Chapitre 1

Introduction

1.1 Reconstruction 3D

L'objectif de la reconstruction tridimensionnelle est de déterminer un modèle géométrique à partir d'un nuage de points. Son développement est de plus en plus important, et ses applications sont nombreuses :

- archivage ;
- conception de carrosserie ;
- prototypage rapide ;
- animation en informatique graphique ;
- etc...

La reconstruction est favorisée par les progrès récents réalisés dans le domaine des systèmes d'acquisition de données 3D à l'aide d'outils non-mécaniques (stéréovision, projection lumineuse).

La reconstruction d'objets permet de passer du modèle physique (maquette, prototype), préalablement numérisé, à un modèle mathématique.

Il se pose alors les problèmes du choix et du calcul du modèle. A ce niveau, nous sommes confrontés à plusieurs contraintes, dont la plus importante est de créer un modèle accepté directement par les systèmes de CAO. Pour cela, il faut que les carreaux de surface aient des formes rectangulaires, car ils sont définis par un produit tensoriel (surface de *Bézier* ou *B-Spline*).

Cela permet, éventuellement, d'intégrer ensuite le modèle surfacique calculé dans un autre modèle, créé par un processus interactif, à l'aide d'un logiciel de CAO par exemple.

1.2 Méthodes de reconstruction

Des méthodes intéressantes de reconstruction ont été développées :

- basée sur la constitution de maillage triangulaire [HDD+92]

– basée sur la subdivision de carreaux [CS94]

Mais ces méthodes ne procurent pas des modèles utilisables par les sytèmes de CAO. Ce sont soit des carreaux triangulaires, soit des modèles possédant un trop grand nombre de facettes ou de carreaux le long des lignes de rupture, car les caractéristiques surfaciques ne sont pas prises en compte.

Au LIGIM, une méthode de reconstruction a été mise au point, dans le cadre du projet ESPRIT 3DSCAN [IPSV94]. Nous allons présenter les grandes lignes de ce travail [Fig 1.1].

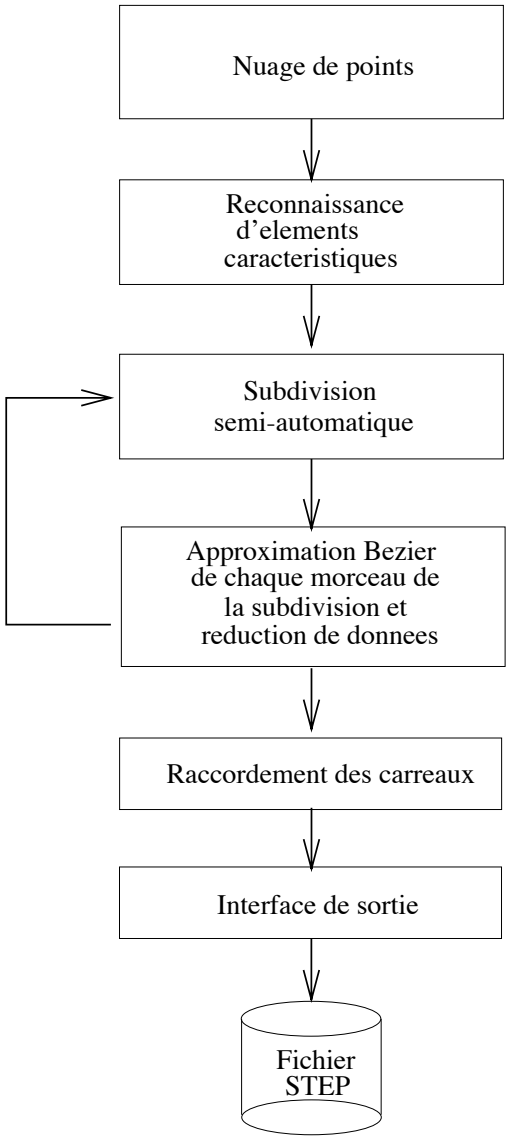

FIG. 1.1 - *Méthode d'approximation par des carreaux de Bézier*

La stratégie de modélisation est basée sur le partitionnement des données initiales en zones homogènes, en fonction des caractéristiques géométriques, telles que coins et arêtes vives. Ensuite, on essaie de trouver, pour chaque sous-carreau, le degré minimum compte tenu de la précision souhaitée. Enfin, on raccorde tous les sous-carreaux, ce qui peut poser quelques problèmes puisqu'ils peuvent être de différents degrés.

La méthode de modélisation utilisée est une approche locale basée sur l'approximation par des surfaces de Bézier.

Cependant, la subdivision de la surface est déterminée manuellement, les contours des carreaux sont fixés par l'utilisateur, qui s'appuie sur les résultats de la segmentation, produits par deux processus complémentaires :

- la croissance de régions
- la détection de contours

Cette segmentation produit des sous-nuages de points non rectangulaires. De plus, les frontières détectées ne sont pas toujours fermées et le bruit peut provoquer la détection de fausses arêtes.

Pour rendre la reconstruction automatique, il faut donc éliminer cette étape manuelle de découpage. C'est à ce niveau qu'intervient l'étude de DEA.

Nous proposons une méthodologie de reconstruction automatique de surfaces qui prenne en compte les caractéristiques géométriques pour partitionner le nuage de points en zones homogènes.

Il s'agit ensuite de former les sous-carreaux, c'est à dire déterminer une structure de carreaux rectangulaires pour se conformer aux contraintes des logiciels de CAO.

Après un paramétrage des points en relation avec chaque sous-carreau, ceux-ci sont approximés par une surface biparamétrique, et enfin, les carreaux sont joints de manière continue pour former la surface totale.

Cette méthode permet donc de reconstruire rapidement des surfaces complexes par des carreaux compatibles avec les systèmes de CFAO.

1.3 Problèmes liés à la reconstruction de surface sur un nuage de points non organisé

Dans le cas d'un nuage de points répartis sur une grille rectangulaire, le contour est parfaitement défini. Le paramétrage est alors facile à déterminer [Annexe B].

Dans notre cas, à l'issue du processus de segmentation, nous obtenons des sous-nuages, non rectangulaires.

Ceux-ci ne peuvent pas être directement approximés par des surfaces biparamétriques. En effet, ces surfaces imposent un paramétrage selon deux directions u et v.

Cela implique que chaque zone soit définie par un contour fermé composé de quatre courbes (un carreau).

On évite le plus possible d'obtenir des formes dégénérées en triangles ou biangles, ce qui n'est pas toujours possible selon les caractéristiques de la surface. On essaie de privilégier les formes quadrangulaires.

Cependant, la forme de chaque zone est quelconque, et il est quelquefois difficile de trouver une orientation dominante dans une forme très accidentée.

Certains critères ou caractéristiques sur le contour fermé quelconque, peuvent être étudiés pour choisir un contour acceptable :

- la courbure gaussienne

– la polygonalisation

– le squelette

– les boîtes englobantes

Il est donc nécessaire de choisir des directions u et v qui s'appuient sur les quatre frontières, pour effectuer un reparamétrage des points à l'intérieur du carreau.

Ensuite, ces points affectés d'un nouveau paramètre, sont lissés par un carreau de *Bézier* ou *B-Spline* :

– le modèle de *Bézier* permet une représentation simple des surfaces et surtout autorise une grande interactivité lors de la construction

– le modèle *B-Spline* possède les mêmes propriétés que le modèle *Bézier*, mais il permet en plus de prendre en compte la répartition des données, au détriment d'algorithmes plus complexes et de temps de calcul plus élevés [Annexe A]

Nous avons choisi d'utiliser le modèle B-Spline, car c'est un modèle plus général qui englobe le modèle Bézier.

Dans la suite de ce mémoire, nous développons les méthodes que nous avons utilisées pour choisir automatiquement un contour «acceptable» ainsi qu'un paramétrage adapté à ce contour.

Chapitre 2

Détermination du contour du carreau

2.1 Problématique

Pour approximer un nuage de points par une surface, la phase de paramétrage doit pouvoir affecter deux paramètres (u_i, v_j) à chaque point P_{ij}.

Si le réseau de points est proche d'une forme rectangulaire, les directions affectées aux paramètres u et v sont aisées à définir [Fig 2.1].

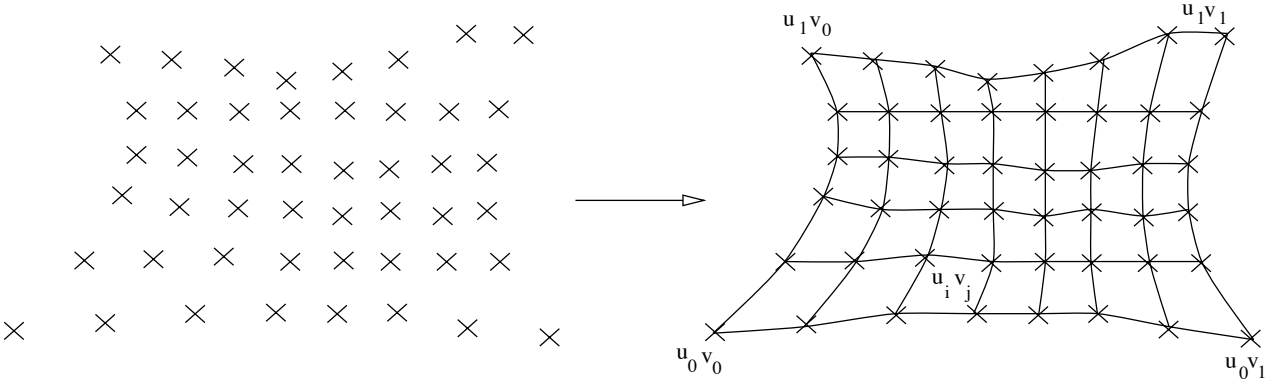

FIG. 2.1 - *Paramétrage d'une surface rectangulaire*

Quand le nuage de points n'est pas organisé en une forme proche d'un rectangle, il est difficile de trouver deux directions privilégiées. Il est donc nécessaire de définir quatre bords dans le carreau pour ensuite fixer un paramétrage [Fig 2.2].

Notre étude porte donc sur le choix d'un contour «acceptable» à partir de zones non-rectangulaires issues d'une phase de segmentation.

Cette segmentation de la surface peut se faire selon plusieurs critères :

- reconnaissance de surfaces planes
- extraction de formes géométriques connues (cônes,sphères,cylindres)
- étude fine de la courbure

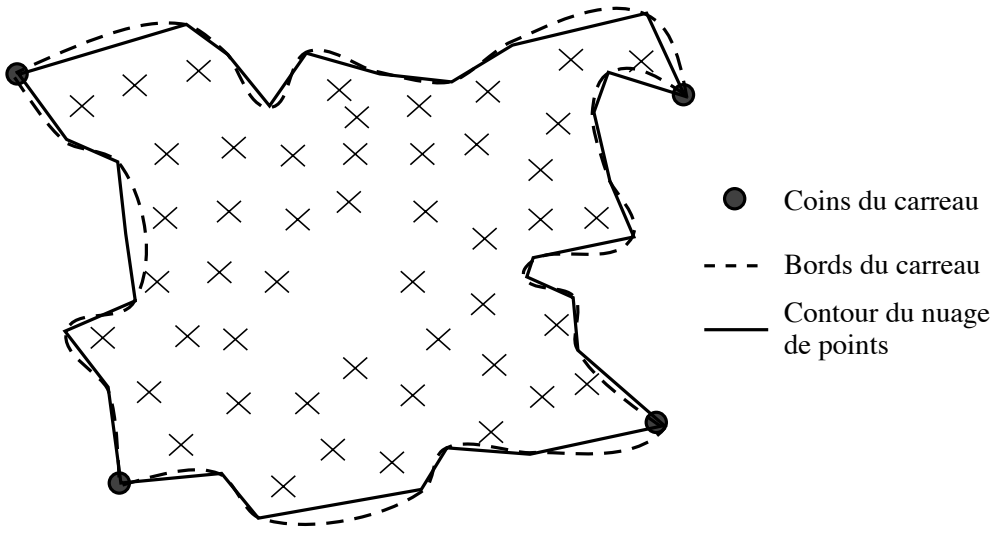

FIG. 2.2 - *Détermination des quatre bords du carreau*

Les résultats produits sont cependant sensibles au bruit.

[HJ87] proposent donc une méthode basée sur une simple décomposition en surfaces concaves, convexes et planes, en utilisant les points de la surface et la normale.

Une phase de segmentation par extension de région à partir d'un noyau est suivie par une classification en fonction des variations de courbure, et enfin une fusion des zones adjacentes ayant les mêmes propriétés.

[BJ88], quant à eux, décomposent la surface en régions élémentaires déterminées par segmentation/fusion des points en fonction du signe de la courbure gaussienne K et de la courbure moyenne H. Ils définissent ainsi huit types de surfaces [Tab 2.1].

	$K > 0$	$K = 0$	$K < 0$
$H < 0$	pic	cylindre convexe	col convexe
$H = 0$	indéterminé	plat	surface minimale
$H > 0$	vallée	cylindre concave	col concave

TAB. 2.1 - *Types de surfaces*

Ainsi le mode de segmentation des nuages de points fournit des zones aux contours complexes [Fig 2.3].

Il s'agit alors de trouver un carreau rectangulaire sur les frontières quelconques déterminées par un suivi de contour [MM94].

2.2 Méthodes existantes

Dans ce paragraphe, nous faisons une présentation des principales méthodes existantes, que l'on peut appliquer à notre problème, qui est l'extraction de quatre courbes à partir d'une liste de points ordonnés formant un contour.

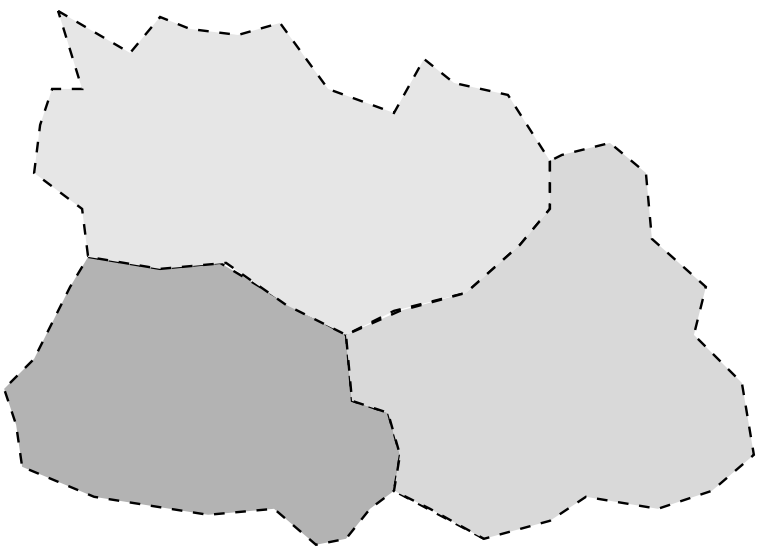

FIG. 2.3 - *Exemple de contours issus de la segmentation*

La plupart du temps, cela revient à extraire des points qui caractérisent au mieux le contour, ou à trouver une ou plusieurs directions privilégiées. A partir de ces informations, nous déterminons les quatres points qui se situent aux coins du carreau, puis nous effectuons un lissage de chaque section de contour par une courbe Bézier ou B-spline, pour obtenir les bords du carreau, avec une forme quasi rectangulaire.

Nous pouvons travailler sur la projection 2D, puisqu'il s'agit d'images de profondeur.

2.2.1 Approximation polygonale

Le but de la méthode est de sélectionner un nombre fixé de points qui caractérisent au mieux la forme donnée [Fig 2.4].

Nous recherchons donc un ensemble de sommets significatifs (ou privilégiés) $V_j,\ j = 1,2,\cdots,k$ parmi les points $P_i,\ i = 1,2,\cdots,n$ du contour original, où k est le nombre de sommets du polygone extrait, ceci pour une distance d'approximation D_s fixée arbitrairement au départ et ajustée tout au long de l'algorithme.

Il existe deux types d'approximation :

- optimiser le nombre de côtés pour une erreur donnée ;
- minimiser l'erreur pour un nombre de côtés fixé.

Nous nous intéressons à la seconde approche, qui se déduit de la première en faisant varier le seuil jusqu'à obtenir les sommets les plus significatifs du contour. Nous l'utilisons pour trouver un quadrilatère dont les sommets seront ceux du carreau. Pour obtenir les quatre courbes frontières, nous réalisons ensuite un lissage des morceaux de lignes brisées entre chacun des quatre sommets privilégiés.

Les techniques, assez semblables, que proposent [MM94] et [PV94], sont basées sur la minimisation de l'erreur d'approximation entre un point P_k et un segment

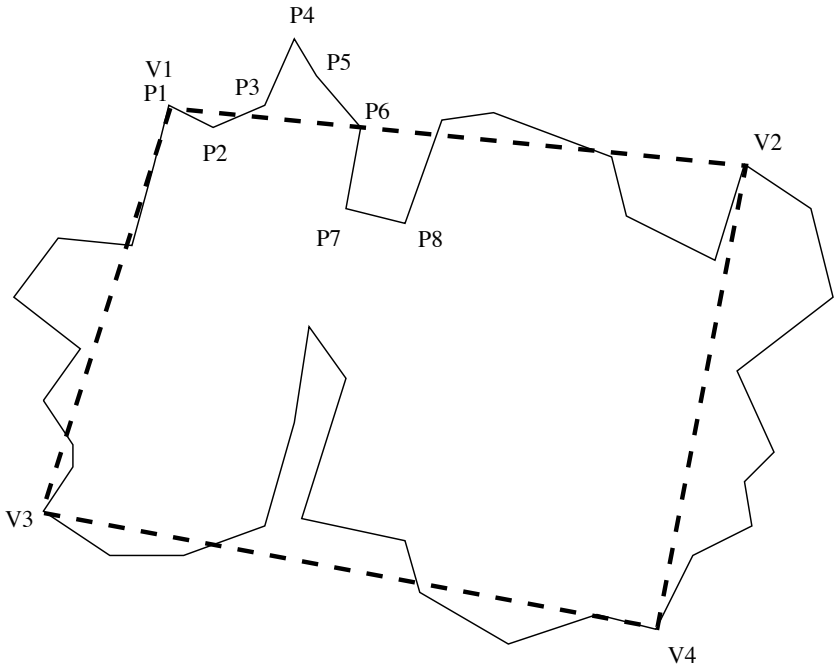

FIG. 2.4 - *Approximation polygonale d'un contour*

$[P_i, P_j]$, qui est définie comme la distance euclidienne de ce point au plus proche point du segment [Fig 2.5]. Cette erreur e_{ij}^k se formule ainsi :

$$e_{ij}^k = \begin{cases} d(P_i, P_k), & si\ \theta_{jk}^i \geq \pi/2 \quad (point P_2) \\ d(P_j, P_k), & si\ \theta_{ki}^j \geq \pi/2 \quad (point P_4) \\ d(P_i, P_k) \sin \theta_{jk}^i & sinon \quad (point P_3) \end{cases}$$

où θ_{ab}^c est l'angle $P_a\widehat{P_c}P_b$ dans le triangle formé par les trois points P_a, P_b et P_c.

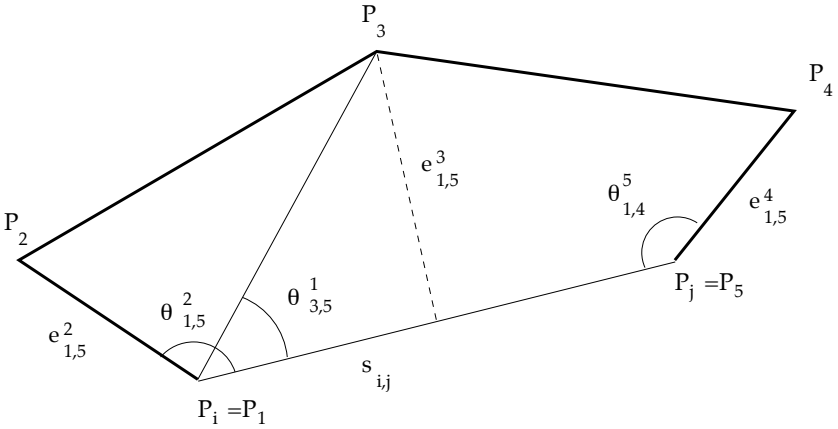

FIG. 2.5 - *Erreur d'approximation*

La première phase est la détermination du point de départ, qui constitue le premier sommet.

Dans une forme régulière (triangle, rectangle, losange...), les sommets sont les points qui maximisent la somme des distances au centre de gravité.

Ici, par extension, nous choisissons le point le plus éloigné du point central.

Soit $T_i = (x_i, y_i),\ i = 1, 2, \cdots, n$ les points du contour, alors le point central est défini par $C = (x_c, y_c)$ tel que $C = \frac{1}{n}\sum_{i=1}^{n} T_i$.

Le point T_k tel que $d(T_k, C) > d(Ti, C)$ pour tout $i \neq k$, où d désigne la distance euclidienne entre deux points, est choisit comme premier sommet. On transpose alors les points pour avoir $P_1 = T_k$:

$$P_i = \begin{cases} T_{i+k-1}, & si\ i+k-1 \leq n \\ T_{i+k-1-n}, & si\ i+k-1-n > n \end{cases}$$

On applique ensuite la méthode d'extraction de points V_j, appelés sommets privilégiés du contour, sur la liste ordonnée P_i.

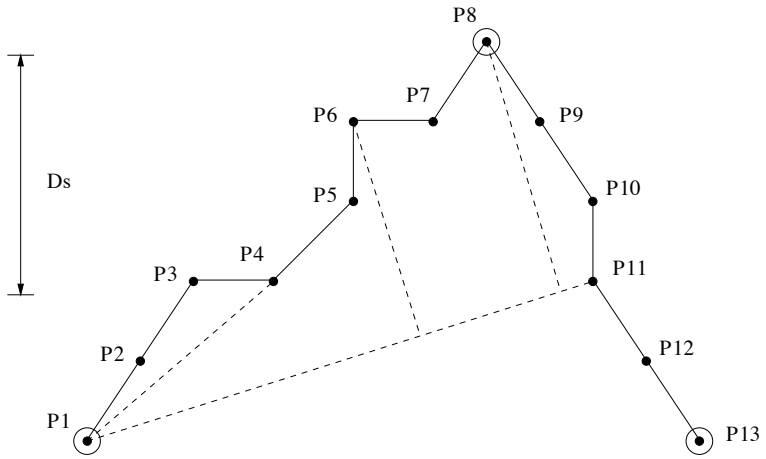

FIG. 2.6 - *Extraction des sommets privilégiés d'un contour*

On considère le contour composé des points P_1 à P_{13} (avec $V_1 = P_1$) [Fig 2.6]. On génère le segment $s_{1,3}$ et l'on regarde si l'erreur $e^2_{1,3}$ est supérieure à D_s. Dans l'exemple $e^2_{1,3} < D_s$, on génère donc $s_{1,4}$ et l'on calcule les erreurs de tous les points entre P_1 et P_4. Les erreurs $e^2_{1,4}$ et $e^3_{1,4}$ sont elles aussi inférieures à D_s, l'algorithme produit alors le segment $s_{1,5}$. Quand on atteint le segment $s_{1,11}$ les erreurs $e^6_{1,11}$ et $e^8_{1,11}$ sont supérieures à D_s, comme on a $e^8_{1,11} > e^6_{1,11}$, on choisit P_8 comme point de rupture et $V_2 = P_8$. On génère alors les segments de $s_{8,10}$ jusqu'à $s_{8,13}$ et l'on cherche comme précédemment les erreurs supérieures au seuil. Le dernier point du polygone est le dernier point du contour (ici $V_3 = P_{13}$).

On a donc caractérisé les treize points du contour par trois sommets significatifs P_1, P_8 et P_{13}.

Pour obtenir un nombre fixé de sommets k, la méthode ne fournit pas de solution optimale mais une solution satisfaisante dans la plupart des cas. Nous choisissons $k = 5$ pour obtenir un quadrilatère (le premier et le dernier point étant identiques).

L'algorithme consiste à choisir une valeur D_s et à extraire les sommets significatifs $V_j,\ j = 1, 2, ..., n$ à partir des points du contour $P_i,\ i = 1, 2, ..., n$.

A l'itération t:

1. Si l'on trouve un nombre m de points égal à k, on s'arrête.
2. Si $k < m$, on augmente le seuil $D_{s,t+1} = D_{s,t} + R_t$ et on relance l'extraction sur la nouvelle liste de sommets significatifs $V_j,\ j = 1, 2, ..., m$.
3. Si $k > m$, on diminue la correction $R_{t+1} = \frac{R_t}{2}$ et le nouveau seuil $D_{s,t+1} = D_{s,t} - R_{t+1}$. On relance alors l'extraction sur l'ancienne liste de sommets significatifs .

Une des difficultés est le choix de la distance seuil de départ. Si R_0 est trop petit, un grand nombre d'itération sera nécessaire, et s'il est trop grand, la première itération prendra beaucoup de temps. On a constaté qu'une bonne valeur pour R_0 est $\frac{\beta n}{d_{max}}$ où $d_{max} = d(C, P_1)$, n est le nombre de points du contour initial et β un paramètre entre 0 et 1.

Il faut aussi prévoir un test d'arrêt dans le cas d'oscillations autour de la solution, lorsque l'on atteint un indice de correction R trop petit.

Ainsi,cette méthode, même si elle ne propose pas de solution optimale, permet une bonne approximation d'un contour par un polygone, dont les sommets nous servirons à construire les quatre courbes frontières du carreau.

2.2.2 Approximation par des boîtes

Boîtes englobantes

Le but de cette méthode est d'englober le contour constitué des points $P_i,\ i = 1, 2, ..., n$, dans une boîte. Pour cela, on cherche le rectangle englobant d'aire minimale ou de périmètre minimal.

[MS88] recense un ensemble d'algorithmes pour mettre des objets dans une boîte, en choisissant de faire tourner l'objet plutôt que la boîte [Fig 2.7].

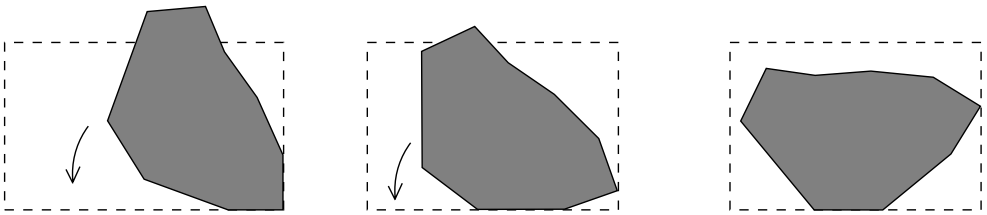

FIG. 2.7 - *Méthode des boîtes englobantes*

Pour les polygones, la méthode est divisée en deux parties :

1. une partie de calcul, qui détermine si le polygone tient à l'intérieur de la boîte dans une configuration donnée
2. une partie de contrôle, qui décide des changements de configuration.

On travaille sur l'enveloppe convexe du polygone, ce qui simplifie l'algorithme et est justifié par le fait que si un polygone est contenu dans une boîte, alors son enveloppe convexe aussi, à cause de la convexité de la boîte. Cette méthode nécessite peu de données, uniquement les coordonnées du point le plus haut (x_t, y_t), le plus bas (x_b, y_b), le plus à droite (x_r, y_r), ainsi que le plus à gauche (x_l, y_l), appelés points actifs du polygone [Fig 2.8].

On fait tourner l'objet d'un angle θ arbitraire, suffisamment petit pour que les mêmes points restent en haut, bas, droite et gauche. Ces points vont devenir $x' = x\cos\theta - y\sin\theta$ et $y' = x\sin\theta + y\cos\theta$.

On calcule ensuite les dimensions de l'objet :

- sa largeur $w = (x_r - x_l)\cos\theta - (y_r - y_l)\sin\theta = a\cos\theta + b\sin\theta$
- sa hauteur $h = (x_t - x_b)\sin\theta + (y_t - y_b)\cos\theta = d\sin\theta + e\cos\theta$

En éliminant θ, on obtient :

$$h = \frac{(ad + be)w + (ae - bd)(a^2 + b^2 + w^2)^{1/2}}{a^2 + b^2}$$

On remplace ensuite w par w_{box} dans l'équation précédente et si h est plus petit ou égal à h_{box}, alors le polygone est contenu dans la boîte. L'algorithme est très simple et la solution peut être trouvée directement.

Il faut maintenant décider de la configuration à choisir, c'est à dire quelle orientation donner au polygone pour qu'il rentre dans la boîte.

On choisit l'orientation initiale du polygone, et on définit la region solution comme tous les angles entre 0 et le plus petit angle requis pour que l'un des points actifs soit remplacé par un nouveau point si le polygone subit une rotation dans le sens trigonométrique. Il s'agit du minimum des quatre angles représentés sur la figure 2.8.

On cherche ensuite si le polygone est contenu dans la boîte. Si c'est le cas, on regarde si la solution obtenue appartient à la region solution :

- si oui, alors le problème à été résolu
- si non, on fait subir une rotation au polygone de l'angle minimum trouvé précédemment

Ce processus est repété jusqu'à ce qu'une solution soit trouvée, ou que le polygone ait subit une rotation de plus de 180°, car aucune solution ne sera plus trouvée (à cause de la symétrie).

Cet algorithme permet d'obtenir une solution en fonction du nombre n de points du contour initial. La rotation de l'objet implique que chaque arête du polygone devienne une fois horizontale et une fois verticale. De plus, lorsque l'on cherche la nouvelle liste de points actifs, il suffit de considérer les successeurs immédiats des points actifs actuels.

Trouver la boîte d'aire minimum ne nécessite pas d'autre procédé, [FS75], puisqu'une arête du polygone est assurée d'être le long de sa boîte minimale. On fait tourner le polygone et on calcule à chaque fois la boîte et son aire. On note l'aire minimale et la configuration correspondante.

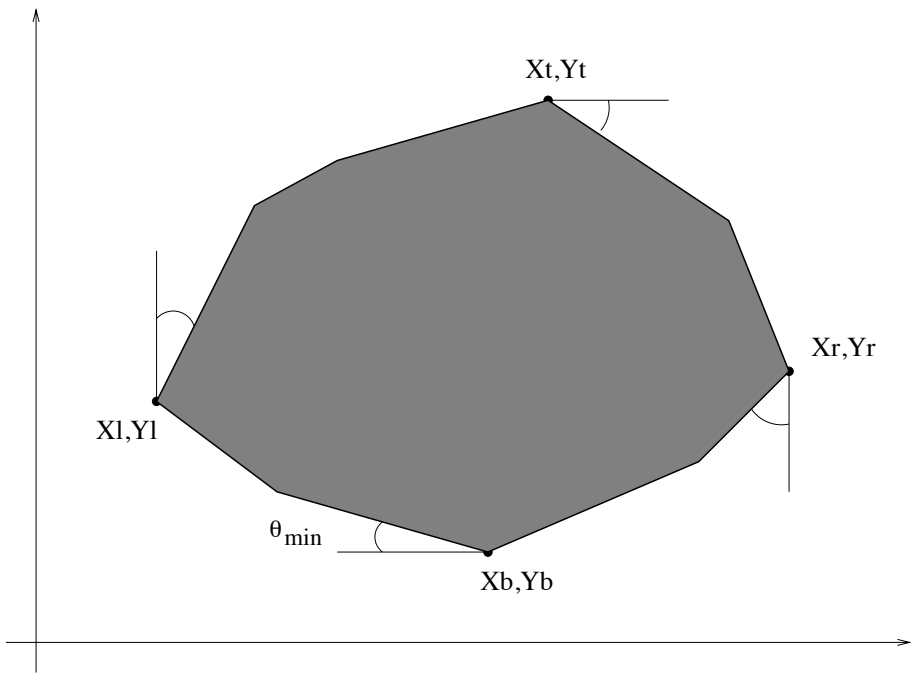

FIG. 2.8 - *Méthode pour trouver les points actifs*

La boîte de périmètre minimum peut être trouvé en remarquant que le périmètre est $P = 2(w(\theta)+h(\theta))$. On obtient un extremum quand $dP/d\theta = 0$, lié à la condition

$$\tan\theta = \frac{e+b}{d+a}$$

C'est un minimum lorsque $d^2P/d\theta^2 > 0$, et donc

$$(e+b)\sin\theta + (a+d)\cos\theta < 0$$

qui se simplifie en $1/\cos\theta < 0$.

On recherche une solution pour $\theta \geq 0$ et certainement pour $\theta \leq 90°$, car alors les points actifs auront changé. Ceci contredit $1/\cos\theta < 0$. Cela veut dire que la solution est donc atteinte pour l'un des angles où les points actifs sont modifiés. Il suffit alors de calculer les boîtes et les périmètres dans ces configurations et de noter le minimum.

La boîte ainsi obtenue forme le contour du carreau, chaque frontière de degré 1 étant délimitée par un côté de la boîte.

Rectangles orientés

L'objectif de cette méthode proposée par [MAN94], est d'approximer le polygone formant le contour par un rectangle. Ce dernier doit répondre à deux critères :

- son aire doit être égale à celle du polygone
- l'orientation de ses cotés doit être proche des directions des cotés du polygone

Pour cela, on minimise deux fonctions de coût f_1 et f_2.

$$f_1 = (\sum S_{pl} - a)^2 + (\sum S_{ps} - a)^2 + (\sum S_{nl} - b)^2 + (\sum S_{ns} - b)^2$$

et

$$f_2 = (\sum S_{pl}\alpha_{pl} - a\alpha_0)^2 + (\sum S_{ps}\alpha_{ps} - a\alpha_0)^2$$
$$+(\sum S_{nl}\alpha_{nl} - b(\frac{\pi}{2} - \alpha_0))^2 + (\sum S_{ns}\alpha_{ns} - b(\frac{\pi}{2} - \alpha_0))^2$$

Les notations sont celles représentées sur la figure 2.9.

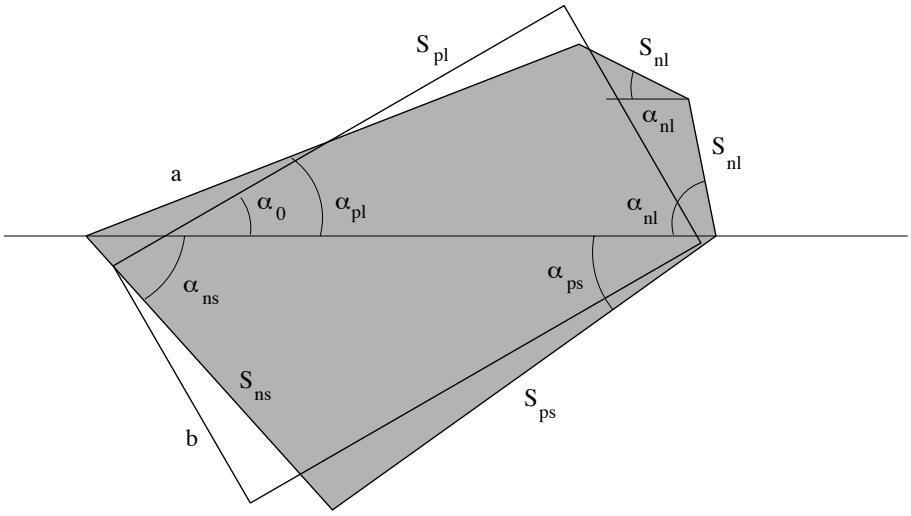

FIG. 2.9 - *Paramètres des fonctions de coûts*

- S_{pl} et S_{nl}, sont les longueurs des arêtes du polygone situées *au-dessus* de la diagonale et ayant une pente positive, respectivement négative

  S_{ps} et S_{ns}, sont les longueurs des arêtes du polygone situées *en-dessous* de la diagonale et ayant une pente positive, respectivement négative

- les $\alpha_{pl}, \alpha_{nl}, \alpha_{ps}, \alpha_{ns}$ désignent les angles que forment les arêtes correspondantes avec la diagonale

- α_0 est l'angle entre l'arête de longueur a du rectangle, avec la diagonale du polygone

- la diagonale est, quant à elle, choisie comme la ligne intérieure joignant les deux sommets les plus éloignés du polygone

- a est une dimension du rectangle et $b = A/a$, Aétant l'aire du polygone

La fonction de coût f_1 est suffisante pour définir la taille du rectangle, mais f_2 est nécessaire pour fixer l'orientation de ce dernier.

Cette méthode propose une bonne approximation des polygones par des rectangles en minimisant deux fonctions de coûts.

Elle peut aussi être appliquée au cas des triangles, qui sont préalablement transformés en quadrilatères par l'ajout d'un sommet arbitraire sur l'un des cotés.

Cependant, son application à notre problème peut s'avérer coûteuse, car le calcul de l'aire ou la détermination de la diagonale pour un polygone ayant un grand nombre de sommets, peut être d'un coût élevé.

De plus, les résultats présentés sont effectués sur des polygones simples (3 à 7 côtés), et le problème des polygones non-convexes n'a pas été abordé.

2.2.3 Approximation basée sur la courbure

Les points pour lesquels la courbe admet une forte courbure sont des points priviligiés pour caractériser la forme du contour [Fig 2.10].

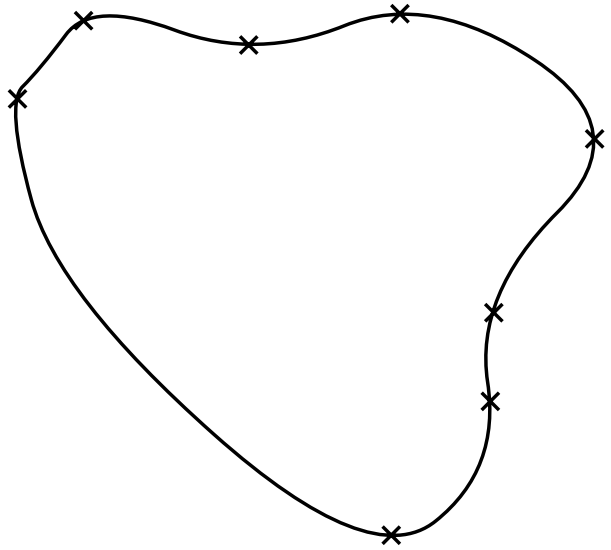

FIG. 2.10 - *Extrema locaux de la courbure*

[AD91] propose un algorithme d'approximation polygonale basé sur l'étude des points $(x(t), y(t))$ pour lesquels il y a un extremum de courbure. Une technique de «split-and-merge» est utilisée pour modifier la liste des sommets significatifs.

Pour éviter les problèmes introduits par de petites oscillations locales, on lisse le contour par un filtre Gaussien :

$$h(t,\omega) = \frac{1}{\sqrt{2\pi}\omega} exp[\frac{t^2}{2\omega^2}]$$

où ω est la longueur du filtre. Les points de la courbe lissée sont tels que :

$$X(t,\omega) = x(t) * h(t,\omega)$$
$$Y(t,\omega) = y(t) * h(t,\omega)$$

où * désigne l'opérateur de convolution.

La courbure est définie par :

$$K(t,\omega) = \frac{\dot{X}\ddot{Y} - \dot{Y}\ddot{X}}{(\dot{X}^2 + \dot{Y}^2)^{3/2}}$$

où $\dot{X}$ et $\ddot{X}$ désignent les dérivées première et seconde de X.
L'algorithme à appliquer à un contour fermé peut se résumer ainsi :

1. lisser le contour avec le filtre Gaussien présenté précédemment
2. déterminer la liste de points ayant un maximum positif et un minimum négatif de courbure
3. appliquer l'algorithme de «split-and-merge» avec comme points initiaux les points trouvés à l'étape 2

Le paramètre ω du filtre doit être fixé. Une valeur élevée enlèvera toutes les oscillations, alors qu'une faible valeur n'en enlèvera pas suffisamment.

Un algorithme de «split-and-merge» est proposé par [TYL94], il permet de déterminer les sommets significatifs du contour de l'objet à partir d'une liste initiale de points présélectionnés. Soit $P_m = \{B_1, B_2, ..., B_{N(m)}\}$ la liste des points à l'itération m.

S_{ij} est le j^{ieme} point du segment délimité par B_i, et B_{i+1} et D_{ij} est la distance de ce point au segment. T_{ij} est le j^{ieme} point du segment délimité par B_i, et B_{i+2} et E_{ij} est la distance de ce point au segment.

1. soit $P_0 = B_0, B_1, ..., B_n$, la liste des points du contour initial
2. pour i=1 à $N(m)+1$, phase de découpage, on teste tous les segments $[B_1, B_2]$
   - déterminer S_{ij} tel que $D_{ij} = max\{D_{i1}, D_{i2}, ..., D_{iK}\}$
   - si $D_{ij} >$ seuil de découpage, $P_{m+1} = P_{m+1} \cup \{S_{ij}\}$, S_{ij} est un sommet significatif
3. pour i=1 à $N(m)$, phase de fusion, on teste tous les segments $[B_i, B_{i+2}]$
   - déterminer T_{ij} tel que $E_{ij} = max\{E_{i1}, E_{i2}, ..., E_{iK}\}$
   - si $E_{ij} <$ seuil de découpage, $P_{m+1} = P_{m+1} \backslash \{T_{ij}\}$, on retire T_{ij} de la liste des sommets significatifs
4. si $P_m \neq P_{m-1}$ alors $P_{m+1} = P_m$, on relance le «split-and-merge»
5. sinon, P_m contient la liste des sommets significatifs

Cette méthode donne une solution optimale pour les paramètres fixés ω et les seuils du «split-and-merge». Mais, elle ne permet pas de fixer le nombre de sommets à extraire, il serait possible de le faire en corrigeant les deux valeurs de seuil (split et merge), mais il est difficile de combiner leurs effets.

Nous pouvons appliquer cette méthode à notre problème, qui est légèrement différent puisque l'on a un liste de points et non plus une courbe. On peut approximer le contour par une *B-Spline* puis faire l'étude de la courbure et appliquer l'algorithme de «split-and-merge».

2.2.4 Utilisation du squelette

Le but de cette méthode est de déterminer le *diagramme de Voronoi* d'un ensemble de points formant une courbe fermée. Le diagramme de Voronoi, appelé aussi axe médian ou squelette, d'une liste de points e_i est la liste de points P, tels que chaque élément soit équidistant de deux ou plusieurs points du contour, et que le cercle centré en P soit inscit dans la forme, sans contenir d'autres points du contour [Fig 2.11].

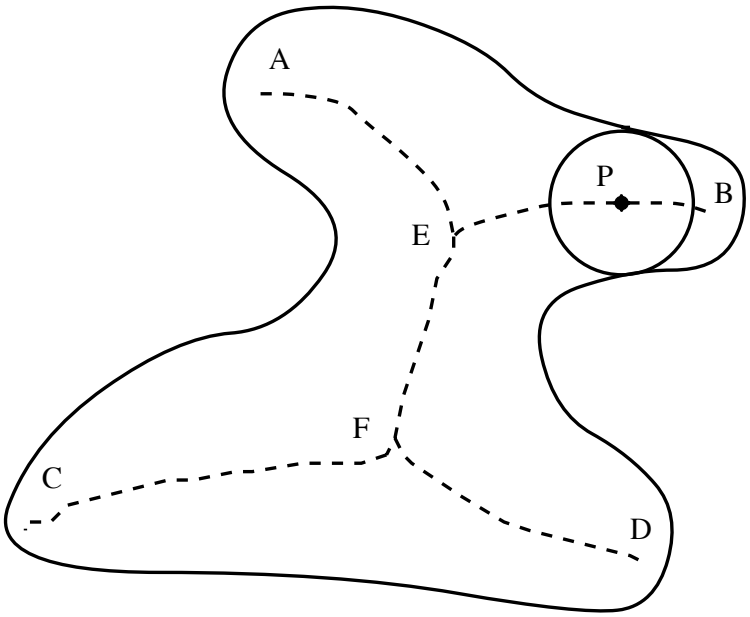

FIG. 2.11 - *Exemple de squelette d'une forme*

Les points P_i satisfont les conditions suivantes [CHO95]

1. P est équidistant de au-moins deux points de la courbe
2. P est à l'intersection des lignes ayant pour direction la normale en ces points
3. P est à l'intérieur du contour
4. le cercle de centre P n'intersecte pas le contour

Les deux premières conditions définissent P comme le centre d'un cercle, les deux suivantes s'assurent que P appartient réellement au squelette [Fig 2.12 et 2.13].

Les diagrammes de Voronoi possèdent des propriétés intéressantes, qui peuvent être utilisées dans l'étude de la forme d'un contour, et pour déterminer les points extrimités d'un carreau.

Le squelette est constitué de plusieurs segments. Chaque segment est limité par deux types de points :

– un point d'embranchement, quand le cercle associé est tangent à plus de trois points du contour (E et F)

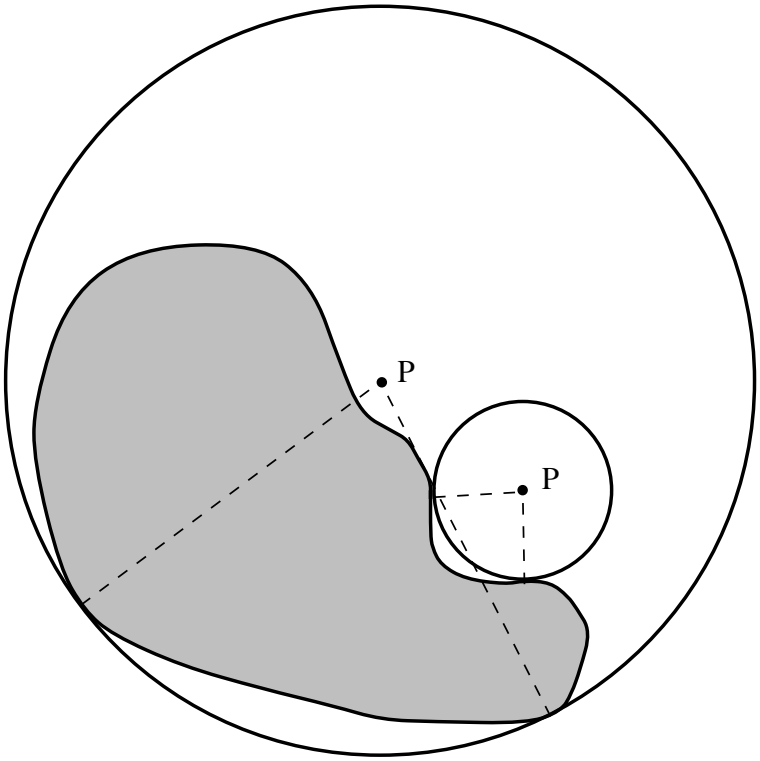

FIG. 2.12 - *Exemple de non respect de la condition 3*

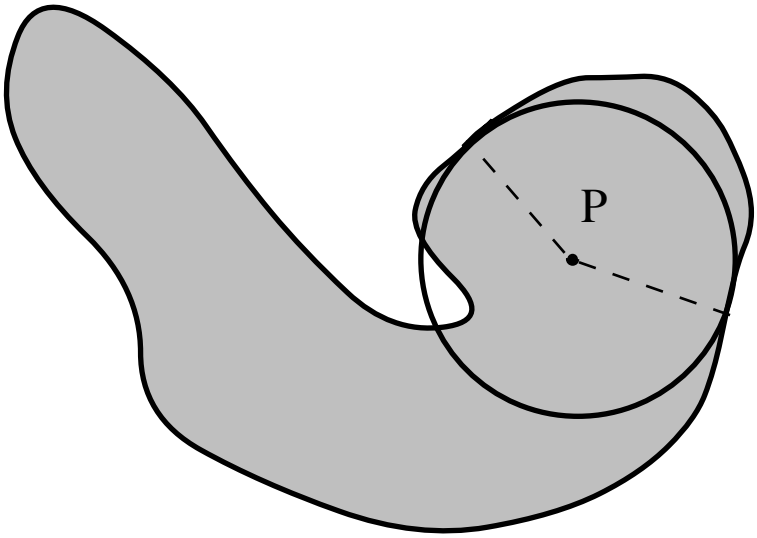

FIG. 2.13 - *Exemple de non respect de la condition 4*

– un point terminal, ou point extrême, on considère que le cercle n'est tangent qu'en un point, qui est la limite quand deux points de tangence se rejoignent (A,B,C et D)

Les autres points du segment sont ceux pour lesquels le cercle touche le contour en exactement deux points (P).

A chaque point des segments, il correspond donc deux sections du contour, une de chaque côté. De plus, chaque point terminal du diagramme correspond à un point où le contour admet un extremum local de la courbure.

Une autre propriété intéressante est la possibilité de mettre le diagramme sous forme d'arbre, les nœuds étant les points d'embranchement, les feuilles étant les points terminaux. [Fig 2.14]

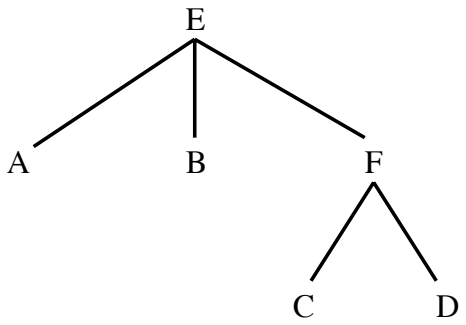

FIG. 2.14 - *Structure d'arbre de la figure 2.11*

Ceci permet de simplifier l'étude du squelette afin de déterminer les points délimitant le carreau, en structurant les données : extremum local de le courbure, correspondance entre les segments et les différentes sections du contour.

2.3 Résultats

Nous avons implémenté une méthode de polygonalisation du contour en deux étapes :

1. extraction de sommets significatifs
2. élimination des points les moins significatifs par fusion des arêtes

Dans la première phase, nous utilisons la méthode présentée au paragraphe 2.2.1, pour extraire k sommets significatifs. Nous choisissons $k = 9$, ce qui correspond à un octogone.

Dans la seconde étape, nous rejetons les sommets qui caractérisent le moins la forme, pour obtenir l sommets significatifs, avec $l = 5$, ce qui donne un quadrilatère (le premier et le dernier point sont identiques pour avoir un contour fermé).

A chaque itération, nous éliminons le sommet le moins significatif, c'est à dire celui dont la distance avec le segment entre le sommet précédent et le sommet suivant, est minimale. Sur la figure 2.15, ϵ^2_{13} (définie au paragraphe 2.2.1) est la plus faible, donc V_2 est retiré de la liste.

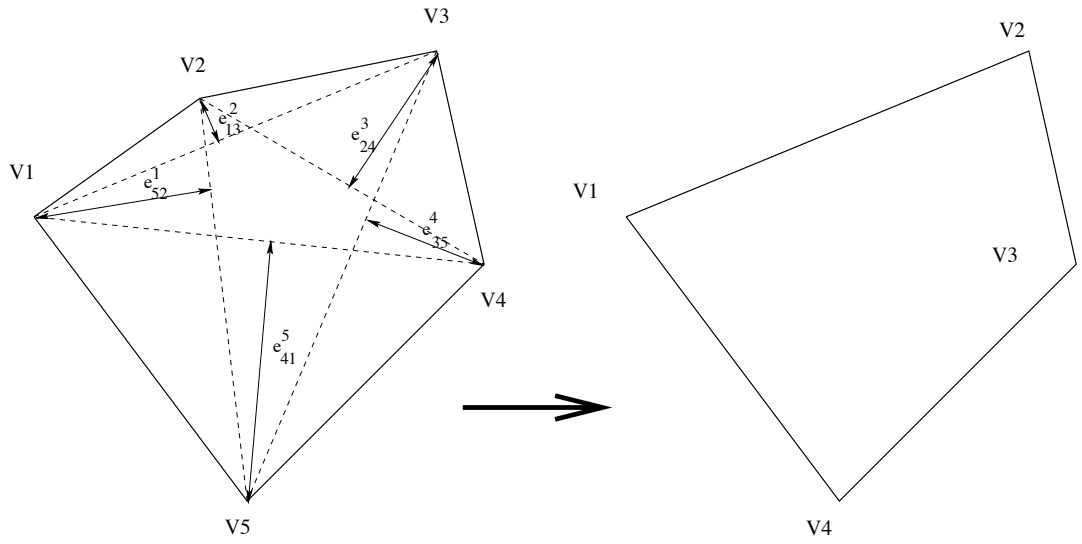

FIG. 2.15 - *Elimination d'un sommet*

L'algorithme proposé est le suivant :

- Soit V_k la liste des k sommets privilégiés du contour, $n = k = 9$ et $l = 5$
- Répéter $(k - l)$ fois
  - déterminer le point $\{V_j\}$ tel que $e^j_{j-1,j+1} = min\{e^i_{i-1,i+1}\}$
  - faire $V_{n-1} = V_n \backslash \{V_j\}$
  - $n = n - 1$
- V_5 contient les cinq sommets privilégiés.

Nous avons donc segmenté le contour en quatre sections, chacune étant ensuite approximée par une courbe B-spline, qui formera une des frontières du carreau. Cela permet d' obtenir une frontière plus proche du contour initial.

Nous présentons les résultats issus de la phase d'extraction des sommets les plus significatifs.

Nous remarquons que la forme du contour initial est conservée dans son ensemble (Figure 2.16), la méthode ne tenant pas compte des petites oscillations (Fig 2.17), comme cela se passe avec une approximation basée sur la courbure.

Cependant, cette méthode présente plusieurs limites. D'une part, elle n'offre pas une segmentation du contour qui permette de minimiser le degré de la courbe B-spline. D'autre part, si elle offre des résultats satisfaisants pour des contours convexes ou quasi convexes, les contours fortement non convexes posent des problèmes. Ainsi, la détection des sommets significatifs sur le contour de la figure 2.18, ne permet pas de réaliser un paramétrage correct.

Les méthodes présentées n'étudient pas le cas de la non convexité, car elles ont été développées dans le cadre de la reconnaissance de formes (principalement sur l'occlusion d'objets). A ce stade, il est difficile de définir un critère de convexité, pour déterminer jusqu'à quel point l'extraction des sommets significatifs est correcte. Il faut attendre la phase de reconstruction pour pouvoir calculer l'erreur de

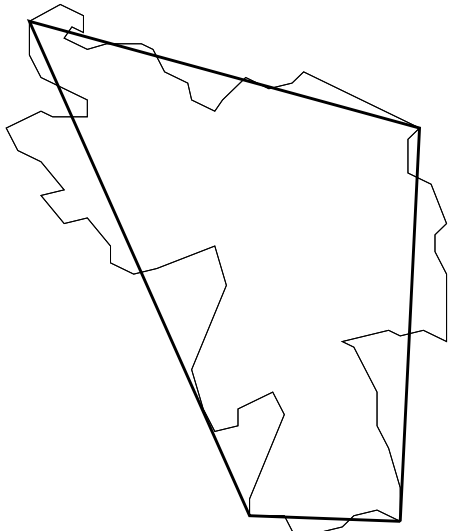

FIG. 2.16 - *Exemple 1 de polygonalisation de contour*

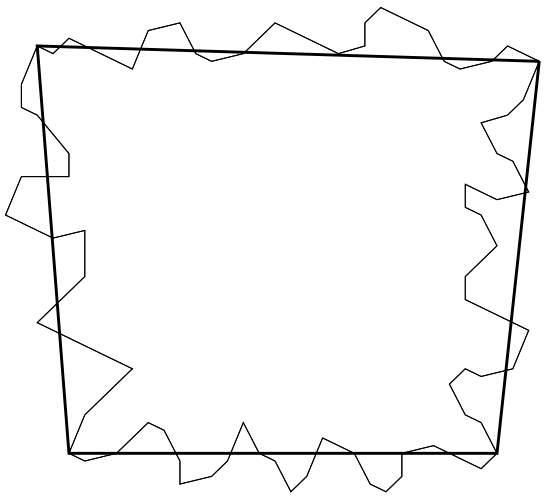

FIG. 2.17 - *Exemple 2 de polygonalisation de contour présentant des oscillations*

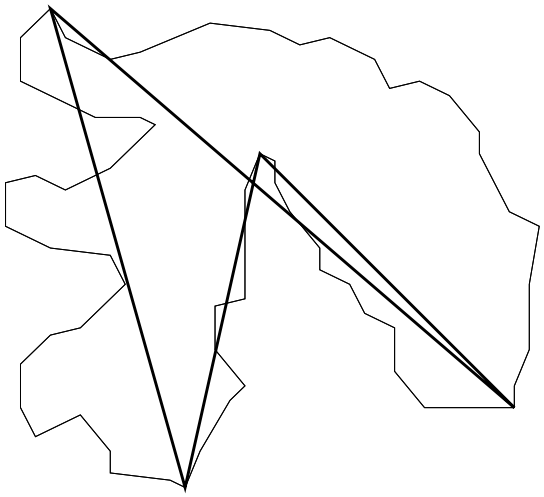

FIG. 2.18 - *Exemple de polygonalisation de contour non convexe*

l'approximation, et alors décider de la subdivision du nuage en deux ou plusieurs sous-nuages convexes. En dernier recours, un utilisateur expert a la possibilité de définir lui même le contour.

Chapitre 3

Paramétrage d'un nuage de points

3.1 Problématique du paramétrage

La phase précédente nous a permis d'obtenir des carreaux de surface parfaitements définis par quatre contours [Fig 3.1].

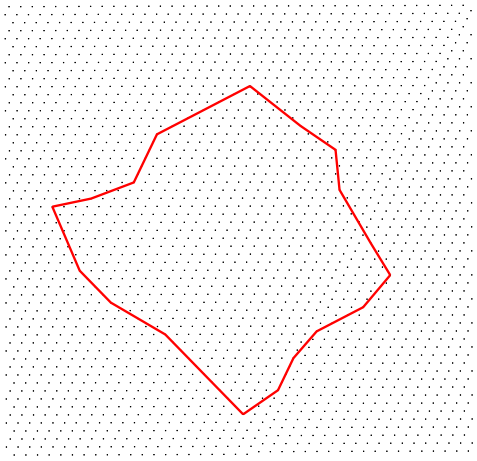

FIG. 3.1 - *Matrice de points initiaux et courbes frontières*

Pour pouvoir approximer ces carreaux, il faut donner des valeurs de paramètre (u_i, v_j) à chaque point P_{ij} pour $i = 0, 1, ..., m$ et $j = 0, 1, ..., n$, avant de les lisser par une surface de type B-spline.

Il existe plusieurs méthodes de paramétrage d'un nuage de points en vue d'une approximation paramétrique.

Nous allons présenter les principales méthodes utilisées pour le paramétrage des

courbes :

- répartition uniforme du paramètre
- répartition proportionnelle à la distance entre les points
- répartition voisine de l'évolution de l'abscisse curviligne
- projection

Nous verrons ensuite comment il est possible d'appliquer ces méthodes au cas des surfaces, nous développerons enfin la méthode utilisée et les résultats obtenus.

3.2 Paramétrage des courbes

Il s'agit ici de paramétrer un ensemble de $(p+1)$ points P_i pour $i = 0, 1, ..., p$, pour les approximer par une courbe paramétrique.

La plus simple approche est de donner une valeur de paramètre uniforme, qui s'écrit :

$$u_i = \frac{i}{p}, \qquad i \in \{0, 1, ..., p\}$$

et qui produit des valeurs sur $u \in [0, 1]$.

On préfère souvent une répartition proportionnelle à la distance entre les points P_i, qui s'exprime par [LEO91] :

$$\begin{aligned} u_0 &= 0 \\ u_{j+1} &= \frac{|P_{i+1} - P_i|}{\sum_{j=0}^{p} |P_{j+1} - P_j|} + u_j, \qquad i \in \{0, ..., p-1\} \end{aligned}$$

et qui produit également des valeurs sur $[0, 1]$ ($|.|$ symbolise la distance euclidienne).

L'emploi de l'une ou l'autre de ces techniques dépend de la répartition des points. Si la paramètre $|P_{i+1} - P_i|$ varie peu, on utilise la répartition uniforme, sinon on préfère une répartition proportionnelle.

En effet, une répartition uniforme, dans le cas où la longueur de la ligne reliant les points successifs varie beaucoup, peut engendrer des oscillations parasites [Fig 3.2].

Une répartition en fonction de l'abscisse curviligne peut être obtenue en remplaçant la ligne brisée par des arcs de cercle ou de parabole entre deux points consécutifs. Mais dans beaucoup de cas, les résultats ne diffèrent guère de ceux que l'on obtient par la méthode précédente.

Si l'on veut un paramétrage plus fin, [BEZ87] propose un méthode projective. On trace la courbe ayant pour polygone caractéristique la ligne brisée, puis on projette sur elle les sommets, et la valeur paramétrique attribuée aux points de passage est celle de leur projection [Fig 3.3].

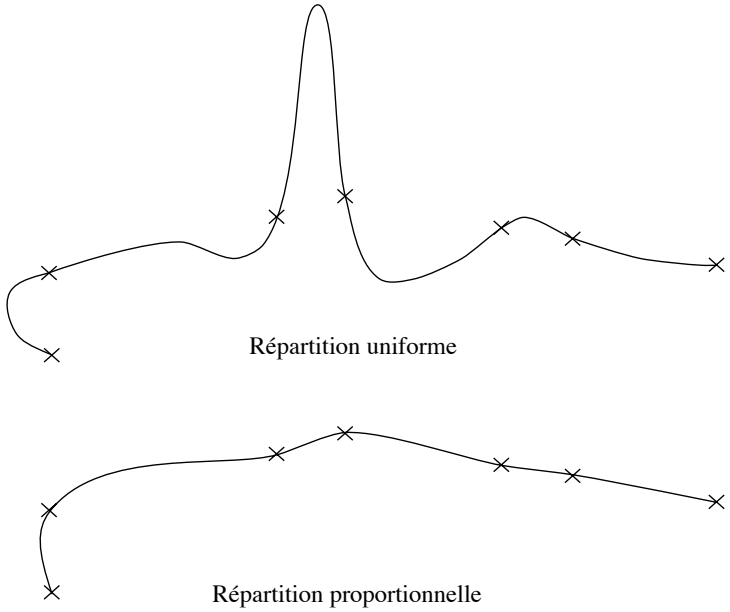

FIG. 3.2 - *Exemple de paramétrage*

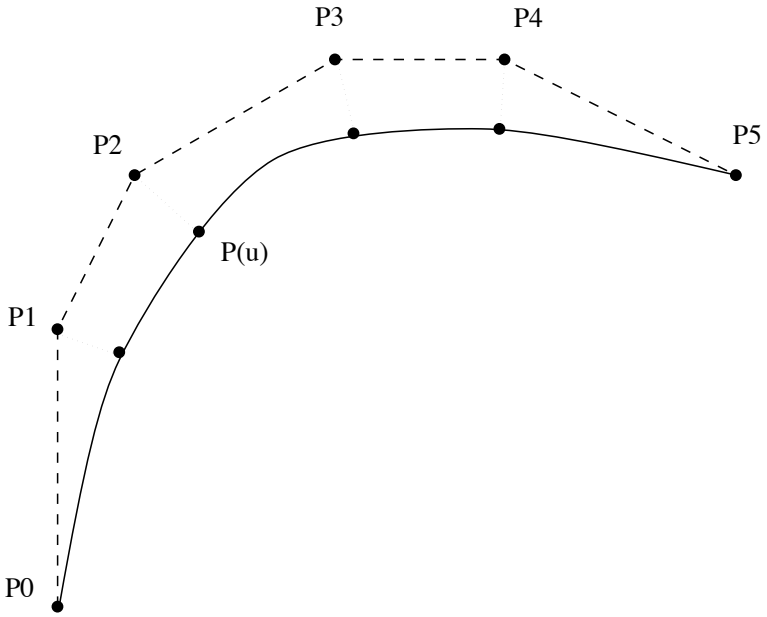

FIG. 3.3 - *Méthode projective de paramétrage*

3.3 Paramétrage des surfaces

Le paramétrage des courbes ne peut pas toujours se généraliser à celui des surfaces.

Il faut ici affecter deux paramètres (u_i, v_j) à chaque point p_{ij} du réseau, selon deux directions de référence [Fig 3.4].

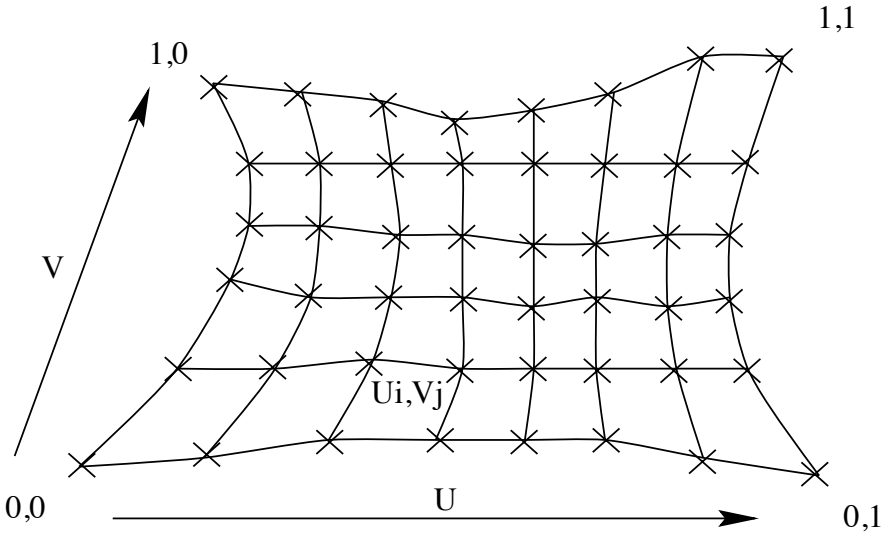

FIG. 3.4 - *Directions de référence*

A partir d'un carreau rectangulaire (obtenu lors de la phase de détermination de contour), il est assez facile de fixer une valeur de paramétrage lorsque les points sont régulièrement répartis et que le nuage possède un nombre constant de points sur chaque ligne et chaque colonne.

Dans notre cas, la répartition des points n'est pas régulière puisque les carreaux ne sont pas forcément orientés dans la direction de la matrice de points initiaux. Et, les points du carreau ne sont pas répartis de la même façon en u et en v.

Nous pouvons fixer pour chaque courbe isoparamétrique (en u par exemple) une paramétrisation, puis effectuer une moyenne pour toutes ces courbes. Mais cela ne sera acceptable que si toutes les isoparamétriques possèdent les mêmes caractéristiques, et donc le même paramétrage, ce qui est rarement le cas.

Il existe peu de véritables méthodes de paramétrage de surface, les autres techniques, de simples extensions des courbes aux surfaces, ne produisant pas des résultats satisfaisants.

[HOS88] propose d'optimiser les valeurs de paramètres, pour obtenir une approximation optimale, par une approche itérative (Newton). Il modifie les paramètres (u_i, v_k) aux points p_{ik} pour $i = 1, ..., N$ et $k = 1, ..., M$ en minimisant la somme :

$$D = \sum_{i=1}^{n} \sum_{k=1}^{m} (D_{ik})^2 = \sum_{i=1}^{n} \sum_{k=1}^{m} (P_{ik} - Y(u_i, v_k))^2$$

avec $n < N - 1$, $m < M - 1$, $Y(u, v)$ étant l'approximation de la surface.

Le but étant de faire converger le vecteur d'erreur D_{ik} vers la normale, on applique la correction suivante aux paramètres [Fig 3.5] :

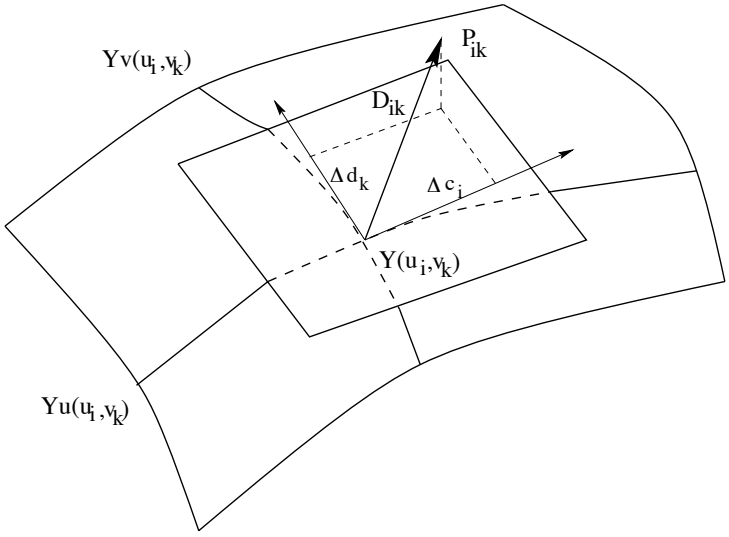

FIG. 3.5 - *Projection de l'erreur sur le plan tangent*

$$\tilde{u}_i = u_i + \frac{\Delta c_i}{\nu}, \qquad \tilde{v}_k = v_k + \frac{\Delta d_k}{\mu}$$

où ν est la longueur de la courbe pour $u_i = cste$, et μ celle pour $v_k = cste$.

[KOS91] propose une méthode de paramétrisation basée sur la courbure, qui consiste à résoudre à chaque itération, un système à $2(N+1)(M+1)$, où $N * M$ est la taille de la matrice des points initiaux. Cette technique implique que le nuage soit régulier, ce qui n'est pas notre cas.

3.4 Méthode développée

Les approches précédentes ne sont pas satisfaisantes car elles n'expliquent pas comment obtenir la première approximation.

Nous avons donc développé une méthode projective pour fixer le paramétrage d'un réseau de points. A partir des quatres courbes frontières, nous construisons un carreau de Coons, sur lequel nous utilisons une méthode de Newton pour projeter les points.

Le carreau de Coons est très bien adapté à la projection en vue d'un paramétrage, car il possède plusieurs propriétés :

- il permet de conserver les contours du carreau
- en projection 2D, les isoparamétriques du carreau de Coons correspondent à celles que nous souhaitons en 3D sur la surface biparamétrée (cf paragraphe 3.5)

Nous définissons ainsi le paramétrage du point sur le carreau de Coons, qui servira à l'approximation par une surface paramétrée.

3.4.1 Carreau de Coons

Les carreaux de Coons [COO74] sont des surfaces biparamétriques qui interpolent quatre courbes frontières, $P(u,0),\ P(1,v),\ P(u,1)\ et\ P(0,v)$ telles que le montre la figure 3.6.

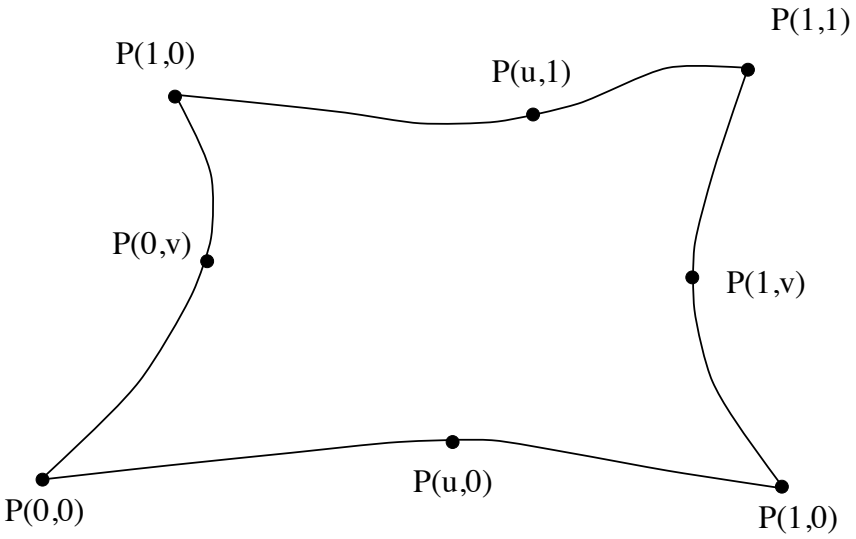

FIG. 3.6 - *Courbes frontières à interpoler*

On utilise aussi deux fonctions de mélange F_0 et F_1 telles que :

- $F_0(t) = 1 - F_1(t)$, pour générer des combinaisons de points barycentriques
- $F_0(0) = F_1(1) = 1$ et $F_0(1) = F_1(0) = 0$, pour interpoler réellement

Nous choisissons :

- $F_0(t) = 2t^3 - 3t^2 + 1$
- $F_1(t) = -2t^3 + 3t^2$

Un carreaux de Coons est défini par :

$$\begin{aligned} P(u,v) \;=\; & P(u,0)F_0(v) + P(u,1)F_1(v) \\ & + P(0,v)F_0(u) + P(1,v)F_1(u) \\ & - P(0,0)F_0(u)F_0(v) + P(0,1)F_0(u)F_1(v) \\ & - P(1,0)F_1(u)F_0(v) + P(1,1)F_1(u)F_1(v) \end{aligned}$$

La tangente en un point d'une courbe isoparamétrique, par exemple $P(u,v)$, est exprimée par la dérivée, prise par rapport à v.

$$\begin{aligned} \frac{\partial P(u,v)}{\partial v} \;=\; & P(u,0)\frac{\partial F_0(v)}{\partial v} + P(u,1)\frac{\partial F_1(v)}{\partial v} \\ & + \frac{\partial P(0,v)}{\partial v}F_0(u) + \frac{\partial P(1,v)}{\partial v}F_1(u) \end{aligned}$$

$$-P(0,0)F_0(u)\frac{\partial F_0(v)}{\partial v} + P(0,1)F_0(u)\frac{\partial F_1(v)}{\partial v}$$
$$-P(1,0)F_1(u)\frac{\partial F_0(v)}{\partial v} + P(1,1)F_1(u)\frac{\partial F_1(v)}{\partial v}$$

3.4.2 Méthode de Newton

Nous utilisons la méthode de Newton pour projeter un point P_{ij} du réseau sur la surface de coons. Le projeté est noté $P(u,v)$, u et v étant les valeurs des paramètres affectés au point.

Nous projetons P_{ij} sur chacune des tangentes en $P(u,v)$ des courbes isoparamétriques en u et v, de directions t_u et t_v [Fig 3.7].

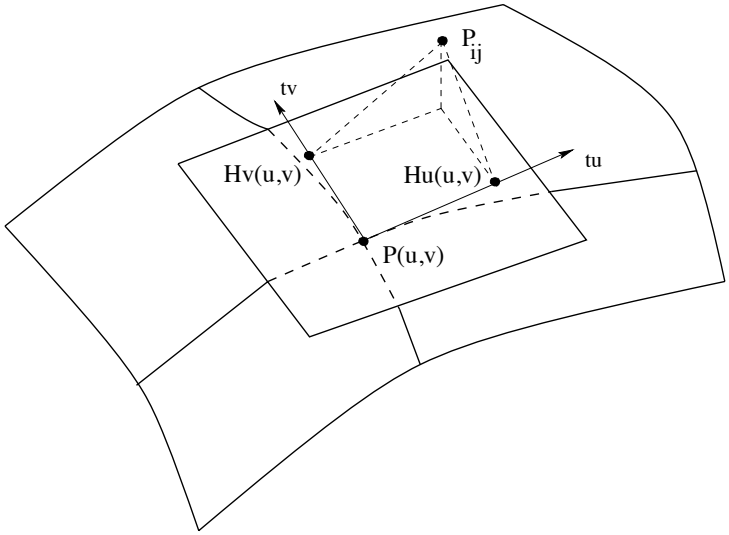

FIG. 3.7 - *Méthode de Newton*

Les projetés s'expriment ainsi :

$$H_u(u,v) = P(u,v) + ((P_{ij} - P(u,v)).t_u)t_u$$
$$H_v(u,v) = P(u,v) + ((P_{ij} - P(u,v)).t_v)t_v$$

Nous appliquons ensuite une correction à u et v pour faire tendre H_u et H_v vers $P(u,v)$. Nous obtenons ainsi un nouveau paramétrage :

$$\tilde{u} = u + \frac{(H_u(u,v) - P(u,v)).e_1}{\frac{\partial P(u,v)}{\partial u}.e_1}$$
$$\tilde{v} = v + \frac{(H_v(u,v) - P(u,v)).e_1}{\frac{\partial P(u,v)}{\partial v}.e_1}$$

où $e_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ ou $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$. Nous choisissons e_1 en fonction des coordonnées non nulles de la dérivée, dans les cas de tangence horizontale ou verticale (la dérivée partielle ne peut avoir ses deux coordonnées nulles que dans le cas d'un carreau dégénéré à un seul point).

$P(u, v)$ est le projeté de p_{ij} lorsque $(\tilde{u} - u) < \varepsilon$ et $(\tilde{v} - v) < \varepsilon$.

Les résultats obtenus sont présentés sur les figures 3.8 et 3.9.

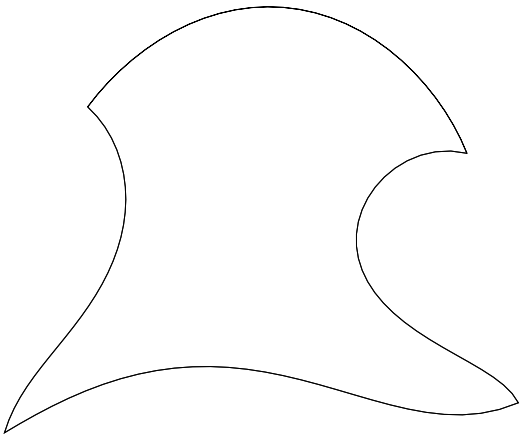

FIG. 3.8 - *Contour du carreau*

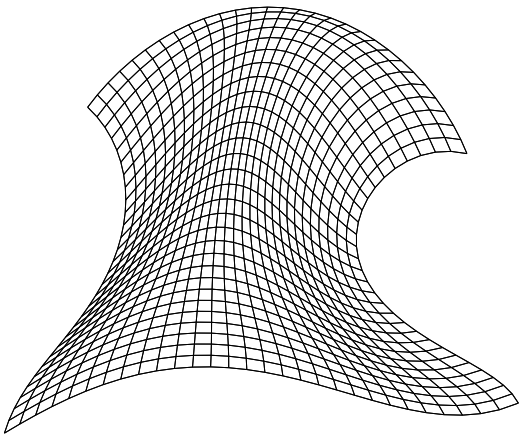

FIG. 3.9 - *Carreau de Coons*

3.4.3 Limites de la méthode proposée

La méthode que nous proposons, offre des résultats satisfaisants dans la plupart des cas.

Cependant, elle peut s'avérer limitée lorsque le contour du carreau est fortement concave dans un coin [Fig 3.10 et 3.11]. La méthode de projection sur un carreau de Coons est mise en défaut, et il faut se tourner vers d'autres techniques plus puissantes [MAL95].

FIG. 3.10 - *Contour concave dans un coin*

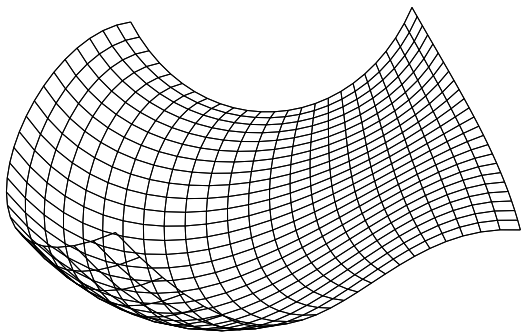

FIG. 3.11 - *Carreau de Coons formant un pli*

Dans la pratique, ce cas n'apparaît pas si les contours sont correctement déterminés.

D'autre part, le choix des paramètres de départ est important, car il peut augmenter le nombre d'itérations de la méthode de Newton, si les valeurs initiales sont trop éloignées des valeurs finales.

De plus, deux paramétrages initiaux différents peuvent produire des surfaces légèrement différentes. Une solution est de prendre un critère d'arrêt très fin, mais cela se fait au détriment du temps de calcul.

Un problème de divergence peut survenir lorsque la surface possède des courbures très prononcées, ce qui est du à la divergence de la méthode de Newton. Cependant, dans notre cas, la divergence est peu probable car les carreaux sont fabriqués de telle façon que la courbure reste faible. Ainsi, la surface de Coons est peu courbée, et le point projeté est très proche de sa projection.

Nous avons observé qu'une valeur initiale de paramètre peut être fixée en projetant les points sur un plan approximant les quatre courbes frontières. Ce paramétrage initial permet d'améliorer l'algorithme par rapport à un paramétrage arbitraire ($u = 0.5$ et $v = 0.5$ par exemple).

3.5 Résultats et comparaisons

Nous présentons ici deux exemples d'approximation par une surface B-spline, d'un nuage de points non organisé. Nous travaillons sur les images de profondeur Mickey (Fig 3.12) et moule (Fig 3.13).

La première étape est la définition du contour (Fig 3.14 et 3.16) pour créer le carreau de Coons (Fig 3.15 et 3.17).

Ensuite, nous projetons chaque point du sous-nuage (Fig 3.18 et 3.19) , sur le carreau de Coons et nous lui affectons une valeur de paramètre.

A partir de là, nous pouvons approximer le nuage par une surface biparamétrique, ici une B-spline (Fig 3.20 et 3.21).

La méthode de paramétrage proposée fournit de bons résultats. A chaque itération de la méthode de Newton, nous calculons :

- un point sur la surface
- deux dérivées en ce point

Nous avons developpé une autre méthode, pour remplacer la méthode de Newton. Elle consiste, à chaque itération, à diviser le carreau en quatre parties et à calculer la plus petite distance entre le point du nuage et les centres de chaque sous-carreau, et à recommencer jusqu'à obtenir la précision voulue (Fig 3.22).

Ainsi, à chaque itération, nous calculons :

- quatre points sur la surface
- quatre distances avec le point du nuage

Cette méthode converge plus lentement mais plus sûrement pour fournir des résultats pratiquement équivalents.

FIG. 3.12 - *Image de Mickey*

FIG. 3.13 - *Image du moule*

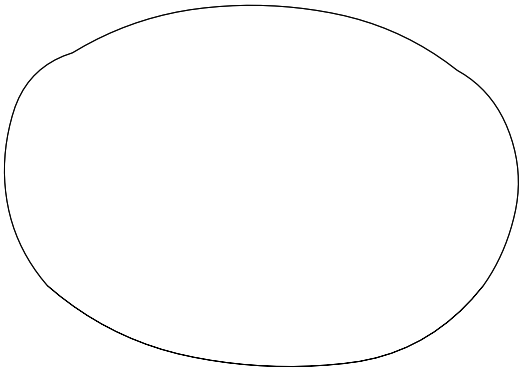

FIG. 3.14 - *Contour du nez du Mickey*

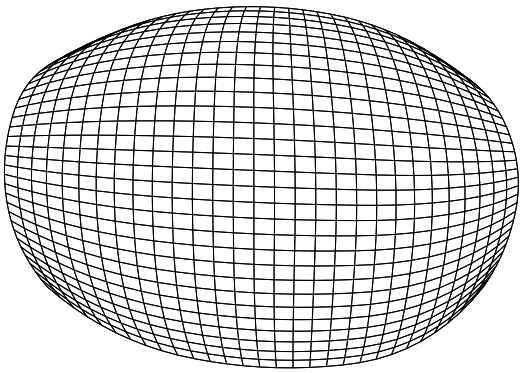

FIG. 3.15 - *Carreau de Coons du Mickey*

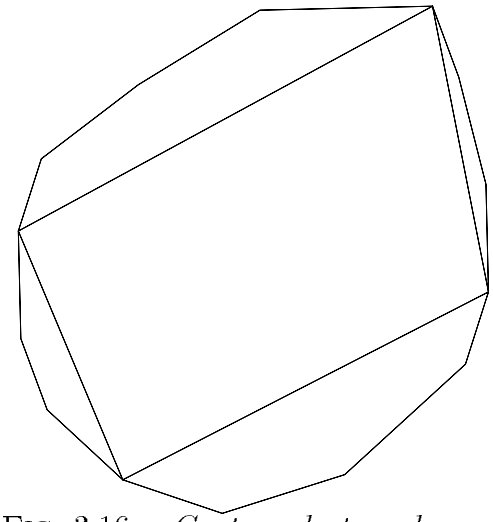

FIG. 3.16 - *Contour du trou du moule*

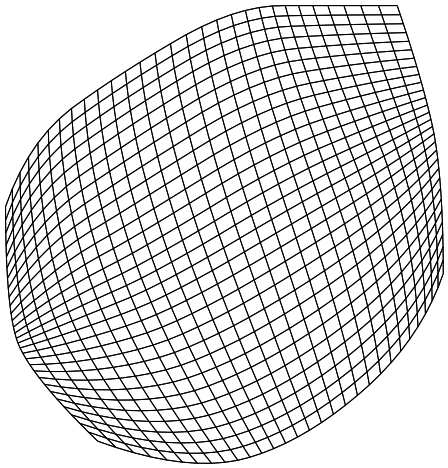

FIG. 3.17 - *Carreau de Coons du moule*

FIG. 3.18 - *Nuage de points du Mickey*

FIG. 3.19 - *Nuage de points du moule*

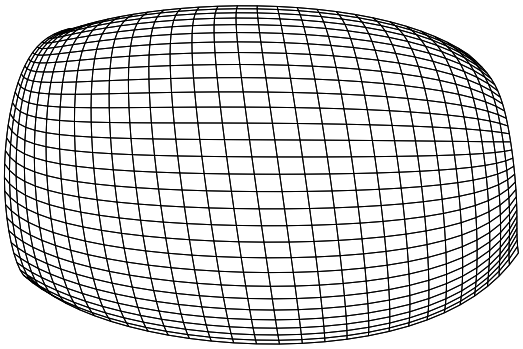

FIG. 3.20 - *Approximation de la surface du nez du Mickey*

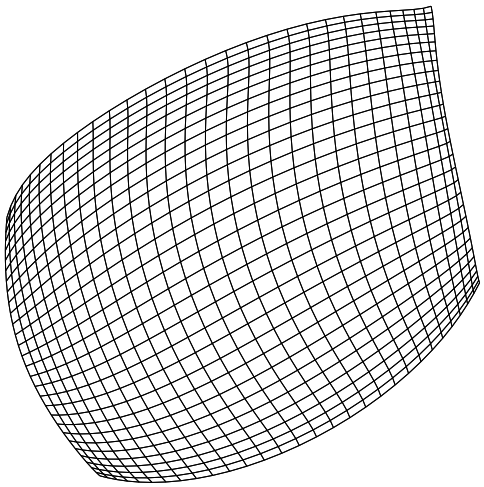

FIG. 3.21 - *Approximation de la surface du trou du moule*

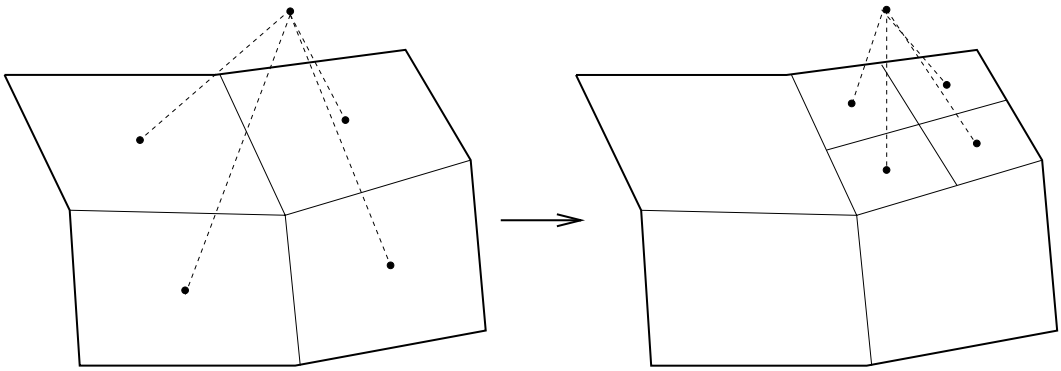

FIG. 3.22 - *Méthode de subdivision en quatre sous-carreaux*

Chapitre 4

Conclusion

L'étude de DEA a porté sur l'automatisation de la reconstruction de surfaces 3D à partir d'un nuage de points.

Elle se situe au niveau des étapes 3 et 4 de la stratégie de modélisation développée dans le cadre du projet ESPRIT (Fig 1.1) :

- subdivision semi-automatique
- approximation de chaque carreau

La méthode que nous proposons, évite à l'utilisateur d'intervenir au cours du processus de reconstruction, en enchaînant la définition des contours du carreau (après une phase de reconnaissance des caractéristiques de la surface) et leur approximation.

Cependant, ce processus n'est pas encore totalement automatique, car nous avons pris en compte un seul carreau à la fois.

Il serait maintenant intéressant de généraliser à plusieurs carreaux (Fig 4.1).

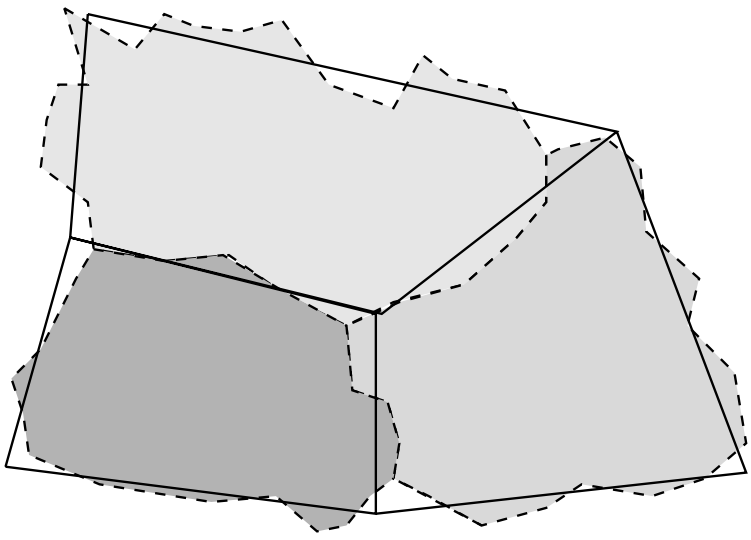

FIG. 4.1 - *Contours adjacents*

Il se pose alors le problème de la subdivision du nuage en une grille de carreaux contigus. De nombreuses contraintes viennent s'ajouter au problème de base, qui est

la reconstruction d'une surface définie sur un contour fermé. Ces contraintes sont notamment la création de carreaux adjacents et comment ces derniers interagissent entre eux.

De plus, pour répondre aux contraintes d'approximation et de raccordement par des surfaces biparamétriques, les carreaux doivent toujours être à topologie rectangulaire et doivent partager chacune de leurs frontières avec un seul autre carreau.

Ainsi, les perspectives de notre étude sont la subdivision d'un nuage de points, en tenant compte de contraintes sur les contours des carreaux adjacents, c'est à dire comment déterminer directement des carreaux rectangulaires en fonction des caractéristiques de la surface. Ceci permettrait de fournir une subdivision rectangulaire automatique de la surface, acceptable directement par les logiciels de CAO.

Annexe A

Courbes Paramétrées

Nous rappelons rapidement dans cette annexe, comment sont définis les courbes et surfaces dans le formalisme *Bézier* et *B-Spline*.

A.1 Modèle Bézier

Une courbe de Bézier est une forme polynomiale paramétrique définie par l'équation :

$$\mathcal{P}(u) = \sum_{i=0}^{m} s_i B_{i,m}(u), \qquad u \in [0,1].$$

- la courbe est de degré m ;
- les s_i sont les points de contrôle de la courbe ou encore appelé polygone caractéristique ;
- les fonctions $B_{i,m}(u)$ sont les polynômes de Bernstein définis par:

$$B_{i,m}(u) = \begin{pmatrix} m \\ i \end{pmatrix} u^i (1-u)^{m-i}, \qquad i \in \{0,1,...,m\}.$$

où les $\begin{pmatrix} m \\ i \end{pmatrix}$ sont les coefficients du binôme de Newton :

$$\begin{pmatrix} m \\ i \end{pmatrix} = \frac{m!}{(m-i)!i!}.$$

Les courbes de *Bézier* possèdent des caractéristiques qui les rendent facile à utiliser avec une grande intéractivité :

- le degré de la courbe dépend du nombre de points de contrôle
- la courbe passe par le premier et le dernier point
- les tangentes sont fixées pour les extrémités
- le polygone de contrôle donne l'allure générale de la courbe

– la courbe est entièrement incluse dans l'enveloppe convexe du polygone

On peut définir de même une surface de *Bézier* qui est représentée par :

$$\mathcal{P}(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} s_{ij} B_{i,m}(u) B_{j,n}(v), \qquad u,v \in [0,1].$$

où les s_{ij} forment le réseau de points de contrôle de la surface.
Les surfaces de *Bézier* possèdent les mêmes proprités que les courbes.

A.2 Modèle B-Spline

Une courbe B-Spline est une courbe paramétrique définie par morceaux et d'équation :

$$\mathcal{P}(u) = \sum_{i=1}^{M} s_i N_{i,k}(u), \qquad u \in [0,1].$$

– la courbe est dite d'ordre k (ou de degré *k-1*) ;

– les s_i sont les points de contrôle de la courbe appelé aussi un polygone caractéristique ;

– les fonctions de base $N_{i,k}(u)$ sont des fonctions polynômiales par morceaux telles que :

$$N_{i,1}(u) = \begin{cases} 1 & si\ t \in [t_i, t_{i+1}] \\ 0 & sinon \end{cases}$$

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

– Les fonctions de base sont définies sur un vecteur nodal, qui est une suite de réels croissants. On choisit la suite $(t_i)_i$ des nœuds :

$$u_1 = ... = u_k < u_{k+1} < u_{k+2} < .... < u_{k+n} = u_{k+n+1} = ... = u_{2k+n+1}$$

où $n = M - 2k - 1$, ceci pour que les extrémités de la courbe coïncident avec le premier et le dernier point de contrôle (de façon analogue aux Béziers).

Ainsi, avec cette définition, la courbe *B-spline* peut se décomposer en n courbes de *Bézier*.
L'ordre k des fonctions de base ne dépend pas du nombre de sommets *M-1* du polynôme caractéristique comme c'est le cas pour les courbes de *Bézier* .

On peut définir de même une surface *B-Spline* qui est représentée par :

$$\mathcal{P}(u,v) = \sum_{i=0}^{M} \sum_{j=0}^{N} s_{ij} N_{i,k}(u) N_{j,l}(v), \qquad u, v \in [0,1].$$

où les s_{ij} forment le réseau de points de contrôle de la surface.

Les courbes et les surfaces *B-spline* possèdent les mêmes propriétés que les *Béziers*, avec en plus la possibilité de contrôler localement la forme.

Annexe B

Méthode de lissage

La reconstruction d'une courbe ou d'une surface est souvent dépendante d'un ensemble de contraintes géométriques qui peuvent être :

- passer exactement par un point
- passer au voisinage d'un point
- respecter une direction donnée (par exemple fixer la tangente en un ou plusieurs points)
- s'approcher d'une direction donnée

Dans cette annexe, nous rappelons une méthode de construction de courbe (ou de surface) *B-Spline* par une approximation au sens des moindres carrés.

B.1 Approximation d'un ensemble de points par une courbe

L'objectif de cette opération est d'approcher un ensemble de *(p+1)* points P_i, $i \in \{0, 1, ..., p\}$ par une courbe *B-Spline*. Pour cela, on cherche les points de contrôle et éventuellement la séquence nodale de la courbe *B-Spline* en utilisant comme critère la minimisation du carré des écarts entre les points P_i et la courbe $\mathcal{P}(u)$:

$$\mathcal{P}(u) = \sum_{i=0}^{q+m-1} s_i N_{i,m}(u), \qquad u \in [0, 1].$$

Ces écarts sont définis par :

$$\varepsilon_i = \mathcal{P}(\mu_i) - P_i, \qquad i \in \{0, 1, \cdots, p\}.$$

Cette méthode laisse le choix des paramètres suivants :

- degré m de la courbe;
- nombre et répartition des nœuds.

Les paramètres sont liés par la relation :

$$q = \frac{p-m+1}{k+1}, \qquad p,q,m > 0 \text{ et } k \geq 0.$$

où k est un paramètre de contrôle de l'erreur et *(q+1)* est le nombre de nœuds de multiplicité 1 de la séquence nodale :

$$\underbrace{u_0,\cdots,u_0}_{(m+1)fois}, u_1, u_2, \cdots, u_{q-1}, \underbrace{u_q,\cdots,u_q}_{(m+1)fois}.$$

Nous choisirons pour la suite $k = p-m$ qui conduit à $q = 1$. La séquence nodale de la courbe $\mathcal{P}(u)$ devient identique à celle definissant une courbe de *Bézier*. Dans ce cas, on réalise une identification par lissage avec un paramètre uniforme.

Après avoir fixé les valeurs de k, m et p, un paramètrage μ_i est affecté à chaque point P_i :

$$\mu_i = \frac{1}{m+k}\sum_{j=1}^{m+k} u^*_{i+j}, \qquad i \in \{0,1,...,p\}.$$

où les u^* sont les nœuds pris successivement.

$$\begin{array}{cccccc} u_0 & ,\cdots & ,u_0 & ,u_1 & \cdots,\cdots & ,u_q \\ u^*_0 & ,\cdots & ,u^*_p & ,u^*_{p+1} & \cdots,\cdots & ,u^*_{2p} \end{array}$$

La minimisation de la somme des carrés entraîne :

$$\min_{s_{ij}} G_j = \min_{s_{ij}} \sum_{t=0}^{p} \varepsilon_{tj}^2,$$

On en déduit le système linéaire suivant :

$$\frac{\partial G_j}{\partial s_{ij}} = 0, \qquad i \in \{0,1,\cdots,(q+m-1)\},\ j \in \{1,2,3\}.$$

Soit :

$$\sum_{t=0}^{q+m-1} s_{tj}\left(\sum_{l=0}^{p} N_{tm}(\mu_l)N_{im}(\mu_l)\right) = \sum_{l=0}^{p} P_{lj}N_{im}(\mu_l).$$

Ce qui donne sous forme matricielle :

$$\begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{q+m-1} \end{bmatrix} = ([\mathcal{N}m,p,q]^T[\mathcal{N}m,p,q])^{-1}[\mathcal{N}m,p,q]^T \begin{bmatrix} P_0 \\ P_1 \\ \vdots \\ P_p \end{bmatrix}$$

avec :

$$[\mathcal{N}m,p,q] = \begin{bmatrix} N_{0m}(\mu_0) & N_{1m}(\mu_0) & \cdots & 0 \\ N_{0m}(\mu_1) & N_{1m}(\mu_1) & \cdots & 0 \\ \vdots & & & \vdots \\ 0 & 0 & \cdots & N_{q+m-1,m}(\mu_p) \end{bmatrix}$$

B.2 Approximation d'un nuage de points par des surfaces

Le processus de lissage peut s'adapter a un réseau de points. Pour un ensemble de points paramétrés $P_{ij},\ i \in \{0,1,\cdots,p\},\ j \in \{0,1,\cdots,q\}$, le réseau de points de contrôle s'obtient en minimisant la somme des carrés des écarts :

$$\min_{s_{kl}} \sum_{i=0}^{p} \sum_{j=0}^{q} \varepsilon_{ij}^2 = \min_{s_{kl}} \sum_{i=0}^{p} \sum_{j=0}^{q} (P(u_i, v_i) - P_{ij})^2$$

Les paramètres de lissage sont définis par :

$$a = \frac{p-m+1}{k+1} \text{ et } b = \frac{q-n+1}{k+1}$$

Ce qui conduit à la forme matricielle :

$$\begin{bmatrix} s_{00} \\ s_{10} \\ \vdots \\ s_{a+m-1,0} \\ s_{01} \\ \vdots \\ s_{a+m-1,b+n-1} \end{bmatrix} = ([\mathcal{N}mn, pq, ab]^T [\mathcal{N}mn, pq, ab])^{-1} [\mathcal{N}mn, pq, ab]^T \begin{bmatrix} P_{00} \\ P_{10} \\ \vdots \\ P_{p0} \\ P_{01} \\ \vdots \\ P_{pq} \end{bmatrix}$$

Bibliographie

[AD91] Nirwan ANSARI and Edard DELP. On detecting dominant points. *Pattern Recognition*, 24(5), 1991.

[BEZ87] Pierre BEZIER. *Mathematiques et CAO, Courbes et surfaces*. HERMES, 1987.

[BJ88] P.J. BESL and R.C. JAIN. Segmentation through variable-order surface fitting. *IEEE transactions on PAMI*, 10(2), march 1988.

[CHO95] Jin J. CHOU. Voronoi diagrams for planar shapes. *IEEE Computer Graphics and Applications*, march 1995.

[COO74] S.A. COONS. Surface patches and b-splines curves. *CADG*, 1974.

[CS94] Xin CHEN and Francis SCHMITT. Surface modelling of rang data by constrained triangulation. *Computer-AIded Design*, 26(8), august 1994.

[FAR92] Gerald FARIN. *Courbes et surfaces pour la CGAO*. MASSON, 1992.

[FS75] H. FREEMAN and R. SHAPIRA. Determinig the minimum-area encasing rectangle for an arbitrary closed curve. *Communication ACM*, 18(7), 1975.

[HDD+92] HOPPE, DEROSE, DUCHAMPS, McDONALD, and STUETZLE. Surface reconstruction from unorganized points. *Computer Graphics (SIGGRAPH '92 Proceedings)*, 26(2), july 1992.

[HJ87] R. HOFFMAN and A.K. JAIN. Segmentation and classification of range images. *IEEE transactions on PAMI*, 9(5), sept. 1987.

[HOS88] J. HOSCHEK. Intrinsic parametrization for approximation. *Computer Aided Geometric Design*, (5):pp 27–31, 1988.

[IPSV94] O.D. ISSELMOU, E. PERNA, B. SHARIAT, and D. VANDORPE. Rita : 3d surface reconstruction. *Fondamentals of Computer Graphics, proceedings of Pacific Graphis '94*, pages 293–305, august 1994.

[KOS91] M. KOSTERS. Curvature-dependent parameterization of curves ans surfaces. *Computer-Aided Design*, 23(8):pp 569–578, october 1991.

[LEO91] Jean-Claude LEON. *Modelisation et construction de surfaces pour la CFAO*. HERMES, 1991.

[MAL95] Isselmou Ould Dellahy MALOUM. Reconstruction de surfaces gauches a partir de donnees non structurees - parametrisation par des transformations conformes. *These de doctorat*, A paraitre en 1995.

[MAN94] A.T. MANNINEN. Orientationnal approximation of convex polygons with rectangles. *Pattern Recognition Letters*, 15(7), july 1994.

[MM94] D.A. MITZIAS and B.G. MERTZIOS. Shape recognition with a neural classifier based on a fast polygon approximation technique. *Pattern Recognition*, 27(5), may 1994.

[MS88] R.P. MARTIN and P.C. STEPHENSON. Putting objects into boxes. *Computer-Aided Design*, 20(9), nov. 1988.

[PV94] Juan-Carlos PEREZ and Enrique VIDAL. Optimum polygonal approximation of digitized curves. *Pattern Recognition Letters*, 15(8), august 1994.

[TYL94] P.W.M. TSANG, P.C. YUEN, and F.K. LAM. Classification of partially occluded objects using 3-point matching and distance transformation. *Pattern Recognition*, 27(1), 1994.

Table des matières