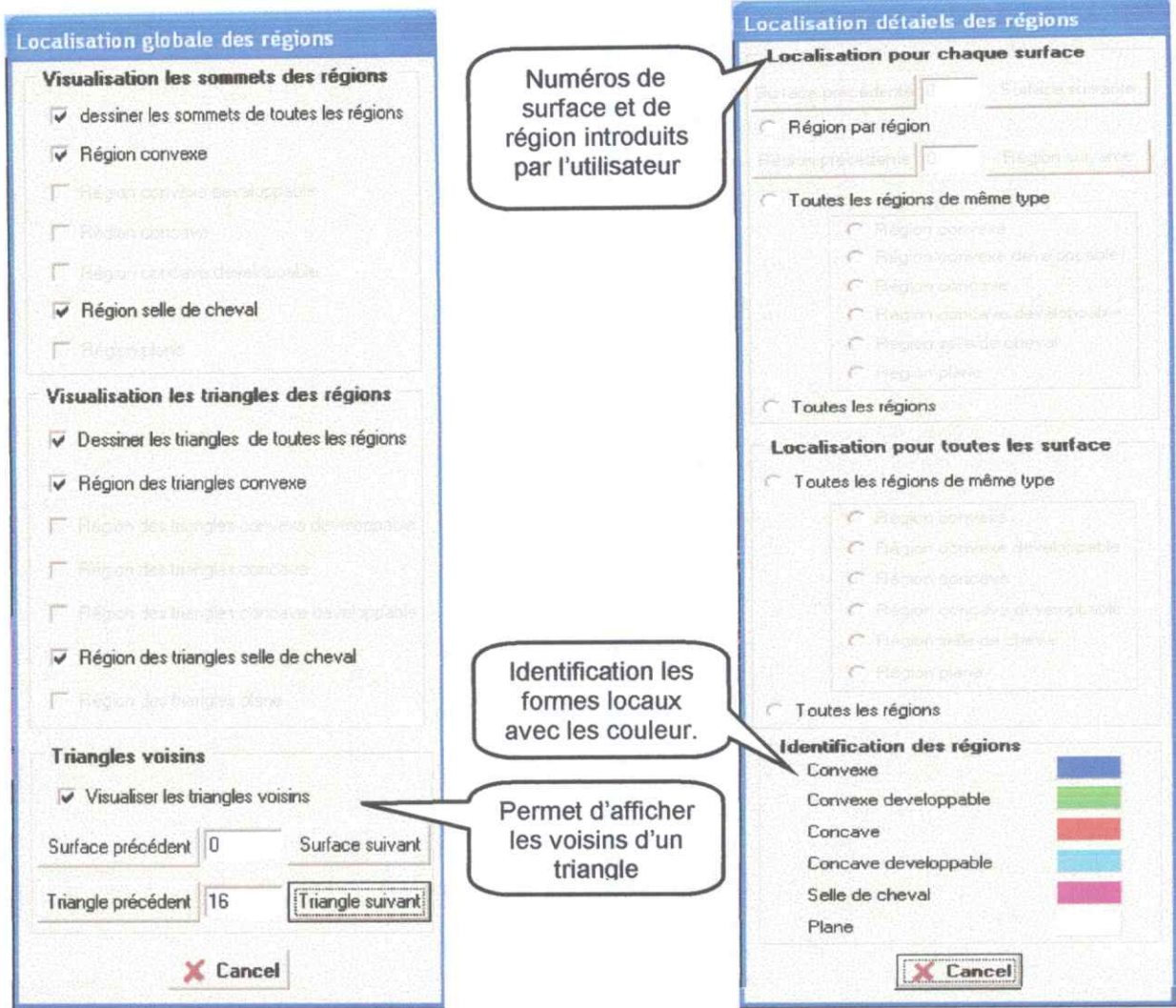


Figure 13 : Fenêtres de Localisation des régions

IV.4. Fenêtre "Les outils adéquats pour l'usinage" :

Une fois que les régions sont créées, l'utilisateur et avec simple clique sur le bouton "Les outils adéquats" affiche la fenêtre "Les outils adéquats pour l'usinage" (Figure 14), qui affiche le rayon d'outil théorique pour chaque région avec la visualisation de la région correspondante.

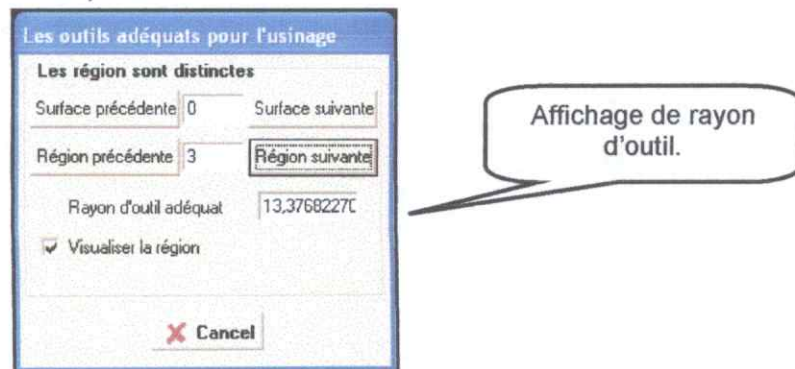


Figure 14 : Fenêtre des rayons d'outils théoriques.

IV.5. Fenêtre "Division des surfaces" :

Dans cette fenêtre, l'utilisateur fait le calcul d'interférence avec le simple clique sur le bouton "calculer les interférences" qui est activé après que l'utilisateur saisisse le nombre de ligne et colonne et appuie sur le bouton "Diviser en zone", ce dernier permet de diviser la surface en des zones distinctes pour optimiser les calculs des points d'interférences.

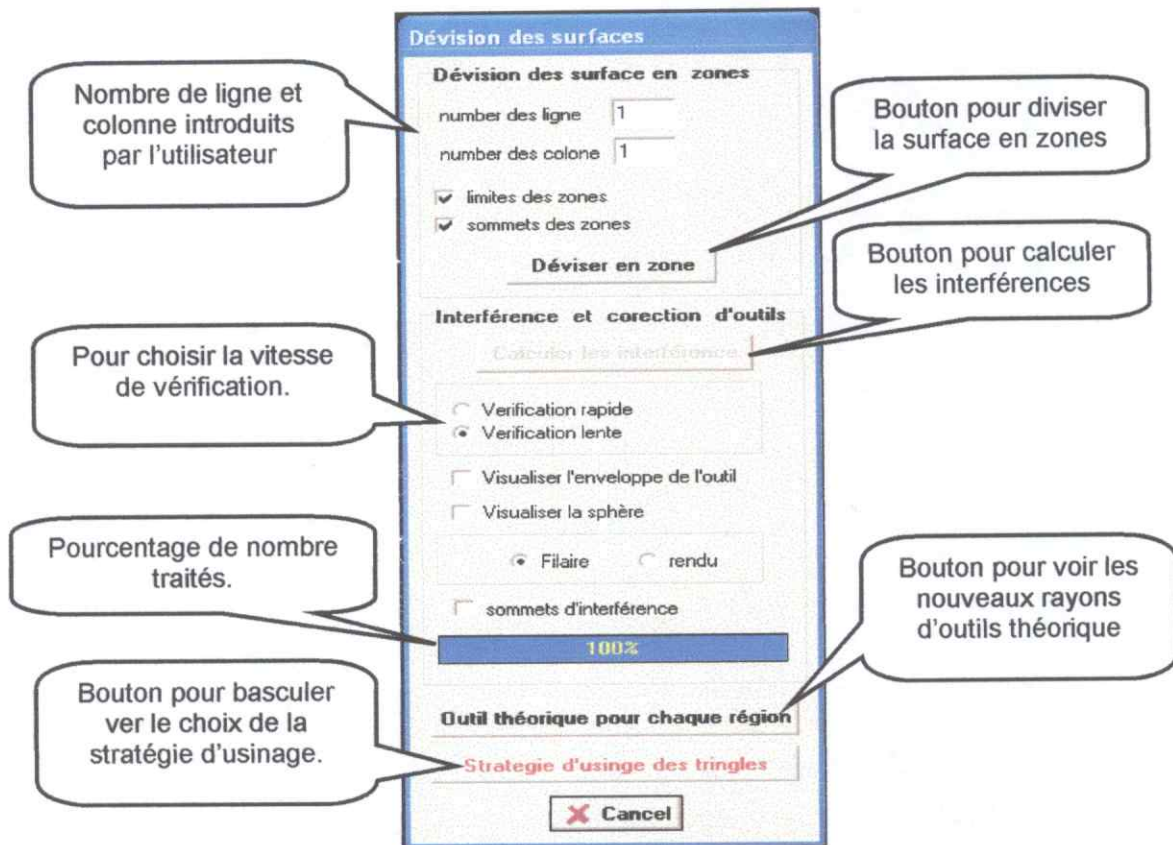


Figure 15 : Fenêtre de calcul des interférences.

Les étapes de calcul des interférences sont détaillées dans l'algorithme suivant :

Début algorithme :

1. Entrer le nombre de ligne.
2. Entrer le nombre de colonne.
3. Diviser la surface en $m \cdot n$ zones.
4. Affecter à chaque zone le x_{min} , x_{max} , y_{min} et y_{max} .
5. affecter à chaque zone la liste des sommets correspondant
6. Pour chaque surface.
 - Pour chaque région
 - Pour chaque sommet (point de contact).
 - a. calculer les cordonnées du centre d'outil.
 - b. Calculer x_{min} , x_{max} , y_{min} et y_{max} de l'enveloppe de la sphère (un cube).

- c. Vérifier s'il existe un chevauchement entre le cube et les zones de division de la surface.
- d. **Si** oui :
 - Pour chaque zone de chevauchement
 - Pour chaque sommet de la liste de cette zone.
 - Tester si le sommet est à l'intérieur de cube.
 - o **Si** oui vérifier la distance entre le centre de sphère et le sommet : **si** inférieur au rayon de sphère (rayon d'outil théorique) alors :
 - Calculer le nouveau rayon outil qui est égale à la distance entre le sommet d'interférence et l'ancien centre outil.
 - Faire la correction de rayon d'outil théorique de la région correspond à ce point.

7. s'il reste des interférences aller à 6.

Fin algorithmes.

Une fois que l'utilisateur calcule les interférences, les rayons d'outils théoriques sont corrigés et le bouton outil "théorique pour chaque région" affiche les nouveaux rayons d'outils (Figure 14) et le bouton "stratégie d'usinage des surfaces" est activé.

IV.6. Fenêtre "visualisation l'angles des triangles" :

Après le calcul des interférences, l'utilisateur clique sur le bouton "stratégie d'usinage des surfaces" qui affiche la fenêtre "visualisation l'angle des triangles"(Figure 16), cette fenêtre permet de visualiser les triangles selon leurs types (triangles horizontaux avec la couleur bleu, triangles proche de l'horizontale avec la couleur rouge, ...etc.) après la saisie des degrés de déviation (horizontale et verticale) et le clique sur le bouton OK.

Ce dernier permet d activer le bouton stratégie d'usinage pour chaque région.

Avec le degré de déviation, le système fait le regroupement des triangles selon leurs déviation en des triangles horizontaux, proche de l'horizontale, verticaux, proche de la verticale et le reste de triangles regrouper dans une liste a part. pour faciliter l'identification de la stratégie.

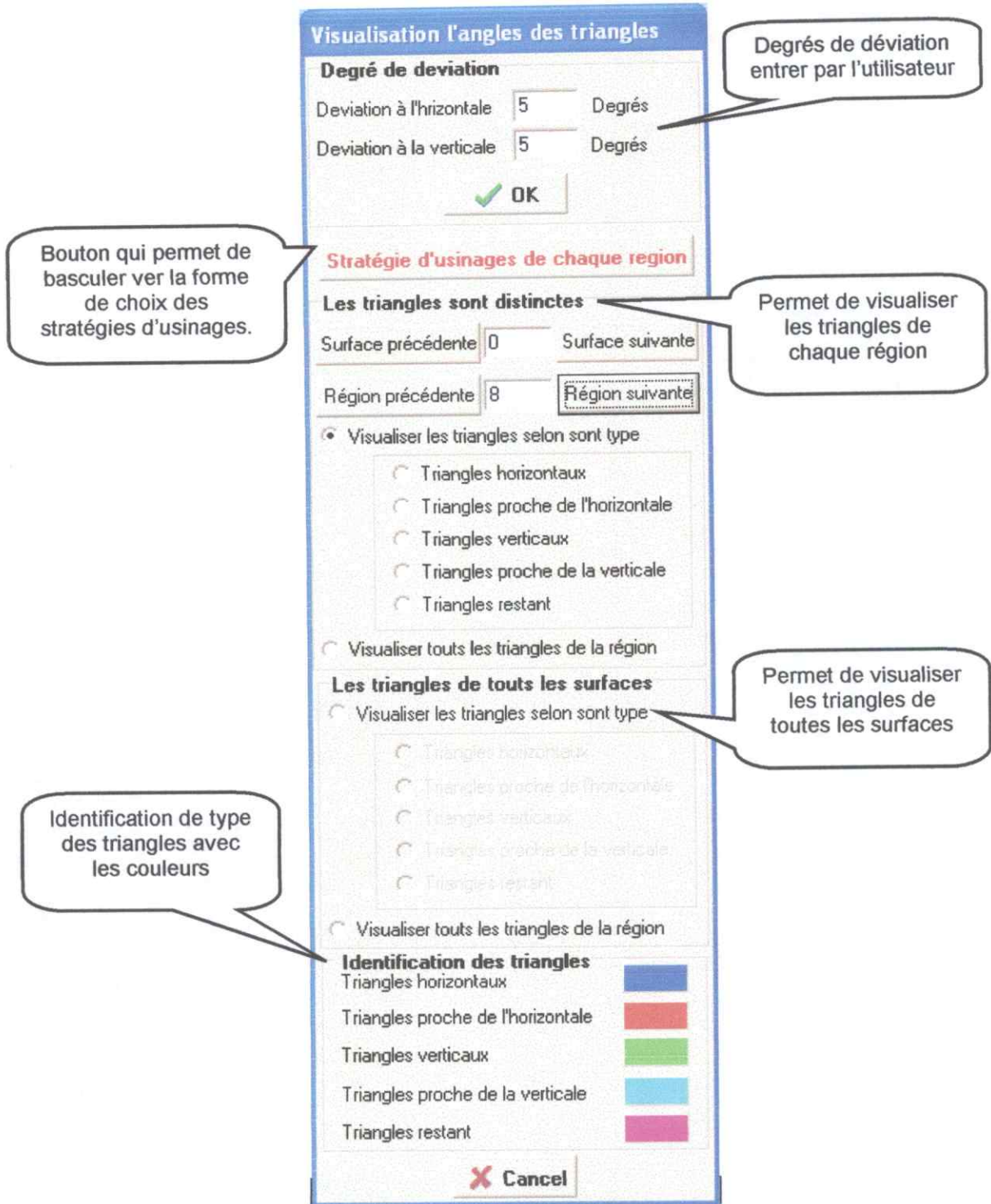


Figure 16 : fenêtre visualisation l'angle des triangles

IV.7. Fenêtre "Stratégie d'usinage de chaque région":

Avec cette fenêtre, l'utilisateur choisi la méthode d'usinage pour l'usinage en finition des surface gauche (Z constant). Et permet de basculer ver la fenêtre de choix des outils avec le bouton "choix d'outil de chaque région".

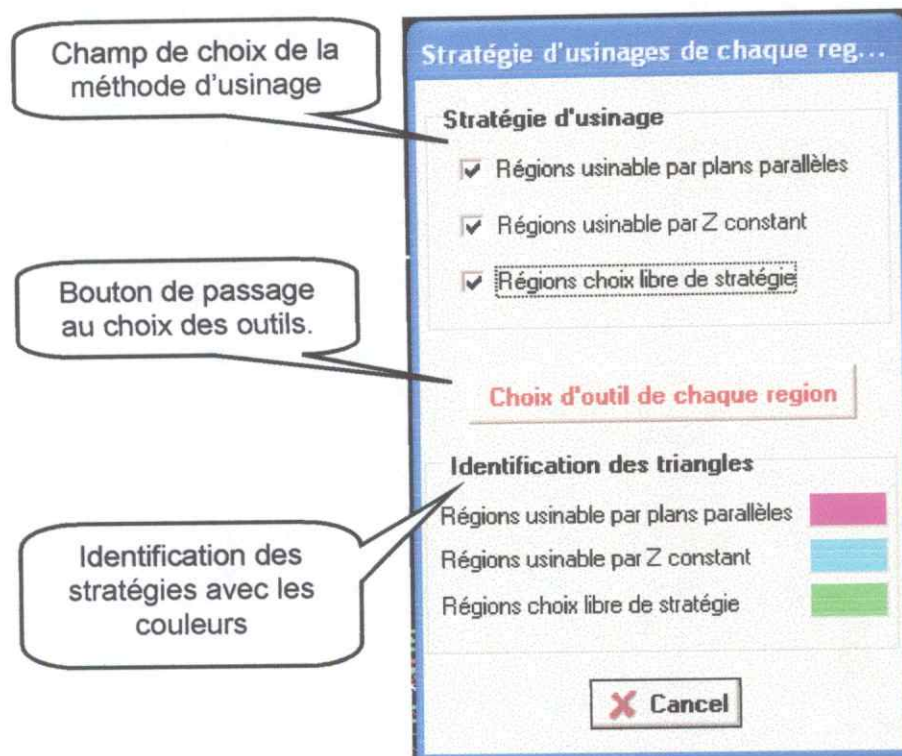
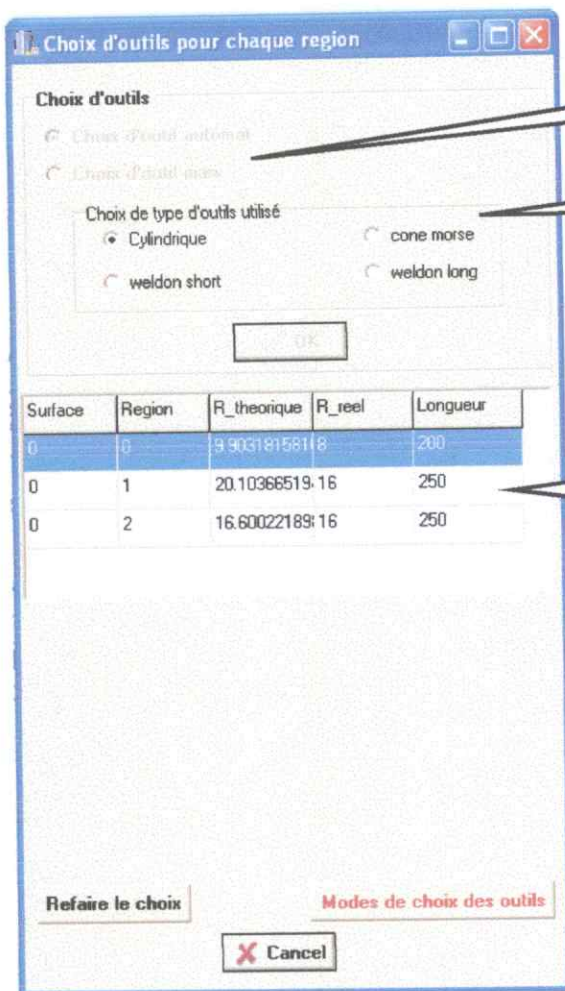


Figure 17 : Fenêtre de choix de la méthode d'usinage.

IV.8. Fenêtre "choix des outils":

Dans cette fenêtre, le choix des outils peut être fait soit automatiquement à partir de la base de données des outils en spécifiant le type d'attachement (voir figure 18.a), soit manuellement en consultant la base de données des outils ou en spécifiant d'autres valeurs du rayon et de longueur (voir figure 18.b).

Une fois que le rayon final est choisi, l'utilisateur a la possibilité soit de refaire le choix d'outil théorique ou final, ou passer à la fenêtre "modes de choix des outils".

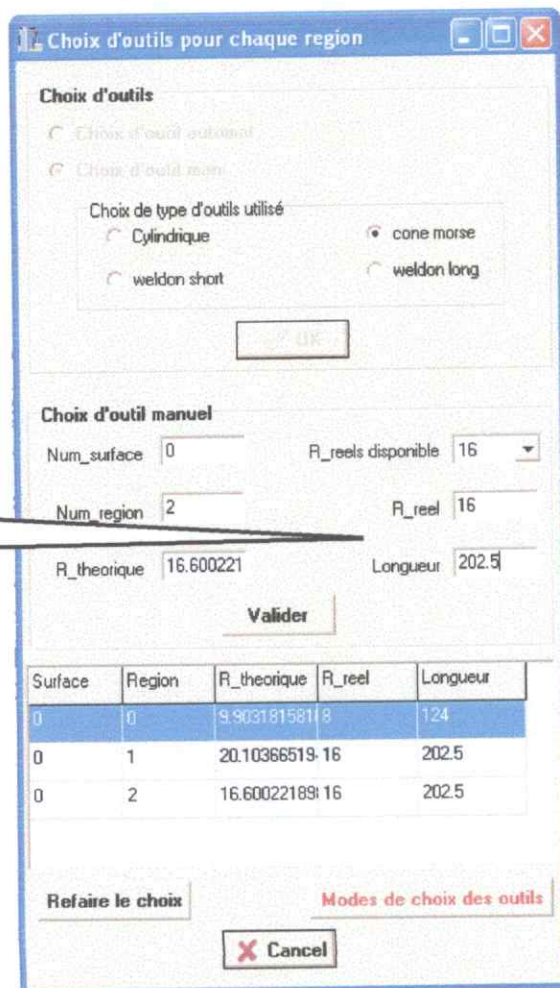


Type de choix

Type d'outil

Le rayon d'outil réel existant dans la base de donnée de chaque région

a. choix automatiques



Dans le choix manuel le R_reel et Longueur est introduits par l'utilisateur.

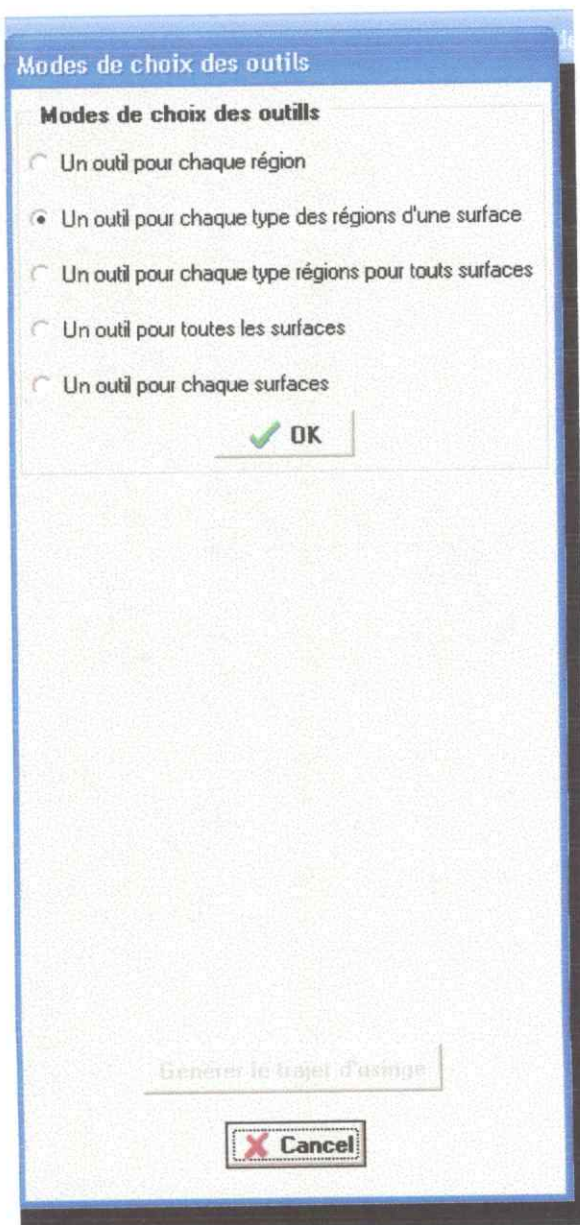
b. choix manuel.

Figure 18 : Fenêtre de choix des outils.

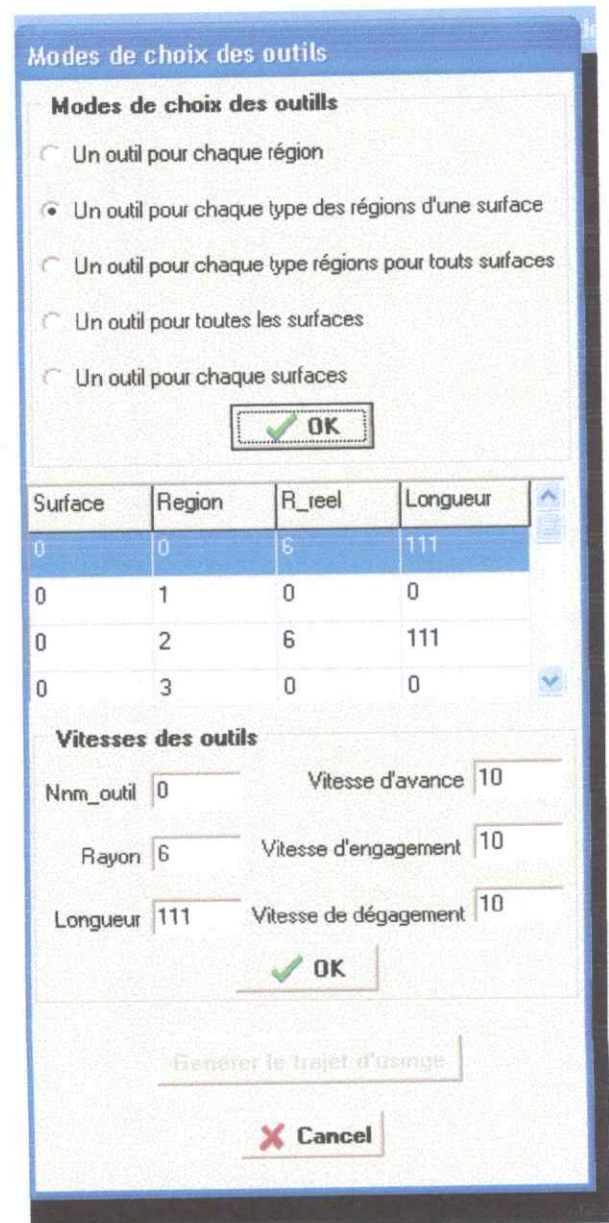
IV.9. Fenêtre modes de choix des outils:

Une fois que l'utilisateur clique sur le bouton "modes de choix des outils" sur la fenêtre précédente, la fenêtre des modes de choix s'affiche, elle offre à l'utilisateur la possibilité de choisir en première lieu le mode de choix des outils (un outil pour chaque région, un outil pour chaque type des régions d'une surface, un outil pour chaque surface, ...etc.) (Voir Figure 19.a), puis cliquer sur le bouton « OK » qui permet d'afficher les outils qui sont spécifique à ce mode et activer les champs de saisie de la vitesse d'avance, vitesse d'engagement et vitesse de dégagement qui sont des paramètres nécessaires pour l'usinage (Figure 19.b).

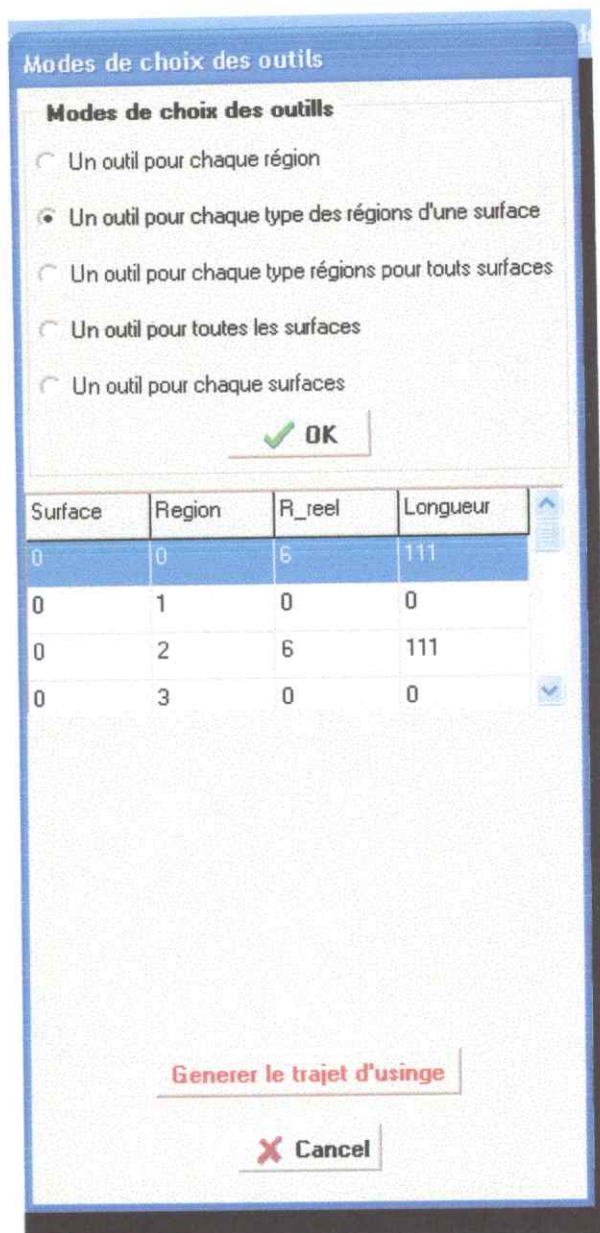
Une fois que l'utilisateur saisie les vitesses et clique sur le 2^{ème} bouton « OK » les vitesses sont affectées à la région correspondante et le champ de vitesse des outils devient invisible, ainsi que le bouton "générer le trajet d'usinage" (Figure 19.c).



a. Mode de choix des outils



b. Affichage des rayons d'outils correspondant.



c. affectation des vitesses

Figure 19 : fenêtre de mode de choix des outils.

IV.10. Fenêtre "génération du trajet d'usinage" :

Dans cette fenêtre, l'utilisateur entre la distance maximale (profondeur de passe) entre deux plans, ensuite lance le calcul des intersections qui permet de trouver les points d'intersection et les insérés dans des listes d'intersections pour chaque plan. Puis entrer la distance entre deux sommets d'intersection pour créer les contours si on clique sur le bouton "calcul des contours".

Une fois que les contours sont calculés, l'utilisateur introduit les paramètres d'usinages (distances d'engagement, de dégagement et de sécurité), ensuite lance le

calcul du trajet, et en fin, après exécution de tous les calculs, l'utilisateur lance l'opération de simulation du mouvement d'outil pour l'usinage en finition des surfaces gauches.

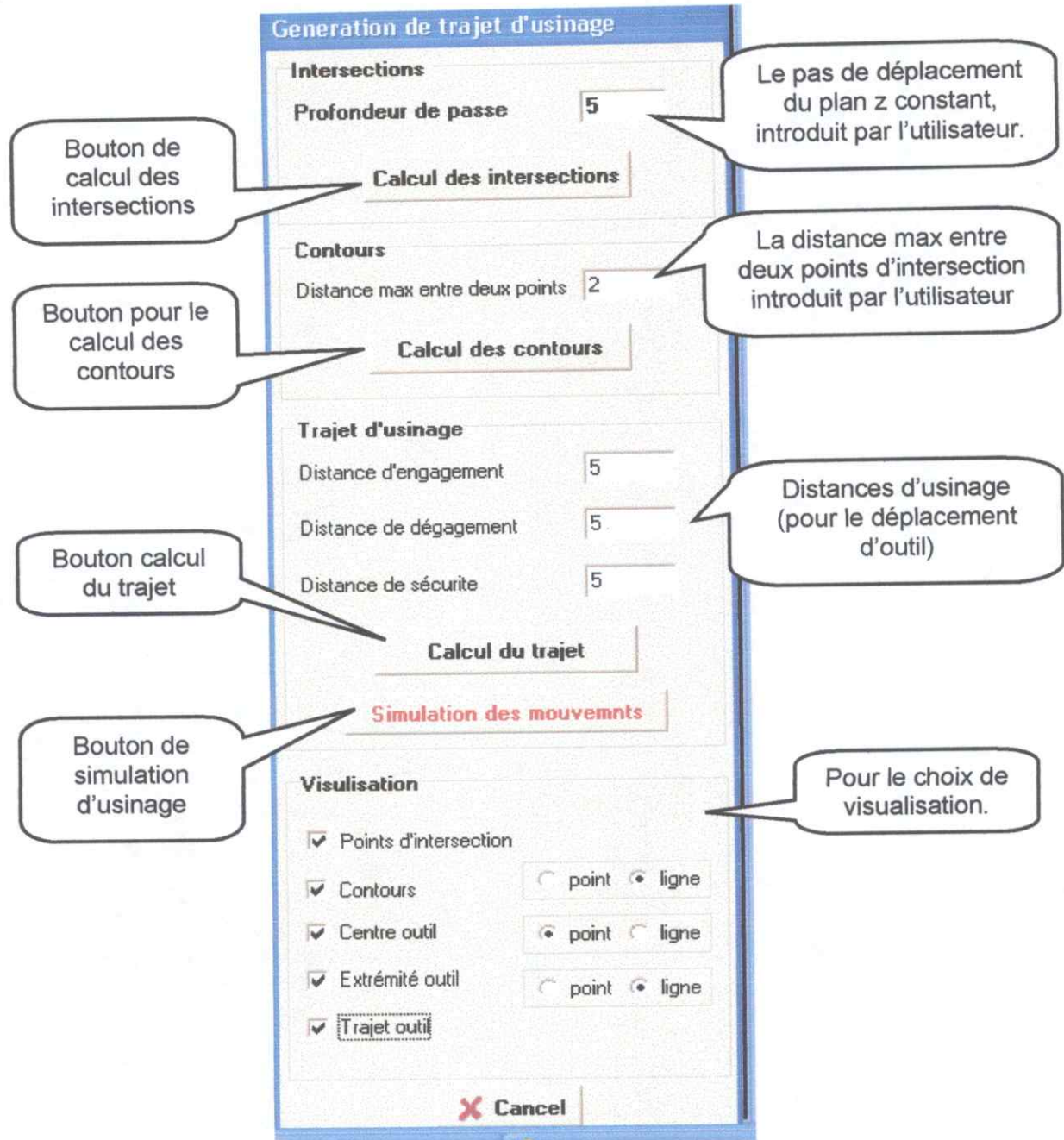


Figure 20 : Fenêtre de génération du trajet d'usinage

V. CONCLUSION :

Dans ce chapitre, nous avons présenté l'environnement de travail, les principaux algorithmes utilisés et les différentes fenêtres de notre application logicielle permettant l'interaction avec l'utilisateur. Dans le chapitre suivant, nous allons tester et valider notre application.



Chapitre **7**

Tests et validation

I. INTRODUCTION :

Après avoir présenté l'implémentation de notre application logiciel, nous passons à la dernière étape, celle des tests et validation.

Cette étape est très importante puisqu'elle permet de tester et de valider les résultats des différentes étapes précédentes et de faire les corrections nécessaires en cas d'apparition de problèmes pendant l'exécution.

II. TESTS ET VALIDATIONS :

II.1. Détermination de la triangulation:

Nous allons tester dans cette phase, le premier cas d'utilisation, c'est la triangulation uniforme initiale de la surface (voir figure1).

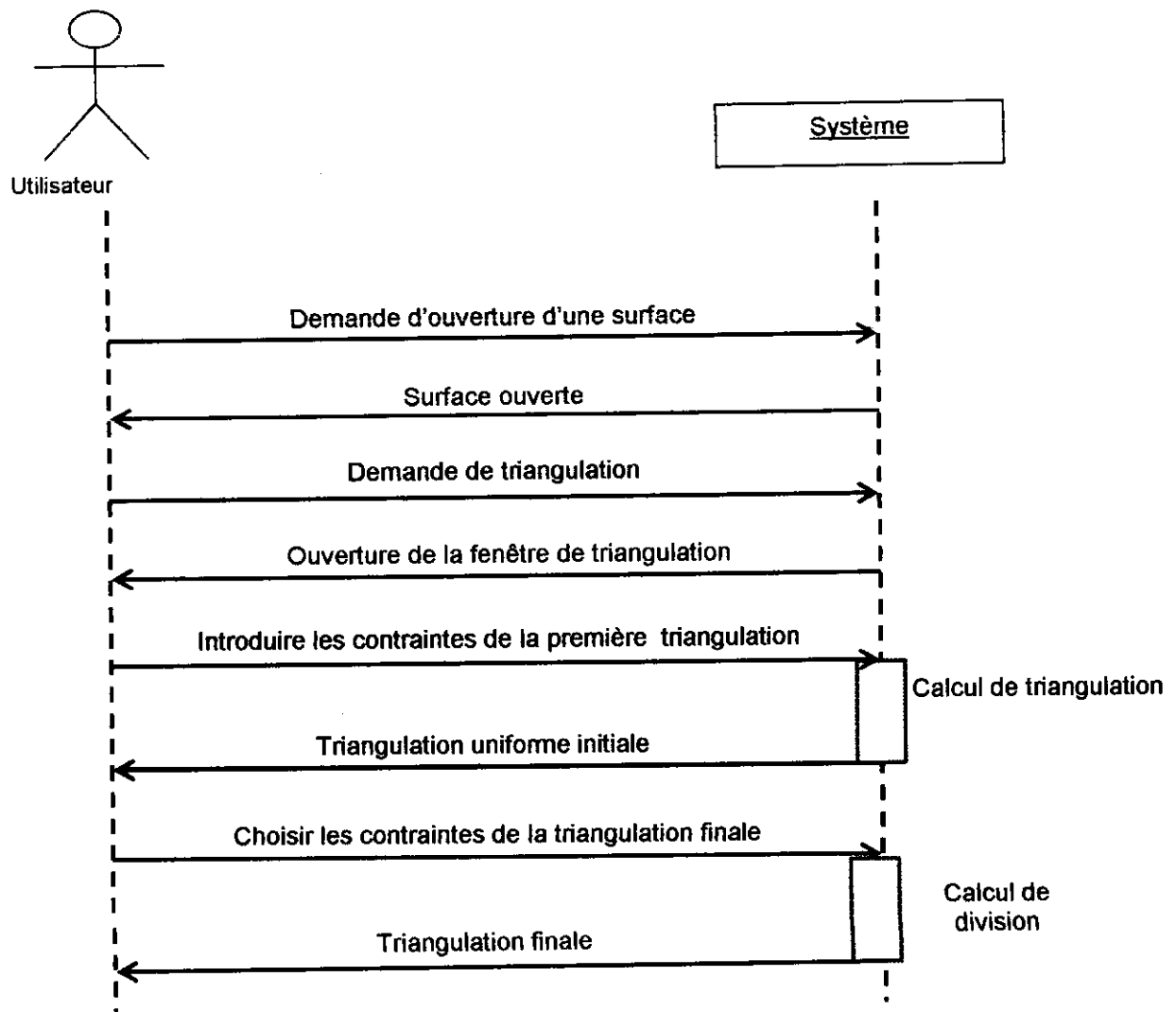


Figure 1 : Scénario de triangulation et de division de la surface.

Nous commençons par l'ouverture d'une surface (voir figure2). Nous avons choisi la surface « ondulée » car elle comporte des parties convexes et des parties concaves et des parties selle de cheval, ce qui permet de localiser les interférences de l'outil dans les différentes régions des surfaces.



Figure 2 : surface ondulée

Après l'ouverture de cette surface, nous activons la forme de la triangulation et nous introduisons les différents paramètres pour obtenir une triangulation initiale avec un nombre minimal des sommets et des segments, et par conséquent un nombre minimal des triangles qui approximes la surface. (Voir figure 3) :

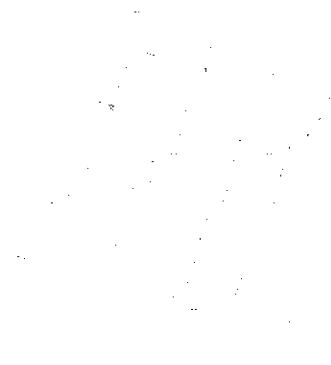
Dans cet exemple nous allons introduire les paramètres suivants :

- Longueur maximale d'un segment est égale à 40 mm.
- L'erreur de flèche des segments est égale à 40 mm.
- L'erreur de flèche des triangles est égale à 40 mm.

Après les calculs automatiques on obtient une triangulation initiale avec un nombre défini des segments dans la direction U et un autre dans la direction V. (Voir figure 3) :



A: sommets initiaux



B : segments horizontaux initiaux

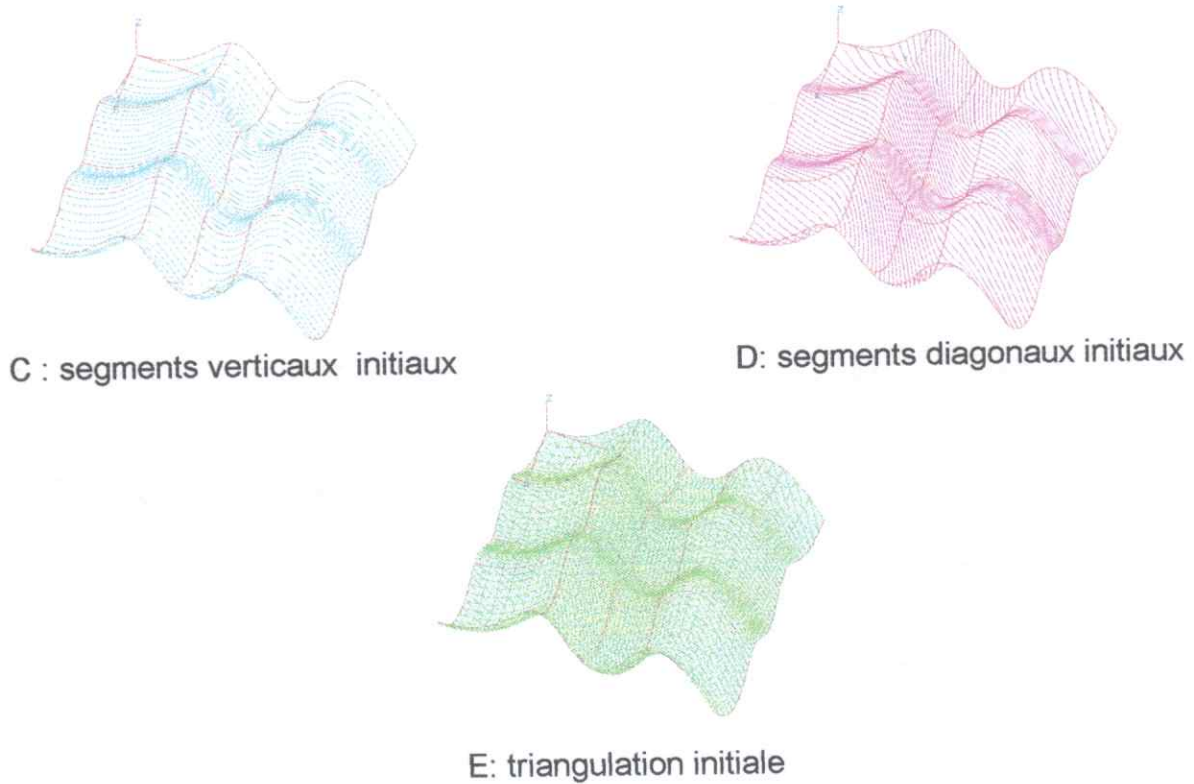


Figure 3 : Les différentes parties de la triangulation initiale.

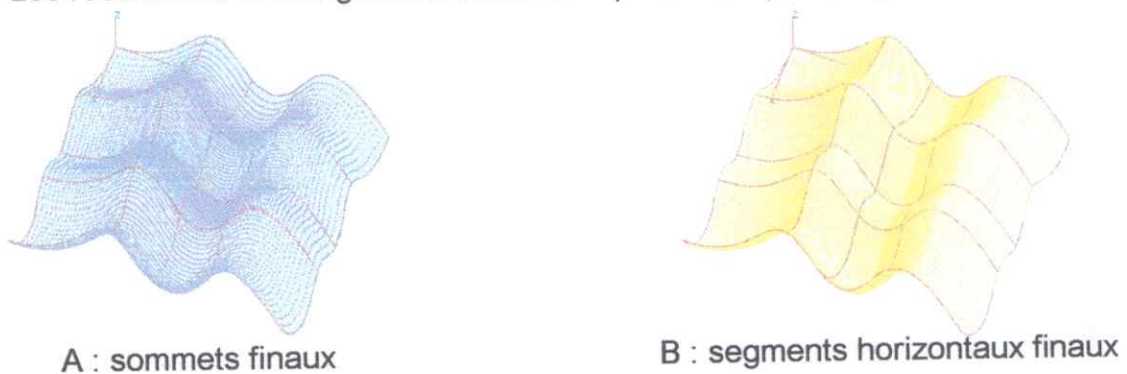
Après l'obtention de la triangulation initiale de la surface on choisi dans la même forme les type de teste qui peuvent s'appliquer à la premier triangulation pour arriver à la triangulation finale.

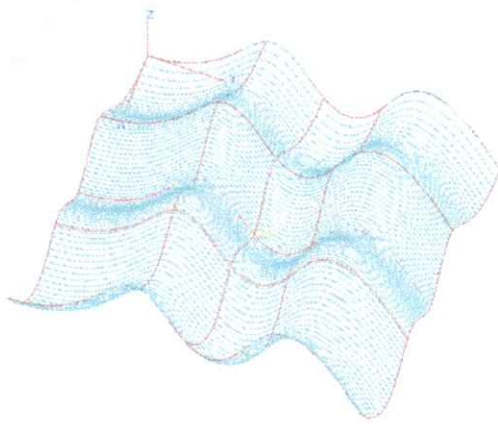
Dans notre exemple on a choisi les quatre type de teste suivant :

- Teste des segments horizontaux.
- Teste des segments verticaux.
- Teste des segments diagonaux.
- Teste des triangles.

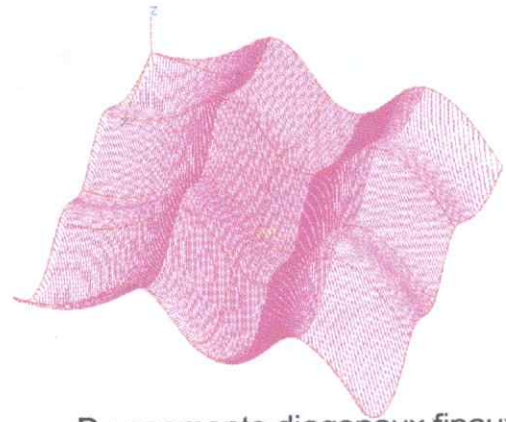
Ces tests permettent d'obtenir une triangulation uniforme optimale avec la respectabilité des différentes contraintes exigées par l'utilisateur.

Les résultats de la triangulation finale sont présentés par la figure 4.

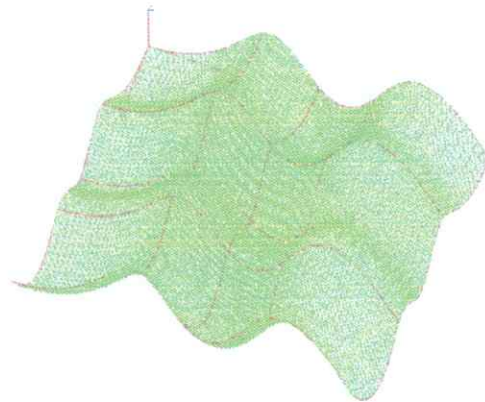




C : segments verticaux finaux



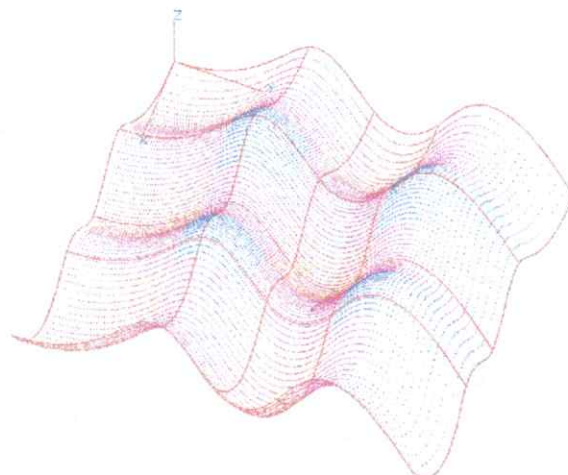
D : segments diagonaux finaux



E : triangulation finale

Figure 4 : les différentes parties de la triangulation finale.

La triangulation finale permet de déterminer et de grouper les sommets de la surface en régions selon les rayons des courbures de chaque sommet. (Voir figure 5)

**Figure 5 :** localisation globale des régions par sommets.

A partir de la localisation des régions par sommets et la méthode des triangles voisines (voir figure 6), On va déterminer une localisation précise des régions. (Voir figure 7)

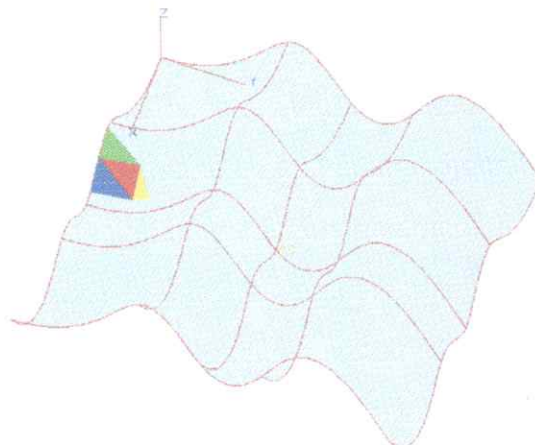


Figure6 : les triangles voisines.

Cette dernière permette de définir les sommets, les triangles et le rayon minimal de chaque région.

Les régions en rouges (0, 2, 4, 8, 9, 10) sont des régions concaves.
 Les régions en bleu (6, 7, 12, 13) sont des régions convexes.
 Les régions en violet (1, 3, 5, 11) sont des régions selle de chevale.

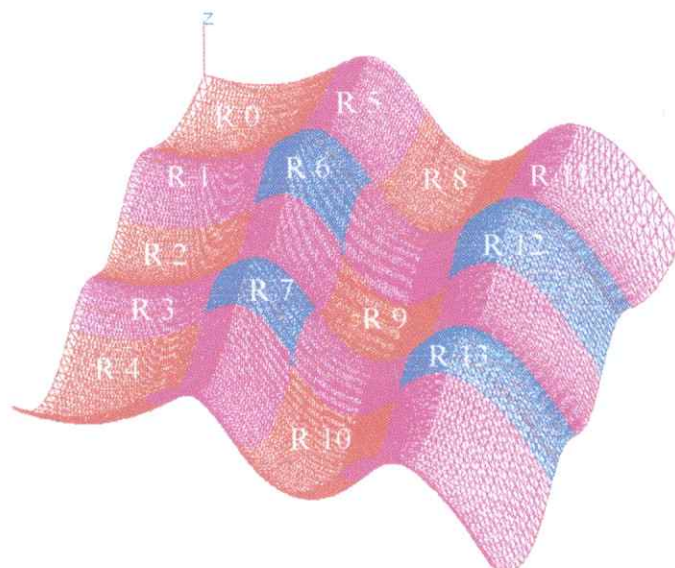


Figure 7 : localisation détaillée des régions.

II.2- DETECTION DES INTERFERENCES ET CORRECTION D'OUTILS :

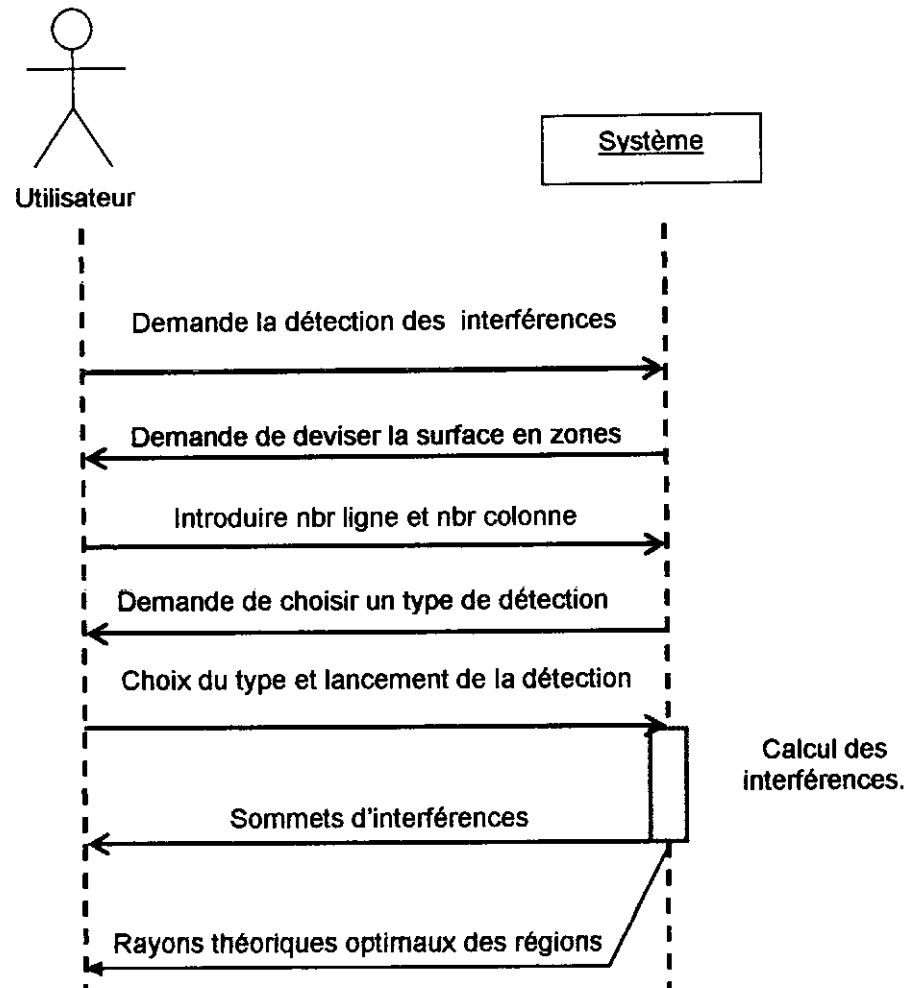


Figure 8 : Scénario de détection des interférences.

Pour lancer la détection des interférences, nous activons la forme de détection des interférences et nous introduisons le nombre des lignes et le nombre des colonnes pour subdiviser les sommets des surfaces dans une matrice des zones (voir Figure 9). Cette matrice est utilisée pour localiser les zones de teste de chaque sommet. Pendant le teste des interférences, les rayons d'outils théoriques des régions sont corrigés. Dans notre exemple nous utilisons les paramètres suivants :

- Le nombre de lignes est égal à 15.
- Le nombre de colonnes est égal à 14.

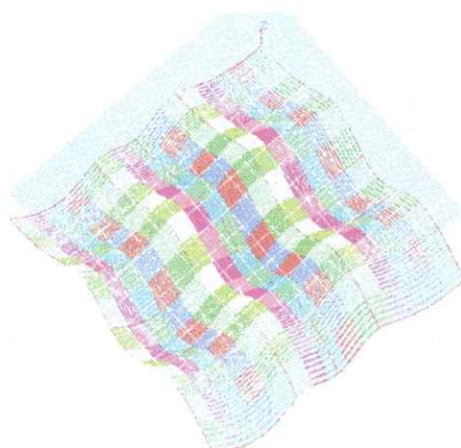


Figure 9 : division des surfaces en zones.

Après la subdivision de la surface, nous allons choisir le teste rapide des interférence et la possibilité de visualiser la sphère de l'outil pendant le teste (Voir figure 10).

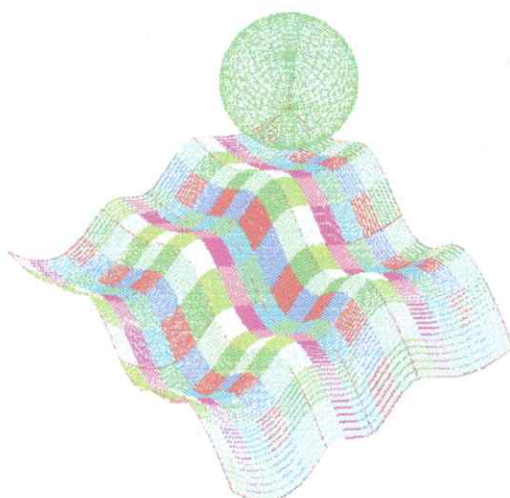


Figure 10 : test des interférences et correction d'outils.

Num région	0	1	2	3	4	5	6
Rayon mm	45.27	5.32	5.18	13.80	13.45	5.34	∞

Num région	7	8	9	10	11	12	13
Rayon mm	∞	8.21	5.19	8.26	52.33	∞	∞

Figure 11 : rayon d'outil de chaque région avant la correction.

Les résultats de teste sont les sommets de la surface qui génèrent des interférences avec au moins un autre sommet d'une autre région (Voir figure 12).

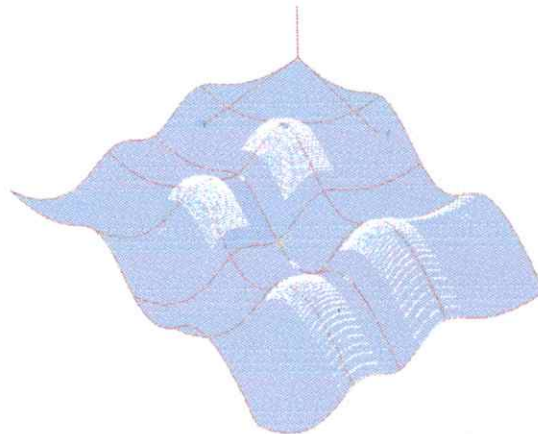


Figure 12 : sommets sources d'interférence.

Les résultats de la correction des rayons d'outils après le test sont affichés dans le tableau suivant (voir figure 13)

Num région	0	1	2	3	4	5	6
Rayon mm	45.27	5.32	5.18	13.80	13.45	5.34	27.65

Num région	7	8	9	10	11	12	13
Rayon mm	27.65	8.21	5.19	8.26	28.50	27.69	27.69

Figure 13 : rayons d'outils après la correction.

II.3- STRATEGIES D'USINAGE :

Cette phase permet de définir la meilleure stratégie d'usinage pour chaque région.

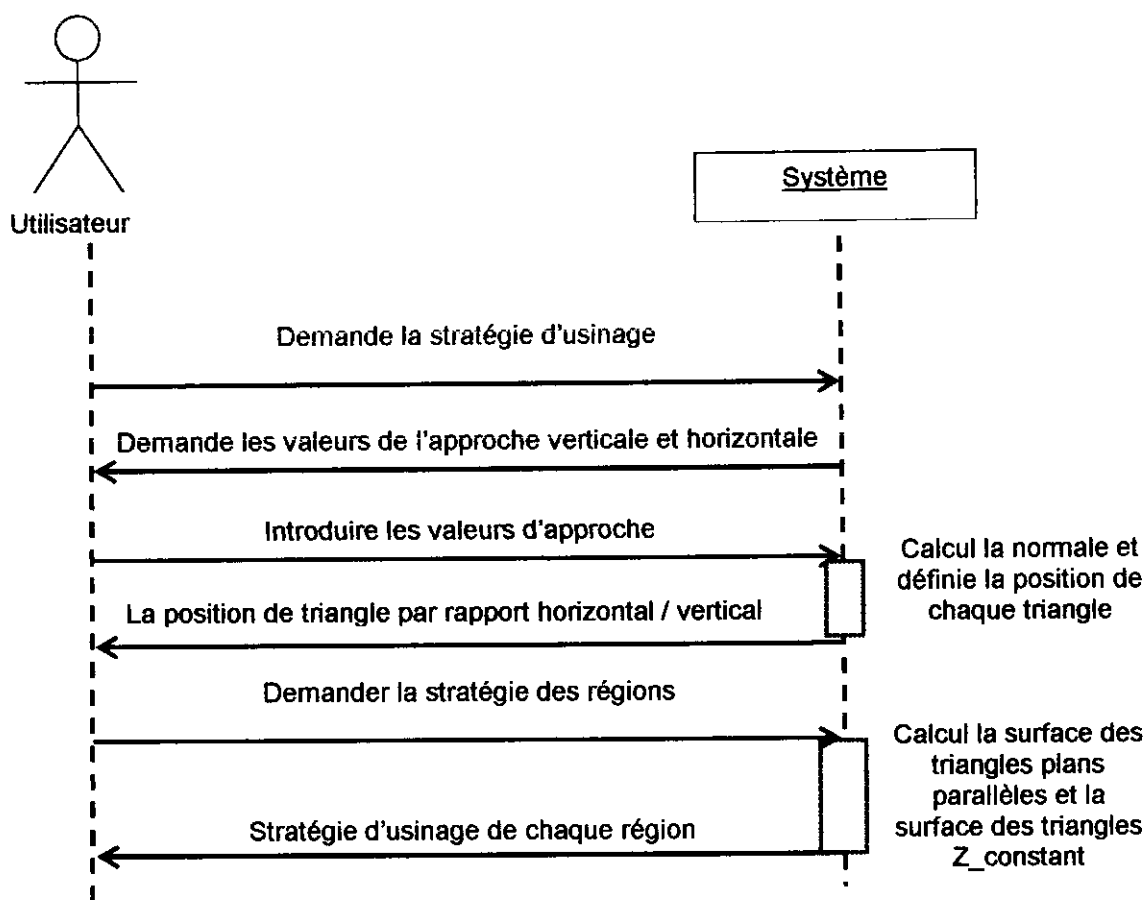


Figure 14 : Scénario de détermination de stratégie d'usinage de chaque région.

Après la phase de détection des interférences et de correction d'outils, nous arrivons à la phase de la définition de la stratégie d'usinage de chaque région.

On ouvre la fenêtre de stratégie d'usinage et on introduit les valeurs d'approche à la verticale et à l'horizontal

Le premier résultat de cette phase est la position de chaque triangle par rapport à l'horizontale et à la verticale (voir figure15.)

Les triangles bleus sont proches de la verticale.

Les triangles rouges sont proches de l'horizontal.

Les triangles violets différents des triangles précédents.

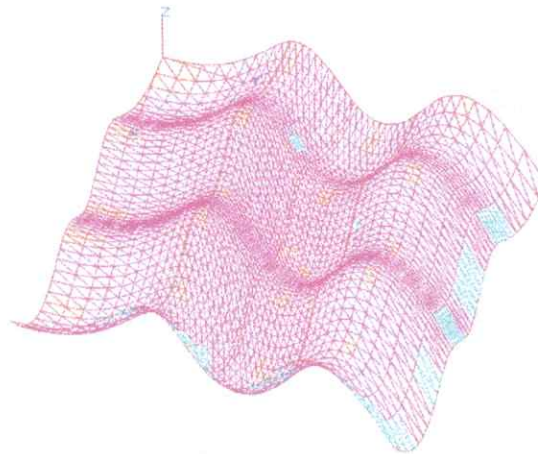


Figure 15 : position de chaque triangle.

A partir de la position des triangle, on calcul la proportion de surface des triangle proche à l'horizontale (stratégie plans parallèle), et la proportion de surface des triangle proche à la verticale (stratégie Z_constant) et on définit la stratégie de chaque région. Si les deux surfaces sont égales, dans ce cas la région est usinable par les deux stratégies (Voir figure 16).

Les régions violettes sont usinables par plans parallèles.

Les régions bleues sont usinables par Z-constant.

Les régions vertes sont usinables par les deux stratégies.



Figure 16 : stratégie d'usinage de chaque région.

II.4- CHOIX DES OUTILS :

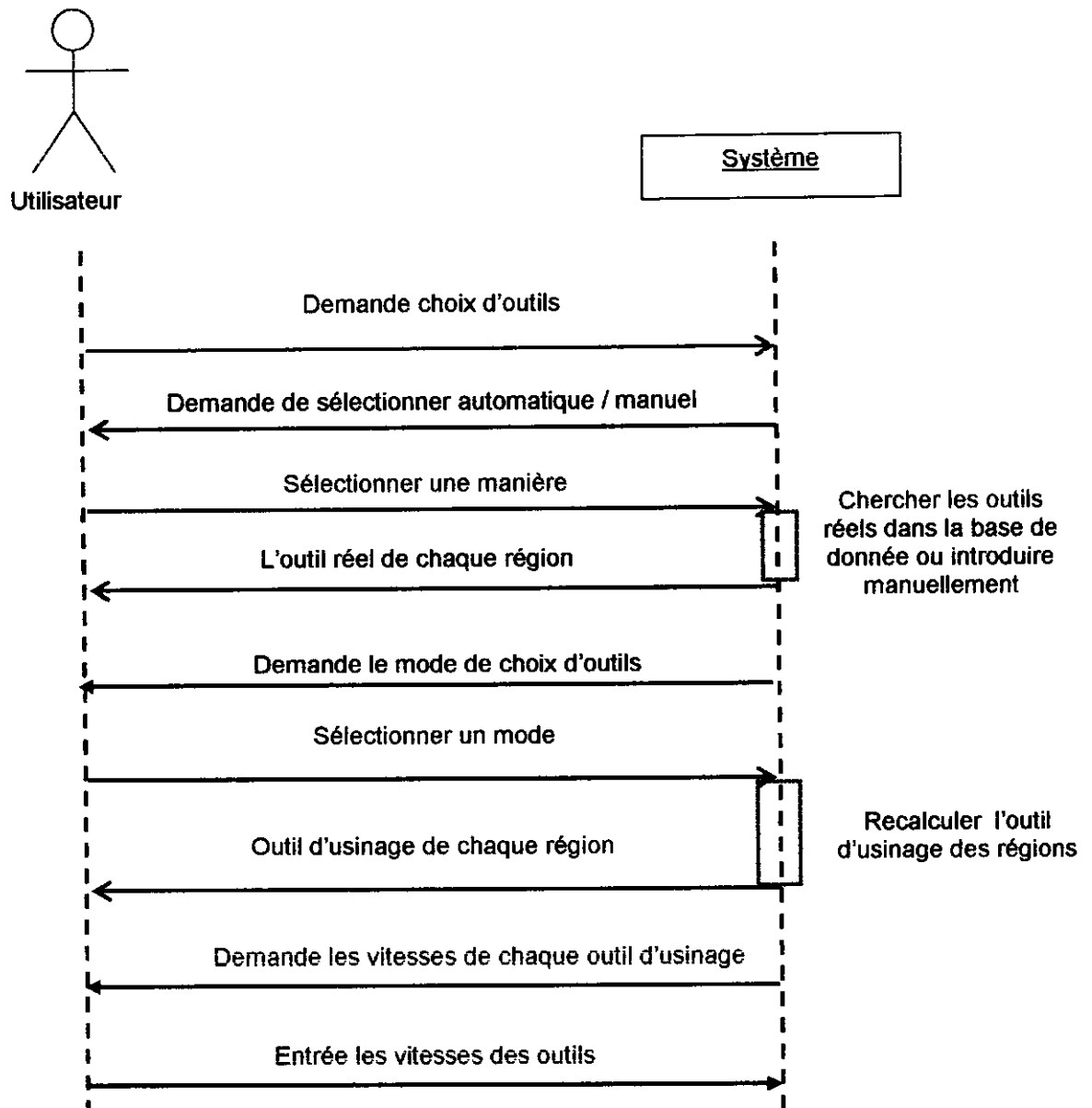


Figure 17 : Scénario de Choix des outils d'usinage des régions.

Pour choisir les outils, nous activons la forme de choix des outils et nous sélectionnons la manière de choix (manuel ou automatique).

Après avoir sélectionné l'outil de chaque région, on détermine l'approche d'utilisation de l'outil. Cinq approches sont utilisées :

- un outil pour chaque région.
- Un outil pour chaque type de région d'une surface.
- Un outil pour chaque type de région de tous les surfaces.
- Un outil pour tous les surfaces.
- Un outil pour chaque surface.

Le mode de choix des outils permet de définir les outils utiliser dans l'usinage, et d'associer à chaque outils d'usinage la vitesse d'avance, vitesse d'engagement et le vitesse de dégagement.

Dans notre exemple, nous avons choisi le mode de sélection d'outils manuel car il y a des régions dont les outils ne figurent pas dans la base de donnée outil, et nous avons opté pour l'approche un outil pour chaque surface.

Le rayon d'outils d'usinage dans ce cas est égal à 5mm.

II.5- GENERATION DE TRAJET D'USINAGE :

Le trajet d'usinage est la phase de validation des résultats précédents.

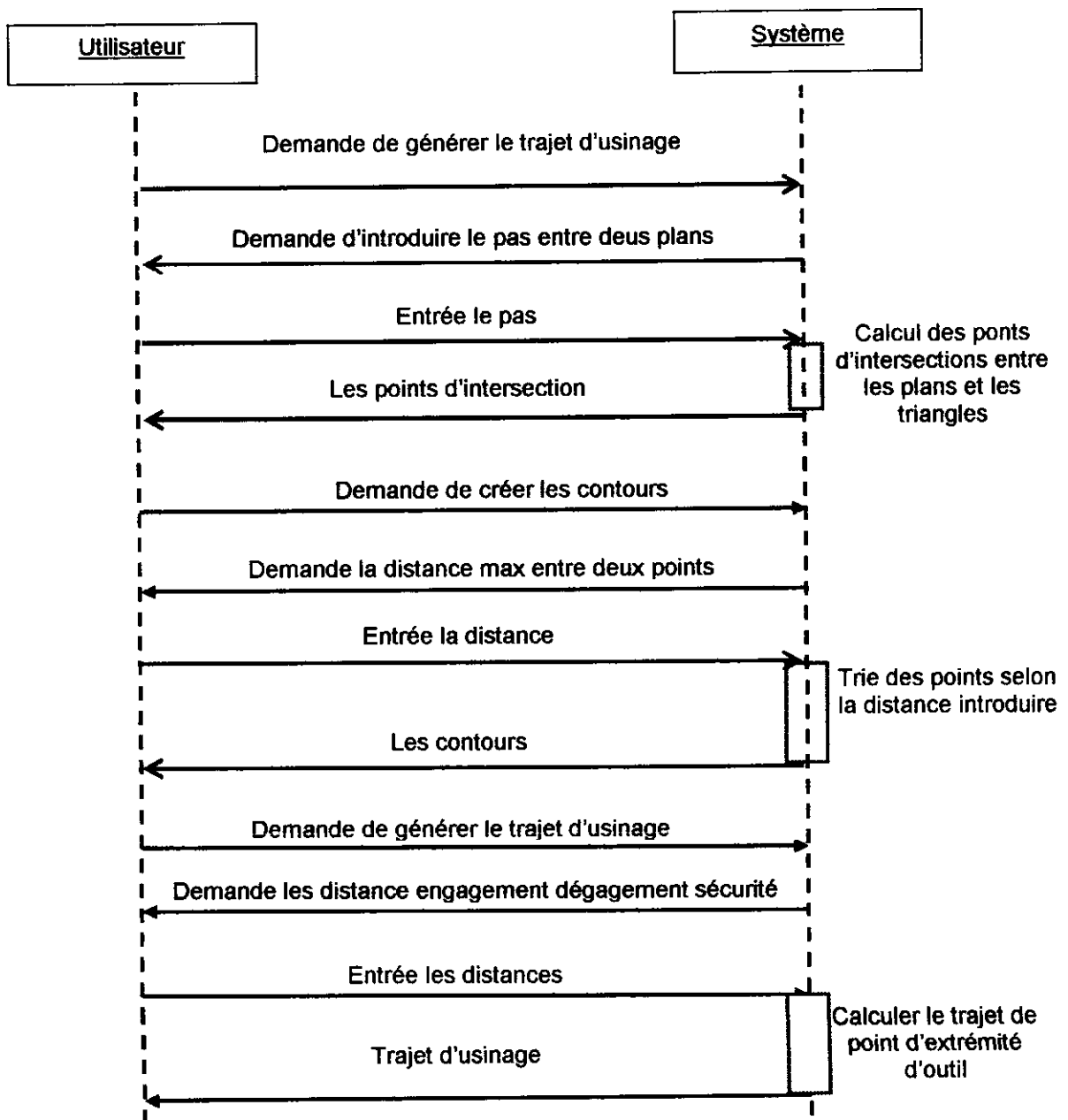


Figure 18 : Scénario de génération de trajet d'usinage.

La génération de trajet d'usinage passe par plusieurs étapes :

- La première, c'est l'ouverture de la fenêtre de la génération de trajet d'usinage et introduction du pas entre deux plans Z_{constant} , dans notre exemple, le pas entre deux plans est égal à 5mm, les résultats de cette étape sont les points d'intersection entre les plans et les triangles. (Voir figure 19)

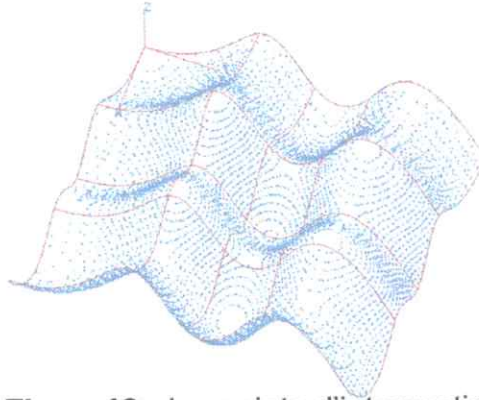


Figure 19 : les points d'intersection.

- La deuxième étape de génération de trajet, c'est le calcul des contours à partir des points d'intersection, pour cela, nous introduisons une valeur de distance maximale entre deux points d'intersection, dans cet exemple nous avons pris une distance égale à 20mm et le résultat est représenté par la figure 20.

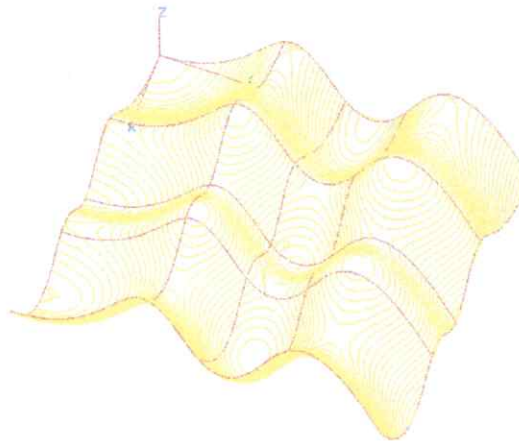


Figure 20 : les contours.

- Après le calcul des contours, nous arrivons à l'étape de calcul du trajet d'usinage. Pour générer le trajet il faut introduire les trois distances :

- La distance d'engagement : égal a 20.
- La distance de dégagement : égal a 20.
- La distance de sécurité : t égal a 30.

La génération de trajet final passe par trois phases :

- A- calcule le trajet de centre outils (voir figure 21).
- B- Calcul le trajet de point extrémité d'outil (voir figure 22).
- C- Calcul le trajet final (voir figure 23).

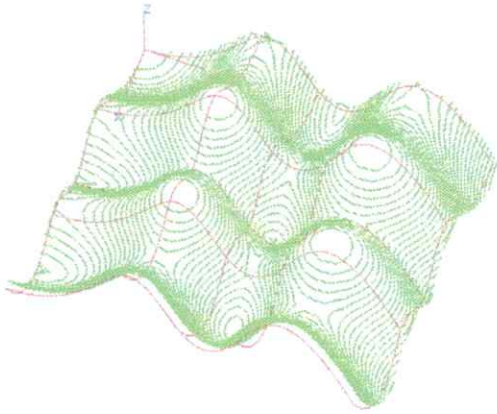


Figure 21 : Trajet de centre d'outil.

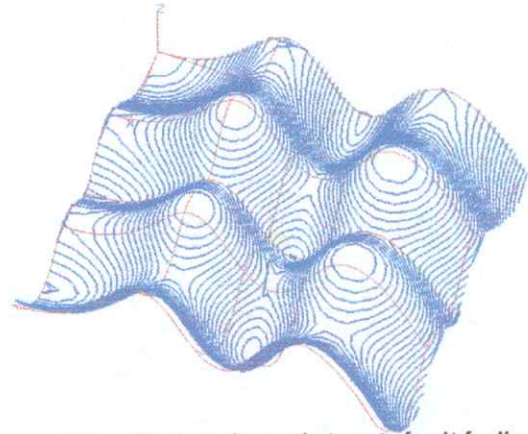


Figure 22 : Trajet de point extrémité d'outil.

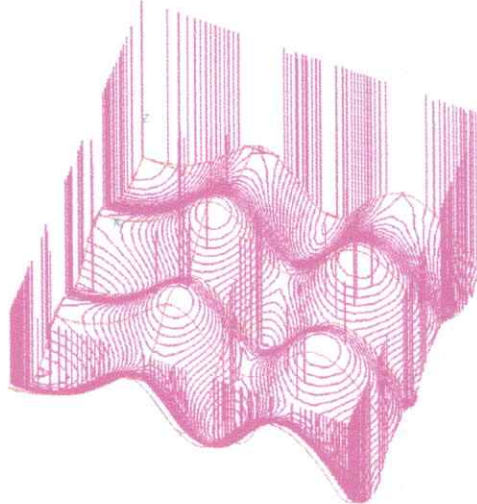


Figure 23 : Trajet d'usinage complet.



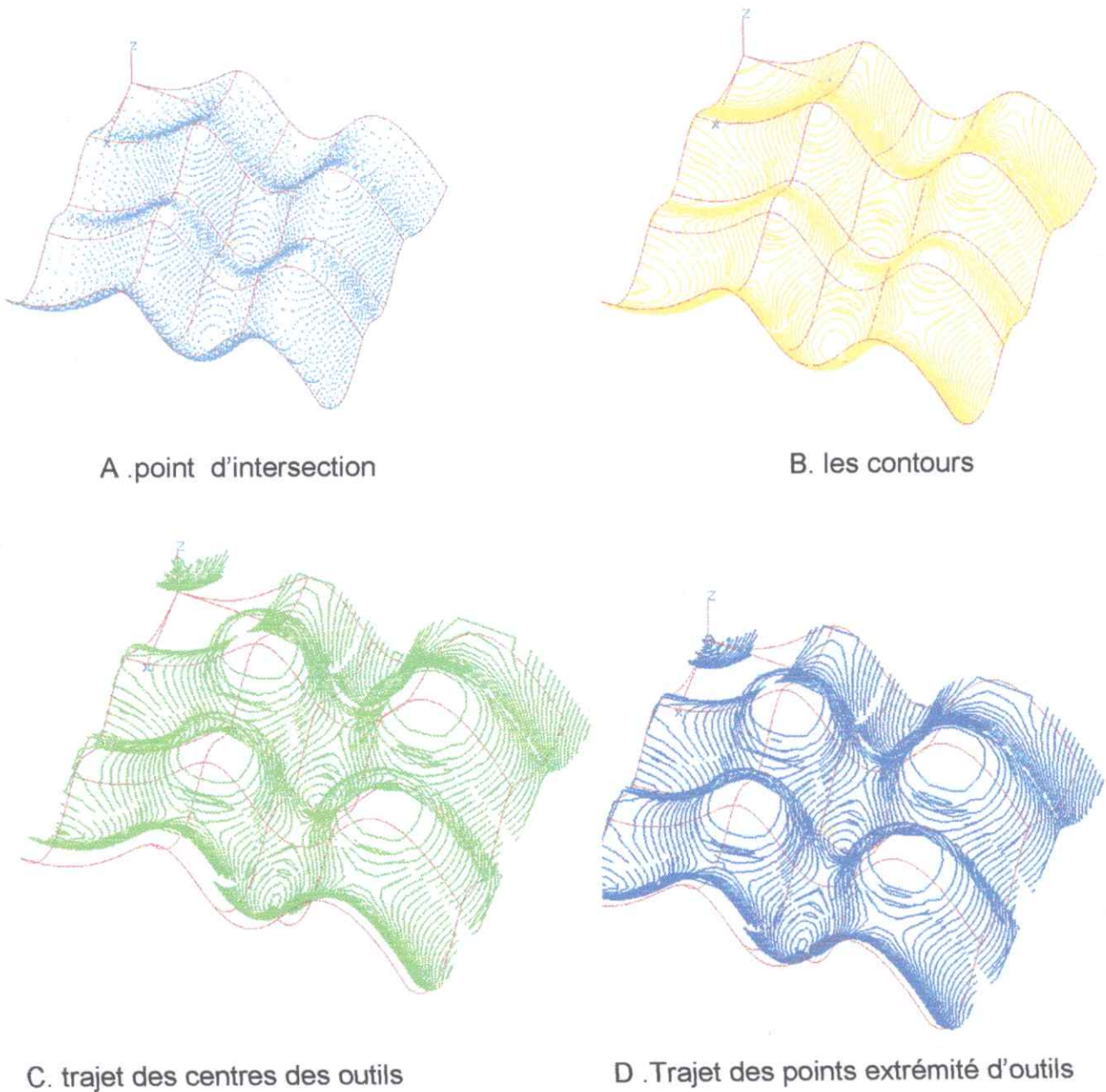
Figure 24 : Simulation de trajet d'outil dans un plan de la surface.

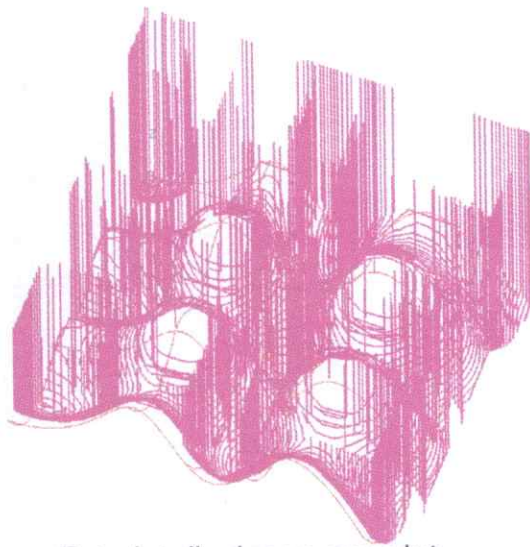
Deux autre cas sont traités :

- 1- un outil pour chaque région. (voir Figures 25 et 26).
- 2- Un outil pour chaque type de région d'une surface (voir Figure 27 et 29).

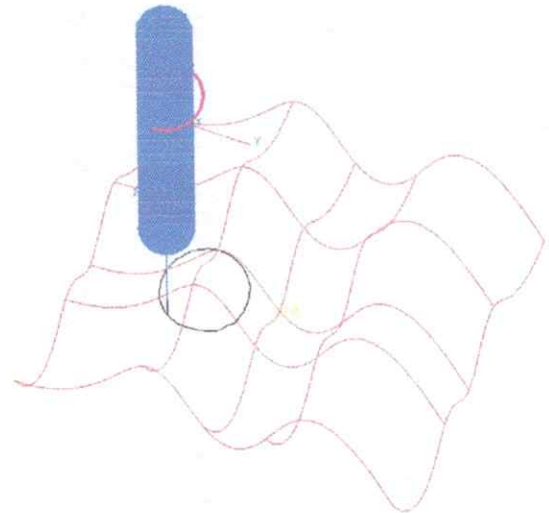
Num région	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Rayon mm	38	5	5	12.5	12.5	5	16	16	8	5	8	16	16	16

Figure 25: tableau des rayons d'outils dans le cas d'un outil pour chaque région.





E. trajet d'usinage complet dans un plan d'une région



F. Simulation de trajet d'outil

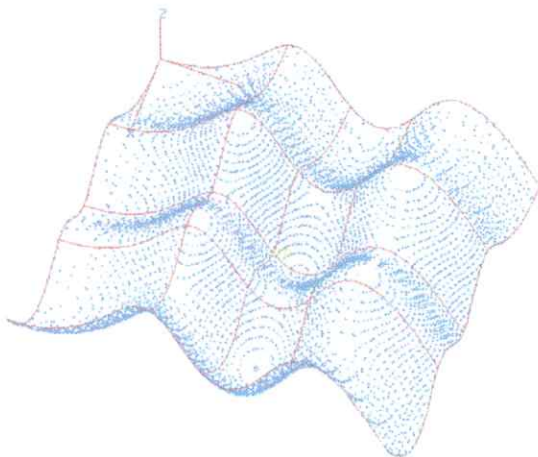
Figure 26 : les résultats de mode d'un outil pour chaque région.

Num région	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Rayon mm	5	5	5	5	5	5	16	16	5	5	5	5	16	16

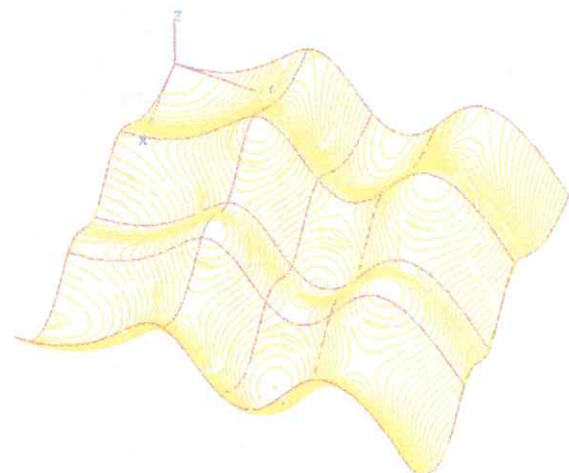
Figure 27: tableau des rayons d'outils dans le cas d'un outil pour chaque Type de région d'une surface.

Dans ce cas :

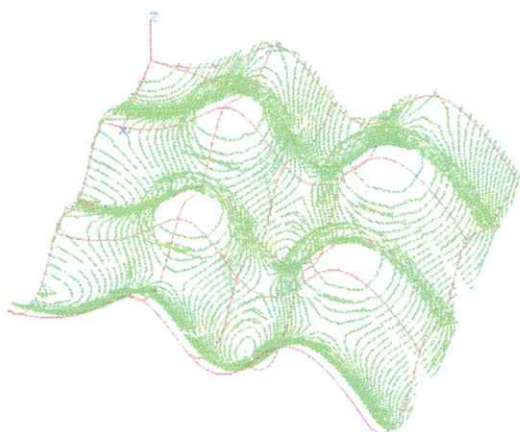
Les régions concaves sont usinées par un outil de rayon égal à 5mm.
 Les régions convexes sont usinées par un outil de rayon égal à 16mm.
 Les régions selle de chevale sont usinées par un outil de rayon égal à 5mm.



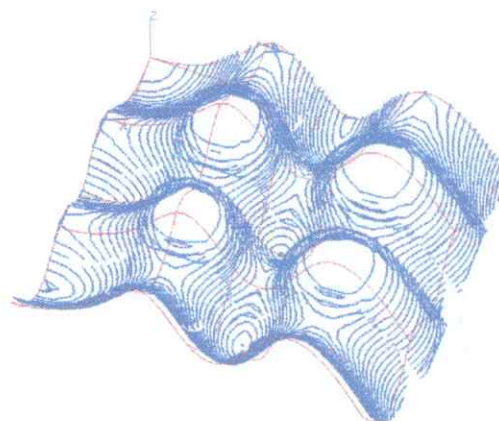
A. Points d'intersections



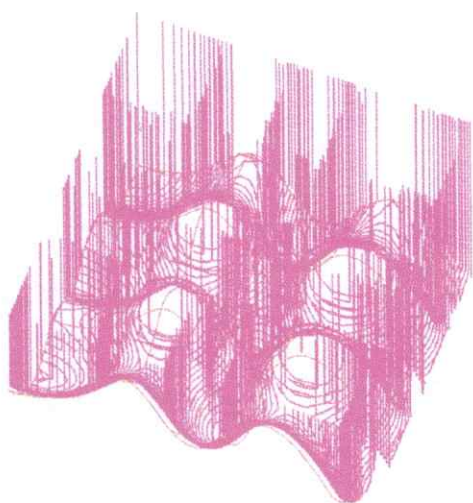
B. les contours



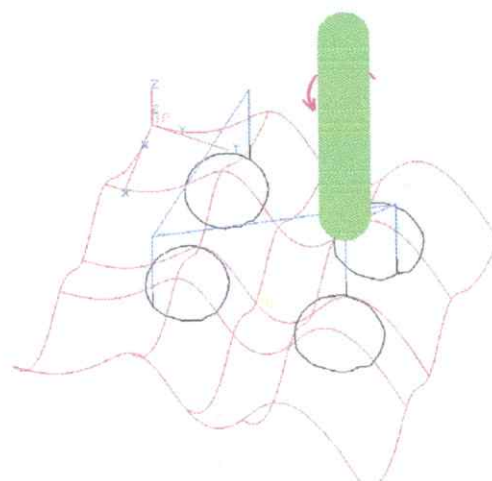
C. trajets des centres d'outils



D. trajets des points extrémités d'outils



E. trajet d'usinage complet



F. Simulation de trajet d'outil

Figure 28 : les résultats de mode d'un outil pour chaque type de région d'une surface.

II.6- GENERATION DE PROGRAMME D'USINAGE PAR « G-CODE » :

Après la génération du trajet d'usinage, nous abordons la dernière étape avant l'usinage, c'est la génération du programme d'usinage « G-Code » (voir figure 29). Ce programme peut être enregistré et transmis à la machine pour réaliser l'usinage effectif de la surface.

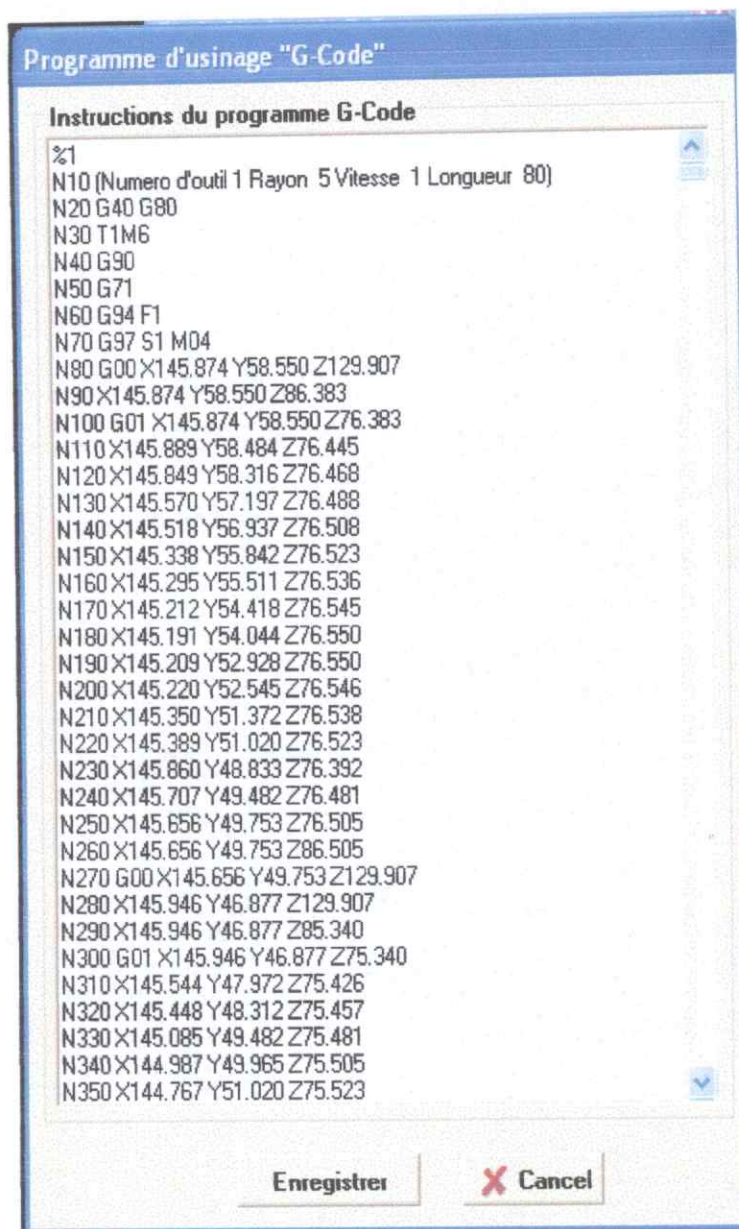


Figure 29 : Génération du programme « G-Code ».

III. CONCLUSION :

Dans ce dernier chapitre, nous avons présenté les différentes fenêtres de notre application logicielle et nous avons testé toutes les fonctionnalités de notre système sur une surface comportant plusieurs types de régions afin de valider les résultats obtenus.

A decorative border with a repeating floral motif surrounds the page. The border consists of small, stylized flowers connected by thin lines, forming a continuous frame around the central text.

Conclusion générale

CONCLUSION GENERALE

Le travail que nous avons présenté dans ce mémoire consiste en la conception et le développement d'une application logicielle d'aide au choix des outils optimums et de stratégies d'usinage lors de la finition des surfaces gauches sur des fraiseuses à commande numérique à trois axes.

La première partie du mémoire a été réservée à l'étude bibliographique des méthodes de conception et de représentation des surfaces gauches en focalisant notre effort sur l'étude des surfaces B-Spline et NURBS. Ensuite, nous avons présenté les machines et les stratégies utilisées dans l'usinage de ces surfaces.

Dans la deuxième partie, nous avons présenté la conception et l'implémentation de notre application logicielle ainsi que les tests de validation.

Le résultat de notre application est l'enrichissement de l'environnement de C.F.A.O développer par l'équipe de CFAO du C.D.T.A par l'intégration des modules permettant de réaliser les tâches suivantes :

- ❖ Triangulation uniforme et semi adaptative des surfaces gauches.
- ❖ Division des surfaces en des régions selon leurs propriétés géométriques.
- ❖ Détection et correction des interférences.
- ❖ Calcul des outils optimums permettant d'éviter les problèmes d'interférences.
- ❖ Groupement des régions.
- ❖ Calcul des points d'intersection.
- ❖ Calcul des contours.
- ❖ Génération de la trajectoire d'outil.
- ❖ Génération des programmes d'usinage « G-Code ».
- ❖ Simulation des mouvements d'outils.

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

- ❖ Reconstruction des surfaces à partir d'un nuage de points.
- ❖ Usinage en ébauche des surfaces à partir d'un nuage de points.
- ❖ Usinage en finition des surfaces à partir d'un nuage de points.
- ❖ Usinage des surfaces gauches sur des fraiseuses à 05 axes.
- ❖ Adaptation des vitesses d'avance lors de l'usinage en finition.

Annexe 

***Bibliothèque graphique
OpenGL***

Annexe A

Bibliothèque Graphique OpenGL [35]

1. Définition :

L'OpenGL est un système graphique qui permet de visualiser une scène 3D (et aussi 2D). Les objets de cette scène peuvent être composés de points, de lignes, de polygones, de quadriques, de NURBS. Ils possèdent des attributs graphiques : paramètres de réflexion, couleur, texture. L'éclairage de la scène est réalisé par des lumières de différents types (spot, lumière à l'infini). La bibliothèque OpenGL a été créée par Silicon Graphics et bénéficie sur des machines à accélération matérielle.

2. Eléments de l'OpenGL :

2.1. Bibliothèques co-existant avec OpenGL :

GLU : (OpenGL Utility Library) contient les routines de bas niveau pour gérer les matrices de transformation et de projection, la facettisation des polygones et le rendu de surface. Les bibliothèques d'extension du système de fenêtres permettent l'utilisation du rendu OpenGL. Il s'agit de GLX pour X Windows (fonctions ayant pour préfixe **glX**) et de WGL pour Microsoft Windows (fonctions ayant pour préfixe **wgl**).

GLUT : OpenGL Utility Toolkit est une boîte à outils indépendante du système de fenêtrage, écrite par Mark Kilgard pour simplifier la tâche d'utilisation des systèmes différents (fonctions ayant pour préfixe **glut**).

GLUT possède des routines pour afficher les objets suivants : cone, cube, tétraèdre, octaèdre, icosaèdre, dodécaèdre, sphère, tore, théière.

2.2. Etats :

C'est le principe général d'OpenGL : On positionne un état interne, et sa valeur est ensuite utilisée comme valeur courante : les ordres de dessin suivants utiliseront cette valeur-ci. Prenons un exemple : pour dessiner un sommet rouge, on positionne d'abord l'état correspondant à la couleur courante à la valeur rouge, et ensuite on demande le dessin d'un sommet. Tous les sommets qui seront ensuite dessinés seront rouges, tant que l'on n'a pas modifié la couleur courante. Et ce principe que nous avons illustré à partir de l'état "couleur" s'applique à tous les états, comme l'épaisseur des traits, la transformation courante, l'éclairage, etc.

2.3. Animation :

Une animation peut se réaliser simplement en utilisant la technique du **double buffer** : montrer à l'écran une image correspondant à une première zone mémoire (buffer) et dessiner les objets dans une deuxième zone mémoire qui n'est pas encore à l'écran, et qui sera affiché lorsque la scène entière y sera calculée, et à une fréquence régulière. L'échange des buffers peut se faire par la commande **glutSwapBuffers(void)**.

2.4. Couleurs :

Les couleurs sont définies en OpenGL de deux manières :

- Couleurs indexées : 256 couleurs sont choisies, et on se réfère au numéro de la couleur (son index). C'est un mode qui était intéressant lorsque les écrans d'ordinateurs ne savaient afficher que 256 couleurs simultanées.
- Couleurs RVBA : une couleur est définie par son intensité sur 3 composantes Rouge, Vert, Bleu. La quatrième composante est appelée *canal Alpha*. La couleur d'un objet est spécifiée par l'appel à **glColor(...)**.

2.5. Primitives géométriques :

La déclaration d'une primitive géométrique se fait par les deux commandes suivantes :

- **glBegin(GLenum mode)** ouvre la déclaration des sommets de la primitive.
- **glEnd(void)** termine cette déclaration.

Il y a dix types de primitives différents, qui correspondent à des points, des lignes, des triangles, des quadrilatères, et des polygones convexes.

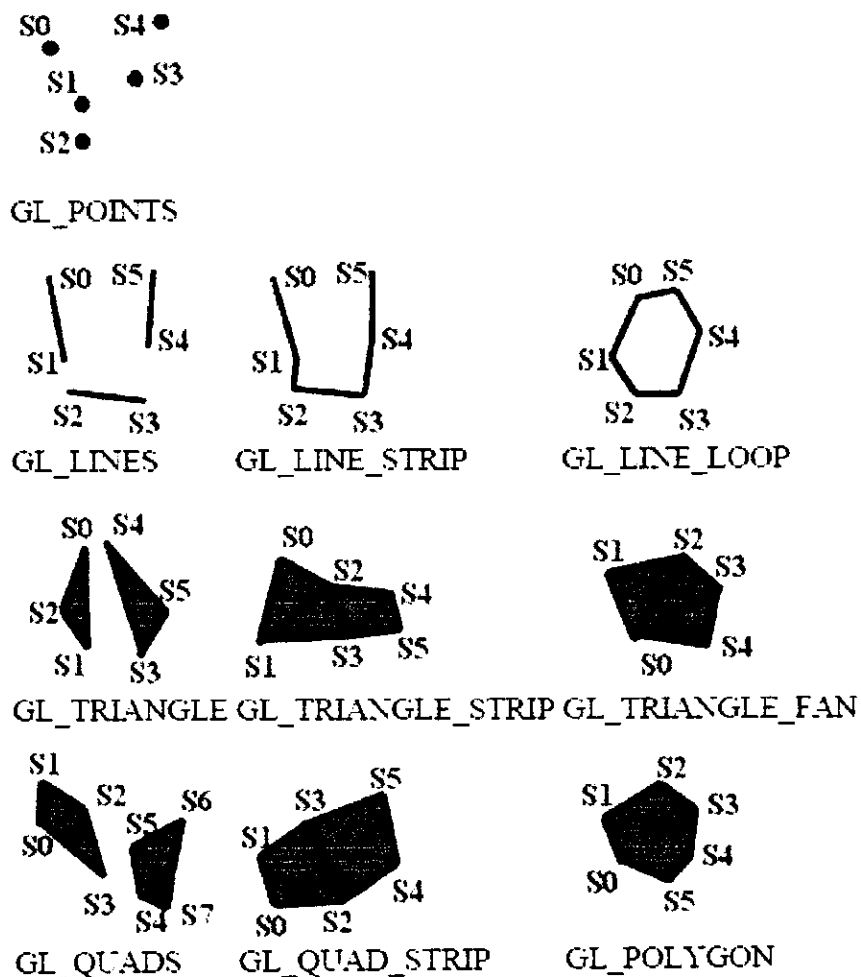


Figure C.1 : Primitive géométrique.

2.6. Vecteur normal :

Un vecteur normal (aussi appelé normale) à une surface en un point de cette surface est un vecteur dont la direction est perpendiculaire à la surface.

Pour une surface plane, les vecteurs normaux en tous points de la surface ont la même direction. Ce n'est pas le cas pour une surface quelconque.

C'est grâce au vecteur normal que l'on peut spécifier l'orientation de la surface dans l'espace, et en particulier l'orientation par rapport aux sources de lumière. L'appel à **glNormal*()** donne une valeur à la normale courante. Elle sera associée aux points spécifiés par les appels suivants à **glVertex*()**.

2.7. Vision :

Le processus de transformation qui produit une image à partir d'un modèle de scène 3D est analogue à celui qui permet d'obtenir une photographie d'une scène réelle à l'aide d'un appareil photo. Il comprend quatre étapes (voir figure C.2).

	Avec un appareil photo	Avec un ordinateur	type de transformation
1	Positionner l'appareil photo	Placer la caméra virtuelle	transformation de vision
2	Arranger les éléments d'une scène à photographier	Composer une scène virtuelle à représenter	transformation de modélisation
3	Choisir la focale de l'appareil photo	choisir une projection	transformation de projection
4	Choisir la taille de la photographie au développement	choisir les caractéristiques de l'image	transformation de cadrage

Figure C.2 : Processus de transformation.

La transformation de vision se fait en utilisant les procédures OpenGL de translation et de rotations appliquées aux objets, ces procédures sont :

- **glTranslate*(TYPE x, TYPE y, TYPE z)** translate le repère local de l'objet du vecteur (x, y, z).
- **glRotate*(TYPE angle, TYPE x, TYPE y, TYPE z)** opère une rotation de l'objet autour du vecteur (x, y, z).
- **glScale*(TYPE a, TYPE b, TYPE c)** opère un changement d'échelle sur 3 axes. Les coordonnées en x sont multipliées par a, en y par b et en z par c.

2.8. Pile de transformation :

Un modèle hiérarchique permet de décrire facilement des objets complexes composés d'objets simples. La scène est organisée dans un arbre tel que les objets ne sont plus définis par leur transformation absolue par rapport au repère de toute la scène, mais par leur transformation relative dans cet arbre.

- A chaque noeud est associé un repère. Le repère associé à la racine est le repère de la scène.
- A chaque arc est associé une transformation géométrique qui positionne l'objet fils dans le repère de son père.

L'OpenGL utilise les *coordonnées homogènes* pour manipuler ses objets. Il maintient trois matrices 4x4 distinctes pour contenir les différentes transformations.

Pour coder l'arbre de description de la scène, il faut utiliser la pile de transformation, en empilant la matrice de transformation courante (sauvegarde des caractéristiques du repère local associé) avant de descendre dans chaque noeud de l'arbre, et en dépilant la matrice en remontant (récupération du repère local associé).

Les primitives de manipulation des matrices de transformation sont :

- **glPushMatrix()** Empile la matrice courante pour sauvegarder la transformation courante.
- **glPopMatrix()** Dépile la matrice courante (La matrice du haut de la pile est supprimée de la pile, et elle devient la matrice courante).
- **glMatrixMode(GLenum mode)** spécifie quelle matrice sera affectée par les commandes suivantes de manipulation de transformations : la matrice de modélisation vision, de projection ou de texture.
- **glLoadIdentity()** donne à la *Matrice courante* la valeur identité.
- **glLoadMatrix*(const TYPE * m)** donne à la *Matrice courante* la valeur de la matrice m.
- **glMultMatrix*(const TYPE * m)** multiplie la *Matrice courante* par la matrice m.

2.9. Eclairage :

La perception de la couleur de la surface d'un objet du monde réel dépend de la distribution de l'énergie des photons qui partent de cette surface et qui arrivent aux cellules de la rétine de l'oeil. Chaque objet réagit à la lumière en fonction des propriétés matérielles de sa surface.

Le modèle d'éclairage d'OpenGL considère qu'un objet peut émettre une lumière propre, renvoyer dans toutes les directions la lumière qu'il reçoit, ou réfléchir une partie de la lumière dans une direction particulière, comme un miroir ou une surface brillante.

Les lampes, elles, vont envoyer une lumière dont les caractéristiques seront décrites par leurs trois composantes : ambiante, diffuse ou spéculaire.

OpenGL distingue quatre types de lumières :

- **Lumière émise** : Ne concerne que les objets. Les objets peuvent émettre une lumière propre, qui augmentera leur intensité, mais n'affectera pas les autres objets de la scène.
- **Lumière ambiante** : Concerne les objets et les lampes. C'est la lumière qui a tellement été dispersée et renvoyée par l'environnement qu'il est impossible de déterminer la direction d'où elle émane. Elle semble venir de toutes les directions. Quand une lumière ambiante rencontre une surface, elle est renvoyée dans toutes les directions.
- **Lumière diffuse** : Concerne les objets et les lampes. C'est la lumière qui vient d'une direction particulière, et qui va être plus brillante si elle arrive perpendiculairement à la surface que si elle est rasante. Par contre, après avoir rencontré la surface, elle est renvoyée uniformément dans toutes les directions.

- **Lumière spéculaire** : Concerne les objets et les lampes. La lumière spéculaire vient d'une direction particulière et est renvoyée par la surface dans une direction particulière. Par exemple un rayon laser réfléchi par un miroir.
- **Brillance** : Ne concerne que les objets. Cette valeur entre 0.0 et 128.0 détermine la taille et l'intensité de la tâche de réflexion spéculaire. Plus la valeur est grande, et plus la taille est petite et l'intensité importante.

2.10. Lampes :

OpenGL offre d'une part une lampe qui génère uniquement une lumière ambiante (lampe d'ambiance), et d'autre part au moins 8 lampes (GL_LIGHT0, ..., GL_LIGHT7) que l'on peut placer dans la scène et dont on peut spécifier toutes les composantes.

La lampe GL_LIGHT0 a par défaut une couleur blanche, les autres sont noires par défaut.

La lampe GL_LIGHT*i* est allumée par un **glEnable(GL_LIGHT*i*)**.

Il faut également placer l'instruction **glEnable(GL_LIGHTING)** pour indiquer à OpenGL qu'il devra prendre en compte l'éclairage.

2.11. Matérielles :

Les propriétés matérielles d'un objet sont celles qui ont été évoquées dans la partie *éclairage* : la lumière émise, la réflexion ambiante, diffuse et spéculaire du matériau dont est fait l'objet. Elles sont déterminées par l'instruction suivante :

- **glMaterial*(GLenum face, GLenum pname, TYPE param)** Où *pname* vaut GL_AMBIENT, GL_DIFFUSE, GL_AMBIENT_AND_DIFFUSE, GL_SPECULAR, GL_SHININESS, ou GL_EMISSION.

2.12. Liste d'affichage :

Il s'agit d'un mécanisme pour stocker des commandes OpenGL pour une exécution ultérieure, qui est utile pour dessiner rapidement un même objet à différents endroits.

Les instructions pour stocker les éléments d'une liste d'affichage sont regroupées entre une instruction **glNewList(GLuint numList, GLenum mode)** et une instruction **glEndList()**. Le paramètre *numList* est un numéro unique (*index*) qui est généré par **glGenLists(GLsizei range)** et qui identifie la liste.

Le paramètre *mode* vaut GL_COMPILE ou GL_COMPILE_AND_EXECUTE. Dans le deuxième cas, les commandes sont non seulement compilées dans la liste d'affichage, mais aussi exécutées immédiatement pour obtenir un affichage.

Les éléments stockés dans la liste d'affichage sont dessinés par l'instruction **glCallList(GLuint numList)**.

Une fois qu'une liste a été définie, il est impossible de la modifier à part en la détruisant et en la redéfinissant.

2.13. Quadriques :

La bibliothèque GLU permet de créer certains objets définis par une équation quadratique : des sphères, des cylindres, des disques et des disques partiels. Pour cela il y a cinq étapes :

- 1) Utiliser **gluNewQuadric()** pour créer une quadrique.

- 2) Spécifier les attributs de rendu :
 - **gluQuadricOrientation()** indique la direction des normales vers l'extérieur ou l'intérieur.
 - **gluQuadricDrawStyle()** indique le style de rendu : avec des points, des lignes ou des polygones pleins.
 - **gluQuadricNormals()** permet de générer des normales pour chaque face ou pour chaque sommet.
 - **gluQuadricTexture()** permet de générer les coordonnées de texture.
- 3) Indiquer la fonction qui traite les erreurs dans l'affichage de la quadrique par **gluQuadricCallback()**
- 4) Afficher la quadrique désirée avec **gluSphere()**, **gluCylinder()**, **gluDisk()**, ou **gluPartialDisk()**. Pour une meilleure efficacité, il est recommandé d'utiliser des listes d'affichage.
- 5) Enfin, pour détruire l'objet, appeler **gluDeleteQuadric()**.

2.14. Textures :

Le placage de texture consiste à placer une image (la texture) sur un objet. Cette opération se comprend aisément lorsqu'il s'agit de plaquer une image rectangulaire sur une face rectangulaire.

De manière générale on procédant de la manière suivante :

- 1) Il s'agit d'abord de créer un Objet Texture et de spécifier la texture que l'on va utiliser.
- 2) Ensuite choisir le mode de placage de la texture avec **glTexEnv[f,i,fv,iv]()**
 - Le mode *decal* décalque la texture sur l'objet : la couleur de l'objet est remplacée par celle de la texture.
 - Le mode *modulate* utilise la valeur de la texture en modulant la couleur précédente de l'objet.
 - Le mode *blend* mélange la couleur de la texture et la couleur précédente de l'objet.
- 3) Puis autoriser le placage de textures avec **glEnable(GL_TEXTURE_2D)** si la texture a deux dimensions. (Ou **GL_TEXTURE_1D** si la texture est en 1D).
- 4) Il est nécessaire de spécifier *Les coordonnées de texture* en plus des coordonnées géométriques pour les objets à texturer.

Les coordonnées de texture vont de 0.0 à 1.0 pour chaque dimension (voir figure C.3).



Figure C.3 : Coordonnées de texture.

Les principes primitifs de manipulation des *Objets Texture* sont :

- **glTexCoord*()** : Pour assigné à chaque sommet des objets à texturer un couple de valeurs qui indique les coordonnées de texture de ce sommet.
- **glGenTextures(GLsizei n, GLuint *textureNames)** : Renvoie *n* noms (des numéros, en fait) de textures dans le tableau *textureNames[]*.
- **glBindTexture(GL_TEXTURE_2D, GLuint textureNames)** : à trois effets différents :
 - la première fois qu'il est appelé avec une valeur *textureNames* non nulle, un nouvel *Objet Texture* est crée, et le nom *textureNames* lui est attribué.
 - avec une valeur *textureNames* déjà liée à un *objet Texture*, cet objet devient actif.
 - avec la valeur zéro, OpenGL arrête d'utiliser des *objets textures* et retourne à la texture par défaut, qui elle n'a pas de nom.
- **glDeleteTextures(GLsizei n, const GLuint *textureNames)** : Efface *n* *objets textures* nommés par les éléments du tableau *textureNames*.

2.15. Tableaux de Sommets :

La motivation pour l'utilisation de tableaux de sommets est double : il s'agit d'une part de réduire le nombre d'appels de fonctions, et d'autre part d'éviter la description redondante de sommets partagés par des polygones adjacents.

En général, on procédant de la manière suivant :

- 1) Il s'agit d'abord **d'activer** jusqu'à six tableaux, stockant chacun un des six types de données suivantes :
 - Coordonnées des sommets.
 - Couleurs RVBA.
 - Index de couleurs.
 - Normales de surfaces.
 - Coordonnées de texture.
 - Indicateurs de contour des polygones.

- 2) Ensuite spécifier les données du ou des tableaux.
- 3) Puis déréférencer le contenu des tableaux (accéder aux éléments) pour tracer une forme géométrique avec les données. Il y a trois méthodes différentes pour le faire :
 - Accéder aux éléments du tableau un par un : accès aléatoire.
 - Créer une liste d'éléments du tableau : accès méthodique.
 - Traiter les éléments du tableau de manière séquentielle : accès séquentiel ou systématique.

Six routines permettent de spécifier des tableaux en fonction de leur type :

- void **glVertexPointer**(GLint *taille*, GLenum *type*, GLsizei *stride*, const GLvoid **pointeur*).
- void **glColorPointer**(GLint *taille*, GLenum *type*, GLsizei *stride*, const GLvoid **pointeur*).
- void **glIndexPointer**(GLenum *type*, GLsizei *stride*, const GLvoid **pointeur*).
- void **glNormalPointer**(GLenum *type*, GLsizei *stride*, const GLvoid **pointeur*).
- void **glTexCoordPointer**(GLint *taille*, GLenum *type*, GLsizei *stride*, const GLvoid **pointeur*).
- void **glEdgeFlagPointer**(GLsizei *stride*, const GLvoid **pointeur*).

Le paramètre *pointeur* indique l'adresse mémoire de la première valeur pour le premier sommet du tableau.

Le paramètre *type* indique le type de données.

Le paramètre *taille* indique le nombre de valeurs par sommets, et peut prendre suivant les fonctions les valeurs 1, 2, 3 ou 4.

Le paramètre *stride* indique le décalage en octets entre deux sommets successifs.

2.16. Lissage :

Les lignes qui ne sont ni horizontales ni verticales sont crénelées. Ce phénomène est appelé *aliasing*.

La fonction **glHint**(GLenum *cible*, GLenum *hint*) permet d'indiquer les préférences que l'on a entre qualité de l'image et rapidité de l'affichage.

Le paramètre *cible* peut prendre les valeurs suivantes : GL_POINT_SMOOTH_HINT, GL_LINE_SMOOTH_HINT, GL_POLYGON_SMOOTH_HINT : pour une qualité d'échantillonnage souhaitable, GL_FOG_HINT : pour la qualité ou la rapidité, GL_PERSPECTIVE_CORRECTION_HINT : pour prend en compte la perspective ou non.

Le paramètre *hint* peut prendre la valeur GL_FASTEST, pour privilégier la vitesse, ou GL_NICEST pour privilégier la qualité, ou GL_DONT_CARE pour indiquer l'absence de préférence.

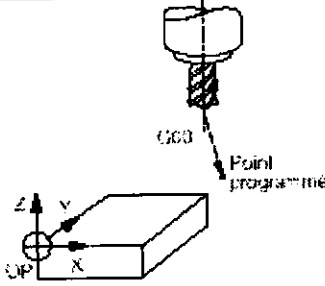
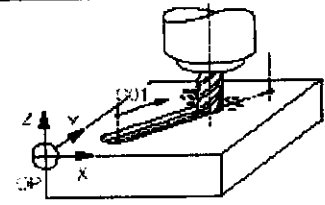
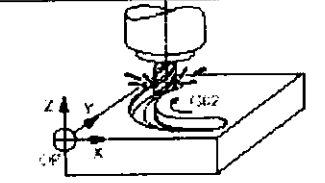
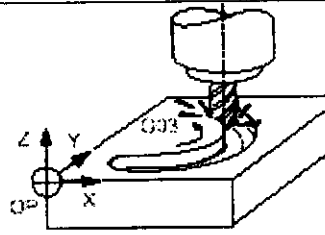

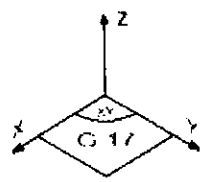


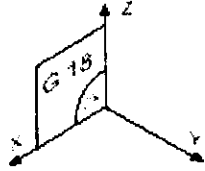
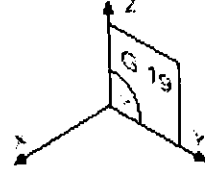
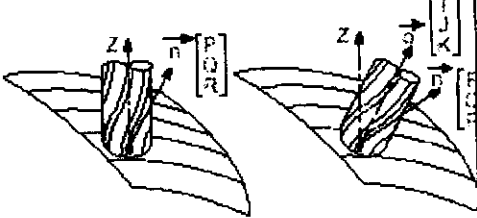
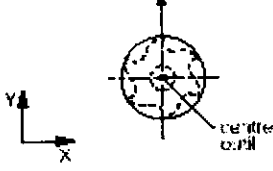
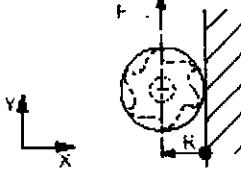
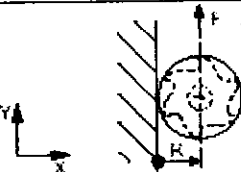


Annexe **B**

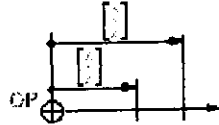
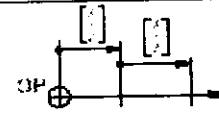
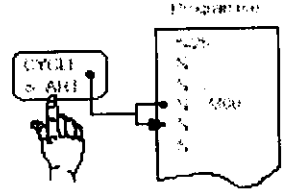
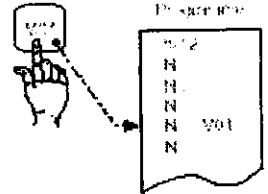
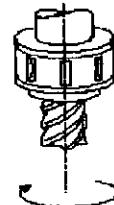
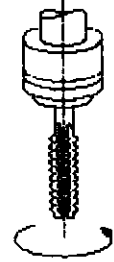
***Principales fonctions du
G-code***

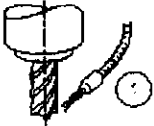
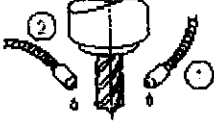
ANNEXE B

PRINCIPALES FONCTIONS DU G-CODE

Fonctions	Figures
<p>G00 : Interpolation linéaire à vitesse rapide. Fonction modale Syntaxe : N.. [G90/G91] G00 [□±] X.. Y.. Z..</p>	 <p>Figure C.1 : La fonction G00.</p>
<p>G01 : Interpolation linéaire à vitesse d'avance. Programmée Fonction modale Syntaxe : N.. [G90/G91] G01 [□±] X.. Y.. Z.. [F..]</p>	 <p>Figure C.2 : La fonction G01.</p>
<p>G02 : Interpolation circulaire sens antitrigonométrique à vitesse d'avance programmée. Fonction modale Syntaxe (plan XY) : N.. [G17] [G90/G91] G02 X.. Y.. I.. J.. ou R.. [F..]</p>	 <p>Figure C.3 : La fonction G02.</p>
<p>G03 : Interpolation circulaire sens trigonométrique à vitesse d'avance programmée. Fonction modale Syntaxe (plan XY) : N.. [G17] [G90/G91] G03 X.. Y.. I.. J.. ou R.. [F..]</p>	 <p>Figure C.4 : La fonction G03.</p>
<p>G04 : Temporisation programmable. Fonction modale Syntaxe : N.. G04 F..</p>	 <p>Figure C.5 : La fonction G04.</p>
<p>G17* : Choix du plan XY Fonction modale Syntaxe : N.. G17</p>	 <p>Figure C.6. La fonction G17.</p>

<p>G18 : Choix du plan ZX Fonction modale Syntaxe : N.. G18</p>	 <p>Figure C.7 : La fonction G18.</p>
<p>G19 : Choix du plan YZ Fonction modale Syntaxe : N.. G19</p>	 <p>Figure C.8 : La fonction G19.</p>
<p>G29 : Correction d'outil dans l'espace 3 ou 5 axes. Fonction modale Syntaxe : N.. [D..] [G01] G29 X.. Y.. Z.. P.. Q.. R.. [I.. J.. K..] [A.. / B.. / C..]</p>	 <p>Figure C.9 : La fonction G29.</p>
<p>G40 : Annulation de correction de rayon Fonction modale Syntaxe : N.. [G00/G01] G40 X.. Y.. Z..</p>	 <p>Figure C.10 : La fonction G40.</p>
<p>G41 : Correction de rayon à gauche du profil à usiner Fonction modale Syntaxe (plan XY) : N.. [G17] [D..] [G00/G01/G02/G03] G41 X.. Y..</p>	 <p>Figure C.11 : La fonction G41.</p>
<p>G42 : Correction de rayon à droite du profil à usiner Fonction modale Syntaxe (plan XY) : N.. [G17] [D..] [G00/G01/G02/G03] G42 X.. Y..</p>	 <p>Figure C.12 : La fonction G42.</p>
<p>G70 : Programmation en pouce Fonction modale Syntaxe : N.. G70</p>	 <p>Figure C.13 : La fonction G70.</p>
<p>G71 : Programmation en métrique Syntaxe : N.. G71</p>	 <p>Figure C.14 : La fonction G71.</p>

<p>G90 : Programmation absolue par rapport à l'origine programme Fonction modale Syntaxe : N.. G90 X.. Y.. Z.. A.. B.. C..</p>	 <p><i>Figure C.15 : La fonction G90.</i></p>
<p>G91 : Programmation relative par rapport au point de départ du bloc Fonction modale Syntaxe : N.. G91 X.. Y.. Z.. A.. B.. C..</p>	 <p><i>Figure C.16 : La fonction G91.</i></p>
<p>M00 : Arrêt programmé . Fonction non modale Syntaxe : N.. [G40] M00 [\$0 ...]</p>	 <p><i>Figure C.17 : La fonction M01.</i></p>
<p>M01 : Arrêt programmé optionnel. Fonction modale Syntaxe : N.. [G40] M01 [\$0 ...]</p>	 <p><i>Figure C.18 : La fonction M02.</i></p>
<p>M02 : Fin de programme. Fonction modale Syntaxe : N.. M02</p>	
<p>M03 : Rotation de broche sens antitrigonométrique Fonction modale Syntaxe : N.. M03</p>	 <p><i>Figure C.19 : La fonction M03.</i></p>
<p>M04 : Rotation de broche sens trigonométrique Fonction modale Syntaxe : N.. M04</p>	 <p><i>Figure C.20 : La fonction M04.</i></p>
<p>M05 : Arrêt de broche Fonction modale Syntaxe : N.. M05</p>	

<p>M08 : Arrosage numéro 1. Fonction modale avant Syntaxe : N.. M08</p>	 <p><i>Figure C.21 : La fonction M08.</i></p>
<p>M09 : Arrêt des arrosages 1 et 2 Fonction non modale après Syntaxe : N.. M09</p>	 <p><i>Figure C.22 : La fonction M09.</i></p>
<p>F : Avance, temporisation, nombre de filets Syntaxe : N.. G93 F.. (Avance en V/L). N.. G94 F.. (Avance en mm/min, degrés/min, pouce/min) N.. G95 F.. (Avance en mm/t, pouce/tour) N.. G04 F.. (Temporisation en secondes) N.. G31 F.. (Nombre de filets)</p>	
<p>S : Nombre de tours/minute, nombre de répétitions de sous programme Syntaxe: N.. G97 S.. (Vitesse de broche en tours/min. N.. G77 [H..] [N.. N..] S.. (Appel et répétitions de sous programme.</p>	
<p>T : Numéro d'outil Syntaxe : N.. T.. M06 (Appel de l'outil)</p>	

Liste des figures

Chapitre 1 :

Figure 1 : Vecteur tangent et vecteur normal en un point sur une surface paramétrique.	4
Figure 2 : Courbes isoparamétriques.	5
Figure 3 : Courbure d'une courbe.	5
Figure 4 : Surface B-Spline.	9
Figure 5 : Surface NURBS.	12

Chapitre 2 :

Figure 1 : structure d'une MOCN.	14
Figure 2 : Schéma général de l'asservissement d'un axe numérique.	18
Figure 3 : règle des trois doigts.	19
Figure 4 : axes de rotation.	20
Figure 5 : Format général des Mots.	22
Figure 6 : Format général des Blocs.	23
Figure 7 : Structure d'un programme ISO.	23
Figure 8 : usinage en bout.	25
Figure 9 : usinage en roulant.	25
Figure 10 : Type de fraisage.	26
Figure 11 : vue générale sur l'usinage.	27
Figure 12 : Trajectoires en positionnement point à point.	28
Figure 13 : Trajectoire en déplacement paraxial.	29
Figure 14 : Trajectoires en continu (contournage en 2D).	29

Chapitre 3 :

Figure 1 : différents types de fraises.	30
Figure 2 : Trajet réel de l'outil pendant l'usinage.	32
Figure 3 : Enveloppe de la surface a usinée.	32
Figure 4 : Position de la fraise sur la surface.	32
Figure 5 : Copiage informatique, guidage de l'outil tangent à la surface.	34
Figure 6 : Erreur de flèche sur les trajets.	34
Figure 7 : Hauteur de crête et distance maximale entre passes.	35
Figure 8 : Outil en position d'interférence et en collision.	36
Figure 9 : Usinage en aller simple.	36
Figure 10 : Usinage en aller retour.	37
Figure 11 : Usinages par la méthode concentrique.	37
Figure 12 : Usinage en Spiral in.	38
Figure 13 : Usinage Spiral out.	38
Figure 14 : Usinage avec la méthode Radiale.	38
Figure 15 : Méthode des plans parallèles.	39
Figure 16 : Méthode de Z-Constant.	39

Chapitre 4 :

Figure 1 : intersection de deux triangles.	41
Figure 2 : Triangulation uniforme d'une surface nominale.	41
Figure 3 : Conditions de subdivision d'un triangle.	42
Figure 4 : Différents cas de subdivision d'un triangle.	42
Figure 5 : Triangulation adaptative d'une surface nominale.	43
Figure 6 : Intersection entre un triangle et un plan.	44
Figure 7 : Droite dans l'espace.	45
Figure 8 : Position des différents points par rapport à AB.	48

Chapitre 5 :

Figure 1 : Diagramme de cas d'utilisation.	53
Figure 2 : Diagramme de classe.	55
Figure 3 : classe TTriangulation des surfaces.	56
Figure 4 : classes principales de la triangulation.	58
Figure 5 : différentes classes de la localisation des régions.	60
Figure 6 : classe Matrice_surface.	61
Figure 7 : classes de la génération de trajet d'usinage.	62
Figure 8 : diagramme de collaboration.	63
Figure 9 : Diagramme d'activité pour la triangulation des surfaces et la localisation des régions.	64
Figure 10 : Diagramme d'activité du choix des outils.	65
Figure 11 : Diagramme d'activité de génération du trajet.	66
Figure 12 : diagramme de séquence du triangulation.	67
Figure 13 : diagramme de séquence de détection d'interférences.	68
Figure 14 : diagramme de séquence de choix des outils.	69
Figure 15 : diagramme de séquence de génération du trajet.	70

Chapitre 6 :

Figure 1 : Fenêtre principale.	71
Figure 2 : Rubrique usinage par régions.	72
Figure 3 : fenêtre de triangulation des surfaces.	73
Figure 4 : Paramètres de triangulations.	74
Figure 5 : schéma de calcul de m.	74
Figure 6 : Schéma d'optimisation de m et n.	75
Figure 7 : Schéma de création des sommets.	77
Figure 8 : Schéma de création des segments.	78
Figure 9 : Création des triangles.	78
Figure 10 : subdivision avec la triangulation uniforme.	80
Figure 11 : fenêtre de modification les paramètres de dessin.	80
Figure 12 : fenêtre d'affichage des résultats.	81
Figure 13 : Fenêtre de Localisation des régions.	82
Figure 14 : Fenêtre des rayons d'outils théoriques.	82
Figure 15 : Fenêtre de calcul des interférences.	83
Figure 16 : fenêtre visualisation l'angle des triangles.	85

Figure 17 : Fenêtre de choix de la méthode d'usinage.	86
Figure 18 : Fenêtre de choix des outils.	87
Figure 19 : fenêtre de mode de choix des outils.	89
Figure 20 : Fenêtre de génération du trajet d'usinage.	90
Chapitre 7 :	
Figure 1 : Scénario de triangulation et de division de la surface.	91
Figure 2 : surface ondulée.	92
Figure 3 : Différentes parties de la triangulation initiale.	93
Figure 4 : Différentes parties de la triangulation finale.	94
Figure 5 : localisation globale des régions par sommets.	94
Figure 6 : Triangles voisines.	95
Figure 7 : localisation détaillée des régions.	95
Figure 8 : Scénario de détection des interférences.	96
Figure 9 : divisions de surface en zones.	97
Figure 10 : teste des interférences et correction d'outils.	97
Figure 11 : rayons d'outil de chaque région avant la correction.	97
Figure 12 : sommets sources d'interférence.	98
Figure 13 : rayons d'outils après la correction.	98
Figure 14 : Scénario de détermination de stratégie d'usinage de chaque région.	99
Figure 15 : position de chaque triangle.	100
Figure 16 : stratégie d'usinage de chaque région.	100
Figure 17 : Scénario de Choix des outils d'usinage des régions.	101
Figure 18 : Scénario de génération de trajet d'usinage.	102
Figure 19 : points d'intersection.	103
Figure 20 : contours.	103
Figure 21 : Trajet de centre d'outil.	104
Figure 22 : Trajet de point extrémité d'outil.	104
Figure 23 : Trajet d'usinage complet.	104
Figure 24 : Simulation de trajet d'outil dans un plan de la surface.	104
Figure 25 : tableau des rayons d'outils dans le mode d'un outil pour chaque région.	105
Figure 26 : résultats de mode d'un outil pour chaque région.	106
Figure 27 : tableau des rayons d'outils dans le mode d'un outil pour chaque type région d'une surface.	106
Figure 28 : résultats de mode d'un outil pour chaque type de région d'une surface.	107
Figure 29 : Génération du programme « G-Code ».	108

REFERENCES BIBLIOGRAPHIQUES

- [1] : Dr. C.-K. Shene "CS3621 Introduction to Computing with Geometry Notes "Michigan Technological University.
- [2] : A.SARIRETE et Y.SAHMI, « Automatisation de l'Usinage des Surfaces Gauches sur des Fraiseuses à Commande Numérique » Mémoire de fin d'étude de U.S.T.H.B
- [3] : B.AGEUNINI et H.MESSSAOUDI « simulation et vérification de l'opération d'ébauchage des surfaces gauches sur des fraiseuses à commande numérique à 3 axes » Mémoire fin d'étude université de blida.
- [4] : BEY Mohamed « modélisation des courbes et des surfaces » Rapport de recherche Avril 2000 CDTA.
- [5] : « *Introduction au fraisage* », rapport de CDTA.
- [6] : Cours de fabrication. « *Usinage par enlèvement de copeaux* ».
- [7] : « *Commande numérique par ordinateur* », 1^{er} année G.M.P 2003/2004.
- [8] : « cours -CFAO - machine.doc » documentation de CDTA.
- [9] : E.Duc E. Lefur « *Machine outils à commande numérique structure modélisation et réglage* » préparation à l'agrégation de génie mécanique 16 Septembre 1997.
- [10] : Bernard Méry. « *Machines à commande numérique* ».
- [11] : « *Manuel de programmation volume 1* », 1020/1040/1060.
- [12] : Roland Maranzana. « *Fabrication Assistée Par Ordinateur*». École de technologie supérieure . Génie de la production automatisée.
- [13] : «Automatisation de l'opération d'ébauchage des surfaces sur des fraiseuses à commande numérique à 3axes» PFE 2005 Université de Blida – réaliser par djamale chabane et lyes mouterfi.
- [14] : Document sur le CFAO « présentation power point de CFAO.ppt » de CDTA.
- [15] : D.C. Anderson C.G Jensen, "A review of numerically controlled methods for finish sculptured surface machining".; School of Mechanical Engineering, Brigham Young University.
- [16] : Techno-Science.net (site web).
- [17] : Rezzak Samia, Taibi Hayette, "Méthode Z-Constant pour l'Usinage des Surfaces Gauches sur des Fraiseuses à Commande Numérique à 3 axes", Mémoire de Fin d'Etudes, Université de Blida, Algérie, 2003.
- [18] : D. Misra, V. Sundararajan, P. K. Wright, « Zig-Zag Tool Path Generation for Sculptured Surface Finishing », Workshop in CAD/CAM, 2003.

[19] : rapport recherche Novembre 2004 « Automatisation de la sélection de la stratégie et de la direction d'usinage pour la finition des surfaces gauches » Bey Mohamed, CDTA division productique et robotique équipe CFAO.

[20] : Thèse de doctorat « Usinage de formes gauches contribution à l'amélioration de qualité des trajectoires d'usinage » de Emmanuel Duc, Ecole normale supérieure de CACHAN.

[21] : H. Nebbat, A. Chourar, « Simulation et vérification des programmes d'usinage des surfaces gauches sur des fraiseuses à commande numérique à 3 axes », PFE, Université Saad Dahlab, Blida, 2004.

[22] : C. Tournier, « Evaluation des Modes de Génération des Trajets Outil pour l'Usinage de Formes Gauches ». Mémoire de Recherche. École Normale Supérieure de Cachan, Juillet 1996.

[23] : A. Cherfi, « Conception et Développement d'une Application d'Optimisation du Trajet d'Usinage par la méthode Z-constant », PFE, Université Saad Dahlab, Blida, 2006.

[24] : S. Boudjouad, N. Tafat-Bouزيد, « Méthodes des plans parallèles pour l'usinage des surfaces gauches sur des fraiseuses à commande numérique à 3 axes », PFE, Université USTHB, 2004.

[25] :P. Martinon. « Résolution Numérique de Problèmes de Contrôle Optimal par une Méthode Homotypique Simplicité ». Thèse de Doctorat, 2005.

[26] :B. Salvatore, « Réseau National de Ressources des Structures Métalliques ».