Diagrammes de Voronoi 2D et 3D, applications en analyse d'images

Etienne Bertin

THESE

présentée par

Etienne Bertin

pour obtenir le titre de DOCTEUR
DE L'UNIVERSITÉ Joseph FOURIER - GRENOBLE I
(Arrêtés ministériels du 5 Juillet 1984 et
du 30 Mars 1992)

"Spécialité : Mathématiques Appliquées"

DIAGRAMMES DE VORONOI 2D ET 3D : APPLICATIONS EN ANALYSE D'IMAGES

Date de soutenance : 25 **Janvier** 1994
Composition du Jury :

Pierre-Jean Laurent	PRESIDENT
Bernard Chalmond	RAPPORTEUR
Francis Schmitt	RAPPORTEUR
Jean-Marc Chassery	EXAMINATEUR
Andre Gagalowicz	EXAMINATEUR
Bernard Lacolle	EXAMINATEUR

Thèse préparée au sein du Laboratoire **TIMC - Institut IMAG**

Autorisation de Soutenance

Doctorat de l'Université Joseph Fourier - GRENOBLE I

Vu les dispositions de l'arrêté du 5 Juillet 1984 et de l'arrêté du 30 Mars 1992.

Vu les rapports de :

Mr. Bernard Chalmond

Mr. Francis Schmitt

Mr. Etienne Bertin est autorisé à présenter une thèse en vue de l'obtention du Doctorat de l'Université Joseph Fourier - Grenoble 1.

Fait à Grenoble, le

Le Président de l'Université
Joseph Fourier - Grenoble 1.
A. NEMOZ

Remerciements

Ce travail a été effectué au sein du laboratoire TIMC (Techniques de l'Imagerie de la Modélisation et de la Cognition) de l'IMAG (Institut de Mathématiques Appliquées de Grenoble).

Je remercie particulièrement le professeur Pierre-Jean Laurent qui a bien voulu présider le jury de cette thèse.

J'adresse mes vifs remerciements aux professeurs Bernard Chalmond et Francis Schmitt qui ont pris la rude charge d'être les rapporteurs de ce mémoire de thèse.

J'exprime toute ma reconnaissance aux professeurs André Gagalowicz et Bernard Lacolle qui ont accepté de participer au jury de cette thèse.

J'ai le plaisir de remercier Monsieur Jean-Marc Chassery Directeur de recherche au CNRS pour avoir pris le risque de me confier un sujet aussi passionnant. Ses directives son entousiasme et sa sympathie ont été des facteurs déterminants pour le bon déroulement de cette thèse.

Patrick Moreau m'a fait profiter de son expérience en géométrie algorithmique et c'est grâce aux nombreuses discussions que nous avons eues que j'ai assimilé les techniques de programmation et ainsi pu mener à bien mon travail de recherche.

Je tiens également à remercier tous les membres du laboratoire pour l'aide qu'ils m'ont toujours prodiguée.

Je réserve toutefois une attention particulière à Annick Montanvert qui de part son dynamisme m'a donne les moyens de poursuivre mes recherches en Autriche et à Raphaël Marcelpoil qui m'a guidé sur les chemins de la sociologie cellulaire ; qu'ils soient tous deux assurés de ma profonde sympathie.

Enfin je remercie toutes les personnes qui ont contribué à la bonne réalisation de ce mémoire.

Résumé

L'objectif de cette thèse est de montrer comment la *géométrie algorithmique* en généralΓet les *diagrammes de Voronoï* en particulier peuvent intervenir en *analyse d'images*Γdiscipline qui est notre centre d'intérêt.

Le propos de l'analyse d'images est la description du contenu d'une imageΓen vue de l'interprétation et de la prise de décision.

La géométrie algorithmique consiste à trouver des algorithmes efficaces en vue de résoudre des problèmes à caractère géométrique.

Nous nous intéresserons ici au problème de représentation des images par des partitionnements plus ou moins complexesΓadaptés ou non au contenu informatif des images. Parmi les partitionnementsΓnous développerons plus particulièrement celui en régions de Voronoï.

Nous aborderons ensuite le problème du codage de formes tridimensionnelles par leur squelette qui est lié au graphe de Voronoï.

La segmentation des images est une partie importante de l'analyse des images. Nous en donnerons une nouvelle approche dans un contexte pyramidalΓinitialisé par une partition en régions de Voronoï et contrôlé par les réseaux de Hopfield.

EnfinΓune utilisation des champs de Markov pour déterminer une partition optimale en un sens à définir sera abordée.

Notations et Conventions

Notations

DVP	Diagramme de Voronoï Ponctuel
DVG	Diagramme de Voronoï Généralisé
Voxel	"Volume element"
Pixel	"Picture element"
S	Ensemble fini de points (ou d'objets)
$VOR(S)$	Diagramme de Voronoï Ponctuel de l'ensemble S
$DEL(S)$	Diagramme de Delaunay de l'ensemble S
$CONV(S)$	Enveloppe convexe des points de S
$Vor_S(p)$	Région de Voronoï associée au site p dans S
$N_S(p)$	Ensemble des sites voisins de p dans $DEL(S)$
$PE(p)$	Polyèdre étoilé associé au site p
$DEL_p(S)$	Tétraèdres de Delaunay ne contenant pas p
$E_p(S)$	Sommets de $VOR(S)$ supprimés en insérant le site p
$B(p_1, p_2, p_3, p_4)$	Sphère circonscrite aux germes $p_1 \Gamma p_2 \Gamma p_3 \Gamma$et p_4
$Card(F)$	Cardinal d'un ensemble fini
$Vol(X)$	Nombre de voxels contenus dans X
$Surf(X)$	Surface de X (Polyèdre ou polygone)
$m(X)$	Moyenne des voxels contenus dans X
$\sigma(X)$	Ecart type des voxels contenus dans X
$Bary(X)$	Barycentre de X
$n(v)$	Niveau de gris d'un voxel
$B(X,Y)$	Bissectrice de deux éléments simples
$Sep(X,Y)$	Séparateur de deux ensembles
$H(X,Y)$	Région des points plus proches de X que de Y
$S_k(X)$	Squelette d'une forme

Conventions

Pour plus de clartéΓ quand nous renvoyons à une section sans mentionner le chapitreΓelle se trouve dans le chapitre ; sinonΓnous indiquons le chapitre. Nous appliquons la même règle aux définitionsΓthéorèmesΓpropositions. S'ils sont référencés dans la section où ils sont définisΓnous ne mentionnons ni la section ni le chapitre ; s'ils sont référencés dans le chapitre où ils sont définis nous ne mentionnons pas le chapitre ; sinon nous mentionnons et le chapitre et la section.

Table des matières

Table des figures

Chapitre 1

Introduction

"La Fièvre d'Urbicande" (Schuiten et Peeters).

"Human intuition is often guided by visual perception. If one sees an underlying structure, the whole situation may be understood at a higher level."

F. Aurhennamer

1.1 Introduction

Dans ce mémoire de thèse nous allons montrer comment utiliser des outils issus de la *la géométrie algorithmique* et plus particulièrement les *diagrammes de Voronoï* en *analyse d'images* [41Γ115].

Nous utiliserons donc les diagrammes de Voronoï dans différents problèmes d'analyse d'images :

- **représentation** d'images en régions de VoronoïΓ
- **squelettisation** d'objets polyédriquesΓ
- **segmentation** d'images.

Nous donnerons aussi une ouverture sur le probème du **partitionnement optimal** d'une image en régions de Voronoï.

La géométrie algorithmique [107Γ 49Γ27] est une science encore jeune et génératrice de nombreux problèmes faisant intervenir des entités géométriques : pointsΓ polyèdresΓsphères... Ces problèmes sont souvent posés en termes simples etΓpourtantΓles solutionsΓquand elles existentΓutilisent souvent des outils complexes issus des mathématiques. L'optimalité recherchée dans les solutions est souvent la source de la principale difficulté. Mais ce souci d'efficacité est justifié par la grande taille des problèmes traités.

De nos jours les diagrammes de Voronoï et de Delaunay [78Γ 5] forment une partie importante de la géométrie algorithmique. Le diagramme qui porte le nom de Voronoï serait apparu pour la première fois dans des écrits de Descartes en 1664 pour l'étude des planètes de notre système solaire. Mais ce n'est que bien plus tardΓ en 1850Γsous l'impulsion de DirichletΓpuis en 1908Γsous l'impulsion de Voronoï que ce diagramme a trouvé ses lettres de noblesse.

Ce n'est qu'en 1934 que Delaunay introduit le diagramme qui porte son nom. Les diagrammes de Voronoï et de Delaunay sont intimement liés puisqu'ils forment deux graphes duaux.

Nous allons voir dans cette introduction pourquoi le diagramme de Voronoï est un outil fondamental. Nous verrons ensuite les applications de ce diagramme à différentes disciplines. Nous présenterons enfin un plan détaillé de ce mémoire de thèse.

1.2 Voronoï... Pourquoi ?

La question naturelle que nous sommes en droit de nous poser avant d'aborder ce mémoire est la suivante : d'où vient l'intérêt porté aux diagrammes de Voronoï ?

Un premier élément de réponse peut être donné en considérant la simplicité de la définition du diagramme de Voronoï donnée dans Preparata [107]. Soit $S\Gamma$ un ensemble de N points dans l'espace. Pour chaque point p_i de $S\Gamma$quel est l'ensemble des points (x, y, z) dans l'espace qui sont plus proches de p_i que de n'importe quel autre point de S ?

La solution au problème posé est de partitionner l'espace en régions de Voronoï.

Un deuxième élément de réponse est donné dans l'article de Aurenhammer [5] en quatre points :

1. Les régions de Voronoï se retrouvent dans la nature : les cellules d'un tissu dans le plan forment approximativement une partition de Voronoï [69]Γainsi que les alvéoles des abeillesΓles bulles de savon [126] (Figure 1)Γles molécules chimiques [42Γ92]Γou encore les cristaux [56].

2. Les régions de Voronoï ont des propriétés mathématiques intéressantes et surprenantes. C'est ce qui fait penser que les diagrammes de Voronoï sont l'une des plus fondamentales constructions géométriques découvertes pour un ensemble de points [5].

3. Il existe des algorithmes efficaces pour construire les diagrammes de Voronoï. De plus ces diagrammes sont très puissants pour résoudre en temps optimal de nombreux problèmes de géométrie algorithmique [107] : enveloppe convexeΓ problèmes de proximitéΓarbre de poids minimum (*minimum spaning tree*)Γ graphe de Gabriel... [107].

4. La structure duale du diagramme de Voronoï (le diagramme de Delaunay) admet elle aussi de très bonnes propriétés géométriques et topologiques [107]. De plus le graphe de Delaunay est le graphe qui contient toutes les informations locales [135].

Toutes ces remarques nous ont amené à exploiter du mieux que nous pouvions les diagrammes de Voronoï et de Delaunay.

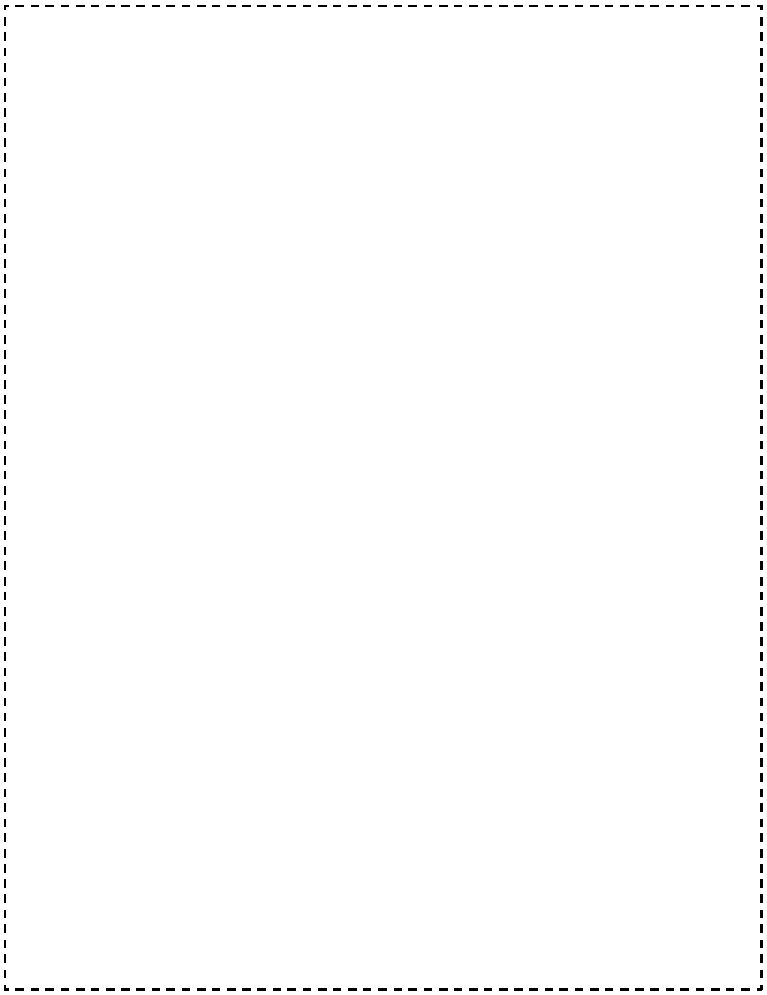


FIG. 1 - Bulles de savon (*tiré de [126]*)

1.3 Applications des diagrammes de Voronoï

1.3.1 Applications en géométrie algorithmique

De nombreuses recherches portent sur la construction des diagrammes de Voronoï. Les deux classes d'algorithme les plus utilisées sont :

- la classe "split and merge"Γ
- la classe "incrémentale".

Ces deux classes sont extrêmement intéressantes puisqu'elles permettent de résoudre d'autres problèmes de géométrie algorithmique commeΓparexempleΓla détermination de l'enveloppe convexe d'un ensemble de points du plan ou de l'espace

[107Γ20Γ50].

De plus le calcul des diagrammes de Voronoï et de Delaunay permet de résoudre en temps optimal des problèmes de géométrie algorithmique commeΓpar exemple : la recherche de tous les plus proches voisinsΓle calcul de l'enveloppe convexeΓle graphe de GabrielΓle MSTΓ..

Les calculs de la complexité de la construction des diagrammes de Voronoï et de Delaunay donnent lieu à de nombreuses recherchesΓnotamment en analyse "randomisée" [46Γ132Γ24]Γou en moyenne [48]. Les techniques de calcul font appel à la théorie des probabilités et sont assez élégants.

1.3.2 Applications diverses

Il est aussi à noter que les diagrammes de Voronoï et de Delaunay sont très utiles dans de nombreux domaines. Nous pouvons citer par exemple :

1. la sociologie cellulaire qui est l'étude structure-fonction des cellules au sein d'un tissu cellulaire [87Γ86]Γ

2. la reconstruction de surfaces à partir d'un modèle numérique de terrain [111Γ34Γ6]Γ

3. la compression des images à l'aide de la théorie des IFS (Iterated Function System) [44Γ43Γ37]Γ

4. la percolation au sein du diagramme de Delaunay ou de Voronoï [137]Γ

5. la reconstruction d'un volume [22Γ23Γ26]Γ

6. l'interpolation de données tridimensionnelles obtenues par stéréo [25]Γ

7. la représentation et la segmentation des images 2D ou 3D [1Γ113Γ136Γ106Γ38Γ40Γ12Γ110]Γ

8. la squelettisation d'objets bidimensionnels ou tridimensionnels [83Γ25Γ121Γ74Γ31Γ94Γ4]Γ

9. la classification vectorielle [79]Γ

10. la résolution des équations aux dérivées partielles en utilisant la méthode des éléments finis sur des partitionnements en triangles ou en tétraèdres ayant de "bonnes" propriétés [128Γ82]Γ

11. la morphologie mathématique [138Γ84]Γ

12. la physique et la chimie [42Γ92Γ56].

Cette liste est bien entendue non exhaustiveΓet nous ne doutons pas un instant que d'autres applications existantes ont échappé à notre attention.

1.4 Plan

Ce mémoire de thèse sera articulé de la manière suivante.

Dans le chapitre 2 nous rappellerons les notions essentielles à propos des diagrammes de Voronoï et de Delaunay. Nous donnerons donc les définitions et propriétés fondamentales qui nous seront utiles par la suite. Puis nous insisterons sur les algorithmes de construction : méthode incrémentaleΓ"divide and conquer"... Nous détaillerons aussi les algorithmes de recherche des points les plus proches. Un algorithme original de suppression de points sera proposé et enfin des calculs de complexité concluront ce chapitre.

Dans le chapitre 3 nous parlerons de partitionnements du plan et de l'espace par des éléments géométriques. Nous distinguerons les partitionnements non adaptatifs (i.e. ne tenant pas compte des informations de niveaux de gris contenus dans les images) des partitionnements adaptatifs (i.e. tenant compte des informations de niveaux de gris contenus dans les images). Dans chacune de ces deux classesΓnous distinguerons les partitionnements rigides (i.e. le nombre d'éléments de base est limité) des partitionnements non rigides (i.e. le nombre d'éléments de base est non limité). Nous étudierons plus particulièrement les partitionnements en carrés ou en cubesΓen quadtrees ou en octreesΓen triangles ou en tétraèdres de DelaunayΓet enfin en polygones ou en polyèdres de Voronoï. Nous apporterons une étude détaillée comparative pour clore ce chapitre.

Dans le chapitre 4 nous étudierons la généralisation des diagrammes de Voronoï à des éléments plus complexes que des points : segmentsΓpolygones de l'espace ou

polyèdres. Ce chapitre est intéressant dans deux domaines :

1. en géométrie algorithmiqueΓpuisque nous donnons les équations des séparateursΓet que nous proposons un algorithme de calcul pour approcher le DVG 3D par le DVP 3DΓ

2. en analyse d'imagesΓpuisqu'il existe une connexion entre les DVG 3D et les squelettes d'objets 3D.

Ce chapitre sera largement illustré par des exemples.

Dans le chapitre 5 nous donnerons une nouvelle approche pour segmenter des images dans un contexte pyramidal. L'originalité vient du fait que nous initialisons la base de la pyramide avec les régions de Voronoï. Ceci permet de réduire le nombre de niveaux de la pyramide au cours du processus de segmentationΓde diminuer les temps de calcul et de pouvoir traiter des images de grande taille. Une autre originalité provient du contrôle du processus de décimation de la pyramide. Nous utilisons en effet l'approche nouvelle de Bischof [17]Γqui construit un stable à l'aide des réseaux de Hopfield. Notre méthode sera validée par quelques exemplesΓmais nous insisterons sur les orientations futures à prendre pour poursuivre notre travail.

Le chapitre 6 est une prospection sur les champs de Markov liés aux diagrammes de Voronoï. Une première application est une généralisation du travail de Geman & all pour segmenter des images texturées [60]. Notre généralisation consiste à utiliser un partitionnement en régions de Voronoï adapté aux images plutôt que d'utiliser un partitionnement rigide en carrés. Une deuxième application consiste à générer un partitionnement en régions de Voronoï optimalΓen un sens que nous préciserons.

Nous donnerons enfin une conclusion générale à tous nos travaux.

Chapitre 2

Diagramme de Voronoï Ponctuel 3D.

Ce chapitre est une présentation approfondie du diagramme de Voronoï ponctuel 3D (DVP 3D)Γbranche de la géométrie algorithmique.

Nous insistons sur les définitions formelles et les propriétés fondamentales du DVP 3D. Une large part est consacrée aux algorithmes de constructionΓen particulier aux algorithmes incrémentaux quiΓpar leur nature mêmeΓsont intrinsèquement dynamiques. Nous présentons aussi un algorithme de remise à jour par suppression de germesΓalgorithme qui nous sera utile par la suite. Des études de complexité théoriques sont détailléesΓpuisque l'efficacité des algorithmes de construction du DVP 3D en dépend.

Toutes les notions introduites dans ce chapitre seront constamment utilisées par la suite. Notre but est donc aussi de donner les bases nécessaires à la compréhension des autres chapitres.

2.1 Introduction

La géométrie algorithmique concerne l'étude algorithmique de problèmes géométriques. Les deux ouvrages de base concernant la géométrie algorithmique sont ceux de Preparata&Shamos [107] et de Edelsbrunner [49].

L'étude des DVP 3D est une branche importante de la géométrie algorithmique [5Γ78Γ64]. Les raisons en sont la diversité et la richesse des algorithmes de construction des DVP : algorithmes de type "divide and conquer"Γalgorithmes de type incrémental... (section 2.3) et les liaisons avec d'autres problèmes constituant une préocupation de la géométrie algorithmiqueΓnotamment la recherche de tous les plus proches voisinsΓle calcul de la triangulation de DelaunayΓle calcul d'enveloppes convexes...

La raison d'être de ce chapitre est de donner les notions de base des DVPΓqui nous seront utiles tout au long de ce mémoire de thèse.

Nous apportons aussi une contribution sur les points suivants :

- amélioration de l'algorithme de Bowyer (section 2.4)Γ
- amélioration de l'algorithme d'incrémentation quaternaire (section 2.5.3)Γ
- amélioration de l'algorithme de suppression (section 2.6)Γ
- calculs de complexité des algorithmes incrémentaux (section 2.7).

Il est à noter que dans tout ce chapitreΓsauf mention explicite du contraireΓnous nous mettrons sous l'hypothèse (H) suivante :

Hypothèse H. *Soit S un ensemble de n **points** appelés aussi **sites** ou **germes** de l'espace :*

$$S = \{p_i \in I\!R^3, i = 1, \ldots, n\}$$

Nous supposerons que cinq sites ne sont pas cosphériques, et que quatre sites ne sont pas coplanaires

Cette hypothèse est classique en géométrie algorithmiqueΓet évite des ambiguïtés topologiques au sein du graphe de Delaunay (Définition 2.3). Elle assure que le graphe de Delaunay existe et est unique.

En 2DΓl'hypothèse (H) suppose simplement que quatre sites ne sont pas cocirculaires.

L'illustration 2 donne un exemple dans $\mathbb{R}^2$ de 4 points cocirculaires : dans cet exemple on ne sait pas dire si les sites p_i et p_j sont voisins (de même pour les sites p_k et p_l).

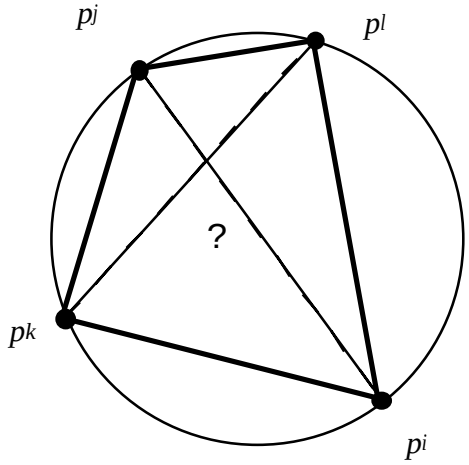


FIG. 2 - Exemple de quatre sites cocirculaires donnant une ambiguïté sur la topologie du graphe de Delaunay.

Le plan de ce chapitre est le suivant. Dans un premier temps (section 2.2)Γnous donnons les définitions et propriétés nécessaires à la compréhension de notre exposé. La section 2.3 est une présentation générale des algorithmes de construction du DVP 3D. La section 2.4 détaille plus spécifiquement les algorithmes incrémentaux pour la construction du DVP 3D. La section 2.5 montre comment localiser un point dans l'espaceΓen fonction d'une structure de données. La section 2.6 présente un algorithme général pour supprimer un germe dans un DVP 3D. Des calculs de complexité seront faits dans la section 2.7. Enfin nous donnons une conclusion et une conjecture dans la section 2.8.

2.2 Définitions et propriétés

Dans cette sectionΓnous allons exposer les quelques définitions de base et les propriétés fondamentales attachées aux diagrammes de Voronoï. Ce sont ces définitions et propriétés qui rendent les DVP si intéressants à étudier en géométrie

algorithmiqueΓet dans notre cas à utiliser en analyse d'images.

Définition 2.1 (Région de Voronoï) *Soit S un ensemble fini de points de $\mathbb{R}^3$. Soit p un point de S. La région de Voronoï $Vor_S(p)$ associée à p est l'ensemble des points de $\mathbb{R}^3$ plus proches de p que de tous les autres points de S :*

$$Vor_S(p) = \{x \in \mathbb{R}^3, d(x,p) \leq d(x,q), \forall q \in S - p\}$$

où d est la distance euclidienne.

Soit p et q deux points de S. Si on note $\Pi(p,q)$ le plan médiateur de p et de q défini dans la section 4.3.1 du chapitre 4Γet $H(p,q)$ le demi-espace limité par le plan $\Pi(p,q)$ et contenant le point pΓalors on a :

$$Vor_S(p) = \bigcap_{q \in S-p} H(p,q) \tag{1}$$

Par conséquentΓchaque région de Voronoï est polyédrale et convexe comme intersection de demi-espaces. On peut donc écrire :

Propriété 2.1 (Polyèdre de Voronoï) *Toutes les régions de Voronoï sont polyédrales et convexes (Figure 3).*

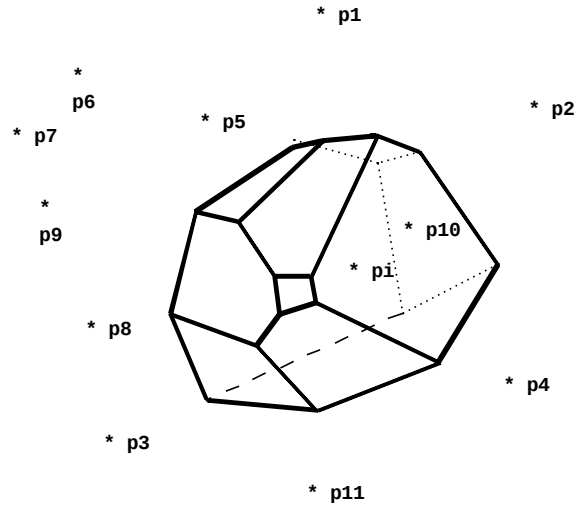


FIG. 3 - Polyèdre de Voronoï associé au site p_i : les sites p_jΓ$j \in \{1, \ldots, 11\}$ sont les sites voisins de p_i (Définition 2.3).

On pourra désormais parler de **polyèdres** de Voronoï. De plus, la définition 2.1 s'étend en dimension quelconque. Les polyèdres de Voronoï deviennent des polytopes de Voronoï.

A partir de la définition 2.1, nous pouvons maintenant introduire la notion de diagramme de Voronoï 3D pour un ensemble fini de points, ou diagramme de Voronoï ponctuel 3D, que nous noterons DVP 3D. Il ne faudra pas confondre cette notion avec la notion plus générale de diagramme de Voronoï pour un ensemble d'éléments simples ou d'objets, ou diagramme de Voronoï généralisé 3D que nous noterons DVG 3D. Cette notion sera introduite dans le chapitre 4.

Définition 2.2 (Diagramme de Voronoï Ponctuel) *Le diagramme de Voronoï (ou partition de Voronoï) d'un ensemble fini de points S est l'ensemble de tous les polyèdres de Voronoï de S :*

$$VOR(S) = \bigcup_{p \in S} Vor_S(p).$$

A partir de la définition 2.2, il est intéressant de noter que le DVP 3D est une partition de l'espace. Cette notion sera très importante dans le chapitre 3 où nous nous intéresserons aux partitions volumiques adaptées à des images tridimensionnelles.

Le diagramme de Voronoï induit une notion de voisinage à travers les faces des polyèdres de Voronoï. Cette notion est codée dans le graphe de Delaunay. Formellement, nous pouvons définir le graphe de Delaunay comme suit :

Définition 2.3 (Graphe de Delaunay) *Le graphe dual du diagramme de Voronoï d'un ensemble fini S de points est le graphe de Delaunay. Deux points de S, p et q, créent une arête dans le graphe de Delaunay (i.e. p et q sont voisins) si, et seulement si, $Vor_S(p)$ et $Vor_S(q)$ sont adjacents dans le diagramme de Voronoï :*

$$DEL(S) = \langle S, E = \{(p,q) \in S^2, Vor_S(p) \cap Vor_S(q) \neq \emptyset\}\rangle.$$

La définition 2.3 permet de donner la définition du voisinage d'un point p au sens de Delaunay.

Définition 2.4 (Voisinage) *Le voisinage au sens de Delaunay d'un point $p \in S$ est défini par :*

$$N_S(p) = \{q \in S \text{ tel que } (p,q) \in E\}$$

où E est l'ensemble des arêtes du graphe de Delaunay.

On peut noter que dans le voisinage (au sens de Delaunay) de p nous n'intégrons pas p.

Une propriété extrêmement intéressante et importante du graphe de DelaunayΓ que nous exploiterons dans le chapitre 3Γ montre que ce graphe peut aussi être considéré comme une partition en tétraèdres de l'enveloppe convexe de l'ensemble des points de S (Figure 5). La propriété suivante précise cette notion :

Propriété 2.2 (Diagramme de Delaunay) *Sous l'hypothèse (H), le graphe de Delaunay de S peut être aussi considéré comme l'unique tétraédrisation de l'enveloppe convexe de S telle que l'intérieur des sphères circonscrites aux tétraèdres $(p_i, p_j, p_k, p_l) \in S^4$ ne contiennent aucun point de S :*

$$DEL(S) = \{(p_i, p_j, p_k, p_l) \in S^4 \text{ tel que } B(p_i, p_j, p_k, p_l) \cap (S - p_i - p_j - p_k - p_l) = \emptyset\}.$$

Démonstration : Cette propriété est démontrée dans [107]. ■

On pourra donc définir un tétraèdre de Delaunay comme suit :

Définition 2.5 (Tétraèdre de Delaunay) *Un tétraèdre $T = (p_i, p_j, p_k, p_l)$ où p_i, p_j, p_k, et p_l sont dans S est un tétraèdre de Delaunay si, et seulement si, l'intérieur de la sphère circonscrite à T ne contient aucun point de S :*

$$B(p_i, p_j, p_k, p_l) \cap (S - p_i - p_j - p_k - p_l) = \emptyset.$$

Dans un tel cas, les arêtes $[p, q]$, $p, q \in (p_i, p_j, p_k, p_l)$, $p \neq q$, sont des arêtes de Delaunay.

Nous pourrons donc désormais parler de **tétraèdres** de Delaunay (Figure 4). Comme pour les régions de VoronoïΓ la définition 2.5 s'étend en dimension quelconque. Les tétraèdres de Delaunay deviennent des simplexes de Delaunay.

La figure 5 est une illustration du diagramme de Voronoï et de la triangulation de Delaunay pour un ensemble de points du plan.

Introduisons maintenant d'autres propriétés qui seront importantes pour la suite.

Propriété 2.3 *Sous l'hypothèse (H), chaque sommet de Voronoï provient de l'intersection d'exactement quatre polyèdres de Voronoï.*

Propriété 2.4 *Soit s un sommet de Voronoï de $VOR(S)$. Sous l'hypothèse (H), il existe exactement 4 sites générateurs de s et 4 sommets voisins.*

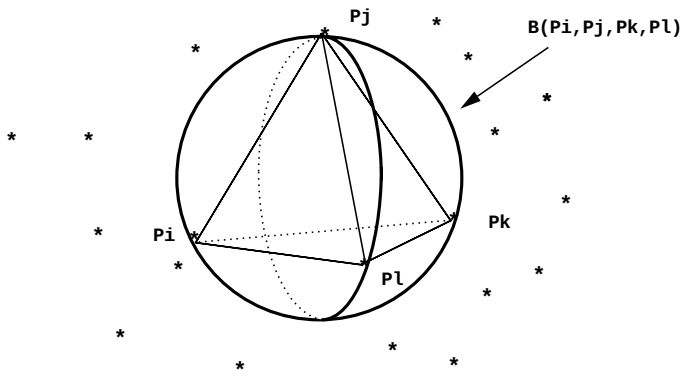


FIG. 4 - Tétraèdre de Delaunay : la sphère $B(p_i, p_j, p_k, p_l)$ est vide de tout point de S.

Démonstration : s est voisin de 4 sommets de Voronoï car (propriété 2.3) s est à l'intersection d'exactement quatre polyèdres de Voronoï. Les quatre sites générateurs de ces quatre polyèdres sont les quatre sites créant s (i.e les quatre sites définissant le tétraèdre dual de s). ■

Cette propriété a permis à Bowyer [30] de trouver un codage astucieux de la structure du diagramme de Voronoï comme nous le verrons dans la section 2.3.

Propriété 2.5 *Soit p un point de S. Si q est le point le plus proche (au sens de la distance euclidienne) de p parmi les points de $S - p$, alors l'arête [p,q] est une arête de DEL(S).*

Cette propriété est très utile pour les algorithmes incrémentaux comme nous le verrons dans la section 2.3.

Une autre propriété intéressante montre qu'il est facile de construire l'enveloppe convexe des points de $S\Gamma CONV(S)\Gamma$à partir de $VOR(S)$.

Propriété 2.6 *Soit p un point de S. Le polyèdre convexe $Vor_S(p)$ est non borné si, et seulement si, p appartient à $CONV(S)$.*

C'est cette propriété qui permet de dire que $DEL(S)$ est une partition en tétraèdres de $CONV(S)$.

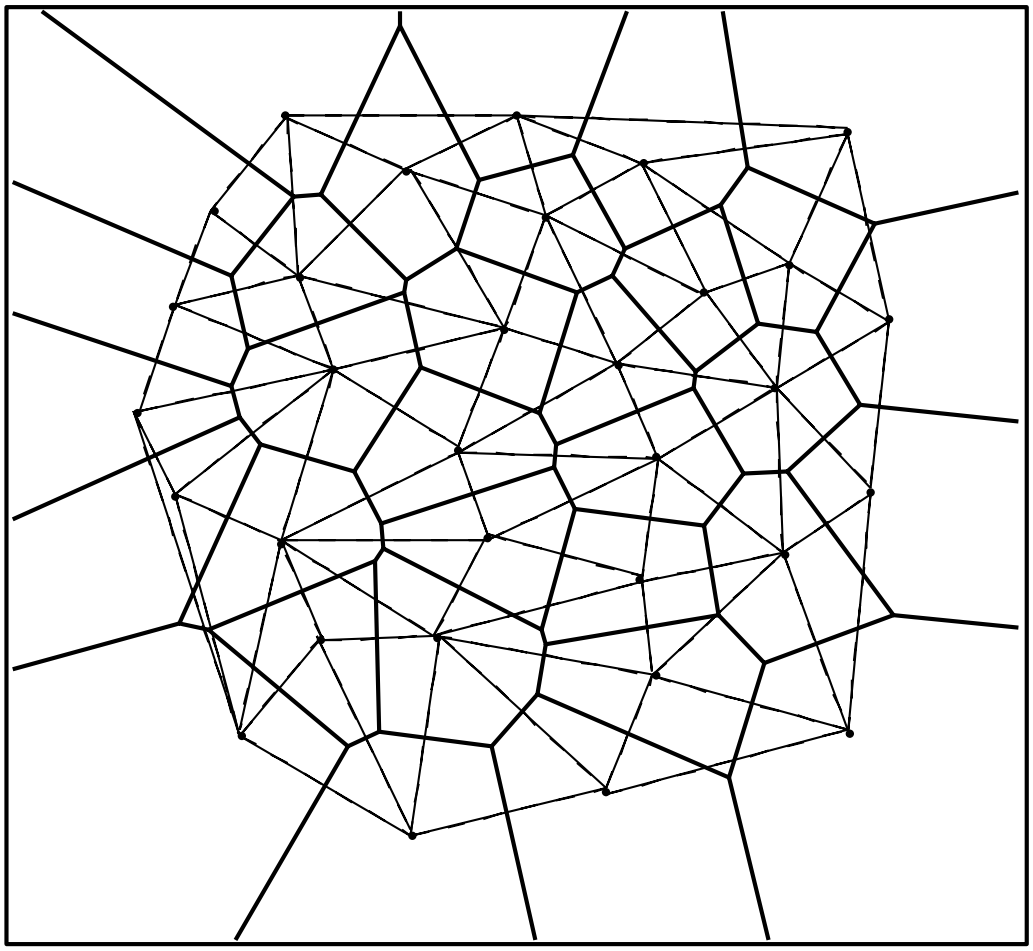


FIG. 5 - Diagramme de Voronoï et triangulation de Delaunay.

2.3 Construction du DVP 3D

Dans la littérature on trouve de nombreuses méthodes de construction du diagramme de Voronoï dans le plan [62Γ 101Γ 127Γ 89Γ 61Γ 102]. Dans l'espace (3D) et en dimension dΓ les articles sont moins nombreux. Les méthodes utilisées sont essentiellement de deux types :

- méthodes globalesΓ
- méthodes incrémentales.

Les méthodes seront dites globales si la connaissance de tous les points doit être donnée avant de commencer la construction du diagramme de Voronoï.

Les méthodes seront dites incrémentales si cette connaissance préalable n'est pas nécessaire. Contrairement aux méthodes globalesΓles méthodes incrémentales sont

intrinsèquement dynamiques. Ces deux types de méthode sont donc fondamentalement différents et leur utilisation dépend entièrement des applications envisagées.

Pour ce qui nous concerneΓnous verrons dans le chapitre 3 qu'une gestion dynamique du diagramme de Voronoï sera nécessaire. C'est pourquoi nous mettrons l'accent dans cette section sur les algorithmes incrémentaux.

Avant de décrire plus en détail différentes méthodes de construction du DVP 3DΓ on peut noter que l'idée maîtresse est le souci de s'affranchir de toute notion d'ordre. En effetΓl'algorithme de Green et Sibson proposé dans [62] est intrinsèquement 2D puisqu'il utilise fortement la possibilité d'ordonner les sommets d'un polygone sous la forme d'un polygone simple comme le montre la figure 6Γce qui ne peut pas se généraliser en dimension $d\Gamma d \geq 3$.

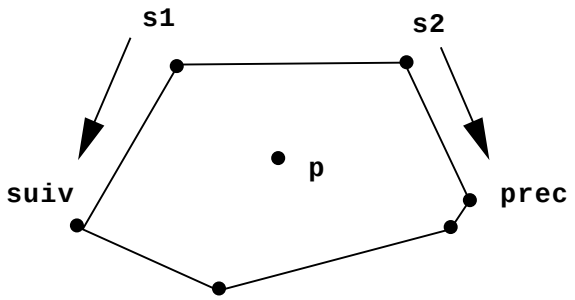


FIG. 6 - Polygone simple et structure de données 2D. *Extrait de [39].*

2.3.1 Méthodes globales

Nous allons donner ici une liste (non exhaustive) des algorithmes globaux demandant la connaissance de tous les sites avant de calculer le diagramme de Voronoï.

"Divide and Conquer"

La célèbre méthode "divide and conquer" initialement proposée par Preparata dans [107] est une méthode globale. L'idée de base de cet algorithme est de diviser récursivement le problème en deux sous-problèmes de même tailleΓpuis de fusionner les sous-diagrammes de Voronoï ainsi obtenus. Cet algorithme peut être décrit dans ses grandes lignes comme suit (pour plus de détails voir [107Γ51]) :

1. Diviser récursivement le problème en deux sous-problèmes.

2. Calculer les DVP des sous-problèmes.

3. Fusionner les DVP ainsi obtenus (Figure 7).

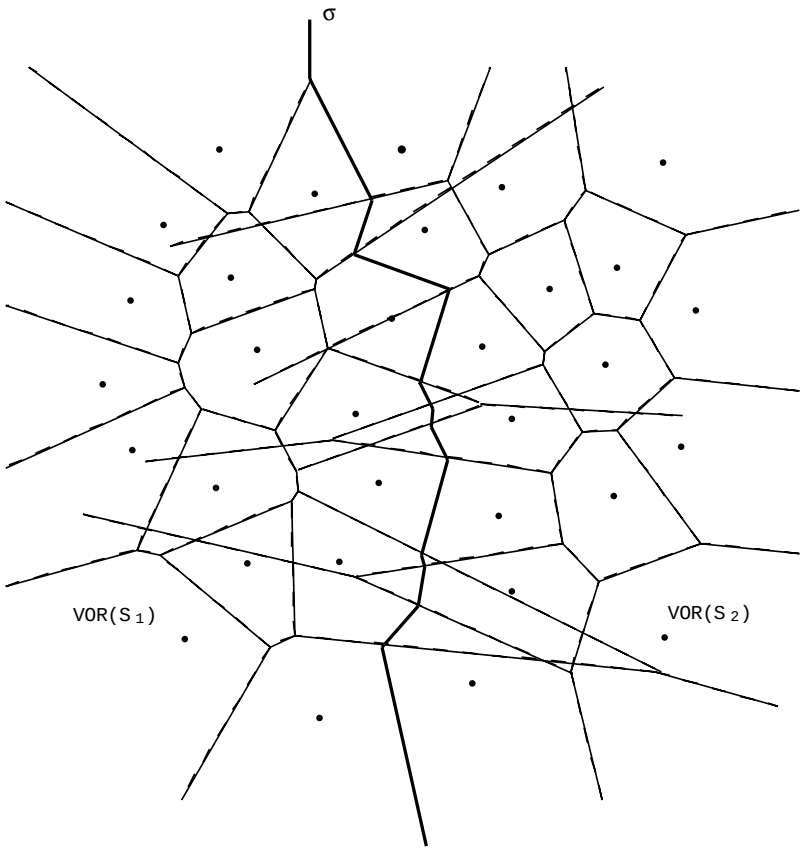


FIG. 7 - Fusion des deux diagrammes de Voronoï $VOR(S_1)$ et $VOR(S_2)$ (arête σ)Γ pour calculer $VOR(S_1 \cup S_2)$ dans l'algorithme "divide and conquer".

C'est dans cette étape de fusion que l'algorithme est le plus difficile à mettre en œuvre. Il a été montré que cet algorithme est optimal en 2D avec une complexité en $O(n \log n)$ [107]. Il est aussi optimal en 3D avec une complexité en $O(n^2)$ [51] (Voir aussi 2.7.1 Corollaire 2.2). L'algorithme "divide and conquer" a été récemment implémenté en 3D par Elbaz [51] à l'aide d'une notion de carte introduite par Spehner [125]. Il est à noter que l'optimalité de cet algorithme dans le pire des cas est associée à de grandes difficultés dans la mise en oeuvre informatique.

Projection d'une enveloppe convexe

Cette méthode consiste à ramener la construction du DVP d'un ensemble S de points de $\mathbb{R}^d$ à la construction d'une enveloppe convexe dans un espace de dimension supérieure ($\mathbb{R}^{d+1}$).

L'initiateur de cette méthode est Brown [33]. Il utilise la projection stéréographique d'une sphère sur le plan pour la construction du diagramme de Voronoï en 2D.

Dans [49] Edelsbrunner utilise la projection orthogonale d'un paraboloïde de révolution sur le plan pour la construction en 2D. Une implémentation effective en 3D a été réalisée par Buckey [35].

En dimension d cette méthode consiste à relever les points de S sur un hyperparaboloïde de révolution dans $\mathbb{R}^{d+1}$. On calcule alors l'enveloppe convexe de ces points ainsi relevés. On obtient donc un polytope de $\mathbb{R}^{d+1}$ dont les hyperfaces sont des d-simplexes. On démontre que la projection orthogonale des d-simplexes du polytope constituant l'enveloppe convexe de $\mathbb{R}^{d+1}$ donne les d-simplexes de Delaunay cherchés dans $\mathbb{R}^d$.

La figure 8 illustre ce principe pour le calcul du diagramme de Delaunay dans le plan. Cet algorithme peut être décrit dans ses grandes lignes et en 3D comme suit (pour plus de détails voir [33 35]) :

1. Relever les points de S sur un hyperparaboloïde de révolution dans $\mathbb{R}^4$.

2. Calculer alors l'enveloppe convexe C de ces points ainsi relevés.

3. Projeter orthogonalement les faces du polytope C de $\mathbb{R}^4$ (i.e. des tétraèdres) dans $\mathbb{R}^3$ (ces tétraèdres ainsi projetés sont les tétraèdres de Delaunay).

Dans son article Buckey [35] a choisi un algorithme de type "divide and conquer" pour le calcul de l'enveloppe convexe dans $\mathbb{R}^4$ mais d'énormes difficultés apparaissent pour l'implémentation effective.

A notre sens les algorithmes utilisant la projection d'une enveloppe convexe en dimension supérieure présentent un intérêt théorique puisqu'on arrive à relier le nombre de $(d-k)$-faces de Voronoï aux k-faces d'un polytope en dimension supérieure [49].

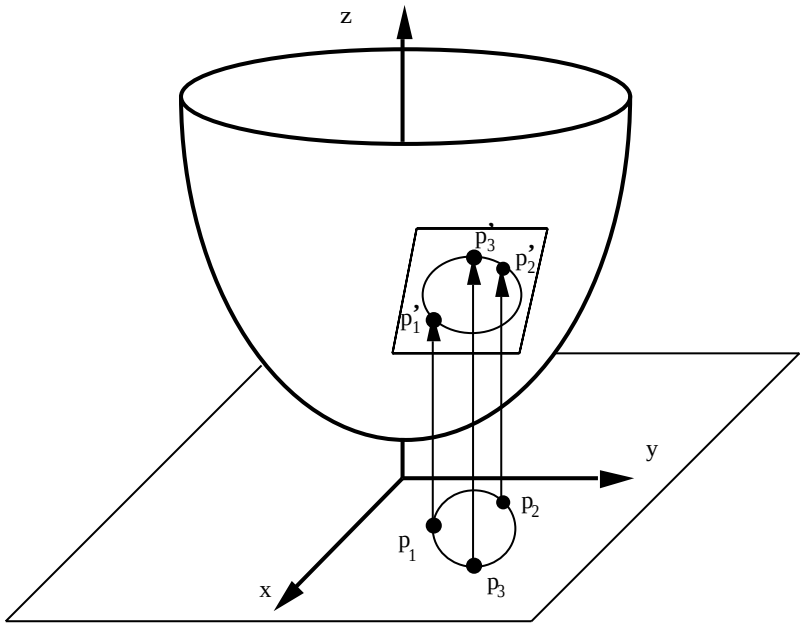


FIG. 8 - Projection d'une enveloppe convexe pour calculer le diagramme de Delaunay.

Dans l'algorithme que nous venons de décrire, les 3-faces (tétraèdres) du polytope de $\mathbb{R}^4$ sont mises en relation avec les tétraèdres de Delaunay et donc, par dualité, avec les 0-faces (sommets) de Voronoï.

L'avantage est que les résultats sur la complexité des polytopes sont relativement nombreux, notamment en ce qui concerne le théorème de la borne supérieure (*upper bound theorem*) [91], liant le nombre maximum de k-faces d'un polytope à celui obtenu par un polytope cyclique.

Ce théorème permet d'obtenir le calcul de la complexité, dans le pire des cas, du nombre de k-faces d'un polytope de $\mathbb{R}^d$, $0 \leq k \leq d$.

L'intérêt pratique ne nous semble pas évident, puisque la tendance serait plutôt de ramener des problèmes de géométrie de $\mathbb{R}^d$ dans un espace de dimension inférieure ($\mathbb{R}^{d-1}$ par exemple) afin de réduire la taille du problème [116].

Autres algorithmes

On peut également citer l'algorithme 2D de Mauss [89], généralisé en dimension quelconque et analysé en moyenne par Dwyer [48] (section 2.7.2). Cet algorithme a aussi été parallélisé sur une machine à quatre transputers par Moreau [100]. La

possibilité de paralléliser qu'offre cette méthode est sans doute son point fort.

D'autres méthodes moins connues et aussi moins récentes ont été publiées. Le problème de ces méthodes est leur mauvaise complexité théorique (complexité $\geq O(n^3)$). Cela s'explique car le caractère local de Voronoï n'est pas suffisamment exploité dans leurs approches. Parmi ces algorithmes on trouve l'algorithme de Tanemura & All [130] celui de Finney [53] et celui de Brostow [32]. Ces trois derniers algorithmes ont été détaillés dans [7]; nous ne les expliciterons donc pas ici.

Méthodes discrètes

Des méthodes dérivées de la géométrie discrète peuvent être utilisées pour calculer une approximation du diagramme de Voronoï. Une illustration en a été donnée dans [94].

Les distances discrètes utilisées sont variables : la distance aux quatre plus proches voisins notée d4 (Cityblock) aux huit plus proches voisins notée d8 (Chessboard) toutes les distances du chanfrein (par exemple : chanfrein 3-4 chanfrein 5-7-11) [39 133].

Une extension immédiate de ces algorithmes en 3D est possible. En effet les distances discrètes en 3D ont été étudiées par Borgefors dans [28].

Les problèmes posés par l'utilisation des distances discrètes en géométrie algorithmique sont multiples. Premièrement le résultat discret est une approximation du résultat continu obtenu par les méthodes précédentes et les méthodes incrémentales (section 2.4). Il est donc nécessaire d'avoir une bonne estimation de l'erreur ainsi faite au vu des applications envisagées. Deuxièmement le diagramme de Voronoï n'est pas structuré dans un environnement de graphes (comme nous le verrons dans la structure de données que nous présentons dans la section 2.4.2) permettant le calcul rapide de paramètres [11]. Et troisièmement le calcul n'est pas dynamique. L'ajout d'un point nécessite de nouveau le calcul de tout le diagramme de Voronoï.

Toutes ces remarques font que nous avons préféré utiliser les algorithmes incrémentaux que nous présentons maintenant.

2.3.2 Méthodes incrémentales

Les algorithmes incrémentaux peuvent être résumés comme suit. Supposons que nous ayons construit le DVP de S. L'algorithme consiste à insérer un nouveau site

p dans $VOR(S)$ pour obtenir $VOR(S \cup p)$:

- Insérer un nouveau site dans $VOR(S)$ (Figure 9).

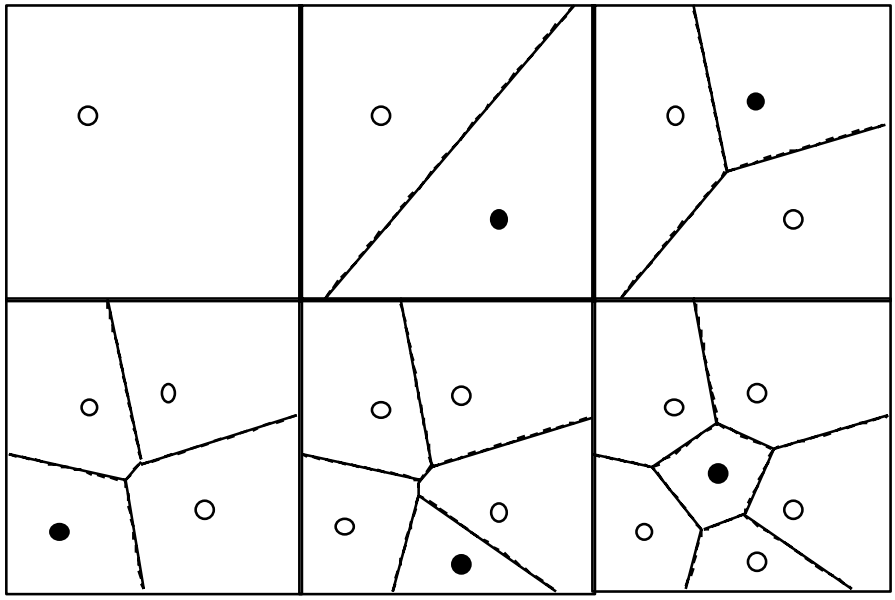


FIG. 9 - Construction du diagramme de Voronoï par ajouts successifs de germes et modification locale du diagramme. Le germe noir correspond au dernier germe ajouté entre deux étapes successives de la construction du diagramme.

L'insertion d'un nouveau site modifie $VOR(S)$ de manière à pouvoir créer la région de Voronoï notée $Vor_{S \cup p}(p)$ dans $VOR(S \cup p)$: des polyèdres de Voronoï de $VOR(S)$ seront à modifier et de manière duale des tétraèdres de Delaunay seront à supprimer. Cette insertion se fait en 2 grandes étapes :

1. Chercher un premier tétraèdre à modifier (Watson) ou un premier polyèdre à modifier (Bowyer).

2. Mettre à jour $VOR(S \cup p)$.

L'étape 1 est une étape qui n'est pas locale (section 2.5) alors que l'étape 2 est locale en général (proposition 2.7 section 2.7.2). L'étape 1 sera détaillée dans la section 2.5. Le calcul effectif de la complexité des algorithmes incrémentaux est reporté dans la section 2.7.

Notons que nous avons développé un nouvel algorithme incrémental en 1990 [7Γ9]. Nous ne le détaillerons pas dans ce mémoire de thèse puisque l'algorithme de BowyerΓ avec les modifications que nous y avons apportéesΓs'est avéré plus performant.

Nous développerons donc dans la section suivante les algorithmes de Watson [140] pour la construction de la partition de DelaunayΓet de Bowyer [30] pour la construction de la partition de Voronoï.

2.4 Méthodes incrémentales

2.4.1 Algorithme de Watson

L'algorithme de Watson concerne plus spécialement la construction directe du diagramme de Delaunay (le diagramme de Voronoï est retrouvé avec une structure de données sous-jacente suffisamment riche). Cet algorithme a été proposé par Watson en 1981 [140]. Il a été implémenté en 3D par Field en 1986 [52]. Une amélioration lui a été apportée par Schmitt et Borouchaki en 1990Γnotamment en ce qui concerne la structure de données [120].

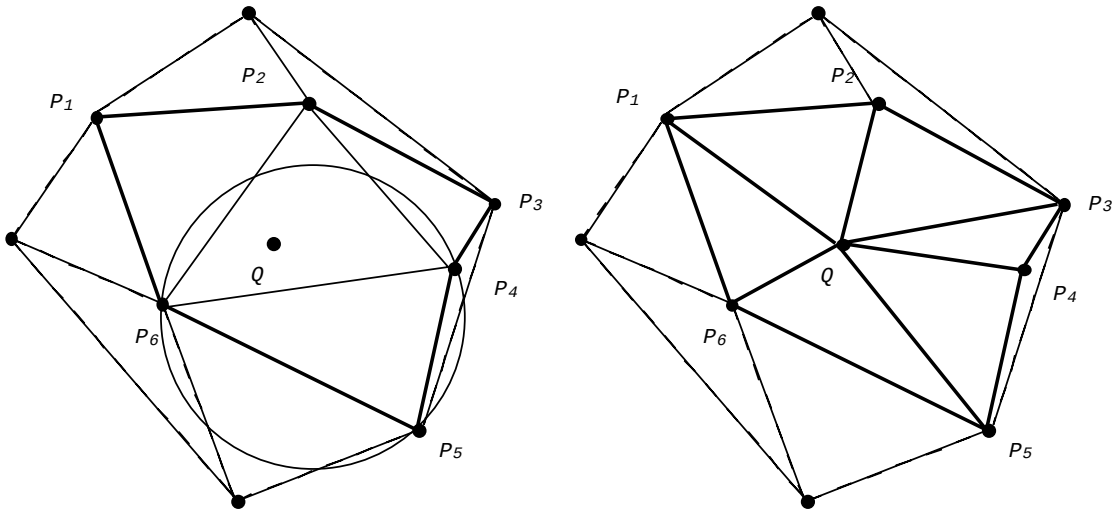
On peut brièvement décrire l'algorithme original de Watson de la manière suivante. Supposons qu'on ait construit le diagramme de Delaunay pour n sites. On veut ajouter dans la structure un nouveau site p_{n+1}. L'algorithme de Watson procède alors comme suit en 3D (pour plus de détails [140Γ52Γ120]) :

1. Chercher un tétraèdre T de $DEL(S)$ dont la sphère circonscrite contient le nouveau site.

2. Chercher tous les tétraèdres dont la sphère circonscrite contient le nouveau site. L'ensemble de ces tétraèdres forme le polyèdre étoilé $PE(p_{n+1})$ (Définition 2.6Γ section 2.6).

3. Retétraédriser le polyèdre étoilé.

Dans l'étape 1Γle tétraèdre T existe bien (Propriété 2.5Γ2.2)Γmais sa recherche n'est pas locale. Cette recherche peut se faire par un algorithme de descente en gradient que nous détaillons dans la section 2.5.



(a) (P_2, P_4, P_6) est le triangle contenant le site de départ (étape 1), (P_1, P_2, P_6), (P_2, P_3, P_4), (P_4, P_5, P_6) sont les autres triangles à supprimer. Par conséquent les sites $(P_1, P_2, P_3, P_4, P_5, P_6)$ forment $PE(Q)$.

(b) On démontre que $PE(Q)$ est étoilé par rapport à Q et que pour calculer la nouvelle triangulation il suffit de relier le site Q à tous les P_i, $i \in \{1, \ldots, 6\}$.

FIG. 10 - Insertion du nouveau site Q dans la structure de Delaunay par la méthode de Watson.

L'étape 2 se fait par un parcours (local en général) dans la structure de données. Les tétraèdres trouvés ne sont plus des tétraèdres de Delaunay (Définition 2.5Γsection 2.2). Ces tétraèdres sont donc à supprimer et forment le polyèdre étoilé associé au nouveau site inséré.

Pour l'étape 3Γon démontre que retétraédriser le polyèdre étoilé $PE(p_{n+1})$ revient à supprimer toutes les arêtes intérieures à $PE(p_{n+1})$Γet à créer toutes les arêtes liant le nouveau site à tous les sommets de $PE(p_{n+1})$. La technique de démonstration de ce fait rejoint celle utilisée dans la proposition 2.5 de la section 2.6.

La figure 10 est une illustation en 2D de l'étape 3 de l'algorithme de Watson.

Nous n'avons pas implémenté cet algorithme puisque nous nous intéressons davantage au calcul du DVP et que des améliorations ont déjà été réalisées par Schmitt et Borouchaki [120] en 1990.

2.4.2 Algorithme de Bowyer

L'algorithme de Bowyer construit directement le diagramme de Voronoï. L'originalité de cet algorithme réside dans la structure de données proposée.

Structure de données

La structure de données de Bowyer est liée à la propriété 2.4 exposée dans la section 2.2Γà savoir qu'un sommet de Voronoï est associé à 4 sites générateurs et à 4 sommets voisins. En effetΓl'information contenue dans cette propriété est codée dans la structure de données. Par conséquentΓà un sommet de Voronoï sΓon associe :

- les 4 sommets de Voronoï voisins de sΓ
- les 4 sites de Delaunay créateurs de s.

L'intérêt immédiat est que l'information à coder est bornée. La figure 11 est une illustration de cette structure de données.

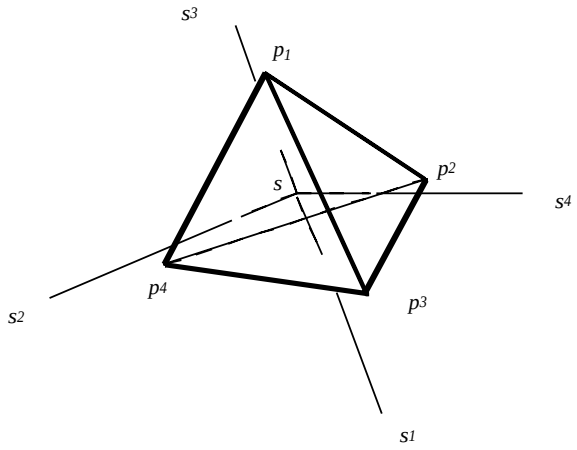


FIG. 11 - Structure de données de Bowyer. Le sommet de Voronoï s a pour sommets voisins s_1Γs_2Γs_3Γet s_4 et pour germes créateurs p_1Γp_2Γp_3Γet p_4

De plusΓpour pouvoir accéder au graphe de Voronoï à partir des germes de DelaunayΓnous avons associé à un germe pΓun sommet s quelconque appartenant à $Vor_S(p)$.

Avec une telle structure de données on peut non seulement retrouver le graphe de DelaunayΓmais aussi :

1. l'ensemble des sommets constituant un polyèdre de VoronoïΓ$Vor_S(p)$Γ

2. l'ensemble des sommets constituant une face quelconque de $Vor_S(p)$Γ

3. les deux sommets constituant une arête quelconque de $Vor_S(p)$.

Les algorithmes pour calculer ces ensembles exploitent très fortement la structure de données. La connaissance de tous ces ensembles est indispensable pour une exploitation efficace du DVP 3D que nous présentons dans les chapitres qui suivent. Les faces obtenues dans le point 2 sont des polygones qui sont représentés sous forme de polygones simples pour des raisons de visualisation.

Algorithme

Cet algorithme incrémental a été proposé et implémenté par Bowyer en 1981 [30]. Nous lui avons apporté quelques modifications en 1992 [9] notamment pour la recherche du premier sommet à supprimer (étape 1 des algorithmes incrémentaux). En effet dans [30]Γpour effectuer une telle rechercheΓBowyer est obligé de maintenirΓ pour chaque siteΓla liste (à priori non bornée) de tous ses voisins au sens de Delaunay.

Notre amélioration permet d'alléger la structure de données présentée par Bowyer en supprimant toutes ces listes (section 2.5.2).

AinsiΓil nous a été possible de calculer le DVP 3D pour un ensemble de 80000 sitesΓreprésentant environ 560000 tétraèdresΓsur une Silicon Graphics Indigo. Les performances pratiques sont indiquées en fin de section.

L'algorithme de Bowyer peut être décrit comme suit. On suppose qu'on insère un (n+1)ième site p_{n+1} au DVP calculé sur n sites ($VOR(S)$).

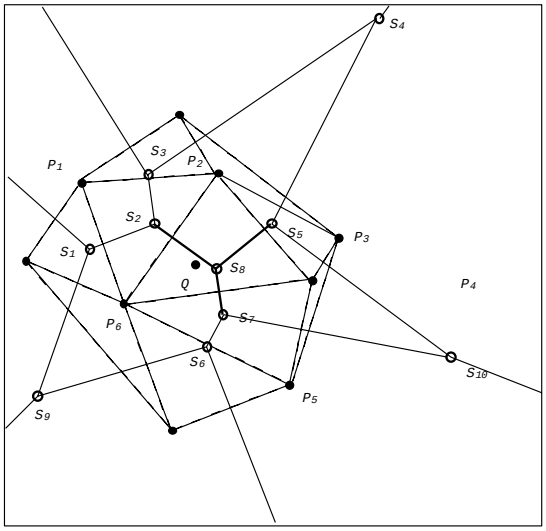
1. Chercher un premier sommet de VoronoïΓsΓà supprimer.

2. Chercher tous les autres sommets de Voronoï à supprimer. Nous noterons l'ensemble de ces sommets : $E(p_{n+1})$.

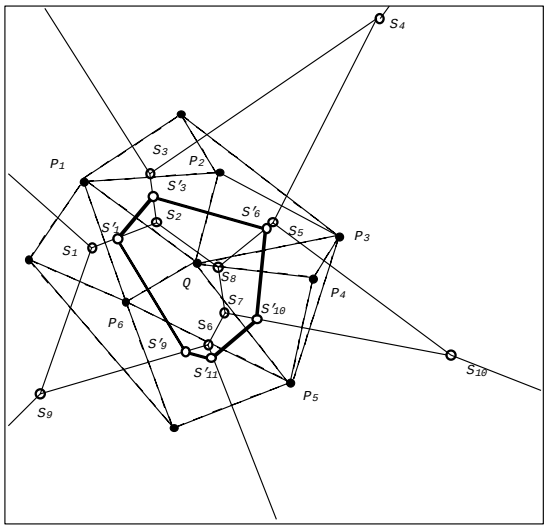
3. Créer tous les nouveaux sommets de Voronoï et leurs relations de voisinage.

Comme dans l'algorithme de WatsonΓdans l'étape 1Γs existe bien (Propriété 2.5Γ section 2.2)Γ mais sa recherche n'est pas locale. Les détails sont donnés dans la section 2.5.

Dans l'étape 2 la recherche de tous les sommets de Voronoï à supprimer s'effectue par un parcours (local en général) dans la structure de données. De tels sommets sont plus proches du nouveau site que des sites créateurs de s.



(a) Le sommet de Voronoï S_8 est intérieur au triangle contenant Q. Les autres sommets de Voronoï à supprimer sont les sommets (S_2, S_5, S_6, S_7).



(b) Le sommet S_2 est à supprimer. Il est voisin de S_1, S_3, et S_8. On montre que les sommets à créer sont situés sur les arêtes de Voronoï $\{S_i, S_j\}$, où S_i est à supprimer et S_j est à conserver. Dans cet exemple, on crée donc un nouveau sommet $S_1^{'}$ sur $\{S_2, S_1\}$. Ce sommet est le centre du cercle circonscrit au triangle (P_1, P_6, Q). De même, on crée un sommet $S_3^{'}$ sur $\{S_2, S_3\}$. Enfin, on ne crée pas de sommet sur l'arête $\{S_2, S_8\}$, car les deux sommets sont à supprimer.

FIG. 12 - Insertion du nouveau site Q dans la structure de Voronoï par la méthode de Bowyer.

L'étape 3 consiste à parcourir tous les sommets de $E(p_{n+1})$. Soit s un sommet de $E(p_{n+1})\Gamma s_1\Gamma s_2\Gamma s_3\Gamma$et $s_4\Gamma$ses quatre sommets voisins et $p_1\Gamma p_2\Gamma p_3$ et p_4 ses quatre germes créateurs. Si $s_i\Gamma i = 1, \ldots, 4\Gamma$n'appartient pas à $E(p_{n+1})\Gamma$on crée un nouveau sommet s centre du nouveau tétraèdre défini par les trois germes $p_i\Gamma p_j$ et p_k créant l'arête $[s_i, s]$ et par le germe inséré p_{n+1}. Si $s_i\Gamma i = 1, \ldots, 4$ appartient à $E(p_{n+1})\Gamma$ alors on ne fait rien.

Une illustration en 2D de l'étape 3 est donnée en figure 12.

La différence entre l'algorithme de Bowyer et l'algorithme de Watson réside dans le critère utilisé pour savoir si un sommet de Voronoï est à supprimer de $VOR(S)$ ou non. Watson cherche si les simplexes de Delaunay sont à hypersphère vide et Bowyer

cherche si les sommets de Voronoï sont plus proches du nouveau site à insérer que de ses germes créateurs. Les deux ensembles obtenusΓ $E(p_{n+1})$ et $PE(p_{n+1})$ sont duaux. On peut donc dire que l'algorithme de Watson qui construit le diagramme de Delaunay est équivalent à l'algorithme de Bowyer qui construit le diagramme de VoronoïΓen supposant que la structure de données est suffisamment riche.

Résultats

Nous avons implémenté l'algorithme de Bowyer en apportant la modification que nous avons déjà mentionnée. Les temps CPU indiqués ici ont été obtenus sur une Silicon Graphics Indigo R 4000 (85 MIPS). La distribution des points est uniforme dans le carré unité en 2DΓet dans le cube unité en 3D. L'étape 1 de l'algorithme de Bowyer est accélérée par une localisation de type octreeΓque nous détaillons dans la section 2.5. Ces temps sont reportés dans les deux tableaux suivants :

Temps CPU obtenus sur une Silicon Graphics Indigo en 3D.

Points	1000	5000	10000	40000	50000	60000	70000	80000
CPU	< 1"	4"	10"	1'10"	1'30"	1'50"	2'10"	2'50"

Temps CPU obtenus sur une Silicon Graphics Indigo en 2D.

Points	10000	40000	50000	60000	70000	80000
CPU	2"	8"	10"	12"	14"	16"
Points	100000	200000	300000	400000	500000	600000
CPU	20"	42"	1'10"	1'34"	2'9"	4'

2.5 Localisation d'un point dans l'espace

Dans les algorithmes incrémentauxΓnous avons vu que l'étape 2 est une étape globale dans la construction du diagramme de Voronoï. Cette étape consiste à localiser un point dans l'espace. En effet le problème peut être posé en ces termes : étant donné $VOR(S)$ et un nouveau point p_{n+1}Γtrouver le polyèdre $Vor_S(q)$ contenant p_{n+1}Γou trouver le tétraèdre $T \in DEL(S)$ contenant p_{n+1}.

L'objet de cette partie est de décrire cette étape de localisation liée à une structure de données. Plusieurs méthodes peuvent être utilisées :

1. localisation dans le graphe de DelaunayΓ

2. localisation dans le graphe de Voronoï

3. localisation dans un octree.

Pour les deux premières méthodes, une descente en gradient dans le graphe de Delaunay (cas 1) ou de Voronoï (cas 2) sera utilisée. Dans la troisième méthode, nous emploierons une gestion dynamique dans un octree.

Les meilleurs résultats sont obtenus par la dernière méthode qui a une complexité théorique plus faible. Le gain de temps obtenu s'effectue aux dépens d'une structure de données plus importante en mémoire.

2.5.1 Localisation dans le graphe de Delaunay

Cette méthode a été proposée dans l'article de Green et Sibson [62]. Elle revient à chercher le point p de S le plus proche du point p_{n+1} à insérer. L'algorithme est le suivant :

1. Initialisation : choisir un site p quelconque de S.

2. Si pour tout q dans $N_S(p)$, $d(p, p_{n+1}) \leq d(q, p_{n+1})$, alors p est le point cherché et l'algorithme a convergé.

3. Sinon prendre q dans $N_S(p)$ tel que $d(p, p_{n+1}) > d(q, p_{n+1})$, poser $p = q$ et retourner en 2.

La validité de cet algorithme repose sur les deux propositions suivantes :

Proposition 2.1 *Soit S un ensemble de points. Soit $p_{n+1} \notin S$. Si $\forall q \in N_S(p)$, $d(p, p_{n+1}) \leq d(q, p_{n+1})$, alors p est le point le plus proche de p_{n+1}.*

Démonstration : En effet, la condition de cette proposition montre que p_{n+1} appartient à $Vor_S(p)$ (Définition 2.1, section 2.2). ■

De plus, comme p_{n+1} se trouve dans $Vor_S(p)$, $Vor_S(p)$ est à modifier.

Proposition 2.2 *L'algorithme converge en $O(\sqrt[3]{n})$ dans le cas où la distribution des points suit une loi de probabilité uniforme dans $\mathbb{R}^3$ et en $O(n)$ dans le pire des cas.*

Démonstration : p_{n+1} n'a qu'un seul point le plus proche (en excluant les cas d'égalités). De plus, à chaque étape, $d(p_{n+1}, p)$ décroît et il n'y a qu'un nombre fini de germes dans S. Dans le cas où la distribution des points suit une loi de probabilité uniforme dans $\mathbb{R}^3$, l'algorithme converge en $O(\sqrt[3]{n})$ en moyenne, où n est le cardinal de S. En effet, la recherche du point le plus proche aura tendance à suivre la "ligne de plus grande pente", et donc à suivre des diagonales, la plus grande étant en $O(\sqrt[3]{n})$. Dans le pire des cas, la convergence s'effectura en $O(n)$. Par exemple, dans le cas où tous les points sont alignés. ■

Une illustration de l'algorithme est donnée en figure 13.

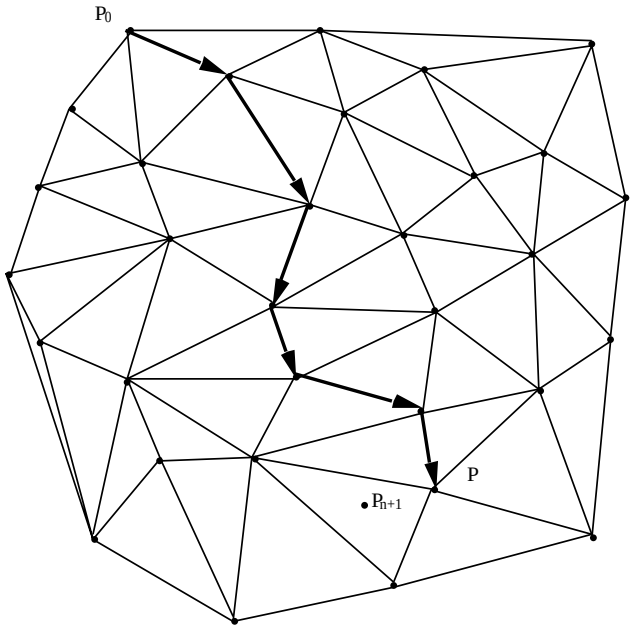
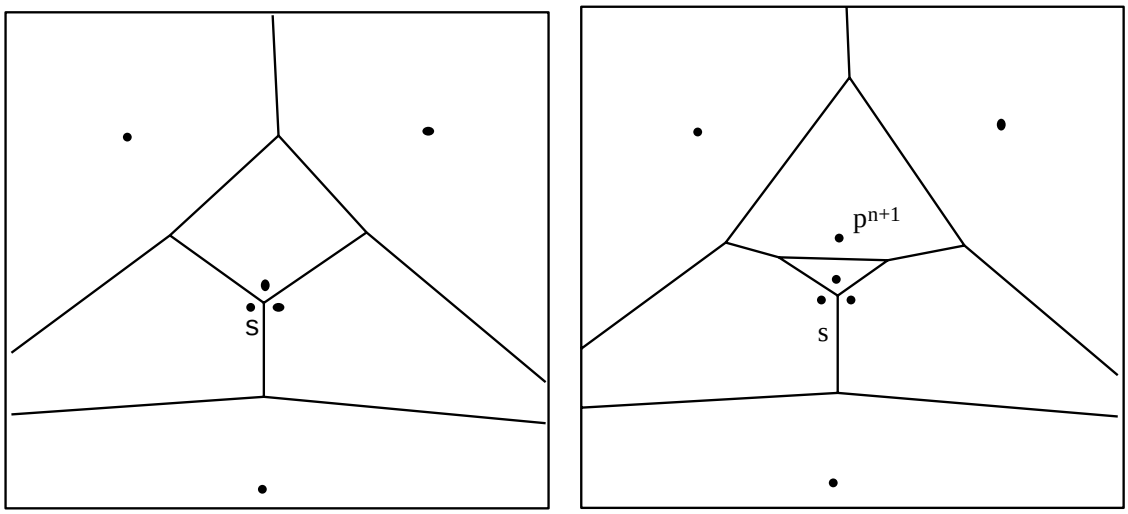


FIG. 13 - Recherche à partir de p_0 et dans le graphe de Delaunay du site p le plus proche de p_{n+1}.

2.5.2 Localisation dans le graphe de Voronoï

Cette méthode a été proposée dans l'article de Schmitt et Borouchaki [120]. Elle consiste à rechercher un sommet à supprimer dans $VOR(S)$ quand on insère un nouveau site p_{n+1}.

Comme l'a remarqué Bowyer [30], le sommet le plus proche du nouveau point n'est pas forcément à supprimer ainsi que le montre l'exemple de la figure 14.



(a) Diagramme de Voronoï

(b) Le sommet s n'est pas à supprimer bien qu'il soit le plus proche du germe p_{n+1}.

FIG. 14 - Problème de localisation dans le diagramme de Voronoï.

Si on utilise le même principe que celui employé dans l'algorithme précédentΓsur le graphe de VoronoïΓil peut donc y avoir un échec. Par conséquentΓBowyer a été obligé de maintenir une structure supplémentaire de voisinage entre les points pour se ramener à l'algorithme précédent relatif au graphe de Delaunay.

Pour éviter de garder cette structure en mémoireΓnous avons utilisé l'algorithme de Schmitt et Borouchaki [120] relatif aux sommets de Voronoï.

Cet algorithme se présente de la manière suivante (on suppose ici que le nouveau site p_{n+1} est inséré dans $CONV(S)$) :

1. Initialisation : choisir un sommet quelconque s de Voronoï dans $VOR(S)$.

2. Soit $p_1\Gamma p_2\Gamma p_3\Gamma$et p_4 les germes créateurs de s. Si pour tout $(p_i, p_j, p_k)\Gamma i \neq j\Gamma$ $j \neq k\Gamma$et $k \neq i\Gamma i, j, k \in \{1, 2, 3, 4\}$ $\langle \vec{ss'}, \vec{p_ip_{n+1}} \rangle < 0\Gamma$où s' est tel que l'arête $[ss']$ ait pour germes créateurs $p_i\Gamma p_j\Gamma$et $p_k\Gamma$alors s est le sommet cherchéΓet l'algorithme converge.

3. Sinon prendre s' dans les sommets voisins de s tel que $\langle \vec{ss'}, \vec{p_ip_{n+1}} \rangle \geq 0\Gamma$poser $s = s'$ et retourner en 2.

La validité de cet algorithme repose sur les deux propositions suivantes :

Proposition 2.3 *Soit S un ensemble de points. Soit $p_{n+1} \notin S$, $p_{n+1} \in CONV(S)$. Soit $s \in VOR(S)$, et p_1, p_2, p_3, et p_4 les germes créateurs de s. Si pour tout (p_i, p_j, p_k), $i \neq j$, $j \neq k$, et $k \neq i$, $i, j, k \in \{1, 2, 3, 4\}$ $\langle \vec{ss'}, \vec{p_i p_{n+1}} \rangle < 0$, où s' est un sommet de Voronoï tel que l'arête $[ss']$ ait pour germes créateurs p_i, p_j, et p_k, alors s est un sommet à supprimer.*

Démonstration : La condition de la proposition implique que p_{n+1} est intérieur au tétraèdre T dual à sΓpar conséquentΓ$p_{n+1} \in B(T)$. DoncΓpar définitionΓT n'est pas un tétraèdre de Delaunay de $DEL(S \cup p_{n+1})$ et le sommet s est bien un sommet à supprimer. ■

Proposition 2.4 *L'algorithme converge en $O(\sqrt[3]{n})$ dans le cas où la distribution des points suit une loi de probabilité uniforme dans $\mathbb{R}^3$ et en $O(n^2)$ dans le pire des cas.*

Démonstration : Il existe un tétraèdre unique contenant p_{n+1} parce qu'on insère p_{n+1} dans $CONV(S)$ et que $DEL(S)$ forme une partition en tétraèdres de $CONV(S)$. De plusΓsoit s un sommet de VoronoïΓet s' un de ses quatre voisins tel que $\langle \vec{ss'}, \vec{p_i p_{n+1}} \rangle \geq 0$. Cette relation définit un demi-espace $H = \{\langle \vec{ss'}, \vec{p_i x} \rangle < 0\}$ avec la propriété $p_{n+1} \notin H$. Si on repasse par s dans l'algorithmeΓalors on produit un cycle dans le graphe de VoronoïΓdonc l'ensemble des demi-espaces H obtenu couvre tout $\mathbb{R}^3$Γce qui est en contradiction avec l'existence d'un tétraèdre contenant p_{n+1}. Par conséquentΓon ne visite qu'une seule fois les sommets de VoronoïΓetΓcomme le nombre des sommets de Voronoï est finiΓl'algorithme converge.

Dans le cas où la distribution des points de S suit une loi de probabilité uniforme dans $\mathbb{R}^3$Γl'algorithme converge en $O(\sqrt[3]{n})$ en moyenneΓoù n est cardinal de S (même argument que dans la proposition 2.2). Dans le pire des casΓla convergence s'effectuera en $O(n^2)$ (section 2.7Γcorollaire 2.1). ■

Une illustration de cette recherche dans le graphe de Voronoï est donnée en figure 15.

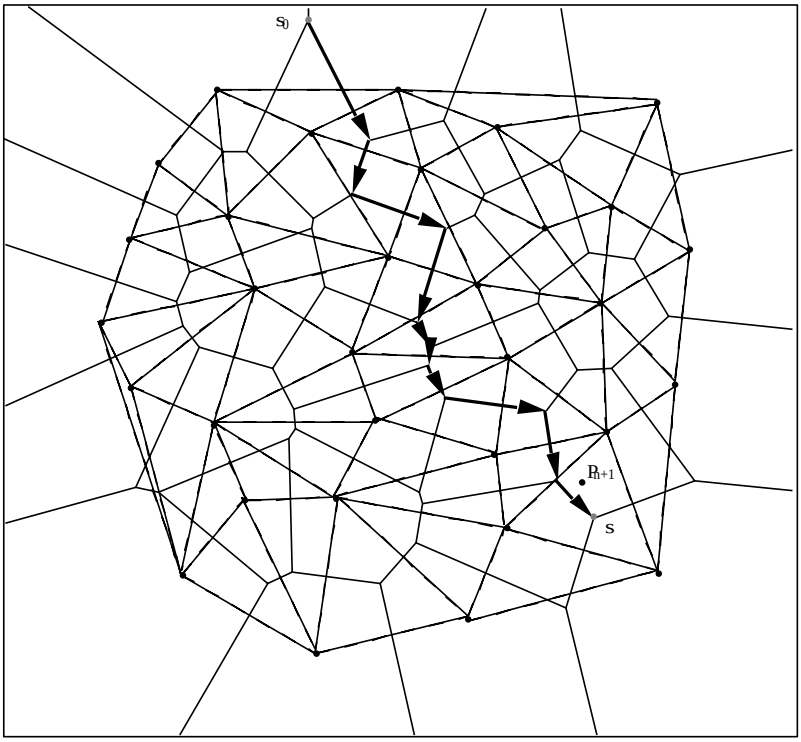


FIG. 15 - Recherche à partir de s_0 dans le graphe de Voronoï du sommet s dont le triangle dual contient p_{n+1}.

2.5.3 Localisation dans un octree

L'idée d'utiliser un octree vient de la méthode d'incrémentation quaternaire de Ohya-Iri et Murota [101] pour le cas 2D. Leur algorithme consiste à placer les points dans un quadtree ce qui leur permet de trouver un ordre pour insérer ces points. Le problème de cette méthode est que la donnée initiale des points est nécessaire et qu'on perd donc le côté dynamique des algorithmes incrémentaux.

Nous proposons par conséquent de construire l'octree dynamiquement. Ainsi l'algorithme peut être décrit comme suit dans ses grandes lignes :

1. Initialisation : insérer p_{n+1} dans l'octree et trouver son site "père" p.
2. Utiliser l'algorithme relatif au graphe de Delaunay en l'initialisant avec p ou l'algorithme relatif au graphe de Voronoï en l'initialisant avec un sommet quelconque de $Vor_S(p)$.

L'étape 1 est réalisée en $O(\log_8 n)$ en supposant que l'octree soit bien équilibré (i.e. la distribution des points de S suit une loi uniforme). L'étape 2 peut se faire

en $O(1)$Γsi le site p est bien localisé par l'octree. Ainsi la recherche du plus proche voisin se fait en $O(\log_8 n)$.

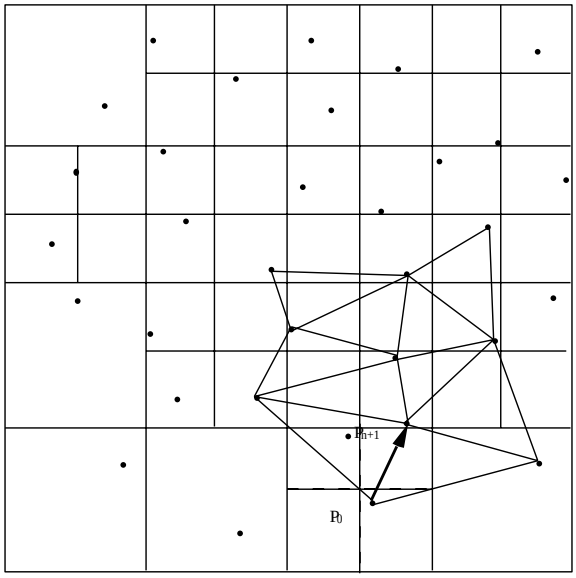


FIG. 16 - Après insertion dans l'octree de p_{n+1}Γrecherche à partir de p_0Γdans le graphe de DelaunayΓdu site le plus proche de p_{n+1}. La localisation préalable par l'octree permet de rendre plus rapide la localisation dans le graphe de Delaunay.

Pour insérer p_{n+1} et lui attribuer son "père" noté pΓdeux cas se présentent :

1. L'insertion de p_{n+1} se fait sur une feuille vide de l'octree. Soit N le noeud père de cette feuille. Par constructionΓ N contient au moins un site dans son arborescence. Le père de p_{n+1} est un de ces sites.

2. L'insertion de p_{n+1} se fait sur une feuille contenant un site p. On crée un nouveau noeud dans l'octree et le père de p_{n+1} est p.

PratiquementΓnous avons constaté que la structure d'octree accélère sensiblement l'algorithme de Bowyer. Pour 70000 germes nous obtenons un gain d'une minute sur 3 minutes environ.

Notre algorithme est illustré en figure 16.

2.6 Suppression

Comme il était intéressant de pouvoir insérer dynamiquement des points dans la structure du diagramme de Voronoï à l'aide d'un algorithme incrémental (section

2.4)Γil nous semble avantageux de pouvoir également supprimer des points interactivement et ce en temps optimal.

Supposons donc que nous ayons construit le DVP de S. L'algorithme consiste à supprimer un site p de $VOR(S)$ pour obtenir $VOR(S-p)$:

- Supprimer un site de $VOR(S)$.

L'étape 1 de localisation des algorithmes incrémentaux (recherche globale) n'existe plus ici. La suppression peut donc se faire localement en général (Théorème 2.1 de cette section). C'est la différence fondamentale entre *suppression* et *insertion*.

L'objet de cette section est de proposer un algorithme de suppression et de donner une analyse de sa complexité. Nous aurons besoin pour commencer de la notion de polyèdre étoilé.

Définition 2.6 (polyèdre étoilé) *Soit S un ensemble de points de $\mathbb{R}^3$ et $DEL(S)$ la tétraédrisation de Delaunay des points de S. Soit p un point de S. Le polyèdre étoilé associé à p, $PE(p)$, est composé des points voisins de p au sens de Delaunay.*

$$Sommets\ de\ PE(p)\ = N_S(p)$$

Les faces de $PE(p)$ sont triangulaires et sont définies comme suit :

$$Faces\ de\ PE(p)\ = \{(p_i,p_j,p_k) \in (S-p)^3\ tel\ que\ (p_i,p_j,p_k,p) \in DEL(S)\}$$

Le polyèdre $PE(p)$ est bien étoilé par définition même de la tétraédrisation de Delaunay. Il est à noter que ce polyèdre n'est pas forcément convexe. L'algorithme de suppression d'un point p peut alors être résumé comme suit :

1. Rechercher le polyèdre étoilé $PE(p)$.
2. Retétraédriser l'intérieur du polyèdre étoilé $PE(p)$.

La validité de notre algorithme repose sur la proposition 2.5 :

Proposition 2.5 *Soit S un ensemble de points de $\mathbb{R}^3$ et $VOR(S)$ le DVP 3D des points de S. Supprimer un point p de S est équivalent à retétraédriser l'intérieur du polyèdre étoilé $PE(p)$.*

Démonstration : Soit $DEL_p(S)$ l'ensemble des tétraèdres de Delaunay de $DEL(S)$ ne contenant pas p.

$$DEL_p(S) = \{(p_i, p_j, p_k, p_l) \in DEL(S) \text{ tel que } p_i, p_j, p_k, p_l \neq p\}$$

Par définition de $DEL(S)$, avant suppression de p, l'intérieur de toutes les sphères circonscrites aux tétraèdres de $DEL_p(S)$ est vide de tout point de S. Par conséquent l'intérieur de toutes ces sphères est vide de tout point de $S-p$. Donc, après suppression du point p dans $DEL(S)$, tous les tétraèdres de $DEL_p(S)$ restent des tétraèdres de $DEL(S-p)$. De plus la tétraédrisation de Delaunay des points de $S-p$ existe et est unique. Donc pour calculer $DEL(S-p)$, il suffit de retétraédriser $PE(p)$. ■

Le caractère local de notre algorithme est contenu dans la proposition suivante :

Proposition 2.6 *Soit S un ensemble de points de $\mathbb{R}^3$. Si un tétraèdre T, intérieur à $PE(p)$, est un tétraèdre de Delaunay de $DEL(N_S(p))$, alors T est aussi un tétraèdre de Delaunay de $DEL(S-p)$.*

Démonstration : D'après la proposition 2.5, on sait que la tétraédrisation $DEL(S-p)$ est l'union des tétraèdres de $DEL_p(S)$ et des tétraèdres intérieurs à $PE(p)$. Mais, comme la tétraédrisation de Delaunay d'un ensemble de points est unique, les tétraèdres de Delaunay **intérieurs** à $PE(p)$ plongés dans $DEL(S-p)$ et les tétraèdres de Delaunay **intérieurs** à $PE(p)$ plongés dans $DEL(N_S(p))$ sont identiques. Ceci achève la démonstration. ■

Par conséquent, pour retétraédriser le polyèdre étoilé, il suffit de tester si les nouveaux tétraèdres calculés sont vides des points de $PE(p)$, d'où le caractère local de l'algorithme.

Pour conclure sur la suppression des points dans la structure de Voronoï, nous pouvons donner la complexité de notre algorithme dans le cas où les points de S suivent une densité de probabilité uniforme dans la boule unité.

Théorème 2.1 *Soit S un ensemble de points générés suivant une densité de probabilité uniforme dans la boule unité de $\mathbb{R}^3$. Supprimer un point dans le diagramme de Voronoï s'effectue en temps constant : $O(1)$.*

Démonstration : D'après la proposition 2.5Γsupprimer un point p revient à retétraédriser l'intérieur de $PE(p)$. D'après la proposition 2.6Γpour retétraédriser l'intérieur de $PE(p)$Γil suffit de ne considérer que les points de $PE(p)$. EnfinΓd'après le corollaire 2.4 de la section 2.7.2Γle nombre de sommets de $Vor_S(p)$ est bornéΓdonc le nombre de points de $PE(p)$ (i.e. le nombre de points de $N_S(p)$) est borné. Par conséquentΓsupprimer un point s'effectue en temps constant. ■

La définition 2.6Γ les propositions 2.5Γ 2.6 et le théorème 2.1 de cette section ont volontairement été donnés en 3DΓpar souci de clarté. Le passage en dimension quelconque est évident et n'est qu'un jeu d'écriture.

2.7 Complexité

Dans cette sectionΓnous allons étudier la complexité théorique des algorithmes de construction du DVP 3D. Nous donnerons la complexité de l'algorithme "divide and conquer" dans le pire des casΓet nous donnerons une analyse originale de la complexité des algorithmes incrémentaux dans le cas où les points sont distribués uniformément dans la boule unité.

2.7.1 Analyse du pire des cas

Théorème 2.2 (Edelsbrunner) *Soit S un ensemble de points de $\mathbb{R}^d$, $d \geq 1$. Le nombre N_k de k-faces du diagramme de Voronoï de S est majoré de la manière suivante :*

$$N_k \leq O(n^{min\{d+1-k, \lceil \frac{d}{2} \rceil\}}),\ 0 \leq k \leq d.$$

Les bornes supérieures sont atteintes.

Démonstration : Elle est détaillée dans [49]. Comme nous l'avons déjà remarqué dans la section 2.3.1 (projection de l'enveloppe convexe)Γcette démonstration est liée au calcul du nombre de $(d+1-k)$-faces d'un polytope de $\mathbb{R}^{d+1}$. ■

Le théorème 2.2 permet de donner le corollaire suivant en 3D :

Corollaire 2.1 *Soit S un ensemble de points de $\mathbb{R}^3$. Les nombres N_k, $0 \leq k \leq 3$, de k-faces du diagramme de Voronoï de S sont majorés de la manière suivante :*

$$N_0 \ \leq \ O(n^2)$$

$$\begin{aligned} N_1 &\leq O(n^2) \\ N_2 &\leq O(n^2) \\ N_3 &\leq O(n) \end{aligned}$$

où N_0 représente le nombre de sommets de Voronoï, N_1 le nombre d'arêtes de Voronoï, N_2 le nombre de faces de Voronoï et N_3 le nombre de polyèdres de Voronoï. De plus les bornes supérieures sont atteintes.

Démonstration : Il suffit d'utiliser le théorème 2.2 et de constater que $\lceil \frac{3}{2} \rceil = 2$ et que $4 - k \geq 2$Γpour $0 \leq k \leq 2$Γet $4 - k = 1$Γpour $k = 3$. De plus les bornes supérieures sont atteintes dans les exemples que nous citons ci-après. ■

Corollaire 2.2 *L'algorithme "divide and conquer" pour construire le diagramme de Voronoï (section 2.3.1) est optimal en 3D.*

Démonstration : D'après le corollaire 2.1Γla taille du diagramme de Voronoï en 3D est en $O(n^2)$ dans le pire des cas. D'après [51]Γl'algorithme "divide and conquer" est en $O(n^2)$Γdonc il est optimal. ■

L'exemple donné dans le chapitre 4Γsection 4.3.1Γcas 4Γest une illustration du pire des cas : les points sont distribués sur deux segments orthogonaux et non coplanaires. Un autre exempleΓissu de la théorie des graphesΓest constitué par le K_5 (graphe complet à 5 noeuds). En effet un K_5 peut représenter un diagramme de Delaunay à 5 noeuds comme l'illustre la figure 17.

Plus généralementΓil existe des graphes de Delaunay 3D complets dont le nombre de noeuds est nΓn entier quelconque [51].

Remarque 2.1 *En 2D, la taille du diagramme de Voronoï est en $O(n)$ dans le pire des cas et l'algorithme "divide and conquer" reste optimal mais sa complexité est en $O(n \log_4 n)$.*

2.7.2 Analyse du cas uniforme

Dans toute cette section on considèrera la distribution de points S suivante : soit S un ensemble de points suivant la densité g de probabilité **uniforme** dans la boule

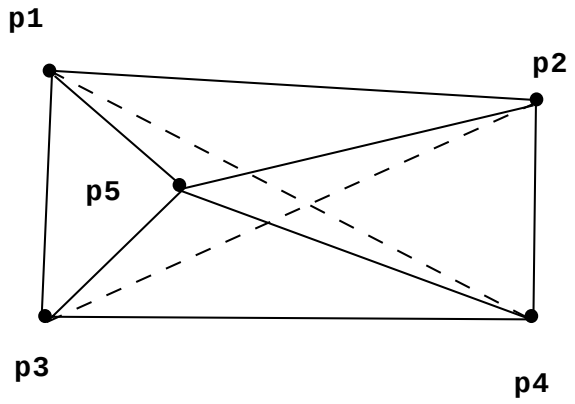


FIG. 17 - Dans cet exemple le K_5 est un graphe de Delaunay : tous les tétraèdres $(p_i, p_j, p_k, p_l)\Gamma$où $i, j, k, l \in \{1 \ldots 5\}\Gamma$et $p_i\Gamma p_j\Gamma p_k$ et p_l sont distinctsΓsont des tétraèdres de Delaunay.

unité de $\mathbb{R}^d\Gamma d \geq 1$. Le nombre moyen ES_n de simplexes de Delaunay de $DEL(S)$ est donné par l'équation :

$$ES_n = \begin{pmatrix} n \\ d+1 \end{pmatrix} P_n \tag{2}$$

où P_n est la probabilité définie par Dwyer [48] :

$$P_n = \int_{\mathbb{R}^d} \ldots \int_{\mathbb{R}^d} (1 - \Gamma_d)^{n-d-1} g(x_1) \ldots g(x_{d+1}) dx_1 \ldots dx_{d+1} \tag{3}$$

Γ_d étant le volume normalisé de l'intérieur de l'hypersphère circonscrite aux $x_i\Gamma$ $(i = 1, \ldots, d+1)$.

A partir des équations 2 et 3ΓDwyer a démontré le théorème suivant :

Théorème 2.3 (Dwyer) *Le nombre moyen ES_n de simplexes de Delaunay de $DEL(S)$ est :*

$$ES_n \approx \frac{d! d^{d-1} V_d M_d n}{d+1} \tag{4}$$

où V_d est le volume de la boule unité et M_d le volume moyen occupé par un simplexe inscrit sur la d-sphère unité.

Démonstration : La démonstration est très technique et calculatoire. Tous les détails sont donnés dans [48]. ■

Corollaire 2.3 *En 3D, le nombre moyen ES_n de tétraèdres de Delaunay de $DEL(S)$ est : $ES_n \approx 6,77n$.*

Démonstration : Le théorème 2.3 et le calcul des constantes V_3 et M_3 donnent le résultat. ■

Le résultat du corollaire 2.3 se retrouve parfaitement en pratique.

En 2DΓon trouve que le nombre moyen ES_n de triangles de Delaunay est : $ES_n \approx 2n$Γce qui peut aussi se démontrer par la formule de Gauss et qui se retrouve en pratique.

On peut noter que le théorème 2.3 montre queΓcontrairement au pire des casΓla taille finale du problème est linéaire :

$$Card(DEL(S)) = O(n).$$

Une analyse de la complexité de l'algorithme incrémental pour la construction du diagramme de Voronoï 3D nécessite de connaître le nombre total de tétraèdres de Delaunay (ou de manière dualeΓle nombre de sommets de Voronoï) construits au cours de l'algorithme.

Proposition 2.7 *Le nombre moyen EP_n de sommets par polyèdre de Voronoï ne dépend pas de n. Plus précisément on a :*

$$EP_n \approx d! d^{d-1} V_d M_d \tag{5}$$

où V_d et M_d sont définis dans le théorème 2.3.

Démonstration : On sait que $ES_n = \begin{pmatrix} n \\ d+1 \end{pmatrix} P_n$ (équation 2). On a donc $EP_n = \begin{pmatrix} n-1 \\ d \end{pmatrix} P_n$ (On fixe un pointΓet on cherche le nombre moyen de tétraèdres attachés à ce point). On peut donc écrire :

$$EP_n = \frac{\begin{pmatrix} n-1 \\ d \end{pmatrix}}{\begin{pmatrix} n \\ d+1 \end{pmatrix}} ES_n = \frac{(n-1)!\ (n-d-1)!\ (d+1)!}{(n-1-d)!\ d!\ n!} ES_n.$$

On obtient alors :

$$EP_n = \frac{d+1}{n} ES_n \tag{6}$$

et on trouve $EP_n \approx d! d^{d-1} V_d M_d$ en utilisant l'équation 4. ■

Corollaire 2.4 *Dans $\mathbb{R}^3$, le nombre moyen EP_n de sommets par polyèdre de Voronoï est : $ES_n \approx 28$.*

Démonstration : Le corollaire 2.3 et la proposition 2.7 donnent le résultat. ■

Ce résultat se retrouve très bien dans la pratique en 3D.

En 2DΓon trouve $ES_n \approx 6$Γce qui se retrouve également en pratique.

Proposition 2.8 *Le nombre moyen ET_n de sommets de Voronoï construits à la fin de l'algorithme incrémental est :*

$$ET_n \approx (d+1)ES_n.$$

Démonstration : A chaque insertion d'un nouveau point dans le diagramme de VoronoïΓon construit un nouveau polyèdre de Voronoï; par conséquent d'après la proposition 2.7Γon construit EP_n sommets de Voronoï. Après n insertions on a donc $ET_n = nEP_n$Γet d'après l'équation 6 on trouve le résultat voulu. ■

Dans la pratique on retrouve bien le même résultat : on construit 4 fois trop de sommets de Voronoï pour $d = 3$ et 3 fois trop pour $d = 2$.

On peut maintenant énoncer le théorème final.

Théorème 2.4 *Soit S un ensemble de n points distribués selon une loi uniforme dans la boule unité de $\mathbb{R}^d$. La complexité de la construction incrémentale du diagramme de Voronoï basé sur les points de S est en $O(n \log_{2^d} n)$.*

Démonstration : Quand on insère un nouveau point dans la structure de VoronoïΓ la recherche du plus proche voisin peut se faire dans un 2^d-tree (arbre à 2^d branches). Cette première étape (étape 1 des algorithmes incrémentaux) est donc réalisée en $O(\log_{2^d} n)$ (section 2.5.3). De plus la mise à jour du diagramme est locale d'après la proposition 2.7. Cette deuxième étape s'effectue donc en $O(1)$ (étape 2 des algorithmes incrémentaux). Par conséquentΓl'insertion d'un nouveau point s'effectue en $O(\log_{2^d} n)$ et donc l'insertion des n points se fait en $O(n \log_{2^d} n)$. ■

De nouvelles techniques d'analyse (analyses randomisées) des algorithmes incrémentaux sont étudiées à l'INRIA Sophia Antipolis [132]. L'idée maîtresse est de maintenir une structure parallèle à la structure de VoronoïΓcelle des arbres de DelaunayΓpour accélérer la recherche du plus proche voisin dans la première étape de l'algorithme incrémental (section 2.5.3). La clef de l'analyse de la complexité réside dans le fait qu'elle est réalisée en moyenne. Une théorie utilisant de puissants théorèmes de probabilités a été élaborée. Comme dans le cas de la structure d'octree que

nous proposons dans la section 2.5.3Γle problème est de trouver un bon compromis entre place mémoire et complexité.

2.8 Conclusion

Dans tous les cas *pratiques* produits par l'algorithme "split and merge" (Chapitre 3Γsection 3.3.1)Γl'algorithme incrémental vérifie la proposition 2.8Γet les corollaires 2.3 et 2.4. Cette remarque nous amène à énoncer la conjecture suivante :

Conjecture 2.1 *Si S est un ensemble fini de points suivant un densité de probabilité, alors ES_n est en $O(n)$.*

Cette conjecture expliquerait très bien les excellentes performances pratiques des algorithmes incrémentaux dans les cas *courants*.

On peut donc dire queΓmalgré l'optimalité théorique de l'algorithme "split and merge"Γles algorithmes incrémentaux ont un comportement optimal dans tous les cas pratiques que nous avons rencontrés. Nous avons en effet montré que leur complexité était meilleure dans le cas de distribution uniforme de points dans la boule unité.

Les algorithmes incrémentaux présentent donc deux avantages :

- les algorithmes incrémentaux sont **dynamiques** (insertion et suppression)Γ
- les algorithmes incrémentaux sont **optimaux** dans la plupart des cas.

Ces deux conditions sont nécessaires pour rendre possible l'exploitation des DVP en analyse d'images 2D et 3DΓcomme nous allons le voir dans les chapitres suivants.

Chapitre 3

Partitionnements du plan et de l'espace

Ce chapitre est consacré aux partitions du plan et de l'espace.

Nous distinguerons essentiellement deux classes de partitionnements :

- les partitionnements non adaptatifsΓ
- les partitionnements adaptatifs.

Chacune de ces deux grandes classes sera à son tour divisée en deux types de partitionnements :

- les partitionnements rigidesΓ
- les partitionnements non rigides.

Nous développerons plus spécifiquement le partitionnement en polyèdres de Voronoï. Un tel pavage permet de réaliser un codage haut niveau (par les polyèdres de Voronoï)Γstructuré dans le graphe de voisinage de Delaunay et adapté aux données.

D'autres modèles de partitionnement seront étudiés : essentiellement les partitions en octreeΓet de Delaunay.

Nous appliquerons ensuite ces partitionnements à l'analyse d'images : compressionΓreprésentation et mesure.

3.1 Introduction

Ce chapitre a pour objectif de montrer l'utilité des modèles géométriques pour obtenir une partition de données volumiques (images tridimensionnelles).

L'idée principale est de partitionner l'espace en utilisant des régions *convexes*. Pour nous ces régions seront les polyèdres de Voronoï. En effet les diagrammes de Voronoï permettent d'obtenir une partition de l'espace avec de bonnes propriétés géométriques (Chapitre 2 section 2.2).

Par définition même du DVP le codage du partitionnement est très simple. En effet à chaque polyèdre de Voronoï on peut associer un germe et inversement à chaque germe on peut associer un polyèdre de Voronoï. Le contenu d'un polyèdre sera la moyenne des niveaux de gris des voxels qui lui sont intérieurs.

De plus les relations topologiques entre les polyèdres sont contenues dans le graphe de Delaunay. Les relations de voisinage qui en résultent sont très informatives (Chapitre 2 section 2.2).

Ces relations rendront possibles les étapes d'analyse et de synthèse d'images :

- reconstruction 3D
- segmentation
- analyse quantitative
- Visualisation
- animation.

L'étape de segmentation est abordée dans la section 3.3.3 mais elle sera développée plus en profondeur dans le chapitre 5 en relation avec les modèles neuronaux associés aux pyramides et aux DVP et dans le chapitre 6 en relation avec les modèles markoviens associés aux DVP.

Les étapes de visualisation et d'animation sont décrites dans [96]. Nous utilisons les outils classiques issus de la synthèse d'images : lissage de Gouraud lumières (composante ambiante émise diffuse et spéculaire) couleurs algorithme de Z-buffer assombrissement en fonction de la profondeur transparence rotation... Nous ne détaillerons pas davantage ces étapes ce qui nous éloignerait trop de notre propos.

Les relations topologiques peuvent aussi être utilisées pour construire des outils de morphologie mathématique [11Γ84Γ138Γ139].

En résuméΓun partitionnement est constitué d'**éléments de volume** qui peuvent parfaitement être décrits par leur contenu et les relations de voisinage.

Nous mettrons donc l'accent sur les modèles de partitionnement par polyèdres de VoronoïΓmais d'autres partitionnements seront présentés.

Les résultats présentés (section 3.3.4) sont issus d'images obtenues sur des préparations biologiques 3D. Les images proviennent d'un microscope confocal à balayage laser (ZEISS CLSM LSM10) [103] (Figure 18).

FIG. 18 - 18 sections d'une série de 41 sections (256×256) d'un noyau cellulaire obtenue avec un microscope confocal à balayage laser (CLSM) (LSM10 ZEISS). La taille des voxels est $0.14\mu m \times 0.14\mu m \times 0.29\mu m$.

Avec un tel capteur la résolution est de $0,14\ \mu m$ dans chaque direction dans le plan focal et de $0,29\ \mu m$ dans la direction axiale. Le microscope confocal à balayage laser donne la possibilité de réaliser des coupes optiques de spécimens transparents et fluorescents selon un processus non destructif. Les données sont digitalisées par coupe et mémorisées dans un tableau 3D dans lequel les valeurs sont proportionnelles à l'intensité de la fluorescence correspondant au sous-volume du voxel considéré. Ces valeurs sont comprises entre 0 et 255 et représentent des niveaux de gris. AinsiΓpour donner un ordre de grandeurΓun volume de $256 \times 256 \times 256$ représente 16 Méga-Octets en mémoire. Dans ce volumeΓla dimension réelle des voxels est définie par les pas de discrétisation S_iΓS_jΓet S_kΓutilisés durant l'acquisition du volume. Comme la

résolution axiale est différente de la résolution latéraleΓle volume discret est composé de voxels non-cubiquesΓce qui génère un artéfact du système d'acquisition. Dans nos applications en biologie (cellules de coeur ou noyaux cellulaires) où le volume est de $256 \times 256 \times 41$ΓS_i et S_j sont égaux à $0,14$ μm et S_kΓà $0,29$ μm. On peut dire que les principaux artéfacts générés par le système d'acquisition sont :

- des voxels non cubiquesΓ
- un bruit de type haute fréquenceΓ
- une contribution des plans voisins hors focal.

Divers algorithmes pour corriger ces artéfacts existent [104]. Une fois les artéfacts corrigés par des prétraitements liés au système d'acquisitionΓl'algorithme de partitionnement que nous présentons dans la section 3.3.1 peut être utilisé.

Le plan de ce chapitre consacré aux partitionnements sera le suivant. Dans la section 3.2Γnous parlerons des modèles de partitionnement en général : nous distinguerons les partitionnements *non adaptatifs* des partitionnements *adaptatifs*Γet les modèles *rigides*Γdes modèles *non rigides*. Dans la section 3.3Γnous détaillerons notre algorithme de partitionnement en polyèdres de VoronoïΓmodèle adaptif et non rigide en 3D. La section 3.4 s'attachera à donner d'autres modèles de partitionnements (octreeΓsection 3.4.1 ; DelaunayΓsection 3.4.2)Γavec un souci de comparaisonΓ section 3.4.3Γdes différentes méthodes de partitionnement présentées dans ce chapitre. Nous donnerons enfin (section 3.5)Γune conclusion. Dans ce chapitreΓtous les algorithmes ont été implémentés en 2D et en 3D. Les illustrations seront donc données indifféremment en 2D ou en 3D.

3.2 Partitionnements : généralités

Il existe une infinité de pavages du plan $\mathbb{R}^2$ et de l'espace $\mathbb{R}^3$. De nombreux dessins d'Esher sont des exemples de pavage du plan où les éléments de surface s'emboîtent à l'infini. Mais ces partitionnements sont difficiles à exploiter. En effetΓdans le but précis de la représentation d'imagesΓdes contraintes particulières peuvent être imposées pour des raisons évidentes de formation des images et d'implémentation algorithmique [39].

Les partitionnements peuvent être classés en deux grandes catégories : les pavages adaptatifs et les pavages non adaptatifs. Chacune de ces deux classes peut être à son tour divisée en deux : les modèles rigides et les modèles non rigides. C'est autour de ces différents modèles que nous allons articuler cette section.

3.2.1 Partitionnements non adaptatifs

Ce type de partitionnement n'est pas adapté au contenu du volume de données. Tout se passe comme si le volume à partitionner était homogène. Par conséquent chaque élément de volume constituant la partition est très peu informatif. De même les liaisons structurant ces éléments ne seront pas significatives.

Nous pouvons donner une définition plus précise des partitionnements non adaptatifs comme suit :

Définition 3.1 (Partitionnements non adaptatifs) *Un partitionnement non adaptatif d'une image I (2D ou 3D) est un partitionnement où la position de chaque élément de partition est géré par une fonction externe au contenu informatif de l'image (e.g. niveaux de gris).*

Dans cette classe nous distinguons :

- les modèles rigides non adaptatifs
- les modèles non rigides non adaptatifs.

Modèles rigides

Par modèles rigides nous entendons les modèles dont les motifs de base sont figés et en petit nombre. Parmi les pavages non adaptatifs et réguliers nous avons les fameux pavages dessinés par Escher (Figure 19) et en ce qui nous concerne les pavages par des polygones réguliers en 2D ou par des polyèdres réguliers en 3D.

En 2D il existe une infinité de polygones réguliers (angles égaux deux à deux et longueurs des arêtes égales deux à deux) et on sait qu'il n'existe que trois partitionnements réguliers (i.e composés de polygones réguliers) :

- le partitionnement carré
- le partitionnement triangulaire

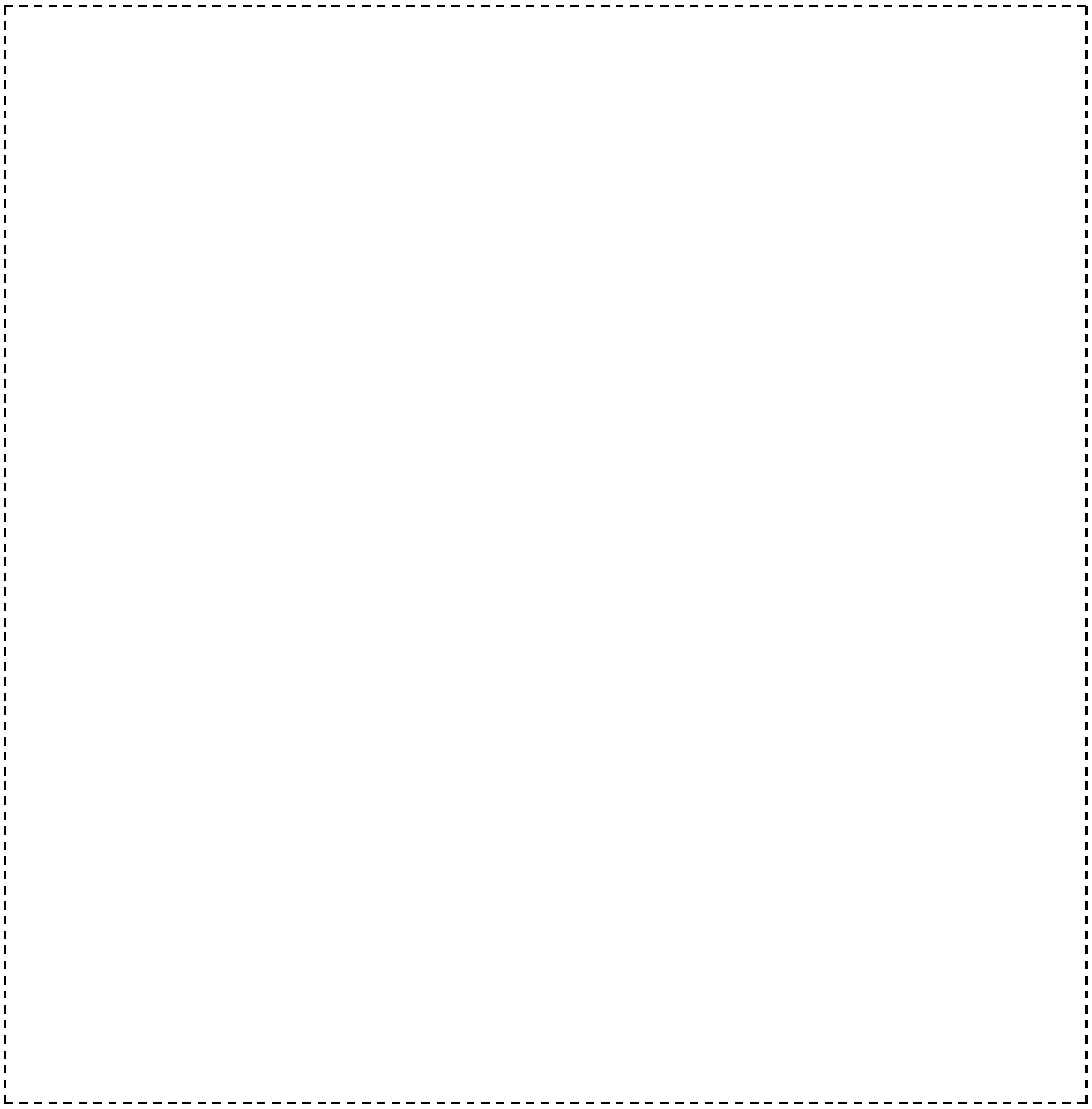


FIG. 19 - Le ciel et l'enfer. M. C. Escher.

- le partitionnement hexagonal

et 8 pavages semi-réguliers (composés de deux polygones réguliers de base) [39].

En 3D il n'existe plus que cinq polyèdres réguliers : le tétraèdre le cube l'octaèdre le dodécaèdre et l'icosaèdre et il n'existe plus qu'un seul pavage régulier de l'espace :

- le partitionnement cubique.

Le modèle rigide le plus utilisé est sans doute le partitionnement par des carrés en 2D et le partitionnement par des cubes en 3D pour une raison essentielle : la

facilité algorithmique. On peut décrire le processus de la manière suivante : Soit une image 3D de taille $(2^n)^3\Gamma$et $p = 2^m\Gamma 0 \leq m \leq n\Gamma$le pas de discrétisation.

Algorithme : Partitionnement cubique

1. Partitionner l'image 3D en $(2^{n-m})^3$ cubes C de taille $(2^m)^3$.

2. Pour tous les cubes $C\Gamma$calculer la moyenne des niveaux de gris.

Les figures 20 et 21 montrent le résultat d'un tel partitionnement en 2D.

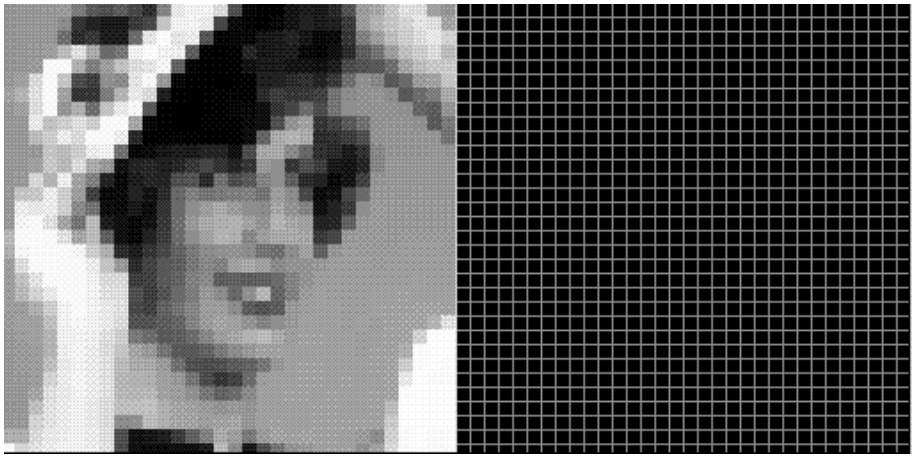


FIG. 20 - Partitionnement en 1024 carrés $8 * 8$ de l'image "femme".

FIG. 21 - Partitionnement en 4096 carrés $4 * 4$ de l'image "femme".

Modèles non rigides

Par modèles non rigides, nous entendons les modèles qui ne possèdent pas des motifs de base. Parmi ces partitionnements non adaptatifs et non rigides, nous pouvons citer les partitionnements en régions de Voronoï (polygones en 2D, polyèdres en 3D), où les positions des germes associés aux régions de Voronoï ne sont pas adaptées aux données [1]. L'algorithme peut être décrit très simplement de la manière suivante :

Algorithme : Partitionnement de Voronoï non adaptatif

1. Ensemencer l'image par un **grand** nombre de germes selon un processus de Poisson.
2. Calculer le diagramme de Voronoï associé à ces germes.
3. Pour tous les polyèdres de Voronoï, calculer la moyenne des niveaux de gris.

L'étape 1 montre que la position des germes ne dépend pas de l'image. Comme nous le verrons dans la section 3.3, ce principe ne sera plus vérifié dans un environnement de type "split and merge".

On voit bien que pour un partitionnement non adaptatif, un algorithme dynamique (i.e. incrémental) de construction du DVP est inutile. En effet, nous avons comme donnée initiale, la connaissance de la position et du nombre total de tous les germes. Par conséquent un algorithme de construction du DVP de type global (e.g. "divide and conquer", Chapitre 2 section 2.3.1) pourra être utilisé. Une illustration d'un partitionnement de Voronoï non adaptatif 2D est donnée dans la figure 22.

Un autre partitionnement non adaptatif et non rigide est le partitionnement en tétraèdres de Delaunay, les tétraèdres étant distribués uniformément dans l'image. Le principe algorithmique est similaire à celui que nous venons de présenter pour le partitionnement de Voronoï non adaptatif. La figure 23 est une illustration du partitionnement 2D en triangles de Delaunay non adaptatif.

3.2.2 Partitionnements adaptatifs

Ce type de partitionnement est adapté au contenu du volume de données. Par conséquent, chaque élément de volume constituant la partition contiendra une information très riche, et les liaisons structurant ces éléments seront très significatives, surtout en ce qui concerne les partitions de Voronoï adaptatives.

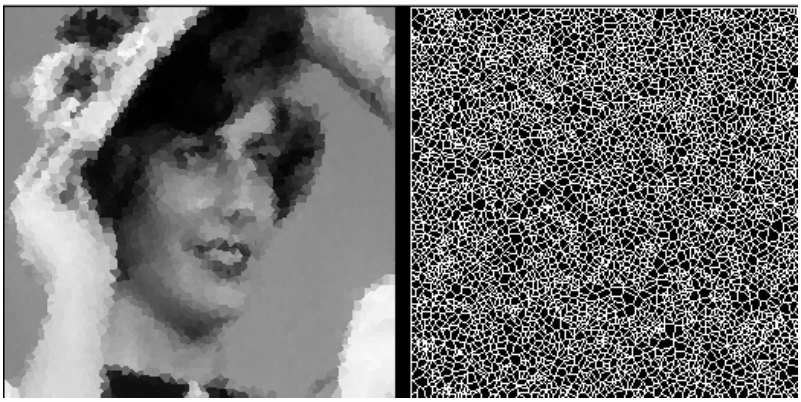


FIG. 22 - Partitionnement non adaptatif avec 4000 polygones de Voronoï de l'image "femme".

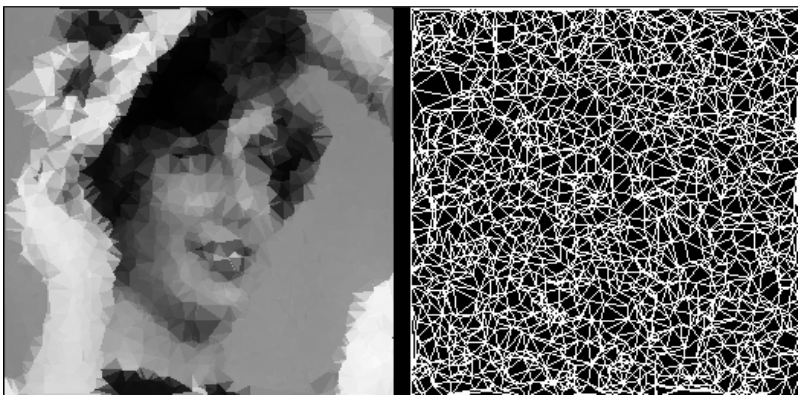


FIG. 23 - Partitionnement non adaptatif avec 4000 triangles de Delaunay de l'image "femme".

Définition 3.2 (Partitionnements adaptatifs) *Un partitionnement adaptatif d'une image I (2D ou 3D) est un partitionnement où la position de chaque élément de partition est gérée par une fonction dépendant du contenu informatif de l'image (e.g. niveaux de gris).*

Comme dans la section précédente, nous distinguons dans cette classe :

- les modèles rigides adaptatifs,
- les modèles non rigides adaptatifs.

L'idée directrice est de trouver un codage haut niveau adapté au volume de données. Une façon correcte de procéder est de travailler dynamiquement, de manière à

ce que l'image puisse guider la localisation des éléments de volume. Pour répondre à une telle contrainte nous utiliserons un algorithme général de type "split and merge" ou division et fusion [38] :

1. Phase de division
2. Phase de fusion

C'est dans la phase de division que la localisation des éléments est guidée par le contenu de l'image. Il est donc nécessaire de trouver des règles simples pour **insérer** des éléments de volumes lors de la phase de **division**.

Modèles rigides

Les partitionnements en quadtrees et en octrees représentent une solution aux partitionnements adaptatifs [118 108]. Toutefois nous qualifierons ce type de partitionnement de **rigide** puisque que les éléments de volume sont exclusivement cubiques. L'étude détaillée des partitionnements en octrees est réalisée dans la section 3.4.1.

Les partitionnements Horizontal Vertical (HV) font aussi partie des modèles rigides. Ils consistent à diviser récursivement l'image en deux alternativement horizontalement et verticalement suivant un critère d'homogénéité. Le découpage récursif en deux parties peut être réalisé de manière optimale (par exemple la somme des variances des deux parties doit être minimale). Ces partitionnements semblent mieux adaptés aux données que les octrees [54].

Le côté dynamique de ces deux partitionnements se retrouve dans le côté récursif des algorithmes associés.

Modèles non rigides

Les partitionnements en polyèdres de Voronoï (section 3.3) ou en tétraèdres de Delaunay (section 3.4.2) sont eux aussi bien adaptés au problème.

En effet nous savons insérer des sites dans un diagramme de Voronoï ou de Delaunay dynamiquement par les algorithmes incrémentaux (Chapitre 2 section 2.4). De plus contrairement aux cubes contenus dans le partitionnement en octrees les polyèdres de Voronoï et les tétraèdres de Delaunay présentent l'avantage d'être **non rigides**.

Dans la section 3.4.3Γnous tenterons d'établir une comparaison entre ces différents partitionnements.

3.3 Partitionnement par le modèle de Voronoï

Nous allons détailler dans cette section notre algorithme de partitionnement d'images volumiques en niveaux de gris par le modèle de Voronoï. Cette méthode est adaptative et non rigide. Notre algorithme est une généralisation du "split and merge" sur les structures régulières de type quadtree (rigide) [55] et sur la structure irrégulière de type Voronoï (non rigide) en 2D [38] étendue à la structure irrégulière de type Voronoï (non rigide) en 3D [12].

3.3.1 Algorithme "split and merge"

Notre algorithme procède en deux étapes : les deux étapes de l'algorithme "split and merge".

Dans l'étape de divisionΓdes polyèdres sont ajoutés dans l'image jusqu'à convergence. Cette étape demande une gestion dynamique du DVP. Dans l 'étape de fusionΓ des polyèdres sont supprimés. Les polyèdres sont ajoutés et supprimés selon des critères que nous allons définir ci-après.

Définition 3.3 (Polyèdre homogène) *Un polyèdre de Voronoï, $Vor_S(p)$, est dit homogène si, et seulement si, l'écart type des niveaux de gris des voxels contenus dans $Vor_S(p)$, noté $\sigma(Vor_S(p))$, est inférieur à un seuil s_1 donné.*

$$Vor_S(p) \text{ homogène } \Longleftrightarrow \sigma(Vor_S(p)) \leq s_1$$

L'homogénéité est donc calculée sur un simple critère statistique. Des fonctions plus robustesΓ comme le gradient morphologique [122Γ 123]Γont été testées sans apporter une amélioration significative à l'algorithme "split and merge".

Définition 3.4 (Polyèdre inutile) *Un polyèdre de Voronoï, $Vor_S(p)$, est dit inutile si, et seulement si, la différence des niveaux de gris de $Vor_S(p)$ et de chaque polyèdre voisin de $Vor_S(p)$ est plus petite qu'un seuil s_2 fixé.*

Si on note $m(Vor_S(p))$ le niveau de gris moyen associé à un polyèdre $Vor_S(p)$, on a :

$$Vor_S(p) \text{ inutile } \Longleftrightarrow \forall p_i \in N_S(p),\ |m(Vor_S(p)) - m(Vor_S(p_i))| \leq s_2$$

GlobalementΓsi un polyèdre est non homogène (Définition 3.3)Γon ajoute des germes selon un critère géométrique (Définition 3.5)Γsinon on ne fait rien. On continue ainsi jusqu'à la convergence (tous les polyèdres sont homogènes). Dans l'étape de fusionΓon suppime les polyèdres inutiles (Définition 3.4).

Définition 3.5 (Ajout polyèdres) *Soit $Vor_S(p)$ un polyèdre non homogène. Un germe q est ajouté sur chaque barycentre des faces séparant $Vor_S(p)$ de $Vor_S(q)$, $q \in N_S(p)$. Si la surface est trop petite par rapport à un pourcentage s_3 fixé de la surface totale de $Vor_S(p)$, le germe q n'est pas inséré.*

$$\forall p_i \in N_S(p),\ q = Bary(Vor_S(p) \cap Vor_S(p_i)) \text{ si } \frac{Surf(Vor_S(p) \cap Vor_S(p_i))}{Surf(Vor_S(p))} > s_3$$

Une illustration dans le plan de la procédure d'ajout est donnée en figure 24. Le barycentre est ici simplement le milieu des segments de Voronoï considérés. On remarque queΓpour un seuil s_3 raisonnableΓon n'ajoute pas de site sur l'arête générée par les germes p et n. On peut noter que cette procédure permet de diviser un polygone etΓde manière similaireΓde diviser un polyèdre.

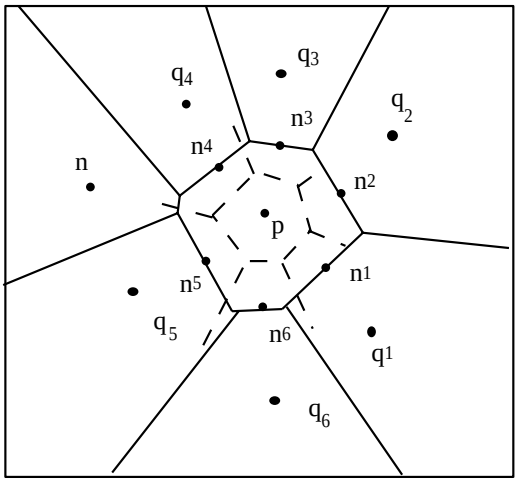


FIG. 24 - Exemple d'ajout de germes en 2D. La région en pointillé représente la nouvelle zone de Voronoï de p.

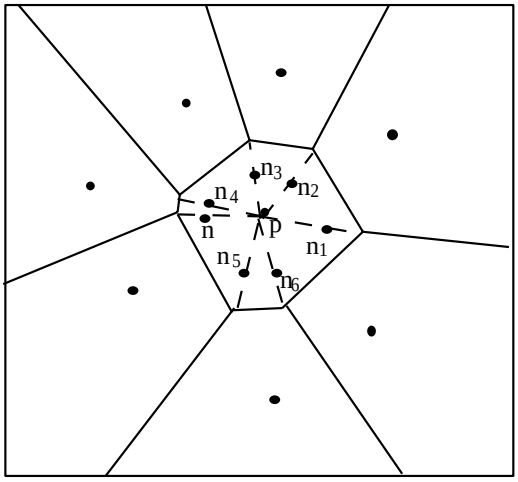


FIG. 25 - Exemple d'ajout de germes en 2D utilisé dans [94].

Avec le processus d'ajout utilisé dans [94]Γles germes sont ajoutés sur les arêtes joignant le germe du polygone à diviser à chacun des sommets constituant ce polygone (Figure 25).

L'avantage de notre méthode est de pouvoir descendre au niveau du pixel etΓde plusΓà chaque étape nous ajoutons globalement deux fois moins de sitesΓce qui nous permet d'être plus adaptatif. D'autre partΓen 3DΓun polyèdre possède beaucoup moins de faces que de sommets de VoronoïΓce qui nous permet de mieux contrôler l'insertion des nouveaux germes. Ces remarques permettent de justifier des résultats de meilleure qualité.

Plus précisémentΓl'algorithme peut être décrit dans ses grandes lignes comme suit :

Algorithme : Partitionnement Voronoï

1. Ensemencer l'image par un **petit** nombre de germes selon un processus de Poisson (ou avec un germe central et un germe dans chaque coin de l'image).

2. Répéter jusqu'à la convergence (tous les polyèdres sont homogènes (Définition 3.3)).

   (a) Calculer le diagramme de Voronoï.

   (b) Calculer la moyenne des niveaux de grisΓl'écart typeΓet le volume de chaque polyèdre.

(c) Pour chaque polyèdre, effectuer une division si le polyèdre est non homogène (Définition 3.5).

3. Fusion : suppression des polyèdres inutiles (Définition 3.4).

L'étape de fusion (étape 6) consiste à chercher tous les polyèdres inutiles, et une fois cette recherche terminée, on procède à leur suppression de la structure. Cette suppression peut se faire en $O(1)$ en utilisant l'algorithme de suppression décrit dans le chapitre 2, section 2.6.

L'algorithme précédent converge. En effet, la procédure d'ajout donnée dans la définition 3.5 divise bien les polyèdres non homogènes. Dans le pire des cas, à la convergence, chaque polyèdre ne contient plus qu'un seul voxel, ce qui donne un écart type de 0, et, dans ce cas, tous les polyèdres sont homogènes.

L'étape 1 des algorithmes incrémentaux (Chapitre 2, section 2.4) est réalisée en $O(1)$ dans un environnement "split and merge". En effet, si $Vor_S(p)$ est un polyèdre non homogène, on peut calculer les germes à insérer q en leur donnant p comme "père". Quand les germes q seront insérés dans la structure, la phase 1 des algorithmes incrémentaux (Chapitre 2, section 2.4) peut être initialisée à partir du "père" p de q. Par construction, p et q ne sont pas très éloignés ($q \in N_{S \cup p}(p)$ dans $DEL(S \cup q)$) : par conséquent la recherche peut se faire en $O(1)$ en moyenne.

De plus, en pratique, l'étape 2 des algorithmes incrémentaux est aussi en $O(1)$; par conséquent, dans un environnement "split and merge", l'algorithme incrémental pour construire le DVP 3D est en $O(n)$.

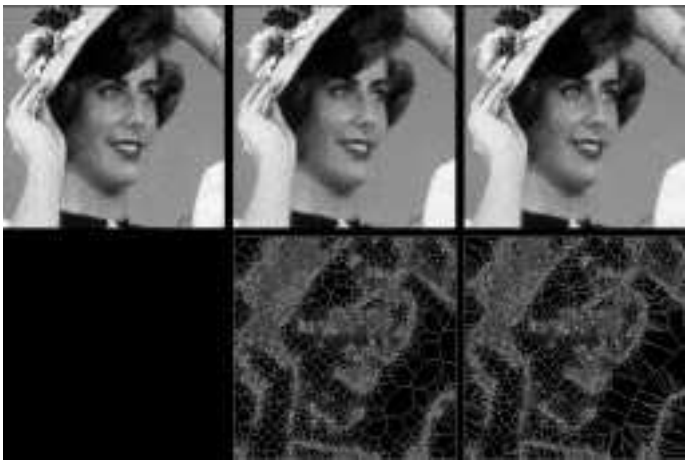
Une alternative à l'algorithme "split and merge" est proposée dans le chapitre 6, section 6.5.

Une illustration de notre algorithme est donnée en 2D dans la figure 26, où la taille de l'image est de 256×256. Le résultat final est obtenu en quelques secondes sur une silicon graphics indigo.

La figure 27 illustre le processus "split and merge" en 3D. La taille de l'image est ici de $128 \times 128 \times 31$ et le temps de calcul est de 3 minutes approximativement sur une silicon graphics indigo.

D'autres illustrations en 3D sont données dans la section 3.3.4.

(a) Division. De la gauche vers la droite et de haut en bas : image originale, 5 polygones, 13 polygones, 41 polygones, 126 polygones, 410 polygones, 1316 polygones, 3906 polygones, 9977 polygones.



(b) Résultat. A gauche, image originale ; au centre, résultat de la division (10585 polygones) et le graphe correspondant au dessous ; à droite résultat de la fusion (9033 polygones) et le graphe correspondant au dessous.

FIG. 26 - Algorithme “split and merge” basé sur le DVP 2D.

FIG. 27 - Convergence de l'algorithme de "split and merge" dans un environnement de Voronoï sur l'image d'un chromosome.

3.3.2 Etiquetage

Une étape d'étiquetage de chaque polyèdre $Vor_S(p)$ est nécessaire pour permettre le calcul du volume $Vol(Vor_S(p))$ de la moyenne

$$m(Vor_S(p)) = \frac{1}{Vol(Vor_S(p))} \sum_{v \in Vor_S(p)} n(v)$$

et de l'écart type

$$\sigma(Vor_S(p)) = \sqrt{\frac{1}{Vol(Vor_S(p))} \sum_{v \in Vor_S(p)} (n(v) - m(Vor_S(p)))^2}$$

de chaque polyèdre.

C'est un problème classique de géométrie algorithmique mais très difficile dans notre cas puisque $256 \times 256 \times 41$ voxels sont à étiqueter dans l'exemple de la cellule (section 3.3.4).

En 2D une solution utilisant les outils de géométrie discrète a été développée grâce à une bonne connaissance des droites discrètes dans le plan. En 3D la notion de plan discret est bien plus complexe [2]. Par conséquent nous avons opté pour une solution issue de la géométrie algorithmique.

La difficulté est donc de réaliser un étiquetage de manière efficace. C'est dans cette étape que les partitionnements réguliers de type octree sont extrêmement performants puisque les éléments de volume sont cubiques et que par conséquent un simple balayage permet de trouver tous les voxels intérieurs au cube. Dans notre cas les éléments de volume sont polyédraux mais structurés dans le graphe de Delaunay. La nouveauté de notre technique d'étiquetage vient justement de l'utilisation de cette structure. Notre algorithme peut être décrit comme suit.

Algorithme d'étiquetage d'un polyèdre

1. Chercher le plus petit parallélépipède rectangle P_p contenant $Vor_S(p)$.

2. Pour chaque voxel $v \in P_p$ chercher le germe q le plus proche de v. Cette recherche est faite dans le graphe de Delaunay. Elle est initialisée par p (Chapitre 2 section 2.5.1).

3. Affecter v à $Vor_S(q)$.

L'intérêt de cet algorithme est que l'étape 2 se fait en $O(1)$. En effet la recherche du germe q le plus proche de v peut être initialisée par le germe p qui n'est pas très éloigné de v puisque v est contenu dans P_p (Figure 28).

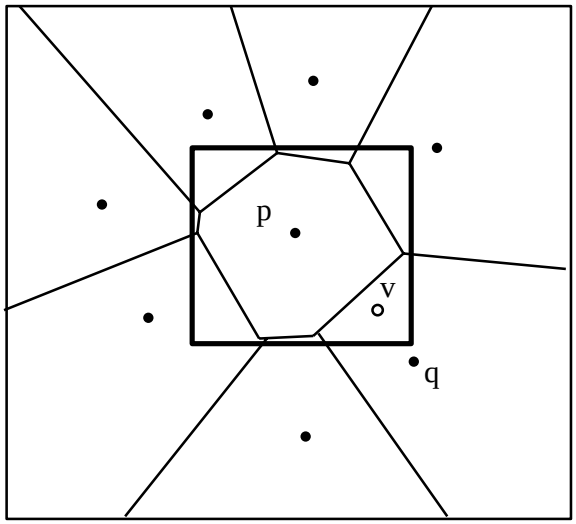


FIG. 28 - Boîte englobante associée au polygone $Vor_S(p)$. Le germe le plus proche de v est q la recherche se fait à partir de p.

Notre algorithme est donc théoriquement efficace. En pratique, la labélisation de tous les polyèdres de Voronoï d'un volume de $256 \times 256 \times 41$ se fait en quelques secondes.

3.3.3 Compression, extraction et mesures

Avant de nous plonger dans des exploitations plus complexes (chapitre 5 et chapitre 6), nous présentons dans cette section les premières applications possibles du partitionnement en polyèdres de Voronoï : compression, extraction et mesures.

Compression

Le codage par partitionnement en polyèdres de Voronoï d'une image 3D donne une représentation compressée du volume de données. Le taux T_C de compression est calculé de la manière suivante :

$$T_C = \frac{b_p xyz}{n(n_x + n_y + n_z + b_p)}$$

où x, y et z représentent la taille en x, en y et en z du volume de données, n représente le nombre de polyèdres obtenu après la phase de fusion de l'algorithme "split and merge" (section 3.3.1), n_x (resp. n_y et n_z) représente le nombre de bytes pour coder les abscisses (resp. les ordonnées et les altitudes) des germes, et enfin b_p est le nombre de bytes nécessaire pour coder l'intensité des niveaux de gris. Le numérateur représente donc le nombre total de bytes pour coder l'image avant la compression et le dénominateur représente le nombre total de bytes pour coder l'image après la compression.

Dans l'exemple de la cellule (3.3.4), l'image est composée de $256 \times 256 \times 41$ voxels. Chaque voxel est codé sur 256 niveaux de gris. Le volume de données aura donc une taille de $256 \times 256 \times 41 \times 8$. Le nombre de polyèdres obtenu, pour un écart type de 7 et après fusion, est d'environ 12000 ; le nombre de bytes nécessaire pour coder les polyèdres avec leurs niveaux de gris moyen est donc égal à $(8 + 8 + 6 + 8) \times 12000$. Le taux de compression est par conséquent de 60.

Extraction

Le principal outil utilisé pour extraire un objet 3D de l'image tridimensionnelle est l'algorithme de recherche d'une composante connexe dans un graphe sous

contraintes. La contrainte est dans ce cas sur les niveaux de gris.

L'algorithme consiste donc à regrouper de manière séquentielle les polyèdres qui ont un niveau de gris similaire. Chaque objet devra par conséquent correspondre à une composante connexe.

Cette méthode se révèle très efficace sur les images provenant du microscope confocal à balayage laser car les objets se distinguent très bien du fond fond non exploitable pour l'utilisateur. Les exemples donnés dans la section 3.3.1 illustrent bien ce fait.

Pour des images plus complexes nous verrons dans les chapitres 5 et 6 que des techniques plus évoluées seront utiles pour extraire les objets.

Nous nous sommes aussi intéressés aux algorithmes issus de la morphologie mathématique sur les graphes [139]. Nous en avons implémenté quelques uns : dilatation binaire et numérique érosion binaire et numérique zones d'influences Ligne de Partage des Eaux (LPE)... [84].

Il est apparu que les résultats n'étaient pas probants en ce qui concerne la segmentation. L'explication que nous en avons donnée est d'ordre théorique : si les boules sont convexes sur des grilles régulières (carrées ou hexagonales par exemple) ce n'est plus le cas sur des graphes aléatoires et en particulier sur le graphe de Delaunay (Figure 29). De ce fait nous n'avons plus l'inclusion des boules de rayon croissant et de même centre (avec la distance graphe deux points voisins sont à distance 1).

L'extraction des objets est nécessaire si on veut par la suite procéder à des mesures de paramètres caractérisant l'objet.

Mesures

En biologie et particulièrement dans l'étude du mode d'organisation du génome humain il est nécessaire de calculer la distance entre une séquence d'ADN (révélée par fluorescence) et une marque caractéristique (centromère) du chromosome. Un tel calcul est difficile puisque le plus souvent les objets (chromosomes) ne sont pas convexes.

Pour résoudre ce problème nous avons utilisé l'algorithme de Dijkstra [117] qui consiste à rechercher le plus court chemin reliant deux noeuds dans un graphe à poids positifs. Pour ce qui nous concerne le graphe utilisé pour un tel calcul est le graphe de Voronoï et le poids associé à deux noeuds du graphe est la distance euclidienne les séparant.

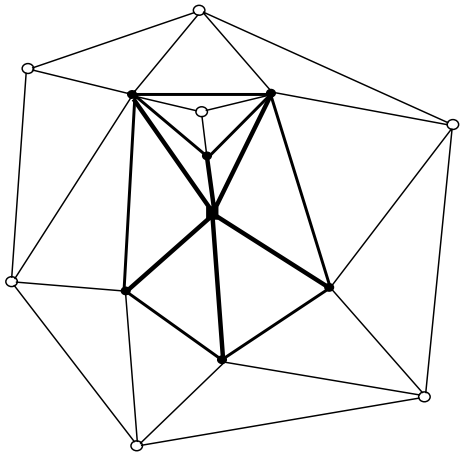


FIG. 29 - Problème posé par la dilatation binaire en morphologie mathématiques sur le graphe de Delaunay. Les points noirs forment la boule de rayon 1 et les points noirs et blancs la boule de rayon 2. Un point de la boule de rayon 2 est contenu dans l'enveloppe convexe des points formant la boule de rayon 1.

La raison fondamentale de ce choix provient d'une propriété importante du diagramme de Voronoï que nous détaillerons dans le chapitre 4 : le squelette d'un objet O est un sous-graphe du DVG de O. Cette propriété permet de lier une approximation du squelette des objets à un sous-graphe Sq du DVP : nous verrons que le DVG peut être approximé par le DVP en discrétisant l'objet O. Les sommets de Voronoï appartenant à Sq seront ceux ayant leurs quatre germes créateurs associés à des polyèdres intérieurs à l'objet (polyèdres appartenant à la composante connexe de l'objet obtenu par extraction).

Le calcul de la distance entre deux sommets s_1 et s_2 de Sq consiste alors à calculer la plus courte distance d_m entre s_1 et s_2 dans le graphe de Voronoï restreint à Sq en utilisant l'algorithme de Dijkstra. La complexité de cet algorithme est en $O(Card(Sq))$. En pratique, le calcul est fait en temps réel. Des études sont effectuées pour calculer l'erreur commise dans le calcul de Sq [4] et donc pour calculer l'erreur commise sur la distance d_m.

La figure 30 montre un résultat que nous avons obtenu en 3D.

FIG. 30 - Calcul de distances intra-chromosomales à l'aide de l'algorithme de Dijkstra. Le chemin est composé d'arêtes de Voronoï et s'approche du squelette du chromosome.

3.3.4 Résultats sur des images 3D

Le premier exemple est donné par l'image 3D d'un chromosome. L'étude de tels chromosomes est intéressante dans le cadre du projet européen *Human Genom* (GENO-CT91-0029). Le volume de données est composé de 31 coupes optiques de 128×128 pixels chacune. L'algorithme "split and merge" converge avec environ 7000 polyèdres en utilisant un écart type de 13. Il reste à peu près 6000 polyèdres après l'étape de fusion. Pour cet exemple le taux de compression est de 25. Le temps de calcul total est de 1 mn 30 s (Figure 27).

Le second exemple est l'image d'un noyau cellulaire. Le volume de données est composé de 41 coupes optiques de 256×256 pixels chacune (Figure 18). L'algorithme "split and merge" converge avec environ 15000 polyèdres en utilisant un écart type de 7. Il reste à peu près 12000 polyèdres après l'étape de fusion. Pour ce deuxième exemple le taux de compression est de 60. Le temps de calcul total est de 3 mn (Figure 31).

Un troisième exemple est donné par l'image d'un autre chromosome. Le volume de données est composé de 21 coupes optiques de 128×128 pixels chacune (Figure 32). On peut remarquer que ces images sont très bruitées. L'algorithme "split and merge" converge avec environ 6000 polyèdres en utilisant un écart type de 7. Il reste à peu prés 5000 polyèdres après l'étape de fusion. Pour ce troisième exemple le taux de compression est de 13 environ. Le temps de calcul total est de 1 mn (figure 33).

FIG. 31 - Différentes vues du résultat de l'algorithme "split and merge" à partir des polyèdres de Voronoï sur un noyau cellulaire

3.4 Etude du modèle octree et du modèle de Delaunay. Comparaison des partitionnements

L'algorithme "split and merge" sur le modèle de VoronoïΓdécrit dans la section 3.3Γpeut être étendu à d'autres modèles géométriques. Nous avons testé cet algorithme sur le partitionnement en octrees où les éléments de volume sont des cubesΓ et sur le partitionnement de Delaunay où les éléments de volume sont des tétraèdres.

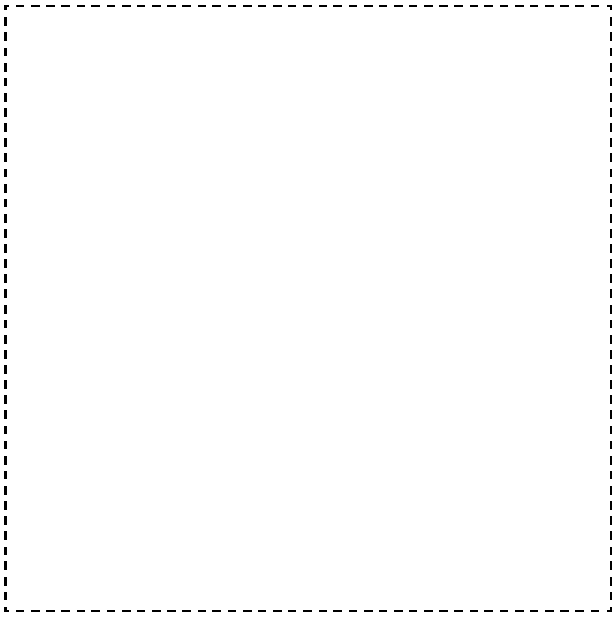


FIG. 32 - 16 sections d'une série de 21 sections (128×128) d'un chromosome obtenu avec un microscope confocal à balayage laser (CLSM) (LSM10 ZEISS).

3.4.1 Partitionnement par le modèle octree

De nombreux articles traitent des partitionnements en quadtrees et en octrees [118]. L'idée ici est d'utiliser ces partitions en analyse d'images. Une récente publication traite du sujet en 3D [108]. Les partitionnements en quadtrees ou en octrees font partieΓcomme nous l'avons déjà soulignéΓdes partitionnement que nous avons appelés *rigides*. L'algorithme peut être décrit séquentiellement et dans ses grandes lignes de la manière suivante :

Algorithme : Partitionnement octree

1. Initialiser l'octree avec toute l'image 3D (une seule feuille pour le volume entier).

2. Itérer jusqu'à convergence (toutes les feuilles sont homogènes).

   (a) Calculer la moyenne des niveaux de grisΓl'écart typeΓet le volume de chaque cube.

   (b) Pour chaque cubeΓsi le cube est non homogène (Définition 3.3Γadaptée à des cubes)Γalors le cube est subdivisé en 8 sous-cubes.

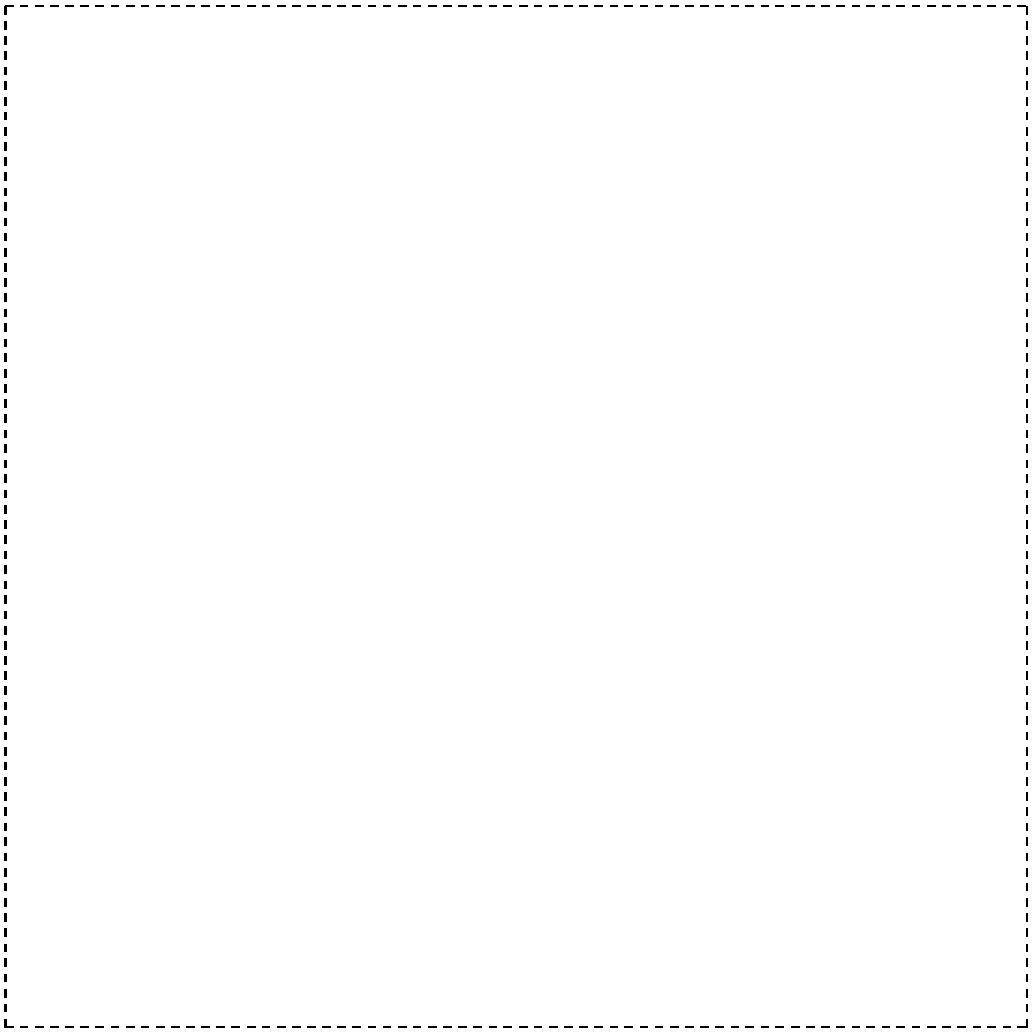


FIG. 33 - Différentes vues du résultat de l'algorithme "split and merge" à partir des polyèdres de Voronoï sur un chromosome.

De nombreuses variantes des octrees sont étudiéesΓnotamment en ce qui concerne le découpage d'une feuilleΓde façon à rendre moins rigide la manière de partitionner un cube. NéanmoinsΓtous les éléments de volume seront des parallélépipèdes rectangles. Telle que nous l'avons définie dans la section 3.3.1Γla fusion n'est pas réalisable pour les quadtrees ou les octrees.

La figure 34 illustre le processus en 2DΓrésultat à comparer avec les résultats des figures 26 et 36. La figure 35 donne le résultat obtenu pour le chromosome en 3DΓ résultat à comparer avec les résultats des figures 27 et 37.

FIG. 34 - Partitionnement en quadtrees de l'image "femme". De la gauche vers la droite et de haut en bas : image originaleΓ4 carrésΓ16 carrésΓ64 carrésΓ244 carrésΓ874 carrésΓ3037 carrésΓ9823 carrésΓreprésentation des carrés.

3.4.2 Partitionnement par le modèle de Delaunay

Comme nous l'avons déjà vu dans le chapitre 2Γsection 2.2Γsi S est un ensemble fini de points de $\mathbb{R}^3$Γle diagramme de Delaunay de S est une partition en tétraèdres de l'enveloppe convexe de S. Comme pour le modèle de VoronoïΓon voudrait utiliser ce partitionnement en analyse d'images. Une récente publication traite du sujet en 2D [40]. La différence essentielle entre ce partitionnement et celui obtenu avec des polyèdres de Voronoï consiste dans le fait que les éléments de volume disposent d'un nombre de faces figé (4 pour un tétraèdre). La procédure d'ajout s'effectuera selon le critère suivant :

Définition 3.6 (Ajout tétraèdre) *Soit T un tétraèdre non homogène. Un germe q sera ajouté sur le barycentre de T.*

L'algorithme peut être décrit de la manière suivante :

FIG. 35 - Partitionnement en octrees d'un chromosome (11500 cubes).

Algorithme : Partitionnement Delaunay

1. Ensemencer l'image par un **petit** nombre de germes selon un processus de Poisson.
2. Répéter jusqu'à convergence (tous les tétraèdres sont homogènes).
   (a) Calculer le diagramme de Delaunay.
   (b) Calculer la moyenne des niveaux de gris, l'écart type, et le volume de chaque tétraèdre.
   (c) Diviser (Définition 3.6) chaque tétraèdre non homogène (Définition 3.3 adaptée à des tétraèdres).
3. Fusion : suppression des tétraèdres inutiles (Définition 3.7).

En l'absence d'un critère aussi simple que celui employé dans le cas du partitionnement en polyèdres de Voronoï pour supprimer un tétraèdre, l'étape de fusion (étape 6) est basée sur la définition suivante :

Définition 3.7 (Tétraèdre inutile) *Soit $p \in S$. Si tous les tétraèdres du polyèdre étoilé $PE(p)$) sont homogènes et de même niveau de gris, alors les tétraèdres de $PE(p)$ sont dits inutiles.*

L'algorithme de fusion consiste alors à chercher tous les tétraèdres inutilesΓcorrespondant aux polyèdres étoilés $PE(p)$Γet à supprimer de la structure de Delaunay les germes pΓassociés à ces polyèdres étoilés.

La figure 36 donne une illustration du processus en 2D à comparer avec les figures 26 et 34 et la figure 37 donne le résultat obtenu sur le chromosome 3D à comparer avec les figures 27 et 35.

3.4.3 Comparaison des modèles de partitionnement

Pour commencerΓprenons l'exemple du noyau cellulaire que nous avons présenté dans la section 3.3.4. Pour chaque modèle de partitionnementΓnous avons comparé certains paramètres.

Modèle	Voronoï	Delaunay	octree
Eléments de volumes	polyèdres	tétraèdres	cubes
Nombre n de faces	$4 \leq n \leq 24$	$n = 4$	$n = 6$
Orientation des faces	arbitraire	arbitraire	OxyΓOxzΓOyz
Type du modèle	non rigide	non rigide	rigide
Nombre d'éléments	12000	15000	20000
Fusion	oui	oui	non
Squelette	oui	oui	non
Temps CPU	$3'$	$4'$	10"

Plusieurs points se dégagent de ce tableau. Nous notons d'abord un fait assez général que nous avons constaté dans la plupart des images que nous avons étudiéesΓ aussi bien en 2D qu'en 3D.

Le partitionnement en polyèdres de Voronoï est le mieux adapté à l'image. En effetΓc'est le partitionnement possédant le moins d'éléments de volume. A titre de comparaisonΓpour le partitionnement cubiqueΓun calcul simple montre qu'il faut 41984 cubes de tailles $4 \times 4 \times 4$ pour partitionner l'image du noyau cellulaire.

De mêmeΓen 2D et pour l'image muscle du GRECOIΓavec un écart type de 13Γ nous obtenons 8203 polygonesΓ12657 trianglesΓ14224 carrés dans un quadtree et 16384 carrés de taille 2×2 dans le partitionnement carré.

(a) Division. De la gauche vers la droite et de haut en bas : image originale, 5 triangles, 13 triangles, 62 triangles, 155 triangles, 365 triangles, 822 triangles, 1703 triangles, 3821 triangles.

(b) Résultat. Image originale à gauche, résultat de la division (5530 triangles) au centre et le graphe correspondant au-dessous, résultat de la fusion (4215 triangles) à droite et le graphe correspondant au-dessous.

FIG. 36 - Algorithme "split and merge" basé sur le diagramme de Delaunay.

FIG. 37 - Partitionnement en tétraèdres de Delaunay d'un chromosome (6500 tétraèdres).

D'après notre étude les partitionnements 3D peuvent être classés du moins adaptatif au plus adaptatifΓet du plus rigide au moins rigideΓcomme suit :

1. partitionnement non-adaptatif cubiqueΓ
2. partitionnement adaptatif en octreesΓ
3. partitionnement adaptatif en tétraèdres de DelaunayΓ
4. partitionnement adaptatif en polyèdres de Voronoï.

Toutefois on peut dire que les partitionnements en tétraèdres de Delaunay et ceux en polyèdres de Voronoï sont assez comparables au niveau des résultats obtenus.

De plusΓces deux modèles non rigides que nous avons présentés permettent de calculer une approximation du squeletteΓde l'exosquelette et du squelette par zone d'infuence d'objets. Une étude complète sera présentée dans le chapitre 4. La représentation en quadtrees ou octrees ne permet pas un tel calcul (Figure 38).

FIG. 38 - Avec le partitionnement de Voronoï il y a émergence du squelette et du SKIZ des objets. Une telle émergence n'existe pas avec un partitionnement en quadtrees.

Ces deux modèles du fait de l'étape de fusion sont beaucoup plus robustes à la translation et à la rotation que le modèle octree. La figure 39 illuste ce fait. De la gauche vers la droite et de haut en bas on obtient sur l'image d'un carré : 16 carrés pour l'image originale avec un quadtree 745 carrés en translatant l'image de 1 pixel en diagonale et 811 carrés avec une rotation de 10 degrés. Dans les mêmes conditions on obtient 408 polygones pour l'image originale 383 polygones pour l'image translatée et 404 pour l'image avec un rotation de 10 degrés. Par conséquent le nombre de polygones reste constant dans une certaine limite alors que le nombre de carrés est instable.

De plus on remarque que dans la représentation par polygones de Voronoï l'approximation du squelette est à peu près conservée.

Le prix à payer est un temps de calcul plus important pour les deux modèles non rigides (Voronoï et Delaunay) en comparaison du modèle rigide (octree).

3.5 Conclusion

Nous avons présenté dans ce chapitre différents modèles géométriques de partitionnement d'une image bidimensionnelle ou tridimensionnelle.

FIG. 39 - Non-invariance en rotation et translation des quadtrees et pseudo invariance des polygones de Voronoï en rotation et translation.

Deux composantes sont essentielles dans les partitionnements :

1. les régionsΓ

2. les relations de voisinage entre les régions.

Le premier point permet d'avoir un codage haut niveau des voxelsΓet le deuxième point permet de structurer les régions dans un graphe extrêmement informatif en ce qui concerne les pavages de Voronoï.

Dans ce casΓle partitionnement sera construit dans un environnement "split and merge". Des modifications peuvent encore être apportées à cet algorithme : le critère d'homogénéité peut être modifiéΓet la procédure d'ajout peut aussi être modifiée dans un environnement markovien comme nous le verrons dans le chapitre 6Γsection 6.5.

Des exploitations ont été présentées (compressionΓsegmentationΓmesures). Nous en verrons d'autres dans les chapitres 5 et 6Γen ce qui concerne la segmentation dans un environnement pyramidal contrôlé par un réseau de HopfieldΓou dans un environnement markovien.

Nous verrons aussi comment obtenir une partition de Voronoï répondant à un critère d'optimalité.

Notre méthode de partitionnement peut être utilisée dans d'autres applications demandant des outils d'analyse d'images 3D (e.g.Γtomographie Γscanner Γrésonnance magnétique...)Γet permet de réduire la taille des données (16 Méga-Octets) avant de procéder aux étapes d'analyse d'images.

Chapitre 4

Diagramme de Voronoï généralisé 3D.

Ce chapitre est consacré à l'étude de la généralisation du diagramme de Voronoï à un ensemble d'objets composés de parties affines (par exemple des polyèdres). Ces objets sont définis dans la section 4.2. Trois axes principaux ont été développés.

Le premier axe est une étude théorique sur les équations des séparateurs entre ces objets. Cette étude débouche sur un théorème de caractérisation.

Le deuxième axe est consacré au calcul d'une solution approchée du diagramme de Voronoï généralisé 3D (DVG 3D) en utilisant le diagramme de Voronoï ponctuel 3D (DVP 3D). Dans un théorème de convergenceΓnous apportons une démonstration de la validité de notre approche algorithmique.

Le troisième axe donne les connexions existantes entre squeletteΓexosqueletteΓ SKIZ (squelette par zones d'influence) et le diagramme de Voronoï généralisé 3D.

Comme dans le chapitre 3Γces connexions permettent de lier à nouveau la géométrie algorithmique à l'analyse d'images.

4.1 Introduction

L'étude de la généralisation du diagramme de Voronoï à un ensemble de polyèdres est liée à l'étude des squelettesΓexosquelettes et SKIZ (squelettes par zone d'influence) (section 4.5). En effetΓil a été démontré [83] que le squeletteΓl'exosquelette et le SKIZ d'un ensemble de polyèdres sont des sous-graphes du graphe de Voronoï généralisé. C'est cette connection entre l'analyse d'image et la géométrie algorithmique qui a motivé notre étude. Nous étudierons ici une classe d'objets plus large que celle des objets polyédriquesΓcelle des objets que nous désignerons "affines" (Définition 4.2Γsection 4.2).

Ce chapitre sera donc consacré aux diagrammes de Voronoï généralisés. Le concept de généralisation peut être vu sous différents angles :

1. généralisation par le choix de la métrique : en analyse d'imagesΓon peut utiliser différentes distances discrètes d4Γd8Γles distances du chanfrein ou la distance euclidienne [39]Γ
2. généralisation au sens de la dimension : 2DΓ3D et nD [9Γ30Γ140Γ49]Γ
3. généralisation en fonction des éléments : pointsΓsegmentsΓpolygones [83Γ74]Γ ou polyèdres [10].

En 2D de nombreux résultats existent déjà. On sait par exemple que les diagrammes de Voronoï basés sur un ensemble de polygones sont constitués de paraboles et de portions de droites (demi-droites et segments de droite). La figure 40 extraite de [74] en est une illustration. Nous donnerons une généralisation de ce résultat en 3D.

De plusΓil existe un algorithme pour calculer le DVG 2D de l'intérieur d'un polygone [74Γ83] et pour l'extérieur d'un ensemble de polygones [74]. Les approches utilisées sont de type "divide and conquer" pour l'intérieur des polygonesΓet incrémentale pour l'extérieur d'un ensemble de polygonesΓapproches que nous avons déjà rencontrées lors de l'étude du DVP 3D (Chapitre 2Γsection 2.3). Un tel algorithme n'existe pas en 3DΓet nous proposerons une approche *non exacte*Γ généralisation tridimensionnelle du travail trouvé en 2D dans [25Γ31Γ94].

Dans ce chapitreΓnous nous occuperons spécifiquement de la généralisation au sens des éléments (polyèdres...) et au sens de la dimension (3D). La métrique considérée dérive de la distance euclidienne (Définition 4.3Γsection 4.2).

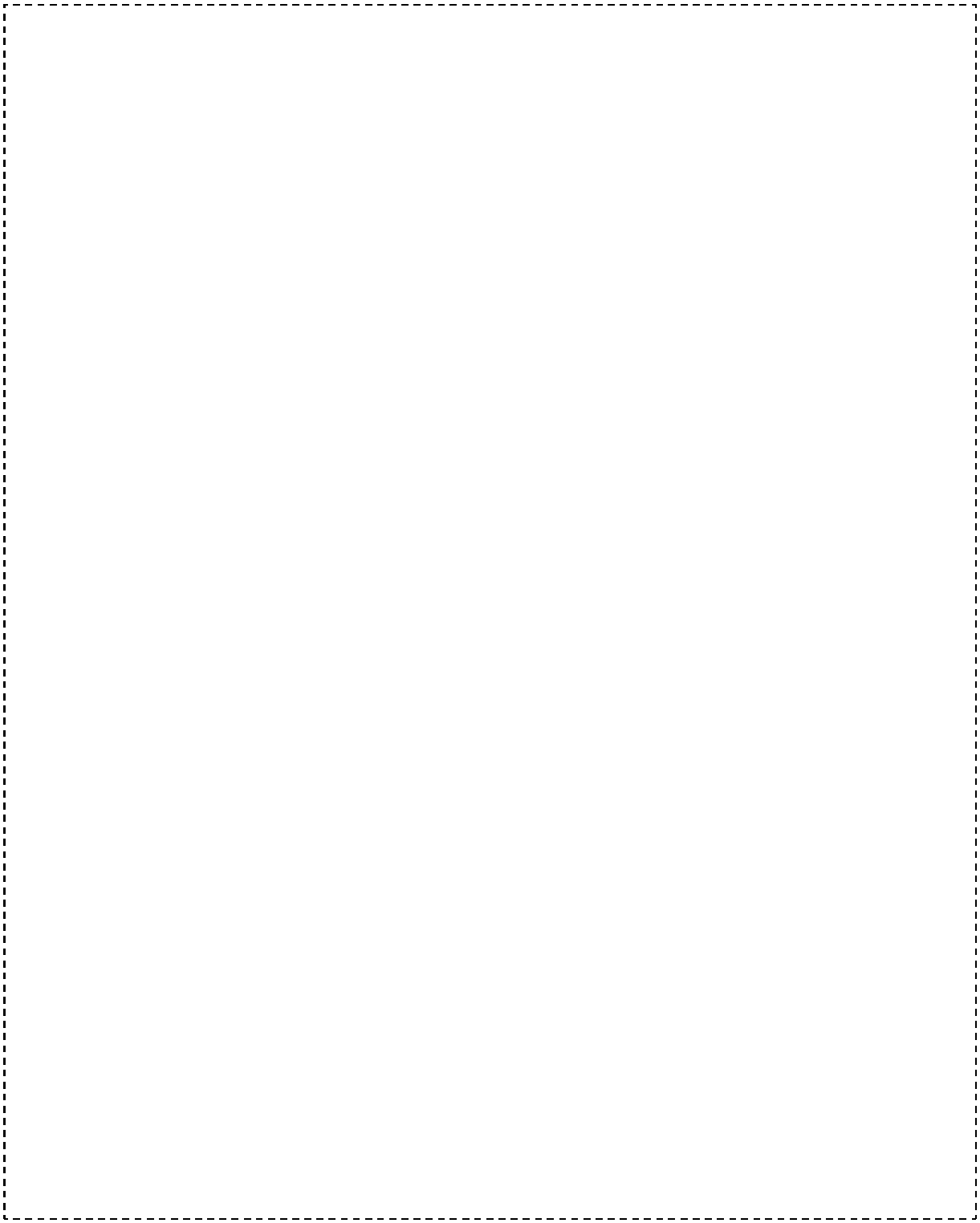


FIG. 40 - Bissectrices de deux éléments de type : point-point (a)Γpoint-segment (b)Γ point-segment dégénérés (c)Γdeux segments ouverts disjoints (d) et deux segments ouverts ayant une extrémité commune (e) (*extrait de [74]*)

Le plan que nous suivrons dans ce chapitre est le suivant. Dans la section 4.2Γ nous donnons quelques définitions formelles. La section 4.3 est consacrée à une étude détaillée du DVG 3D. Cette section aboutit au théorème 4.1. Dans la section 4.4Γun algorithme est présenté pour calculer le DVG 3D à partir du DVP 3D. Une démonstration de la convergence de notre algorithme est apportée à travers le théorème 4.2. Dans la section 4.5Γnous étudions les connexions existant entre squeletteΓexosqueletteΓSKIZ (squelette par zones d'influence) et le diagramme de Voronoï généralisé 3D. La section 4.6 est une annexe à la section 4.3.1 apportant des compléments de calculs. EnfinΓdans la section 4.7 nous donnons une conclusion et quelques perspectives.

4.2 Définitions

Notre but ici est de donner une définition de ce que nous entendons par *élément simple* et *objet*Γet de définir une distance entre les objets. Les bissectricesΓséparateurs et donc le DVG 3D que nous définirons ensuite seront dépendants du choix de cette distance. Cette section est donc destinée à apporter toutes les définitions nécessaires à la compréhension de ce chapitre.

Pour commencerΓla notion de base est celle d'*éléments simples*

Définition 4.1 (Eléments simples) *Points, segments ouverts et polygones ouverts seront appelés éléments simples.*

PuisΓnous passons à la notion d'objet.

Définition 4.2 (Objet) *Un objet (affine) Obj est défini comme un ensemble connexe d'éléments simples.*

Par exemple un segment fermé $[a,b]$ est l'union de trois éléments simples (un segment ouvert et ses deux extrémités) : $[a,b] =]a,b[\cup \{a\} \cup \{b\}$ (Figure 41).

De mêmeΓun polygone fermé$\Gamma\pi\Gamma$dont les sommets sont : $\{a_0,\ldots,a_{N-1}\}\Gamma a_0,\ldots,a_{N-1} \in \mathbb{R}^3\Gamma$est l'union de de son bordΓnoté $\partial\pi\Gamma$et de son intérieur $\mathring{\pi}$. Nous avons donc $\pi = \partial\pi \cup \mathring{\pi}\Gamma$où $\partial\pi$ est une union de segments fermés (Figure 42). Il en découle :

$$\begin{aligned} \partial\pi &= \bigcup_{i=0}^{N-1}]a_i, a_{(i+1) \bmod N}[\cup \bigcup_{i=0}^{N-1} \{a_i\} \\ \mathring{\pi} &= \pi - \partial\pi. \end{aligned}$$

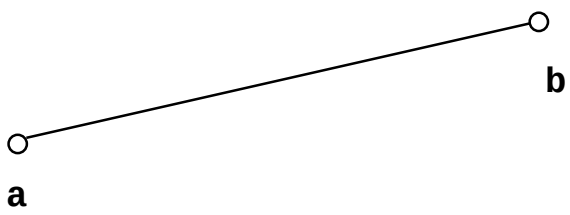


FIG. 41 - Segment fermé $[a, b]$ décomposé en un segment ouvert $]a, b[$ et deux points extrémités a et b.

Par le même procédé de décompositionΓun polyèdre quelconque de $\mathbb{R}^3$ peut être vu comme une union d'éléments simples. Les polyèdres (convexes ou non) seront doncΓde manière naturelleΓconsidérés comme des objets.

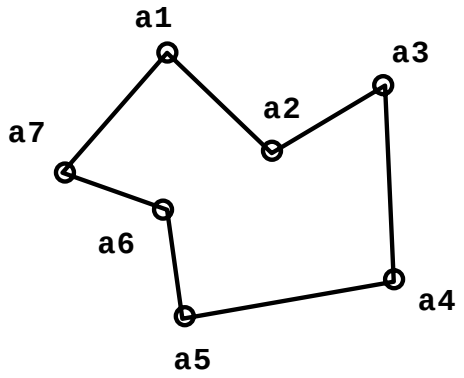


FIG. 42 - Polygone fermé décomposé en éléments simples : polygone ouvertΓsegments ouverts et points.

Définition 4.3 (Distance) *Soit p et q deux points de $\mathbb{R}^3$. On note $d(p, q)$ la distance euclidienne entre p et q. La distance entre un point p et un ensemble non vide S, notée $d(p, S)$, est définie par :*

$$d(p, S) = \inf_{q \in S} d(p, q).$$

Définition 4.4 (Projection) *La projection d'un point p de $\mathbb{R}^3$ sur un polygone π, notée $Pr(p)$, est la projection orthogonale du point p sur le plan Π contenant le polygone π.*

Une définition identique donne la projection d'un point p sur un segment faisant intervenir la droite supportant ce segment.

Définition 4.5 (Image) *L'image d'un point p de $\mathbb{R}^3$ sur un polygone π dont les sommets sont $\{a_0, \ldots, a_{N-1}\}$, $a_0, \ldots, a_{N-1} \in \mathbb{R}^3$, notée $Im(p)$, est la suivante :*

- *$Im(p) = P(p, \Pi)$, si la projection est située sur π, où Π est le plan contenant π (Figure 43, point p_1).*
- *$Im(p) = P(p, D_i)$, s'il existe $i \in \{0, \ldots, N-1\}$ tel que la projection soit sur $]a_i, a_{(i+1) \bmod N}[$, où les D_i sont les droites contenant les $]a_i, a_{(i+1) \bmod N}[$, $i \in \{0, \ldots, N-1\}$ (Figure 43, point p_2).*
- *$Im(p) = a_i$, où a_i est le point extrémité le plus proche, sinon (Figure 43, point p_3).*

La figure 43 illustre cette notion.

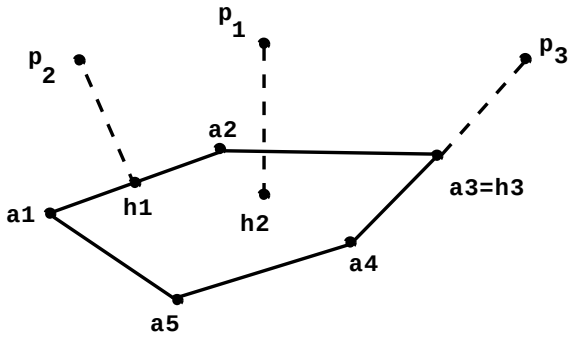


FIG. 43 - L'image de p_1 est sur le polygoneΓcelle de p_2 sur un segment et celle de p_3 sur un point.

D'après cette définition nous avons : $d(p, \pi) = d(p, Im(p, \pi))$.

Définition 4.6 (Bissectrice) *La bissectrice $B(e_i, e_j)$ entre deux éléments simples e_i et e_j est définie comme l'ensemble des points de l'espace équidistants de e_i et de e_j :*

$$B(e_i, e_j) = \{q \in \mathbb{R}^3, d(q, e_i) = d(q, e_j)\}.$$

où la distance est donnée dans la définition 4.3.

Les équations de ces bissectrices seront discutées dans la section 4.3.1.

Définition 4.7 (Séparateur) *Le séparateur entre deux objets distincts Obj_1 et Obj_2, noté $Sep(Obj_1, Obj_2)$, est défini comme l'ensemble des points équidistants de Obj_1 et de Obj_2 :*

$$Sep(Obj_1, Obj_2) = \{q \in I\!R^3, d(q, Obj_1) = d(q, Obj_2)\}$$

où la distance est donnée dans la définition 4.3

Définition 4.8 *Soit $S = \{Obj_p, p = 0, \ldots, N-1\}$ un ensemble de N objets. Le séparateur entre un $Obj_p \in S$ et les autres objets de S est défini comme l'ensemble des points équidistants de Obj_p et de $S - Obj_p$:*

$$Sep(Obj_p, S - Obj_p) = \{q \in I\!R^3, d(q, Obj_p) = d(q, Obj_l),\ Obj_l \in S - Obj_p\}.$$

où la distance est donnée dans la définition 4.3.

Quant aux équations de ces séparateursΓ elles seront discutées dans la section 4.3.2.

Définition 4.9 (Région) *La région $H(e_i, e_j)$ associée à l'élément e_i au vu de l'élément e_j est l'ensemble des points plus proches de l'élément e_i que de l'élément e_j:*

$$H(e_i, e_j) = \{q \in I\!R^3, d(q, e_i) \leq d(q, e_j)\}.$$

Il est à noter queΓcontrairement au cas du DVP 3DΓ$H(e_i, e_j)$ n'est pas un demi-espace en généralΓcomme nous le verrons dans la section 4.3.

Définition 4.10 *Soit Obj_1 et Obj_2 deux objets. La région de Voronoï notée $Cel(Obj_1, Obj_2)$ associée à Obj_1 par rapport à Obj_2 est l'ensemble des points plus proches de Obj_1 que de Obj_2. Pour un élément simple e_i, nous avons :*

$$Cel(e_i, Obj_2) = \bigcap_{e_j \in Obj_2} H(e_i, e_j),$$

donc, pour un objet, nous pouvons écrire :

$$Cel(Obj_1, Obj_2) = \bigcup_{e_i \in Obj_1} Cel(e_i, Obj_2).$$

Plus généralement nous avons la définition suivante.

Définition 4.11 *Soit $S = \{Obj_p, p = 0, \ldots, N-1\}$ un ensemble de N objets. La région de Voronoï notée $Cel(Obj_p, S - Obj_p)$, associée à l'objet Obj_p par rapport aux objets de $S - Obj_p$, est l'ensemble des points plus proches de l'objet Obj_p que des objets de $S - Obj_p$. Pour un élément simple e_i, nous avons :*

$$Cel(e_i, S - Obj_p) = \bigcap_{e_j \in S - Obj_p} H(e_i, e_j),$$

donc, pour un objet, nous pouvons écrire :

$$Cel(Obj_p, S - Obj_p) = \bigcup_{e_i \in Obj_p} Cel(e_i, S - Obj_p).$$

D'après les définitions 4.7 et 4.10 nous pouvons dire que les régions $Cel(Obj_1, Obj_2)$ et $Cel(Obj_2, Obj_1)$ sont adjacentes par le séparateur $Sep(Obj_1, Obj_2)$. Les régions de Voronoï de ces deux objets seront donc connues dès l'instant où l'équation de $Sep(Obj_1, Obj_2)$ sera donnée. C'est pourquoi nous mènerons une étude approfondie sur les séparateurs par le biais des bissectrices dans la section 4.3.

Nous pouvons maintenant définir la notion de diagramme de Voronoï généralisé 3D d'un ensemble d'éléments simples :

Définition 4.12 (DVG 3D basé sur les éléments simples) *Le diagramme de Voronoï généralisé, $VOR(S)$, pour un ensemble S d'éléments simples, est l'ensemble de toutes les régions de Voronoï associées à chaque élément simple :*

$$VOR(S) = \bigcup_{e_i \in S} Cel(e_i, S - e_i).$$

La figure 44 illustre cette notion en 2D où les éléments simples sont des points ou des segments ouverts.

Une autre notion liée à celle du squelette par zone d'influence est la notion du diagramme de Voronoï généralisé 3D d'un ensemble d'objets. Le DVG 3D d'un ensemble d'objets est en fait un sous-graphe du DVG 3D basé sur les éléments simples. Plus précisément nous avons la définition suivante :

Définition 4.13 (DVG 3D basé sur les objets) *Le diagramme de Voronoï généralisé, $VOR(S)$, pour un ensemble S d'objets, est l'ensemble de toutes les régions de Voronoï associées à chaque objet :*

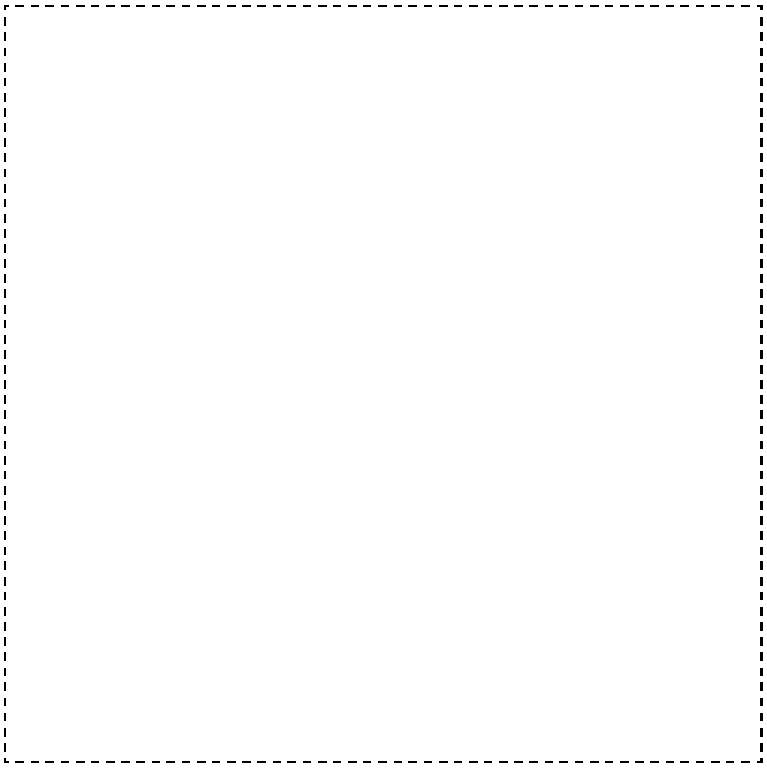


FIG. 44 - DVG 2D basé sur les éléments de l'intérieur d'un polygone (*extrait de [74]*).

$$VOR(S) = \bigcup_{Obj_p \in S} Cel(Obj_p, S - Obj_p).$$

La figure 45 extraite de [74] illustre cette notion en 2DΓoù les éléments simples sont des points ou des segments ouverts.

Le DVG 3D d'un ensemble d'objets peut être calculé à l'aide du DVG 3D de l'ensemble des éléments simples composant les objets.

Nous avons donc deux notions de DVG 3D : l'une est basée sur les éléments simples et l'autre sur les objets. La première notion sera à mettre en correspondance avec les squelettes et les exosquelettesΓet la deuxième notion avec le SKIZ (section 4.5). De plusΓpar constructionΓnous avons la propriété suivante liant ces deux notions :

Propriété 4.1 *Le DVG 3D basé sur un ensemble O d'objets est un sous-ensemble du DVG 3D basé sur les éléments simples composant les objets de O.*

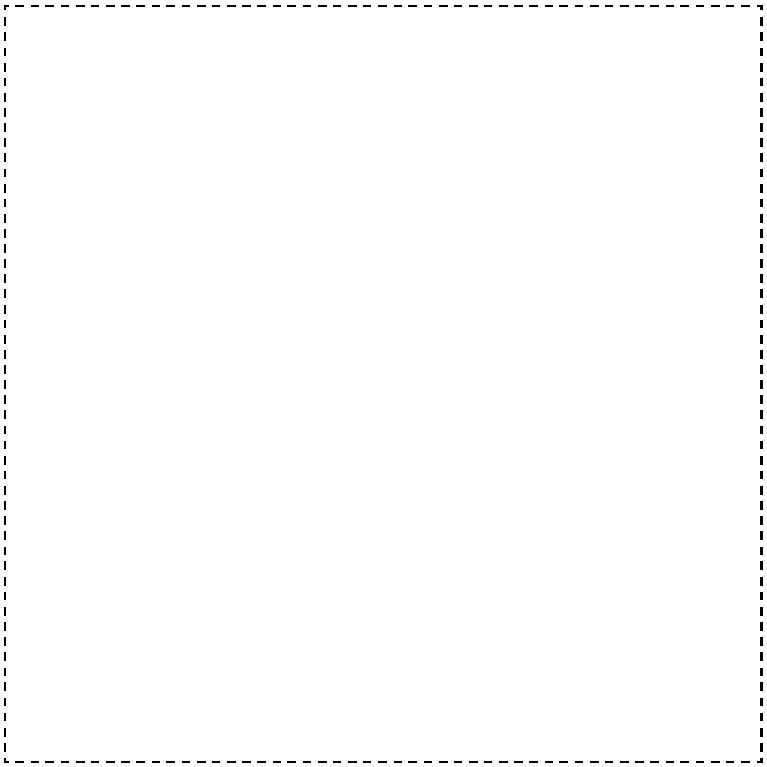


FIG. 45 - A gauche : DVG 2D de l'extérieur d'un ensemble de polygones basé sur les éléments. A droite : DVG 2D l'extérieur d'un ensemble de polygones basé sur les objets (*extrait de [74]*).

Pour résumer : le DVG 3D d'un ensemble d'éléments simples (resp. objets) est l'ensemble des régions associées aux éléments simples (resp. objets) : tel que chaque région R est composée des points plus proches de l'élément simple (resp. objet) associé à R que de tous les autres.

Remarque 4.1 *Si l'ensemble S est composé exclusivement de points, on reconnaît la définition classique du DVP 3D.*

4.3 Equations des bissectrices et des séparateurs

Dans la partie 4.3.1 les équations des bissectrices entre les éléments simples sont données en 3D. L'étude préliminaire que nous effectuons à propos de ces équations débouchera : dans la partie 4.3.2 : sur un théorème général relatif aux équations des séparateurs entre des objets.

4.3.1 Equations des bissectrices

Nous entendons ici par bissectriceΓla surface définie par l'ensemble des points les plus proches de deux éléments simples (Définition 4.6). Cette étude nous amène à considérer différents cas :

1. deux points a et $b\Gamma$
2. un point a et un segment $]b, c[\Gamma$
3. un point a et un polygone $\pi\Gamma$
4. deux segments $]a, b[$ et $]c, d[\Gamma$
5. un segment $]a, b[$ et un polygone $\pi\Gamma]a, b[\cap \Pi \neq \emptyset\Gamma$
6. un segment $]a, b[$ et un polygone $\pi\Gamma]a, b[\cap \Pi = \emptyset\Gamma$
7. deux polygones π_1 et π_2.

Nous ne considèrerons pas les cas dégénérés qui compliqueraient inutilement notre présentation.

Dans toute cette sectionΓ (x, y, z) désigneront les coordonnées d'un point quelconque P de $\mathbb{R}^3$.

• **Cas 1 : deux points a et b.** $B(a, b)$ est le plan médiateur entre a et b (Figure 46).

On obtient donc le lemme suivant :

Lemme 4.1 *La bissectrice entre deux points a et b est le plan médiateur des points a et b.*

Ce cas est le seul rencontré lorsque l'on s'intéresse au DVP 3D.

On retrouve bien qu'une cellule de Voronoï est un polyèdre convexe comme intersection de demi-espaces.

• **Cas 2 : un point a et un segment $]b, c[$.**

Le problème se ramène à un problème bidimensionnel en considérant le plan Π défini par les trois points $a\Gamma b$ et c. Dans ce plan la bissectrice est divisée en trois parties : 2 demi-droites créées respectivement par les deux points a et b et

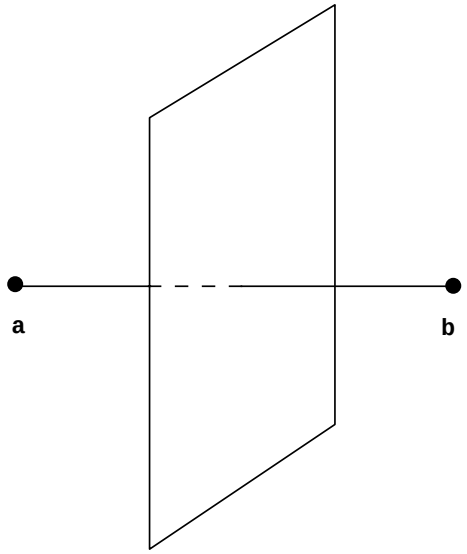


FIG. 46 - Bissectrice de deux points

les deux point b et $c\Gamma$ et un arc parabolique de foyer a et de directrice (a, b) [74]. Par conséquentΓ en 3DΓ la bissectrice devient une portion de surface parabolique complétée de deux parties planes (Figure 47).

FIG. 47 - Bissectrice entre un point a et un segment $]b, c[$ (surface parabolique et deux demi-plans).

Ce résultat nous permet d'écrire le lemme 4.2.

Lemme 4.2 *La bissectrice entre un point a et un segment* $]b, c[$ *est une union de parties planes et d'une portion parabolique.*

• **Cas 3 : un point a et un polygone π.**

Considérons d'abord le cas d'un plan Π et d'un point a. Soit $\{z = 0$ l'équation du plan $\Pi\Gamma$et a le point de coordonnées $(0, 0, 1)$. La projection orthogonale de P sur Π est $H(x, y, 0)$. Les points P situés à égale distance de Π et de a sont caractérisés par l'équation $x^2 + y^2 + 1 = 2z$ qui représente l'équation d'un paraboloïde de révolution (Figure 48).

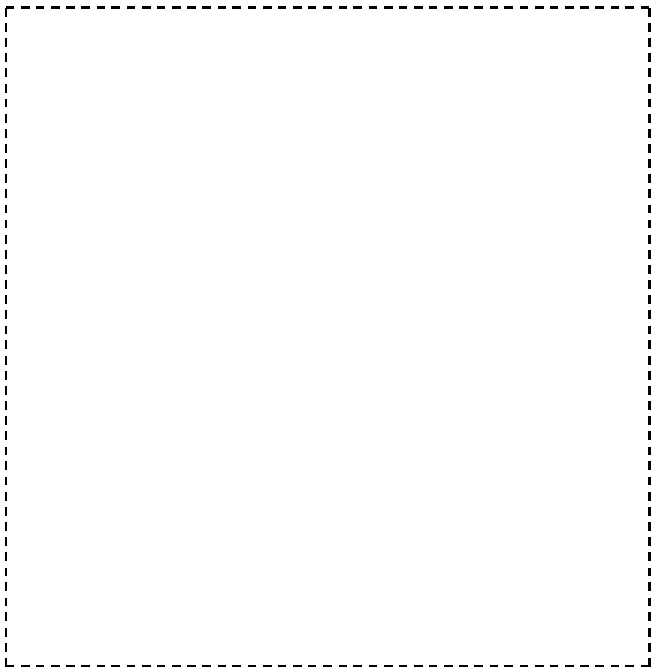


FIG. 48 - Bissectrice entre un point a et un carré (on ne considère ici que la partie paraboloïde de révolution).

Si l'équation du plan Π est plus complexeΓpar réduction de la forme quadratique obtenueΓon retrouve la forme réduite de l'équation d'un paraboloïde de révolution.

Considérons le cas plus général d'un polygone π représenté par ses N sommets $(a_0, \ldots, a_{N-1})$. Le problème se décompose alors en sous-problèmes : les points a et $a_i\Gamma i \in \{0, \ldots, N-1\}\Gamma$donnent des parties planes (cas 1). Le point a et les segments $]a_i, a_{(i+1) \bmod N)}[\Gamma(i \in \{0, \ldots, N-1\})$ génèrent des surfaces paraboliques (cas 2). De plus la bissectrice entre le point a et le polygone ouvert $\overset{\circ}{\pi}$ est une partie d'un paraboloïde de révolution (Figure 48). Pour obtenir la partie de ce paraboloïde de révolution il suffit de considérer la restriction du plan Π (contenant le polygone ouvert π) à $\pi\Gamma$et d'utiliser le résultat précédent. Nous pouvons donc conclure avec le lemme 4.3.

Lemme 4.3 *La bissectrice entre un point et un polygone est une union de parties planes, de parties paraboliques et d'une partie de paraboloïde de révolution.*

• **Cas 4 : deux segments** $]a, b[$ **et** $]c, d[$**.**

Nous supposerons que les deux segments ne sont pas coplanaires (le cas de la coplanarité se ramenant de manière évidente à un problème bidimensionnel).

Avant d'analyser le cas de deux segments, étudions le cas de deux droites orthogonales et non coplanaires D_1 and D_2 données par les équations suivantes :

$$D_1 \ : \begin{cases} y = 0 \\ z = 1 \end{cases} \text{ et } D_2 \ : \begin{cases} x = 0 \\ z = -1 \end{cases} .$$

La projection orthogonale d'un point P sur D_1 est $H_1(x, 0, 1)$ et la projection orthogonale de ce même point P sur D_2 est $H_2(0, y, -1)$. Les points situés à égale distance de D_1 et de D_2 sont caractérisés par l'équation : $y^2 - x^2 = 4z$. Cette équation est celle d'un paraboloïde hyperbolique (appelé aussi col) (Figure 49).

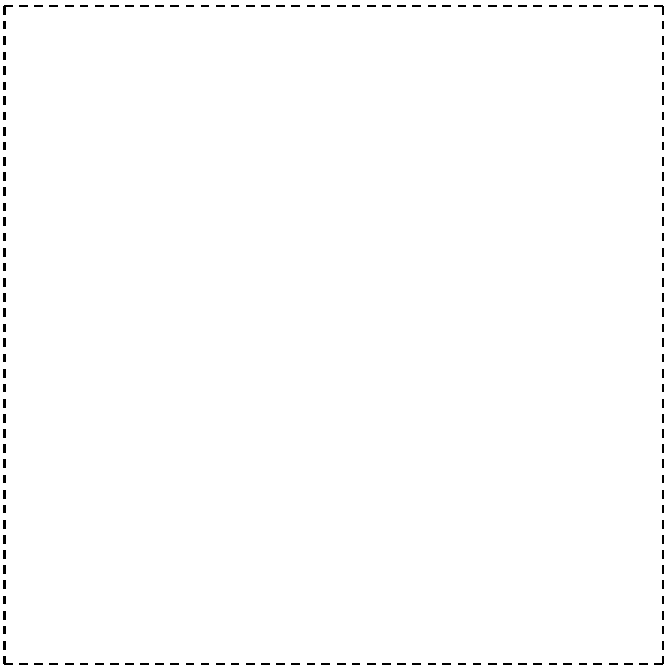


FIG. 49 - Bissectrice entre deux segments (on ne considère ici que la partie hyperbolique).

Le cas général de deux droites D_1 et D_2 non-coplanaires est obtenu par réduction de la forme quadratique obtenue. Le résultat est la forme réduite d'un paraboloïde

hyperbolique. Les détails sont reportés dans l'annexe 4.6.1Γet montrent que nous obtenons toujours un paraboloïde hyperbolique.

Nous pouvons aborder maintenant le cas de deux segments non coplanaires. La bissectrice contient 9 parties. Quatre parties planes générées par les bipoints $\{a, c\}$Γ $\{a, d\}$Γ$\{b, c\}$ et $\{b, d\}$ (cas 1)Γquatre portions de surface parabolique générées par les points extrémités aΓbΓc ou dΓet le segment complémentaire $]a, b[$Γou $]c, d[$ (cas 2). De plusΓnous avons une portion de paraboloïde hyperbolique générée par les deux segments ouverts $]a, b[$ et $]c, d[$ (Figure 49). Ce dernier résultat s'obtient en considérant les restrictions des droites D_1 (resp. D_2) contenant $]a, b[$ (resp. $]c, d[$) à $]a, b[$ (resp. $]c, d[$) et en utilisant le résultat précédent. Le cas 4 permet d'énoncer le lemme 4.4.

Lemme 4.4 *La bissectrice entre deux segments est une union de parties planes, de parties paraboliques, et d'une partie de paraboloïde hyperbolique.*

Dans les cas 5 et 6Γπ désignera un polygone et Π le plan contenant π. De plus le segment $]a, b[$ et le polygone π seront supposés non coplanaires.

• **Cas 5 : un segment $]a, b[$ et un polygone π, avec $]a, b[\cap \Pi \neq \emptyset$.**

Pour commencerΓconsidérons une droite D orthogonale à un plan ΠΓdonnés par les équations suivantes :

$$D \ : \ \begin{cases} x = 0 \\ y = 0 \end{cases} \text{ et } \Pi \ : \ \{ z = 0 \quad .$$

La projection orthogonale d'un point P sur D est $H_1(0, 0, z)$ et sa projection orthogonale sur Π est $H_2(x, y, 0)$. Les points P équidistants de Π et de D vérifient $x^2 + y^2 - z^2 = 0$. On reconnaît un cône de révolution d'axe z (Figure 50).

Le cas général d'une droite D non orthogonale à un plan Π est donné par réduction de la forme quadratique obtenue. Les détails sont reportés dans l'annexe 4.6.2Γet montrent que nous obtenons également un cône.

En ce qui concerne le cas d'un segment $]a, b[$ et d'un polygone πΓsous les hypothèses du cas 5Γpar décomposition du problème et en utilisant les lemmes précédentsΓ comme nous l'avons fait dans les cas 2Γ3Γet 4Γnous obtenons le lemme 4.5.

Lemme 4.5 *La bissectrice entre un segment et un polygone sous les hypothèses du cas 5 est une union de parties planes, de parties paraboliques, de parties de*

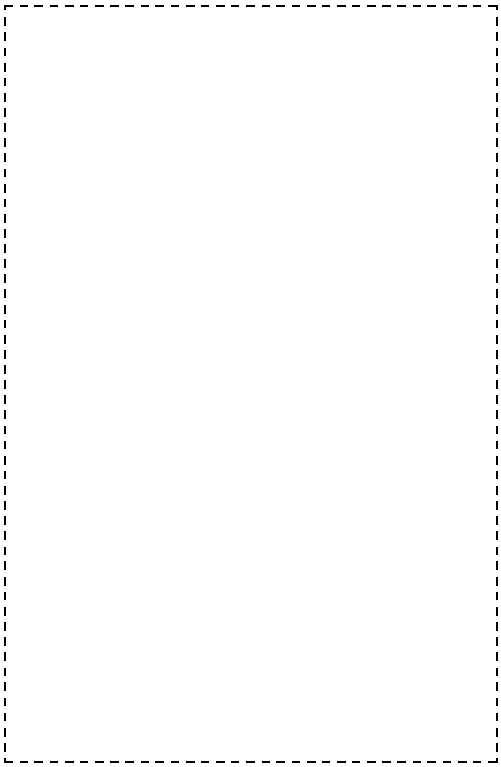


FIG. 50 - Bissectrice entre un carré et un segment orthogonal.

paraboloïdes hyperboliques, de parties de paraboloïdes de révolution, et d'une partie conique.

• **Cas 6 : un segment $]a,b[$ et un polygone π, avec $]a,b[\cap \Pi = \emptyset$.**

Le cas général d'une droite D et d'un plan Π est reporté dans l'annexe 4.6.3Γet montre que nous obtenons comme bissectriceΓune surface parabolique (Figure 51).

En ce qui concerne le cas d'un segment $]a,b[$ et d'un polygone πΓsous les hypothèses du cas 6Γpar décomposition du problème et en utilisant les lemmes précédentsΓ comme nous l'avons fait dans les cas 2Γ3Γ4Γet 5Γnous obtenons le lemme 4.6.

Lemme 4.6 *La bissectrice entre un segment et un polygone sous les hypothèses du cas 6 est une union de parties planes, de parties de paraboloïdes hyperboliques, et d'une partie parabolique.*

• **Cas 7 : deux polygones π_1 and π_2.**

Il est bien connu que la bissectrice entre deux plans est un plan. De mêmeΓpar restrictionΓpour deux polygonesΓla bissectrice contiendra une partie plane générée

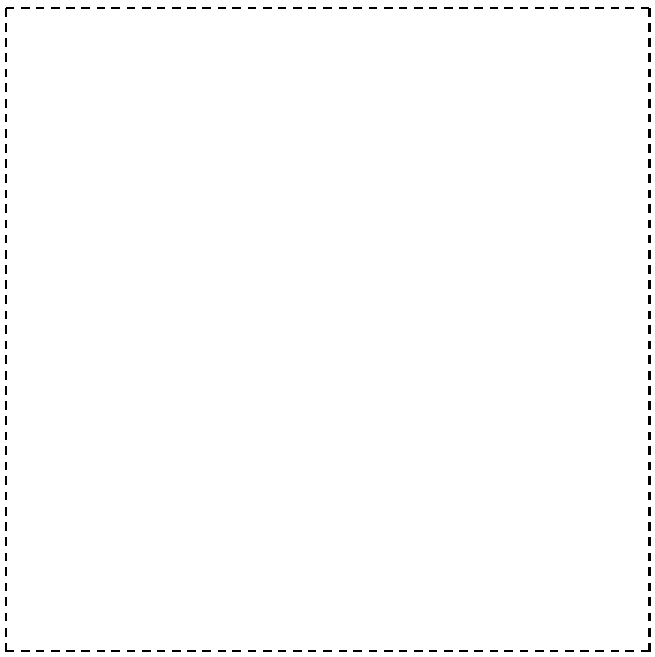


FIG. 51 - Bissectrice entre un carré et un segment "parallèle".

par π_1 et π_2. Les autres parties de la bissectrice résultent des lemmes précédents en utilisant la technique de décomposition. FinalementΓnous avons le lemme 4.7.

Lemme 4.7 *La bissectrice entre deux polygones est une union de parties planes, de parties paraboliques, de parties de paraboloïdes hyperboliques, de parties de paraboloïdes de révolution et de parties coniques.*

4.3.2 Equations des séparateurs

L'étude des différents cas examinés dans la partie 4.3.1 pour rechercher les équations des bissectrices nous conduit au théorème 4.1. Ce théorème est le résultat général relatif aux équations des séparateurs entre des objets tridimensionnels.

Théorème 4.1 (Caractérisation des séparateurs) *Soit S un ensemble d'objets "affines" (Définition 4.2). Les séparateurs entre les objets de S sont composés de parties des quadriques suivantes : paraboloïde de révolution : $x^2+y^2+z=1$, paraboloïde hyperbolique : $x^2-y^2=z$, surface parabolique : $x^2=z$, cône : $x^2+y^2-z^2=0$, et plan : $ax+by+cz+d=0$.*

Démonstration : Tous les objets sont formés d'une réunion d'éléments simples (définition 4.2). D'après l'étude de cas menée dans la partie 4.3.1Γ les équations

des bissectrices entre éléments simples sont celles de parties planesΓde surfaces paraboliquesΓde paraboloïdes hyperboliquesΓde paraboloïdes de révolution et de cônes. La définition 4.11 montre que les séparateurs seront uniquement composés de ces surfaces élémentaires et le théorème 4.1 est démontré. ■

Remarque 4.2 *Si les objets ne sont pas affines, alors les séparateurs ne sont pas nécessairement des surfaces représentées par des quadriques.*

4.4 Approximation du DVG 3D par le DVP 3D

Le calcul exact du DVG 3D pour un ensemble d'éléments simples ou d'objetsΓ est un problème de géométrie algorithmique qui n'est pas encore résolu. En effetΓde nombreuses difficultés algorithmiques apparaissent en 3D : le calcul de l'intersection de quadriquesΓla construction de l'enveloppe convexe en 3DΓle problème de visibilité de deux objets en 3D...

Par conséquentΓnous proposons dans cette section un algorithme qui permet de calculer une approximation du DVG 3D pour un ensemble d'éléments simples ou d'objets.

Le principe est de procéder à une discrétisation O_h des objets (Définition. 4.14) et d'utiliser le DVP 3D de l'ensemble des points obtenus par discrétisation. Cette approche demande donc une définition de la discrétisation des objetsΓla présentation de l'algorithme de calculΓet la démonstration du fait que le résultat de l'algorithme est bien une approximation du DVG 3D. C'est ce que nous nous proposons de faire dans cette section.

Définition 4.14 (Discrétisation : Objet) *Soit S un ensemble d'objets "affines" et O un objet de S. Une discrétisation de O est un ensemble fini de points O_h tel que, pour tout point P appartenant à O, il existe P_h dans O_h tel que $d(P, P_h) < h$, où h est appelé le pas de discrétisation.*

Définition 4.15 (Discrétisation : Ensemble d'Objets) *Soit S un ensemble d'objets. La discrétisation de S, notée S_h, est définie comme suit :*

$$S_h = \bigcup_{O^p \in S} O_h^p$$

Nous aurons également besoin de la notion d'arêtes inutiles dans l'algorithme qui va suivre.

Définition 4.16 (Arêtes inutiles) *Soit S un ensemble d'objets et S_h une discrétisation de S. Les arêtes de $VOR(S_h)$ créées par trois points de l'ensemble S_h appartenant à un même objet sont appelées arêtes inutiles (Figure 52).*

L'algorithme de calcul du DVG 3D à partir du DVP 3D peut être décrit dans ses grandes lignes de la manière suivante :

Algorithme de calcul du DVG 3D à partir du DVP 3D

1. Discrétisation des objets de l'ensemble S (Une telle discrétisation donne un ensemble S_h de points).
2. Calcul du DVP 3D des points de l'ensemble S_h.
3. Suppression des arêtes inutiles (Définition 4.16).

Cet algorithme est une généralisation en 3D d'un algorithme existant en 2D [25Γ 31Γ94]. Le théorème 4.2 montre que le résultat obtenu avec l'algorithme précédent est bien une approximation du DVG 3D de l'ensemble S d'objets.

Théorème 4.2 (Convergence) *Soit S un ensemble d'objets, et S_h une discrétisation de S. Le DVP 3D de S_h converge vers le DVG 3D de S quand le pas de discrétisation tend vers 0.*

Démonstration : Soit O^1 (resp. O^2) deux objetsΓet O_h^1 (resp. O_h^2) leur discrétisation. Soit O_h l'union de O_h^1 et de O_h^2. Soit s_h un sommet de Voronoï du DVP 3D basé sur O_h. s_h est créé par quatre points $p_1\Gamma\, p_2\Gamma p_3\Gamma$et p_4 n'appartenant pas au même objet car O^1 et O^2 sont linéaires. Deux cas se présentent :

1. $p_1\Gamma\, p_2\Gamma$ et p_3 sont sur O^1 et p_4 sur O^2. (p_1, p_2, p_3, p_4) est un tétraèdre de DelaunayΓ donc la sphère B circonscrite à ce tétraèdre est vide. De plus soit H (resp. H') la projection sur O^1 (resp. O^2) de s_h (Figure 53a). On a $d(H, p_i) < h, i \in \{1, 2, 3\}$. En effetΓsi $d(H, p_i) \geq h\Gamma$alors il existe p_n sur O_h^1 tel que p_n se trouve dans B (Définition 4.14)Γce qui donne une contradiction.

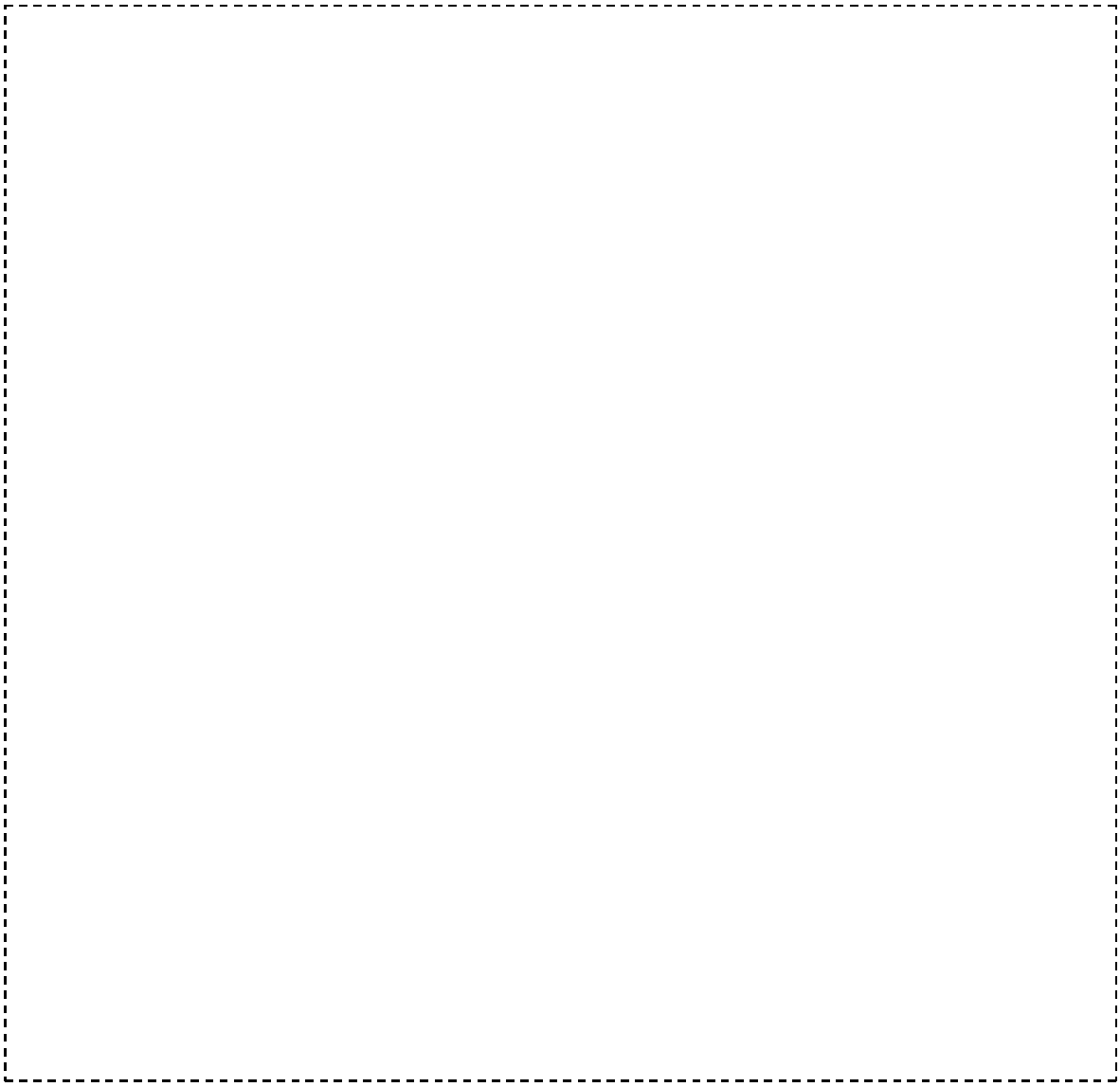


FIG. 52 - Suppression des arêtes inutiles : -a- et -c- avant la suppressionΓ-b- et -d- après la suppression. Les arêtes inutiles sont créées par trois points appartenant au même objet.

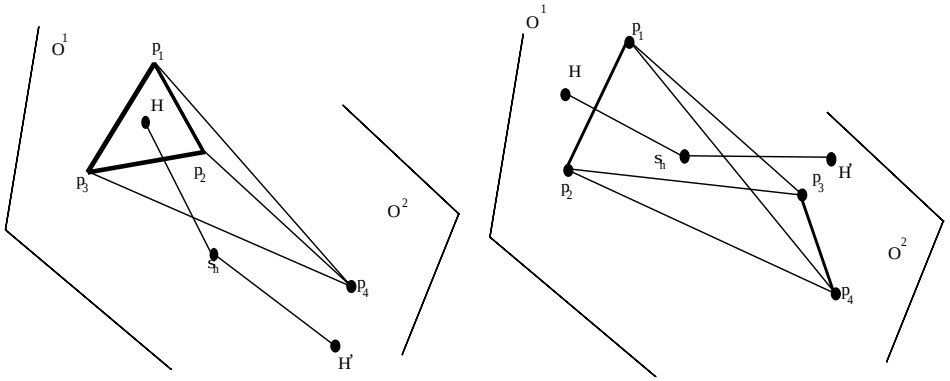
Pour la même raison $d(H', p4) < h$. Donc quand h tend vers zéroΓHΓp_1Γp_2Γ et p_3 convergent vers le même pointΓde même pour H' et p_4. A la limiteΓon a

$$\lim_{h \to 0} d(H', s_h) = \lim_{h \to 0} d(H, s_h)$$

et donc s_h converge vers s quand h tend vers 0Γoù s est un sommet de Voronoï de $VOR(O^1 \cup O^2)$Γce qui complète la preuve pour ce cas.

2. p_1 et p_2 sont sur O^1Γet p_3 et p_4 sont sur O^2 (Figure 53b). Un raisonnement similaire permet de conclure.

L'étude de ces deux cas termine la preuve du théorème 4.2. ■



(a) Premier cas du théorème. (b) Deuxième cas du théorème.

FIG. 53 - Illustration du théorème de convergence.

Il est à noter que ce théorème a été démontré à l'aide d'outils issus de la morphologie mathématique dans l'article de Schmitt [121].

La convergence de notre algorithme peut être visualisée en 3D par la figure 54.

D'autre partΓil faut noter que d'après le théorème 4.2 les surfaces qui émergent dans toutes les figures de cette section ne sont que des approximations des surfaces quadratiques décrites dans le théorème 4.1.

FIG. 54 - Convergence de l'algorithme d'approximation du calcul du DVG 3D par le DVP 3D.

4.5 DVG 3D et squelette 3D

Les squelettes d'objets binaires ont été étudiés par Blum en 1964 dans [21]. L'idée est d'avoir un codage des objets préservant leur connectivité et leur homotopie.

Une manière naturelle de définir le squelette d'un objet binaire tridimensionnel quelconque est formalisée dans la définition 4.17.

Définition 4.17 (Squelette) *Soit O un objet. Le squelette $S_k(O)$ de O est l'ensemble des centres des boules maximales (Définition 4.18).*

Soit $B(p,r)$ la boule ouverte de centre p et de rayon r.

$$B(p,r) = \{q \in \mathbb{R}^3 \text{ tel que } d(p,q) < r\}$$

Une boule maximale est alors définie de la manière suivante :

Définition 4.18 (Boule maximale) *Soit O un objet. $B_M(p,t)$ est une boule maximale de O si, et seulement si :*

- $B_M(p,t) \subset O$.

- $\forall B(q,r)$, *si* $B(q,r) \supset B_M(p,t)$, *alors* $B(q,r) \not\subset O$.

Définissons d'une manière équivalente à la définition 4.17 les squelettes. La nouvelle formalisation qui en découle est utile pour obtenir la proposition 4.1.

Définition 4.19 *Soit O un objet. Le squelette $S_k(O)$ de O est l'ensemble des points s intérieurs à O tel qu'il existe au moins deux points sur le bord de O noté ∂O qui sont équidistants de s et qui sont les plus proches de s. Formellement :*

$$S_k(O) = \{s \in I\!R^3 / \exists p_1, p_2 \in \partial O, d(s,p_1) = d(s,p_2) = d(s,\partial O)\}.$$

En 2D et 3D des études ont été faites pour calculer le squelette d'objets binaires quelconques par des approches discrètes [3Γ 98Γ 112]. La notion utilisée est l'axe médian. L'axe médian est l'ensemble des centres des boules maximales définies sur une grille discrète. L'axe médian est donc dépendant de la distance discrète utilisée. A partir de l'axe médian la notion de graphe de ligne médiane a été définie de manière à obtenir une approximation discrète hiérarchique et connexe du squelette continu [39Γ97].

Dans cette section nous nous placerons dans une approche résolument continue et sur une classe d'objets restreinte aux polyèdres (convexes ou non convexes). L'idée de notre approche est contenue dans la proposition 4.1. Mais avant de présenter cette propositionΓ il nous faut rappeler une propriété propre aux DVG. Cette propriété est une généralisation aux DVG de la définition 2.5 du chapitre 2Γsection 2.2.

Nous désignerons dans la suite par O un objet polyédriqueΓet e_iΓ$i \in \{0,\ldots,N-1\}$ sa décomposition en éléments simples :

$$O = \bigcup_{i=0}^{N-1} e_i$$

Propriété 4.2 *Deux éléments simples de O, e_i, et e_j, $i \neq j$, créent un sommet de Voronoï s si, et seulement si, $B(s, d(s,e_i))$ est vide (par définition, $d(s,e_i) = d(s,e_j)$).*

De cette propriété fondamentaleΓon déduit immédiatement la proposition suivante :

Proposition 4.1 *Le squelette de O est un sous-graphe du DVG 3D basé sur les éléments simples e_i, $i \in \{0,\ldots,N-1\}$.*

Pour obtenir une équivalence il est nécessaire d'introduire la notion de bissectrice dégénérée.

Définition 4.20 *Une bissectrice entre deux éléments simples e_i et e_j est dégénérée si, et seulement si, les deux éléments sont sur la même face, ou sur la même arête (Figure 55).*

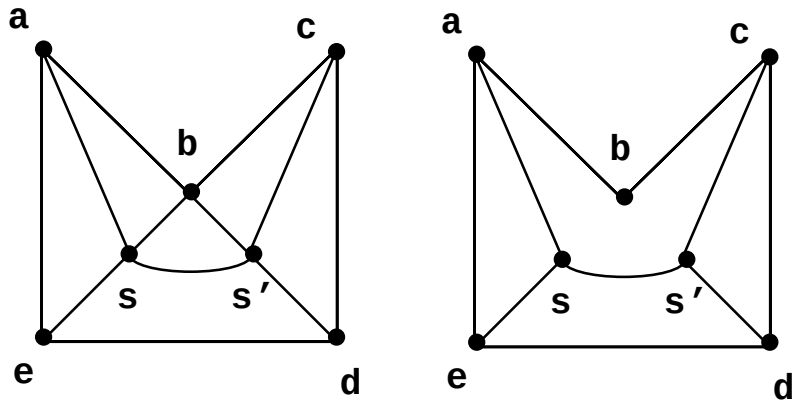


FIG. 55 - L'arête $]s, b[$ (resp. $]s', b[$) est dégénérée car créée par le segment $]a, b[$ (resp. $]b, c[$) et le point b appartenant à l'arête.

Cette définition est une généralisation du concept de "concavité" de Lee [83]. Nous avons préféré opter ici pour une définition moins géométrique car la notion visuelle de "concavité" ne se généralise pas facilement en 3D. Beaucoup plus de cas sont à considérer : sommets concaves convexes hyperboliques... Tous ces cas sont pris en considération dans notre définition.

Nous pouvons maintenant énoncer la proposition liant le squelette 3D et le DVG 3D.

Proposition 4.2 *Le squelette de O est l'ensemble des bissectrices définies par le DVG 3D basé sur les éléments simples e_i, $i \in \{0, \ldots, N-1\}$, dont on a supprimé toutes les bissectrices dégénérées.*

Démonstration : Soit $B(e_i, e_j)$ la bissectrice entre deux éléments simples e_i et e_j $i \neq j$. Pour tout s sur $B(e_i, e_j)$ on a $d(s, e_i) = d(s, e_j) = d(s, \partial O)$. De plus la projection h_i de s sur e_i (resp. h_j sur e_j) se situe par définition sur e_i (resp. sur e_j). Entre autre $h_i \neq h_j$ car la bissectrice $B(e_i, e_j)$ est non dégénérée. D'après

la définition 4.19Γ s appartient bien au squelette. RéciproquementΓsi $B(e_i, e_j)$ est dégénéréeΓalors les projections h_i et h_j sont confondues et on obtient une sphère vide au contact d'un seul point du bord de OΓce qui contredit la définition 4.19. ■

La figure 56 est une illustration du squelette 3D pour l'intérieur d'un cube.

La proposition 4.2 peut s'étendre immédiatement à l'exosquelette de O etΓavec quelques aménagementsΓau SKIZ d'un ensemble d'objets.

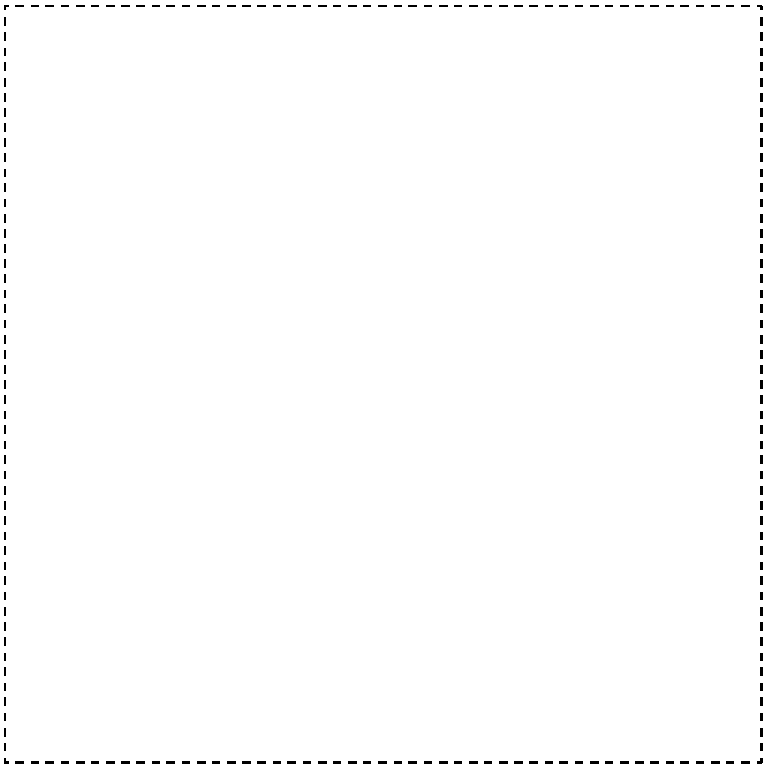


FIG. 56 - Squelette pour l'intérieur d'un cube.

En faitΓle SKIZ d'un ensemble d'objets est exactement l'ensemble des séparateurs obtenus par le DVG 3D basé sur les objets.

L'avantage de cette méthode pour calculer le squelette ou l'exosquelette d'un objet est que nous sommes libérés d'une grille discrète.

Un autre avantage est d'avoirΓde manière naturelleΓle squelette et l'exosquelette codés dans le graphe du DVG 3D basé sur les éléments simples. De plusΓle squelette ainsi calculé est exact.

Bien entenduΓen utilisant la méthode que nous avons présentée dans la section 4.4Γle squelette que nous obtenons est une approximation du squelette réel.

4.6 Annexes

4.6.1 Annexe A

Nous considérons ici l'étude du séparateur de deux droites non-coplanaires D_1 et D_2 de $\mathbb{R}^3$. Dans un tel cas nous pouvons prendre pour équation de D_1 et de D_2 en toute généralité :

$$D_1 : \begin{cases} y = 0 \\ z = -1 \end{cases} \text{ et } D_2 : \begin{cases} ax + by = 0 \\ z = 1 \end{cases} a \neq 0.$$

Soit $P(x, y, z)$ un point quelconque de $\mathbb{R}^3$. La projection de P sur D_1 est $P_1(x, 0, -1)$ et la projection de P sur D_2 est $P_2(x_{P_2}, y_{P_2}, 1)$ où :

$$x_{P_2} = \frac{-aby + b^2x}{a^2 + b^2} \text{ et } y_{P_2} = \frac{-abx + a^2y}{a^2 + b^2}.$$

Après calculs, l'équation de la bissectrice est la suivante : $4z = \alpha(x + \beta y)^2 - \gamma y^2$, et si nous posons $X = x + \beta y$, $Y = y$ et $Z = z$, nous obtenons :

$$4Z = \alpha X^2 - \gamma Y^2 \tag{7}$$

où les expressions de α, β et γ sont les suivantes :

$$\alpha = (\frac{1}{a^2 + b^2})^2 (a^4 + a^2b^2)^2 \tag{8}$$

$$\beta = \frac{a^3b + ab^3}{a^4 + a^2b^2} \tag{9}$$

$$\gamma = (\frac{1}{a^2 + b^2})^2 (a^4 + a^2b^2)[(\frac{a^3b + ab^3}{a^4 + a^2b^2})^2 + 1] \tag{10}$$

Les équations 8 et 10 montrent que α et γ sont strictement positifs ($a \neq 0$). En conséquence, l'équation (7) est bien l'équation d'un paraboloïde hyperbolique.

Si $a = 0$, alors D_1 et D_2 sont coplanaires, et la bissectrice dégénère selon le plan d'équation $\{z = 0\}$.

Dans le cas où les deux droites sont coplanaires (dans le plan Π) et non parallèles, alors nous nous ramenons au 2D en considérant le séparateur bidimensionnel dans le plan Π.

4.6.2 Annexe B

Dans cette annexeΓnous considérons l'étude du séparateur entre une droite D et un plan ΠΓsachant que Π et D ne sont pas coplanaires et sont d'intersection non vide. En toute généralitéΓnous pouvons considérer les équations suivantes pour Π et D :

$$D \ : \begin{cases} ax + bz = 0 \\ y = 0 \end{cases} (a,b) \neq (0,0) \text{ et } \Pi \ : \{z = 0.$$

Soit $P(x,y,z)$ un point quelconque de $\mathbb{R}^3$. La projection de P sur D est $P_1(x_{P_1}, 0, z_{P_1})$ où :

$$x_{P_1} = \frac{-abz + b^2x}{a^2 + b^2} \text{ et } z_{P_1} = \frac{-abx + a^2z}{a^2 + b^2}$$

et la projection de P sur Π est $P_2(x,y,0)$. Après calculsΓnous trouvons l'équation suivante pour le séparateur : $\alpha(x + \beta z)^2 + y^2 - \gamma z^2 = 0$.

Si nous posons $X = x + \beta z$Γ$Y = y$ et $Z = z$ nous obtenons finalement :

$$\alpha X^2 + Y^2 - \gamma Z^2 = 0 \tag{11}$$

Les expressions de αΓβ et γ sont données par les équations 8Γ9 et 10. De mêmeΓ comme α et γ sont strictement positifs $(a \neq 0)$Γl'équation 11 est bien celle d'une quadrique représentant un cône.

Si $a = 0$Γalors D est contenu dans ΠΓet le séparateur dégénère selon le plan d'équation $\{y = 0\}$.

4.6.3 Annexe C

Dans cette annexeΓnous considérons l'étude du séparateur entre une droite D et un plan ΠΓsachant que Π et D ne sont pas coplanaires et sont d'intersection vide. En toute généralitéΓnous pouvons considérer les équations suivantes pour Π et D :

$$D \ : \begin{cases} x = 0 \\ z = 1 \end{cases} (a,b) \neq (0,0) \text{ et } \Pi \ : \{z = 0.$$

Soit $P(x,y,z)$ un point quelconque de $\mathbb{R}^3$. La projection de P sur D est $P_1(0,y,1)$ et la projection de P sur Π est $P_2(x,y,0)$. Après calculsΓnous trouvons l'équation suivante pour le séparateur :

$$x^2 + 1 = 2z \tag{12}$$

L'équation 12 est bien celle d'une quadrique représentant une surface parabolique.

4.7 Conclusions

Dans ce chapitre nous avons donné des résultats généraux sur les DVG 3D. Notamment nous avons montré (Théorème 4.1 de caractérisation) que le DVG 3D d'objets "affines" tels que nous les avons définis dans la définition 4.2 est une union de quadriques : paraboloïdes de révolution paraboloïdes hyperboliques surfaces paraboliques cônes et plans.

En fonction de cette propriété une idée naturelle serait de vouloir construire le DVG 3D de manière exacte avec une approche spécifique comme on procède déjà dans le cas bidimensionnel [74 83].

Malheureusement de nombreuses difficultés algorithmiques apparaissent en 3D : le calcul de l'intersection de quadriques la construction de l'enveloppe convexe en 3D le problème de visibilité de deux objets tridimensionnels...

Par conséquent nous avons proposé un algorithme (partie 4.4) pour calculer une approximation du DVG 3D. Cet algorithme consiste à générer une discrétisation des objets à calculer le DVP 3D de l'ensemble ainsi trouvé et à supprimer les arêtes inutiles.

Nous avons ensuite étudié la connexion qui existe entre le DVG 3D basé sur les éléments d'une part et les squelettes et exosquelettes 3D d'autre part. Nous avons également fait état de la connexion entre le DVG 3D basé sur les objets et les SKIZ.

Les études menées dans [4] montrent que la difficulté essentielle pour étendre notre travail à des objets quelconques afin de calculer le squelette à partir d'un sous-graphe du DVP 3D par la technique que nous avons proposée dans la section 4.4 est dans la discrétisation des points.

Le problème réside dans le fait que l'échantillonnage des points entraîne du bruit sur les arêtes de Voronoï. Par conséquent plusieurs questions se posent :

- Quelle est la robustesse du calcul du DVG 3D à partir du DVP 3D si les points de discrétisation sont bruités ?
- Comment éliminer les arêtes provenant de ce bruit ?
- Comment choisir une "bonne" discrétisation ?

Pour la première question il semble que le calcul du DVG 3D à partir du DVP 3D soit assez robuste. Pour la deuxième question des solutions sont apportées en

2D dans [4Γ31]. EtΓpour la troisième questionΓil semble raisonnable de penser que le pas de discrétisation doit dépendre de la complexité de la forme (i.e. du rayon de courbure).

De plusΓdans le cas d'objets quelconquesΓnous pouvons noter que le théorème 4.1 n'est plus valableΓni même la preuve du théorème 4.2.

Le calcul des squelettes 3D d'objets quelconques à partir du DVG 3D est donc un sujet qui demande encore beaucoup de formalisations théoriques et de réalisations pratiques.

Chapitre 5

Segmentation : approche pyramidale dans un environnement de Voronoï

Dans ce chapitre nous présentons un algorithme de segmentation d'images utilisant une approche pyramidale dans un environnement de Voronoï.

Contrairement à l'approche classique qui consiste à initialiser le processus pyramidal avec la grille 8 connexe des pixelsΓ nous commençons par appliquer à la structure du diagramme de Voronoï l'algorithme "split and merge" que nous avons déjà présenté dans le chapitre 3Γsection 3.3.

Le contenu des nœudsΓà la base de la pyramideΓsera représenté par les polygones de VoronoïΓet leurs relations de voisinage par le graphe de Delaunay.

Ceci présente l'avantage de réduire le nombre de nœuds à la base de la pyramide et donc de réduire le nombre de niveaux du processus pyramidal.

Pour la construction de la pyramide irrégulièreΓ nous présentons un réseau de Hopfield contrôlant le processus de décimation [17].

Notre contribution concerne principalement l'initialisation de la pyramide irrégulière par le graphe de Voronoï et celui de DelaunayΓce qui permet de traiter des images de grande taille (e.g. des images tridimensionnelles). La validité de notre approche est illustrée par des exemples.

5.1 Introduction

Dans un processus de segmentation d'images à base de régionsΓ le but est de trouver une partition de l'image où chaque région vérifie un certain critère d'homogénéité (e.g. homogénéité dans les niveaux de grisΓou dans la texture). L'idéal serait aussi que chaque région obtenue par le processus de segmentation corresponde à un objet (ou une partie d'un objet) de la scène réelle. Pour tendre vers de tels objectifsΓde nombreux de nombreux algorithmes de segmentation ont déjà été proposés [142Γ73Γ119Γ134Γ99Γ63].

Quant à la définition formelle de la segmentationΓelle peut être formulée de la manière suivante :

Définition 5.1 (Segmentation) *La segmentation d'une image I est définie comme étant une partition de I en sous-ensembles I_n, $n = 1, \ldots, N$, tels que, si $Pred$ représente un prédicat d'homogénéité sur les composantes connexes, nous avons :*

- $I = \bigcup_{n=1}^{N} I_n$
- *I_n est une composante connexe, $n = 1, \ldots, N$*
- *$Pred(I_n)$ est VRAI pour tous les $n \in \{1, \ldots, N\}$*
- *$Pred(I_p \cup I_q)$ est FAUX pour $p \neq q$, $p, q \in \{1, \ldots, N\}$.*

Le prédicat dépend du modèle utilisé pour la segmentation : on peut avoir une similarité dans l'intensité des pixels (niveaux de gris similaires)Γou dans la textureΓ ou encoreΓune dissimilarité en termes de formes...

Il existe trois approches principales pour réaliser l'étape de segmentation :

1. l'approche régionΓ
2. l'approche contourΓ
3. l'approche région-contour.

A l'heure actuelleΓles chercheurs semblent s'orienter vers des méthodes qui coopèrent. Ils utilisent plusieurs détecteurs de contoursΓet l'intersection de tous les résultats doit donner la segmentation finale. Une autre voie d'investigation est de

trouver des méthodes à paramètres variablesΓde les faire varier dans un certain domaineΓet de chercher la combinaison optimale des paramètres donnant le meilleur résulat en segmentation. On parle alors d'approche multi-échelle.

L'approche que nous utilisons iciΓest une technique multirésolution utilisant les pyramides irrégulières [99]. Cette technique fait partie de l'approche région.

La technique pyramidale a déjà montré son efficacité pour la segmentation des images. L'apport original de notre approche est d'appliquer au préalable l'algorithme "split and merge" basé sur la structure irrégulière et non rigide du diagramme de VoronoïΓce qui nous donne une partition adaptée à l'image (Cf Chapitre 3Γsection 3.3). On construit alors sur cette structure de Voronoï une pyramide irrégulière. Le contenu des nœudsΓà la base de la pyramideΓest représenté par les polygones de VoronoïΓet leurs relations de voisinage par le graphe de Delaunay.

L'avantage de notre approche réside dans le fait que le graphe de Delaunay donne une description *compressée* de l'image à segmenterΓet que cette description est *adaptée* à l'image. Par conséquentΓnous obtenons moins de nœuds à la base de la pyramideΓet donc moins de niveaux dans le processus pyramidalΓen l'initialisant avec le graphe de Delaunay. Ceci devra nous permettre de traiter des images 3D.

Dans notre algorithmeΓle processus de décimation (Définition 5.4) est contrôlé par un réseau de Hopfield. En effetΓ Bishof a montré que ce type de réseau de neurones [70Γ17] peut être utilisé pour déterminer les survivants à chaque niveau de la pyramide irrégulière.

De plusΓce processus a l'avantage de pouvoir influencer directement la décimation en tenant compte des niveaux de gris pour la segmentation.

Il est aussi important de noter qu'en 2DΓle graphe de Delaunay présente l'avantage d'être planaire (ce qui n'est pas le cas pour la grille 8-connexe) et que la planarité est préservée dans la construction de la pyramide [81]. De plusΓen utilisant le principe des pyramides dualesΓon sait que le degré des faces [1] n'augmente pas [80]Γ ce qui rend possible une implémentation des pyramides duales irrégulièresΓen vue de la segmentationΓen parallèle.

La structure de ce chapitre est articulée de la manière suivante. Dans la section 5.2 nous présentons brièvement les pyramides irrégulières. La section 5.3 a pour objet la présentation du processus de décimation de Horst Bishof [17] utilisant les réseaux de Hopfield. Dans la section 5.4 nous exposons quelques résultats expérimentaux.

[1] Le degré d'une face est le nombre de sommets que contient cette face

Nous proposons dans la section 5.5 des perspectives importantes vers lesquelles s'orienter en vue d'améliorer de manière significative et innovante les résultats de segmentation. Enfin la section 5.6 est consacrée à une conclusion.

5.2 Pyramide

En analyse d'images une pyramide peut être définie comme étant une succession d'images dont la résolution décroît de manière exponentielle [131]. Classiquement la base de la pyramide est l'image originale. Dans le plus simple des cas les différents niveaux de la pyramide sont obtenus à partir des précédents par une simple opération de filtrage suivie d'une opération de décimation [66]. Des fonctions plus générales peuvent être utilisées pour permettre la réduction désirée. Ces fonctions sont appelées *fonctions de réduction.*

Trois propriétés importantes caractérisent une pyramide :

1. la structure : e.g. voisinage relations pères-fils entre les différents niveaux
2. le contenu d'un nœud : e.g. pixel arête...
3. la fonction exécutée par les nœuds : e.g. filtrage.

Dans cette section nous étudierons uniquement la structure des pyramides.

5.2.1 Structure

La structure d'une pyramide est déterminée par les relations de voisinage à un niveau donné et par les relations pères-fils entre deux niveaux consécutifs. Chaque niveau i de la pyramide peut être décrit par le graphe de voisinage $G_i = \langle V_i, A_i \rangle$ où l'ensemble des sommets V_i correspond aux pixels du niveau i et où $A_i \subseteq V_i \times V_i$ exprime les relations de voisinage des pixels. Deux sommets $p, q \in V_i$ sont adjacents dans G_i s'ils sont voisins dans la structure (Figure 57).

Définition 5.2 (Voisinage dans les pyramides ou graphe d'adjacence) *Le voisinage d'un sommet $p \in V_i$ est défini par*

$$\Gamma(p) = \{p\} \cup \{q \in V_i | (p, q) \in A_i\}.$$

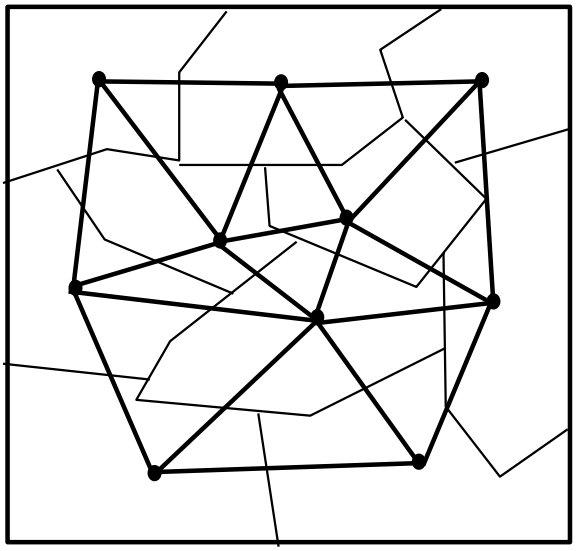


FIG. 57 - Exemple d'un graphe d'adjacence.

Dans l'algorithme de segmentation que nous présentons en section 5.4Γà la base de la pyramideΓle voisinage sera à mettre en relation avec le voisinage de Delaunay défini dans le chapitre 2Γsection 2.2Γdéfinition 2.4.

La structure verticale de la pyramide (i.e. les connexions entre deux niveaux consécutifs) peut être décrite par un graphe bipartie $R_i = \langle (V_i \cup V_{i+1}), L_i \subseteq (V_i \times V_{i+1}) \rangle$.

Toute pyramide avec n niveaux peut être décrite par n graphes de voisinage et $n-1$ graphes verticaux (structure verticale).

La notion de graphe de voisinage (définition 5.2) et de structure verticale permet de donner la notion très importante de champ récepteur (Figure 58).

Définition 5.3 (Champ récepteur) *Le champ récepteur (i.e. l'ensemble des fils) d'un nœud $q \in V_{i+1}$ est défini par :*

$$RF(q) = \{p | \langle p, q \rangle \in L_i\}.$$

On peut distinguer dans les pyramides différentes structures :

- structures régulières
- structures irrégulières

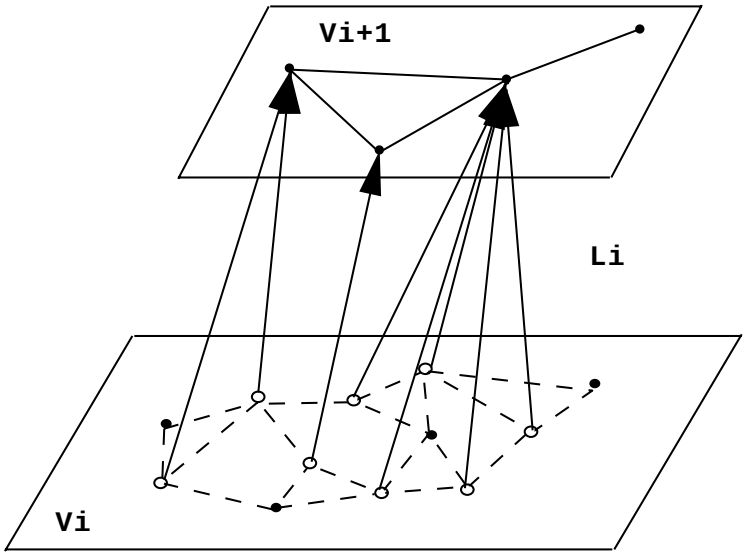


FIG. 58 - Représentation des champs récepteurs.

Pour les structures **régulières** (Figure 59)Γles relations de voisinage sont données par un graphe régulierΓ homogèneΓ de degré fixé (si on ne tient pas compte des problèmes de bords). Dans ce casΓles relations de voisinage sont indépendantes du nœud choisi dans le graphe (ce graphe peut très bien provenir d'un partitionnement non adaptatif et rigide tel que le partitionnement en carrés (chapitre 3Γ section 3.2.1)).

Si ces relations de voisinage sont dépendantes du nœud choisi dans le grapheΓ alors la pyramide est dite **irrégulière**.

FIG. 59 - Pyramide régulière.

Les pyramides régulières peuvent être mises en relation avec les partitionnements non adaptatifs et rigides (chapitre 3Γsection 3.2.1)Γet les pyramides irrégulières avec les partitionnements adaptatifs (chapitre 3Γsection 3.2.2).

Dans ce chapitre nous nous intéressons uniquement aux pyramides irrégulières.

5.2.2 Pyramides irrégulières

Dans ces pyramides les contraintes de régularité des pyramides régulières sont relâchées. Ces pyramides opèrent sur un graphe quelconque.

La principale raison de l'introduction des pyramides irrégulières est le comportement rigide (e.g. non invariance en translation et en rotation) des structures régulières [18]. Les pyramides irrégulières sont beaucoup plus flexibles [99].

Il existe essentiellement deux possibilités de construire des pyramides :

1. la contraction parallèle du graphe de voisinage [114]Γ
2. la décimation du graphe de voisinage [93].

Comme la contraction parallèle est possible uniquement pour certains types de graphes [114]Γnous détaillerons seulement la décimation du graphe de voisinage [93].

Le processus de décimation de Meer [93] divise les nœuds de la pyramide à un niveau donné en deux catégories : les nœuds qui survivent au niveau supérieur de la pyramide (**survivants**)Γet les nœuds qui n'apparaissent plus au niveau supérieur (**non survivants**). Meer [93] a donné deux règles pour le processus de décimation. Ces règles sont les suivantes :

Définition 5.4 (Règles de décimation)

1. *Deux voisins à un niveau i ne peuvent survivre ensemble.*
2. *Un nœud non survivant doit être voisin d'un nœud survivant.*

Une décimation vérifiant ces deux règles est appelée **décimation valide**. Il est bien connu que les règles 1 et 2 sont équivalentes au fait que les nœuds V_{i+1} du graphe G_{i+1} au niveau $i+1$ définissent un **stable** du graphe $G_i = \langle V_i, A_i \rangle$ (Figure 60).

Dans [93]Γil a été montré comment on pouvait obtenir le stable en parallèle par un algorithme stochastique.

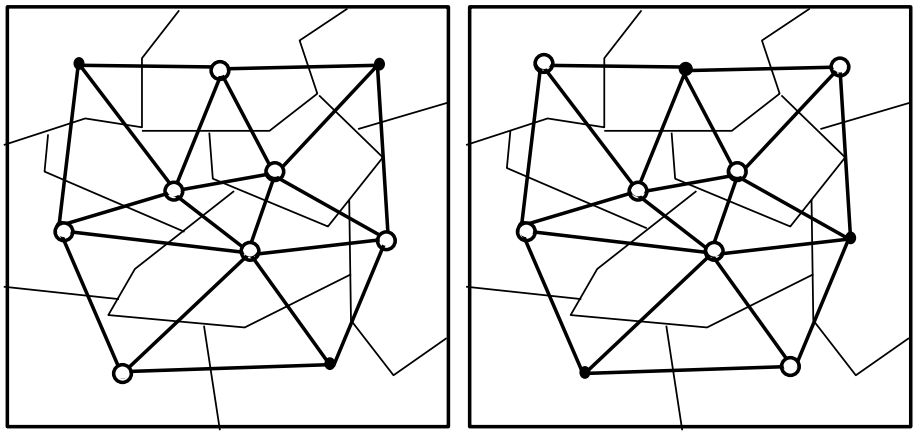


FIG. 60 - Deux exemples de stable pour un même graphe : les points noirs sont des nœuds survivants et les points blancs sont des nœuds non survivants.

L'algorithme de décimation stochastique de Meer peut être décrit dans ses grandes lignes (pour plus de détails voir [80]) comme suit :

Algorithme : Décimation stochastique :

1. Donner un nombre aléatoire suivant une loi de probabilité uniforme à chaque nœud.
2. Sélectionner les maxima locaux comme les nœuds survivants.
3. Boucher les trousΓ (i.e. répéter l'étape 2 tant qu'il existe des nœuds non survivants n'ayant pas de nœuds survivants dans leur voisinage).
4. Chaque nœud non survivant sélectionne un père. Cette construction définit le champ récepteur d'un nœud survivant.
5. Construire le graphe de voisinage du niveau supérieur. Deux nœuds survivants N_1 et N_2 deviennent voisins s'il existe un nœud dans le champ récepteur de N_1 et un nœud dans le champ récepteur de N_2 voisins au niveau inférieur.
6. Répéter les étapes 1–5 pour construire les niveaux supérieurs suivants jusqu'à ce qu'il ne reste plus qu'un nœud.

Cet algorithme peut être modifié pour tenir compte du contenu de chaque nœud (e.g. niveaux de gris). Ceci est réalisé par les pyramides adaptatives [76]Γ qui sont utilisées en segmentation d'image.

Nous allons donner dans la section suivante une alternative à l'algorithme de décimation de Meer par les réseaux de Hopfield. Cette nouvelle approche est due à Bischof [17]Γavec lequel nous avons collaboré pour l'adapter à la segmentation des images en niveaux de gris.

5.3 Réseaux de Hopfield

5.3.1 Réseaux de Hopfield

Une définition des réseaux de neuronesΓdonnée par Hecht-Nielsen [67]Γa été résumée de la manière suivante dans [88] :

"Un réseau de neurones est une structure de traitement de l'information parallèle et distribuéeΓconstituée d'unités de calcul interconnectées par des canaux unidirectionnels appelés *connexions*. Chaque unité de traitement n'a qu'une seule connexion de sortie qui peut être dupliquée en autant d'exemplaires que désiréΓles duplicata transportant tous le même signal. Le traitement effectué par chaque unité peut être défini de manière arbitraire pourvu qu'il soit complètement **local**Γc'est-à-dire qu'il ne dépende que des valeurs des signaux arrivant à l'unité par ses connexions entrantes et du contenu de la mémoire attachée à cette unité (Figure 61)." De plusΓ chaque connexion a un poids qui lui est attachéΓet le traitement local considéré pourra être une simple somme pondérée suivie d'un seuillage.

Il existe une grande variété de modèles de réseaux de neurones. En 1982 Hopfield en a introduit un nouveau [70]. Son réseauΓappelé *réseau de Hopfield*Γest composé d'unités binaires (i.e. chaque unité a deux états : 0 ou 1) totalement interconnectées les unes aux autres (excepté à elles-mêmes) : chaque unité i émet sa sortie vers toutes les unités j du réseau à travers une connexion affectée d'un poids w_{ij}.

HabituellementΓchaque modèle de réseau de neurones est lié à un algorithme d'apprentissage qui lui est spécifique (i.e. un algorithme qui spécifie comment changer les poids des connexions sous l'influence d'un stimulus extérieur). Les réseaux de HopfieldΓeuxΓseront plus volontiers attachés aux problèmes d'optimisation combinatoire [88].

Les états des unités du réseauΓà un moment donnéΓconstituent une matrice qui peut être considérée comme étant la matrice d'**état** du système.

FIG. 61 - Architecture d'un réseau de Hopfield *Extrait de [88].*

Hopfield a montré que le réseau converge toujours vers un état stable si :

- les poids associés aux connexions sont **symétriques** (i.e. $w_{ij} = w_{ji}$ où w_{ij} est le poids de la connexion entre l'unité i et l'unité j)Γ
- les unités sont mises à jour de manière **asynchrone** selon la sortie décrite ci-après.

La sortie d'une unité i est donnée par la procédure suivante :

$$s(i) = \begin{cases} 1 & \text{si } I_i + \sum\limits_{j \neq i} w_{ij} s(j) > U_i \\ 0 & \text{sinon} \end{cases}$$

En effetΓ Hopfield a montré que le réseau est gouverné par l'énergie E donnée dans l'équation 13 :

$$E = -\frac{1}{2} \sum_i \sum_j w_{ij} s(i) s(j) - \sum_i I_i s(i) + \sum_i U_i s(i) \tag{13}$$

où $w_{ij} \in \mathbb{R}$ est le poids entre les nœuds i et jΓ I_i le champ extérieur et U_i le seuil associé au nœud i.

Cette énergie ne peut que décroître au cours du temps. En effetΓsi au cours d'une itération une unité i change d'étatΓ la variation d'énergie ΔE peut s'écrire de la manière suivante :

$$\Delta E = -[\sum_{j \neq i} w_{ij} s(j) + I_i - U_i] \Delta s(i) \tag{14}$$

Or la quantité $\Delta s(i)$ est positive si et seulement si la quantité entre crochets de l'équation 14 est positive et similairement pour le cas négatif. Par conséquent tous les changements d'état du réseau vont contribuer à la décroissance de l'énergie E. De plus E est bornée donc le réseau converge vers un état stable. Cet état correspond à un minimum local de E et non à un minimum global puisque l'algorithme de minimisation correspond à une descente en gradient.

Hopfield a aussi présenté un algorithme d'apprentissage mémorisant quelques formes qui correspondent aux états stables du réseau. Malheureusement la capacité de mémorisation des réseaux de Hopfield est très limitée. Dans [90] il a été montré que dans un réseau de n unités binaires le nombre de formes que l'on peut apprendre est borné asymptotiquement par $\frac{n}{4 \log n}$.

C'est pourquoi l'aspect le plus intéressant des réseaux de Hopfield réside dans leur capacité à converger vers un minimum d'énergie [88]. Cette propriété peut donc être utilisée pour résoudre des problèmes d'optimisation (e.g. voyageur de commerce [72 77]) en les codant dans le réseau. Les variables doivent correspondre aux états des unités et les contraintes ainsi que la fonction à optimiser doivent être codées dans les poids de telle sorte qu'un état d'énergie minimale corresponde à une solution repectant les contraintes.

En 1984 Hopfield a introduit un modèle continu [71] (i.e. l'état des unités peut changer continûment). Ce modèle a les mêmes caractéristiques que le modèle binaire.

Dans cette section nous utiliserons les réseaux de Hopfield comme une procédure d'optimisation pour trouver un stable à un niveau donné de la pyramide. Par contre nous ne serons pas intéressés par le minimum global de l'énergie de l'équation 15. En effet comme il va être montré dans le théorème 5.1 que Bishof a énoncé il nous sera suffisant de trouver un minimum local.

5.3.2 Décimation par réseaux de Hopfield

Dans cette section nous allons montrer comment Bishof a remplacé les étapes 1–3 de l'algorithme de décimation stochastique de Meer par un réseau de Hopfield qui opère sur le graphe de voisinage $G = \langle V, A \rangle$ [2]. Il est à noter qu'une décimation par

[2] L'indice i est supprimé car l'algorithme travaille à un niveau donné

réseaux de Hopfield est plus générale que la décimation stochastique telle qu'elle a été proposée par Meer et qu'elle contient de manière naturelle le concept de pyramide adaptative (section 5.5.1).

Avant de poursuivre introduisons une définition :

Définition 5.5 (Fonction d'état) *L'état d'un nœud $p \in V$ est donné par la fonction :*

$$s : V \mapsto \{0,1\} \text{ avec } s(p) = \begin{cases} 1 & \text{si le nœud } p \text{ survit} \\ 0 & \text{sinon} \end{cases}$$

L'énergie que nous considérerons est la suivante :

$$E = -\frac{1}{2} \sum_{\langle i,j \rangle \in A} w_{ij} s(i) s(j) - \sum_{k \in V} I_k s(k) \tag{15}$$

Bishof a montré le résultat suivant :

Théorème 5.1 (Bishof) *L'énergie E donnée dans l'équation 15 converge vers un minimum local, E_{min}, avec $I_k > 0 \ \forall k \in V, w_{ij} = w_{ji} < 0$ et $I_k < |w_{ij}| \ \forall k \in V, \langle i,j \rangle \in A$ si, et seulement si, l'attribution des nœuds survivants et non-survivants, $s(p)$, donne une décimation valide (i.e. les règles 1 et 2 de la définition 5.4) ou, de manière équivalente, forme un stable de G.*

Démonstration : La preuve est détaillée pour un cas plus particulier dans [17 16] et pour le cas général dans [8] . ■

Le résultat remarquable contenu dans le théorème 5.1 est que pour obtenir un stable à un niveau donné d'une pyramide (irrégulière ou non) il suffit de trouver un **minimum local** de l'énergie donnée par l'équation 15. Bien souvent c'est plutôt le minimum global qui est intéressant (e.g. segmentation par champ de Markov).

Pour minimiser l'énergie E (équation 15) on peut définir un réseau de Hopfield opérant sur le graphe de voisinage G. Il suffit de poser dans l'équation 13 $U_i = 0$ pour obtenir l'énergie donnée par l'équation 15.

Un minimum local donnera le stable (ou décimation valide définition 5.4) à un niveau donné de la pyramide irrégulière. La mise à jour de la valeur d'une unité p dans le réseau est faite comme suit :

$$s(p) = \begin{cases} 1 & \text{si } I_p + \sum_{\langle q,p \rangle \in A} w_{qp} s(q) > 0 \\ 0 & \text{sinon} \end{cases}$$

L'état initial du réseau de Hopfield peut être choisi de manière aléatoire.

Les résultats sur la complexité de la convergence sont importants car on sait que l'algorithme de Meer [93] converge en $O(|V|)$ étapes dans le pire des cas.

Dans [124]Γil a été montré que les réseaux de Hopfield avec des poids négatifs (si un poids est nulΓla connexion sera considérée comme non présente) conjugués à un algorithme séquentiel de mise à jour (i.e. à un temps donné seulement une unité est mise à jour) convergent dans le pire des cas en $2|V|$ étapes.

Dans tous les cas pratiques que nous avons rencontrésΓpour $|V| \approx 10000$Γl'algorithme converge après seulement 6 itérations environ.

La figure 62 est une illustration de la décimation d'un graphe de Delaunay par un réseau de Hopfield.

Dans la section 5.4 nous allons montrer comment choisir les poids entre les arêtes de manière à avoir de bons résultats en segmentation.

5.4 Application en segmentation

Nous présentons ici quelques résultats obtenus illustrant la coopération entre les pyramidesΓles diagrammes de Voronoï et les réseaux de Hopfield. Les poids des arêtes du graphe de voisinage sont choisis comme suit :

$$w_{ij} = -f(d_{ij}) \text{ avec } d_{ij} = |g_i - g_j|$$

où g_i et g_j sont les niveaux de gris des nœuds i et jΓet f une fonction à définir. Nous avons choisi de prendre comme fonction :

$$f(d_{ij}) = \begin{cases} 2 & \text{si } i \text{ et } j \text{ sont voisins et } d_{ij} \leq \Theta \\ 0 & \text{sinon} \end{cases}$$

Une telle fonction permet non seulement de construire la pyramide en tenant compte de l'information des niveaux de gris contenus dans l'image à segmenterΓ mais aussi de vérifier les deux règles de décimation de Meer si on se place dans le cadre du graphe de similarité qui est un sous- graphe du graphe de voisinage.

Définition 5.6 (Graphe de similarité) *Soit $G = \langle V, A \rangle$. Le graphe de similarité est le graphe $R = \langle V, A^R \rangle$, où :*

$$A^R = \{(p,q) \in A \text{ tel que } w_{pq} \neq 0\}$$

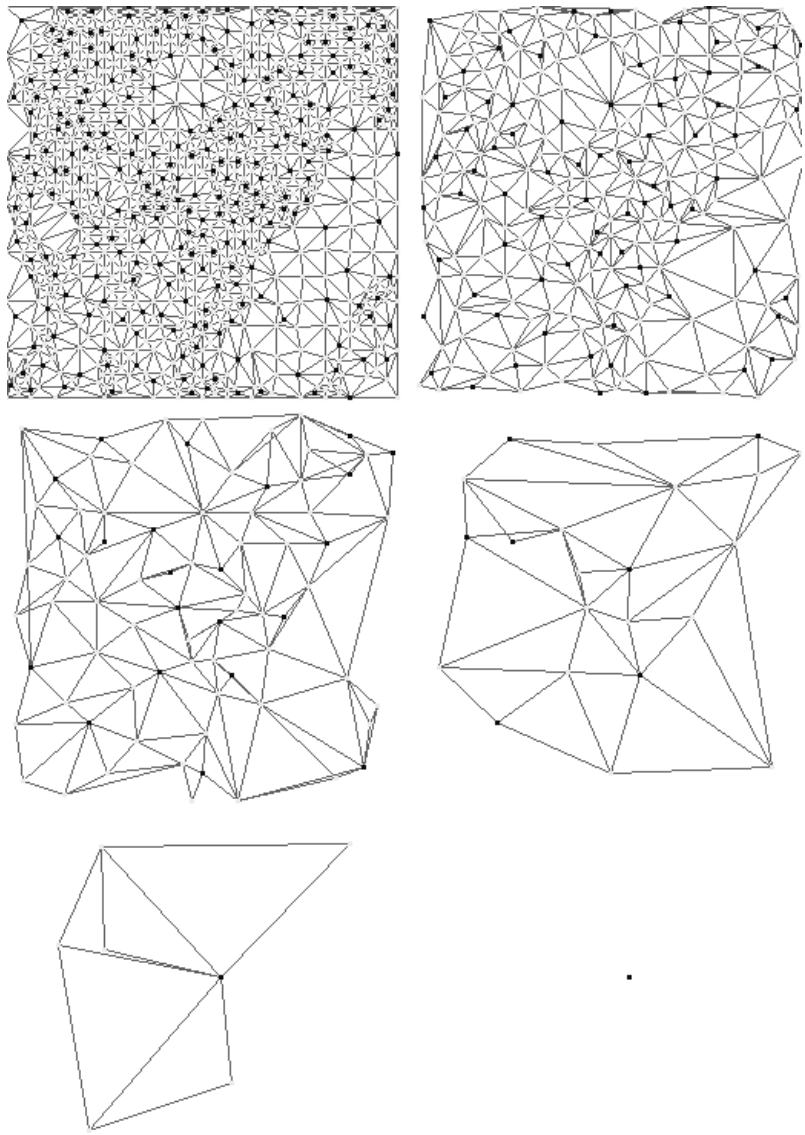


FIG. 62 - Décimation d'un graphe de Delaunay. Les points noirs sont les survivants.

Contrairement au graphe de voisinage, le graphe de similarité n'est pas connexe en général. En effet, au dernier niveau de la pyramide de segmentation, chaque nœud correspond à une composante connexe (i.e. un objet de l'image) et donc le graphe de similarité correspond à un graphe où le nombre de composantes connexes est égal au nombre de nœuds (graphe totalement déconnecté). La contrainte imposée dans A^R montre que, si deux régions i et j ont à peu près le même niveau de gris (i.e. $w_{ij} \leq \Theta$), alors elles auront la possibilité de fusionner ; dans le cas contraire, la dissimilarité des niveaux de gris fait que les deux nœuds ne pourront plus fusionner car ces deux nœuds ne sont plus voisins dans le graphe de similarité.

Si une arête dans le graphe de voisinage lie deux nœuds dont la différence des niveaux de gris est supérieure au seuil (Θ), alors ces deux nœuds ne sont pas voisins dans le graphe de similarité. Ce graphe n'est pas connexe en général, et l'algorithme de Horst Bishof permet de définir un stable sur chaque composante connexe de ce graphe.

Dans ses grandes lignes, l'algorithme pyramidal dans un environnement de Voronoï peut être décrit de la manière suivante :

Algorithme : Pyramide dans un environnement de Voronoï

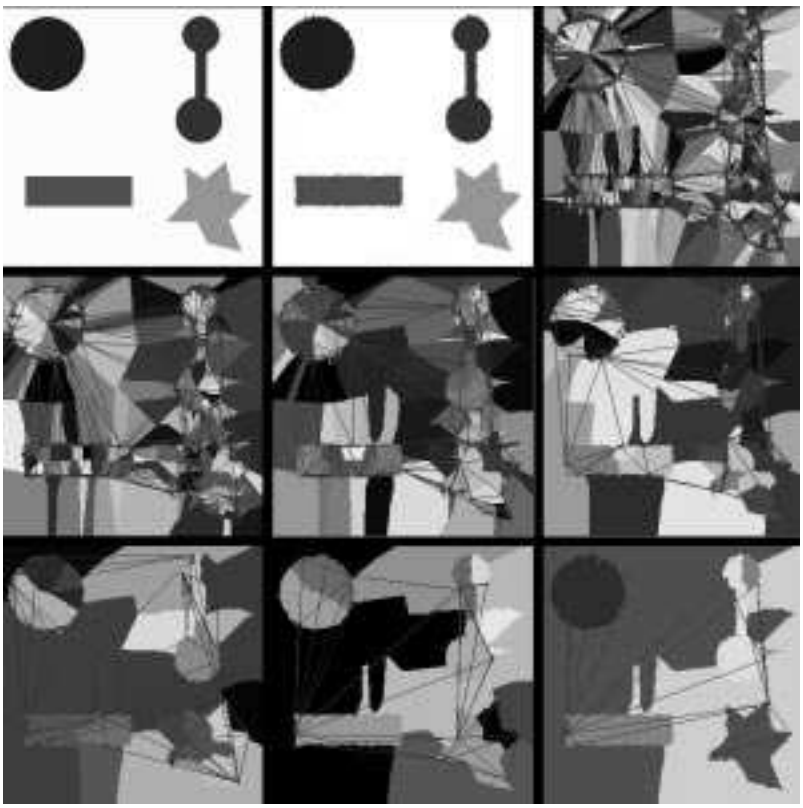
1. Calculer une partition de Voronoï adaptée à l'image avec l'algorithme "split and merge" (Chapitre 3, section 3.3).

2. Construire la pyramide avec la décimation adaptative.

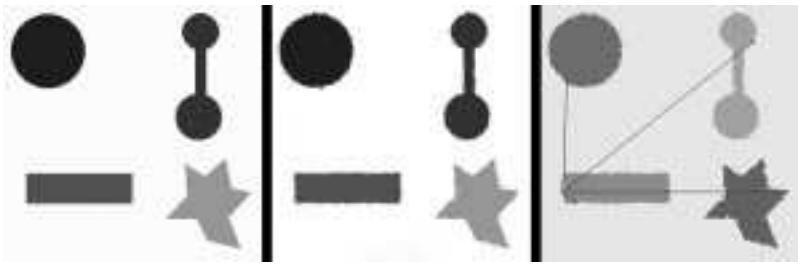
La figure 63 est une illustration de l'algorithme pyramidal dans un environnement de Voronoï contrôlé par un réseau de Hopfield ($\Theta = 7$) (256×256).

La figure 64 est une autre illustration de l'algorithme pyramidal dans un environnement de Voronoï contrôlé par un réseau de Hopfield ($\Theta = 7$) (64×64).

La figure 65 donne un résultat de la segmentation de l'image "femme" en utilisant de fausses couleurs. ($\Theta = 9$)(256×256). Le processus entier (algorithme "split and merge" suivi de l'algorithme de décimation) prend 2 minutes environ sur une Sun 4. Des améliorations peuvent être apportées : par exemple, en ce qui concerne le seuillage local [99].On peut aussi tenir compte de la surface des polygones : beaucoup de polygones de petite surface sont localisés dans des zones à fort gradient, et peu de polygones de grande taille dans des zones à faible gradient (niveaux de gris homogènes).

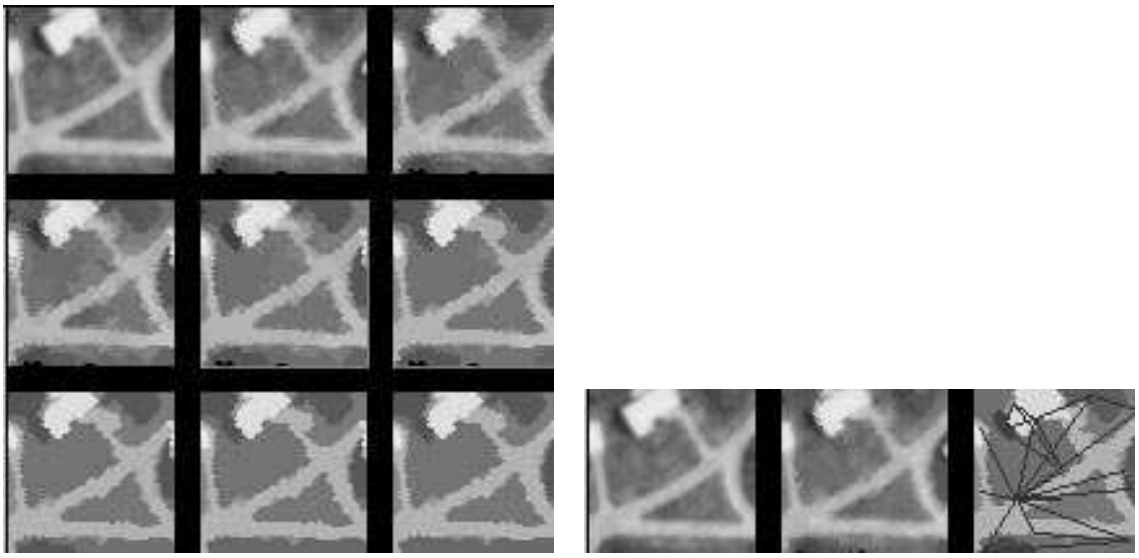


(a) De la gauche vers la droite et du haut vers le bas. Image originale. Diagramme de Voronoï : 1500 polygones. Niveau 1 : 521 régions. Niveau 2 : 225 régions. Niveau 3 : 110 régions. Niveau 4 : 57 régions. Niveau 5 : 33 régions. Niveau 6 : 20 régions. Niveau 7 : 14 régions.



(b) Image originale. Diagramme de Voronoï. Résultat final (niveau 11) : il ne reste plus que 5 régions.

FIG. 63 - Processus pyramidal basé sur les diagrammes de VoronoïΓcontrôlé par les réseaux de Hopfield.



(a) De la gauche vers la droite et du haut vers le bas. Image originale. Diagramme de Voronoï : 1043 polygones. Niveau 1 : 322 régions. Niveau 2 : 122 régions. Niveau 3 : 70 régions. Niveau 4 : 48 régions. Niveau 5 : 38 régions. Niveau 6 : 36 régions. Niveau 7 : 34 régions.

(b) Image Originale. Diagramme de Voronoï. Résultat final (niveau 8) : il ne reste plus que 12 régions.

FIG. 64 - Autre processus pyramidal basé sur les diagrammes de VoronoïΓcontrôlé par les réseaux de Hopfield.

La figure 66 donne un résultat de la segmentation de l'image "Lena". $(\Theta = 7)(256 \times 256)\Gamma$avec approximativement le même temps de calcul que pour l'exemple précédent.

Ce travail contient des perspectives intéressantes. Nous allons brièvement les décire dans la section suivante.

5.5 Perspectives

5.5.1 Décimation adaptative

Dans l'énergie de l'équation (15)Γles poids w_{ij} et les données externes I_i sont des paramètres libres. Ces paramètres peuvent influencer le processus de décimation en

FIG. 65 - Image "femme" et résultat final de la segmentation. Au départ il y a 9033 polygones de Voronoï. Il reste 50 régions.

FIG. 66 - Image "Lena" et résultat final de la segmentation. Au départ il y a 8033 polygones de Voronoï. Il reste 32 régions.

vue de la segmentation des images en niveaux de gris. Les poids w_{ij} expriment une contrainte entre les nœuds i et j.

Si w_{ij} est proche de 2Γ les nœuds i et j pourront se retrouver dans la même région. Au contraireΓsi w_{ij} est proche de 0Γles nœuds i et j ne se retrouveront pas dans la même région. Dans les cas intermédiairesΓle fait qu'un nœud survive ou non dépendra de tout le voisinage.

Il est à noter queΓsi les poids w_{ij} varient continûment entre 0 et 2Γles conditions du théorème 5.1 ne sont plus remplies. Par conséquentΓla règle 1 de Meer peut ne pas être vérifiée à la convergence de l'algorithme de Hopfield. Par contre la règle 2 sera toujours vérifiée. En effetΓsoit p un nœud de V. Si pour tout $q \in \Gamma(p)\Gamma s(q) = 0\Gamma$

alors $I_p + \sum_{\langle q,p\rangle \in A} w_{qp} s(q) = I_p > 0$Γet doncΓd'après la règle de mise à jour que nous nous sommes fixée (section 5.3.2)Γ $s(p)$ doit prendre la valeur 1. Par conséquentΓ même si les conditions du théorème de Bischof sont non remplies Γla règle 2 de Meer reste vérifiéeΓce qui est très important pour garder la localité et donc pour conserver le parallélisme.

Cette relâche dans les contraintes nous permet d'avoir une plus grande latitude quant au choix des poids entre les arêtes en vue de la segmentation.

Par exemple la fonction suivante :

$$f(d_{ij}) = \begin{cases} 2 & \text{si } i \text{ et } j \text{ sont voisins et } d_{ij} \leq \Theta_1 \\ \frac{2(\Theta_2 - d_{ij})}{\Theta_2 - \Theta_1} & \text{si } i \text{ et } j \text{ sont voisins et } \Theta_1 < d_{ij} < \Theta_2 \\ 0 & \text{sinon} \end{cases}$$

devrait permettre d'obtenir une décimation plus adaptée au contenu de l'image.

5.5.2 Coopération région contour

Le partitionnement en régions de Voronoï produit un lissage de l'image et donc provoque une perte d'information dans les contours. Cette perte pourra être supprimée par une coopération de type région contour. Cette notion est déjà présente dans les pyramides [14Γ13]. L'idée est de déterminer dans l'image les contours significatifs qui sont à transmettre et faire évoluer à tous les niveaux de la pyramide. C'est une amélioration qui devrait être importante et complètement adaptée à notre approche.

5.6 Conclusion

Dans ce chapitre nous avons présenté un algorithme pyramidal utilisant les polygones de VoronoïΓ la triangulation de DelaunayΓ et les réseaux de Hopfield. La combinaison de toutes ces notions nous a permis de construire un algorithme de segmentation pour des images de niveaux de gris. L'initialisation de la pyramide par un diagramme de Voronoï adapté à l'image et par son graphe de Delaunay donne la possibilité de réduire le nombre de niveaux contenus dans la pyramide puisque sa base contient un nombre de nœuds réduit. Ceci nous a permis de traiter des images 256×256 et 512×512 rapidement. De nombreuses améliorations sont à apporter comme nous l'avons vu dans la section précédente :

- l'utilisation de fonctions permettant de mieux s'adapter au contenu des imagesΓ

- la coopération région contourΓ
- l'utilisation des informations géométriques des polygones de Voronoï.

Nous pouvons donc dire que la coopération des pyramidesΓdes diagrammes de Voronoï et des réseaux de Hopfield semble extrêmement prometteuse et riche dans le but de segmenter rapidement des images de grande taille.

Chapitre 6

Partition de Voronoï et champs de Markov

Dans ce chapitre nous allons montrer comment nous pouvons lier le contexte markovien au contexte de Voronoï.

Le premier problème auquel nous serons confrontés sera la segmentation des textures. Nous verrons quelques connexions avec le chapitre 5.

L'idée essentielle sous-jacente de ce premier problème est de réduire la taille de l'espace des configurations et de réduire le nombre de relations de voisinage associées au champ de Markov. Le but est donc d'accélérer les algorithmes d'optimisation combinatoire.

Le deuxième problème sera de trouver une partition de Voronoï qui vérifie un certain critère d'optimalité.

Le but auquel nous conduit ce deuxième problème est de trouver une alternative à l'algorithme "split and merge" proposé dans la section 3.3 du chapitre 3.

6.1 Introduction

C'est en 1984 que les champs de Markov font une apparition remarquée dans le domaine de l'analyse d'images [58].

Associés à l'approche bayésienne où le problème de décision est posé en terme de probabilité les champs de Markov fournissent un cadre mathématique cohérent pour l'extraction de primitives à partir d'observations [65].

Dans le domaine de l'analyse d'images les principales applications des champs de Markov sont :

- la classification de textures [60 29 45]
- la détection de contours [19 57 129]
- la restauration et le filtrage [19 58 36]
- la détection du mouvement.

L'idée consiste à trouver pour un système composé d'atomes (ou une image composée de pixels) un état stable correspondant à un **minimum d'énergie**.

L'approche par champs de Markov (Markov Random Field) permet de ne prendre en compte que les **interactions locales** entre les atomes du système (ou les pixels de l'image) en dotant l'ensemble de ses particules d'une topologie.

Les deux notions essentielles pour modéliser une image par un champ de Markov sont donc :

- le graphe de voisinage induisant la notion de clique
- le modèle d'énergie.

L'objectif est alors de trouver le bon modèle d'énergie correspondant au problème d'analyse d'images considéré.

L'équivalence contenue dans le théorème de Hammersley-Clifford entre les champs de Markov et les distributions de Gibbs permet de montrer que l'énergie peut être mise sous la forme d'une somme de potentiels locaux intégrant les relations de voisinage.

L'espace des configurations possibles est en général extrêmement vaste. C'est une des raisons qui nous a motivé pour introduire les modèles markoviens dans les diagrammes de Voronoï.

Nous nous intéresserons donc plus particulièrement ici au graphe de voisinage. L'idée de base est la même que celle trouvée dans le chapitre 5 : coder les images par les informations de haut niveau contenues dans les régions de Voronoï permettant d'avoir des relations de voisinage cohérentes et adaptées aux images obtenues dans le graphe de Delaunay.

Le plan que nous suivrons dans ce chapitre est le suivant : tout d'abord nous donnons les définitions et théorèmes relatifs aux champs de Markov (section 6.2). Puis nous développons quelques algorithmes d'optimisation en vue de minimiser l'énergie contenue dans la distribution de Gibbs : algorithmes déterministes et stochastiques (section 6.3). Dans la section 6.4 nous donnons un exemple d'utilisation des champs de Markov liés au modèle de Voronoï pour la segmentation des textures. Nous expliquons ensuite comment trouver une partition optimale (en un sens qui sera défini) d'une image en région de Voronoï (section 6.5). Finalement nous donnons une conclusion et quelques perspectives (section 6.6).

6.2 Champs markoviens

Nous ne redéfinirons pas ici toutes les notions de base des probabilités. Dans toute la suite nous utiliserons les notations suivantes :

- Soit S un ensemble fini de sites de $\mathbb{R}^2$ ou $\mathbb{R}^3$. En analyse d'images l'ensemble S représente l'ensemble des pixels contenus dans une image. Nous verrons dans la section 6.4 et 6.5 que pour nous ces sites sont associés à une partition de Voronoï.

- Soit Λ l'espace des états qui représente l'ensemble des valeurs prises par un site de S. Nous supposerons que Λ est fini. En général Λ représente les 256 niveaux de gris d'une image ou les différents labels correspondant à des textures par exemple.

- Soit Ω l'ensemble de toutes les configurations de S ($\Omega = \Lambda^S$). Si $x \in \Omega$ alors on peut écrire $x = \{x_s, s \in S\}$.

- Soit $P(\Omega)$ la tribu des parties de Ω.

- Soit P la probabilité définie sur $P(\Omega)$.

L'hypothèse minimale relative à S est sa dénombrabilité [47].

Définition 6.1 (Champ aléatoire) *Le triplet $(\Omega, P(\Omega), P)$ est appelé champ aléatoire ou espace probabilisé.*

Définition 6.2 (Vecteur aléatoire) *$X = (X_s)_{s \in S}$ désignera un vecteur aléatoire si, et seulement si, pour tout $s \in S$, X_s est une variable aléatoire de l'espace $(\Omega, P(\Omega), P)$ à valeur dans Λ.*

Comme dans le chapitre 5ΓS est structuré dans un grapheΓà partir des relations de voisinage.

Définition 6.3 (Système de voisinage) *L'ensemble*

$$V_S = \{V_S(s),\ s \in S,\ et\ V_S(s) \subset S\}$$

définit un système de voisinage sur S si, et seulement si :

- $s \notin V_S(s)$,
- $\forall s, t \in S,\ s \in V_S(t) \Longleftrightarrow t \in V_S(s)$.

Les éléments de $V_S(s)$ sont les voisins de s dans SΓrelativement au système de voisinage considéré. V_S produit donc un grapheΓappelé graphe de voisinage. V_S nous servira à définir la notion très forte de localité dans les champs de Markov.

Comme nous l'avons déjà mentionnéΓles sites s représenteront les germes de Delaunay. Par conséquentΓle graphe précédent sera le graphe de Delaunay et nous aurons $V_S(s) = N_S(s)$Γoù $N_S(s)$ est défini dans le chapitre 2Γsection 2.1.

Nous pouvons maintenant introduire la notion de champ de Markov relativement à un système de voisinage.

Définition 6.4 (Champs de Markov) *Un vecteur aléatoire X est un champ de Markov relativement à V_S si, et seulement si :*

- $\forall x \in \Omega$, *on a :*

$$P(X = x) > 0 \tag{16}$$

- $\forall s \in S$, *on a :*

$$P(X_s = x_s | X_r = x_r, r \neq s) = P(X_s = x_s | X_r = x_r, r \in V_S(s)). \quad (17)$$

Le fait que les champs de Markov sont des processus locaux est contenu dans les relations de voisinage à travers la relation 17. On comprend bien l'intérêt de préserver des relations de voisinage pas trop étendues.

Par la suite une image sera considérée comme la réalisation d'un champ de Markov.

Dans la modélisation mathématique des champs de Markov nous introduisons la notion de clique notion classique de la théorie des graphes.

Définition 6.5 (Clique) *Soit c une partie de S. c est une clique relativement à V_S si, et seulement si, c est un sous-graphe complet du graphe de voisinage. Le cardinal de c s'appelle l'ordre de la clique.*

Il est intéressant de noter qu'en 2D le diagramme de Delaunay est planaire et que par conséquent il n'y a pas de clique d'ordre 5 (Figure 67).

En 3D l'ordre des cliques dans le diagramme de Delaunay n'est pas borné puisqu'il existe des diagrammes de Delaunay complets (chapitre 2 section 2.7.1). Nous avons vu en particulier un exemple de clique d'ordre 5 (figure 17). Dans les cas pratiques l'ordre des cliques sera borné puisque nous avons vu (chapitre 2) que dans de tels cas le nombre de voisins d'un site ne dépend pas de $Card(S)$.

Les cliques vont nous permettre de localiser la notion d'énergie dans les mesures de Gibbs associées à un potentiel. De plus les cliques permettent de relier un champ de Markov à une mesure de Gibbs associée à un potentiel comme nous le verrons dans le théorème de Hammersley-Clifford (théorème 6.1).

Avant d'en arriver à ce théorème il nous reste à introduire la notion de mesure de Gibbs modélisant les systèmes thermodynamiques.

Définition 6.6 (Mesure de Gibbs) *Soit U une énergie définie sur Ω et à valeur dans $\mathbb{R}$. La mesure de Gibbs d'énergie U est la probabilité Π sur Ω définie par :*

$$\forall x \in \Omega,\ \Pi(x) = \frac{1}{Z} \exp(-U(x)) \quad (18)$$

où Z est la constante de normalisation définie par :

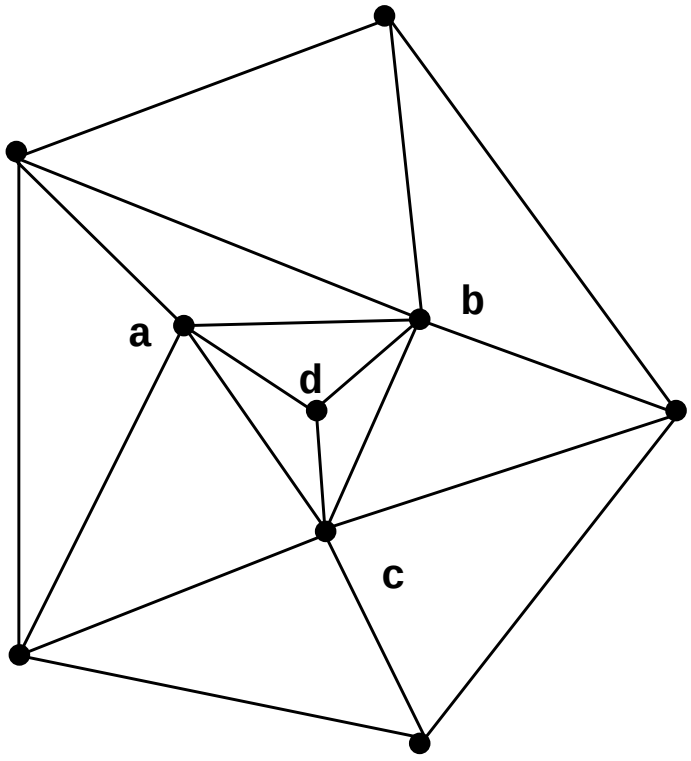


FIG. 67 - Exemple de cliques sur le graphe de Delaunay : $\{a\}\Gamma\{b\}\Gamma\{c\}\Gamma\{d\}$ sont des cliques d'ordre $1\Gamma\{a,b\}\Gamma\{a,c\}\Gamma\{a,d\}\Gamma\{b,c\}\Gamma\{b,d\}\Gamma\{c,d\}$ sont des cliques d'ordre $2\Gamma\{a,b,c\}\Gamma\{a,b,d\}\Gamma\{a,c,d\}\Gamma\{b,c,d\}$ sont des cliques d'ordre $3\Gamma\{a,b,c,d\}$ est une clique d'ordre 4. Il ne peut pas exister des cliques d'ordre ≥ 5 car le graphe de Delaunay est planaire en 2D.

$$Z = \sum_{x \in \Omega} \exp(-U(x)). \tag{19}$$

La fonction d'énergie U précédente peut être **globale** dans le sens où elle peut prendre en compte toutes les interactions entre les points de S.

C'est pourquoi il est intéressant d'examiner le cas où l'énergie s'exprime en ne considérant que des informations **locales** à partir de la notion de clique définie relativement au système de voisinage V_S. Dans un tel casΓon parlera de mesure de Gibbs associée au système de voisinage V_S ou à un potentiel.

Définition 6.7 (Mesure de Gibbs associée à un système de voisinage) *La mesure de Gibbs d'énergie U est dite associée au système de voisinage V_S si, et seulement si :*

$$U(x) = \sum_{c \in C} V_c(x) \tag{20}$$

où C est l'ensemble des cliques relativement à V_S et V_c est une fonction des éléments contenus dans la clique c. $V = \{V_c, c \in C\}$ désigne le potentiel.

L'équation 20 montre que le potentiel V est nul en dehors des cliques. La décomposition de l'énergie U (équation 20)Γen la somme de potentiels V_cΓpermet donc de décomposer l'énergie globale d'une image en une somme de potentiels locaux sur les cliques [109].

A partir du cadre proposé précédemmentΓnous pouvons énoncer le théorème de Hammersley-Clifford liant un champ de Markov à une mesure de Gibbs.

Théorème 6.1 (Hammersley-Clifford) *Soit V_S un système de voisinage. Un vecteur aléatoire X est un champ de Markov relativement à V_S si, et seulement si, $\Pi(x) = P(X = x)$ est une mesure de Gibbs associée à V_S et d'énergie donnée dans l'équation 20.*

La localité des champs de Markov à travers le système de voisinage et l'équation 17 se retrouve donc dans la localité de la mesure de Gibbs associés à un potentiel à travers l'équation 20 qui intègre la notion de clique.

Le théorème 6.1 montre que maximiser la probabilité P (i.e maximiser la probabilité Π)Γ

$$\text{trouver } x \in \Omega,\ \Pi(x) = \max_{x' \in \Omega} \Pi(x'),$$

revient à minimiser l'énergie UΓ

$$\text{trouver } x \in \Omega,\ U(x) = \min_{x' \in \Omega} U(x').$$

Ce théorème permet donc de transformer une approche bayésienne en une approche où le problème est posé en terme énergétique.

Tout revient donc à trouver le minimum de l'énergie U dans l'espace de configuration Ω. En général une telle énergie n'est pas convexeΓet de plus l'espace Ω est extrêmement vaste. Des algorithmes d'optimisation devront donc être mis en oeuvre afin de trouver le minimum global de l'énergie U. C'est ce que nous nous proposons de faire dans la section suivante.

6.3 Algorithmes d'optimisation

Nous n'examinerons pas ici les deux algorithmes déterministes : GNC (Graduated Non Convexity) [19]Γet MFA (Mean Field Annealing) [57] plus adaptés à un certain type d'énergie (modèle de la membrane à continuité lâche) [141].

Nous détaillerons uniquement l'algorithme stochastique (i.e. non déterministe) de recuit simulé [58Γ59]Γet l'algorithme déterministe ICM (Iterated Conditionned Mode) [15]. Notons qu'il existe d'autres algorithmes d'optimisation que nous ne présenterons pas iciΓpar exemple l'algorithme de Creutz qui est présenté dans la thèse de Hérault [68].

6.3.1 Recuit simulé

Cet algorithme a été proposé initialement par Metropolis en 1953 [95]. La démonstration de sa convergence a été fournie par Geman&Geman [58].

Cette approche provient de techniques utilisées en physique statistiqueΓconsistant à trouverΓpour un système donnéΓun état stable correspondant à un minimum d'énergie.

En physiqueΓune bonne façon de trouver les états stables d'un système complexe est de le réchauffer à une température élevée puis de le laisser refroidir lentement.

Dans le cas des métauxΓon porte le système à la température de fusion (état énergétique élevé) avant de le refroidir lentement. On atteint ainsi des états de très faible énergie. C'est cette procédure qui porte le nom de *recuit*.

SiΓau contraireΓle métal est refroidi trop rapidementΓon obtient un état métastableΓqui correspond à un minimum local de l'énergie. Cette procédure porte le nom de *trempe* [68].

Algorithme : Recuit simulé

1. Tirer une configuration initiale aléatoire x dans Ω.
2. Poser $T = T_0$ (température initiale élevée).
3. Pour tous les sites s de S
   (a) Changer la valeur x_s en une valeur x'_s (x_s et x'_s sont dans Λ). Cette opération est appelée transformation élémentaire.

(b) Evaluer la modification induite sur l'énergie par cette transformation élémentaire :

$$\Delta U = U' - U$$

où U' est l'énergie après la transformation élémentaire et U avant une telle transformation.

(c) Si $\Delta U < 0$ accepter la transition (l'énergie décroît).

(d) Si $\Delta U \geq 0$ accepter la transition avec la probabilité

$$\exp(-\frac{\Delta U}{T}).$$

4. Diminuer la température T.

5. Répéter 3-4 jusqu'à la convergence.

On peut démontrer que cet algorithme produit une chaîne de Markov $X^{n,T}$ de mesure invariante Π_T où Π_T est la mesure de Gibbs associée à l'énergie $\frac{U(x)}{T}$ et au système de voisinage V_S :

$$\lim_{n\to\infty} P(X^{n,T} = x) = \Pi_T(x). \tag{21}$$

Par ailleurs l'algorithme converge car on a la proposition suivante :

Proposition 6.1 *Quand $T \to 0$, Π_T converge simplement vers la probabilité uniforme sur les minima de la fonction énergie U. Si U possède k minima globaux $m_1, \ldots, m_k$, alors on a :*

$$\begin{aligned} \lim_{T\to 0} \Pi_T(x) &= 0 \text{ si } x \notin \{m_1, \ldots, m_k\} \\ \lim_{T\to 0} \Pi_T(x) &= \frac{1}{k} \text{ sinon.} \end{aligned}$$

Démonstration : Cette proposition est classique et la démonstration se trouve dans [109]. ■

Cette proposition et l'équation 21 montrent que si on fait tendre simultanément n vers l'infini et T vers 0 alors l'algorithme de recuit simulé converge vers un minimum global de l'énergie.

Il est à noter que l'étape 3 peut être réalisée un certain nombre de fois à température constante (palier de température ou thermalisation).

Geman&Geman ont démontré qu'une décroissance exponentielle de la température (étape 4) après chaque thermalisation est nécessaire pour assurer la convergence de l'algorithme de recuit simulé. SiΓau n^{ieme} palier de températureΓla température notée T_n satisfait la condition suivante :

$$T_n \geq \frac{c}{\log(1+n)}$$

où c est une constante indépendante de nΓalors l'algorithme de recuit simulé converge vers le minimum global de l'énergie quand $n \to \infty$.

Le problème est qu'une telle décroissance de la température est trop coûteuse au niveau des temps de calcul. PratiquementΓon choisit une décroissance linéaire de la température :

- $T_n = 0.93\ T_{n-1}$ si 10% des transformations élémentaires sont acceptées au $(n-1)^{ieme}$ palier.
- $T_n = 0.965\ T_{n-1}$ sinon.

La procédure se termine quand moins de 1% des transformations élémentaires sont acceptées [68].

L'algorithme de recuit simulé est une bonne alternative aux techniques déterministes qui sont rapidement limitées par le trop grand nombre d'informations à prendre en compte. NéanmoinsΓl'algorithme ICM que nous présentons dans la section suivante peut être préféré à l'algorithme de recuit simulé si on a une connaissance a priori sur la solution.

6.3.2 ICM (Iterated Conditionned Mode)

Cet algorithme a été proposé initialement par Besag [15]. Contrairement à l'algorithme de recuit simulé qui contient des tirages aléatoiresΓ cet algorithme est parfaitement déterministe dans le sens où le résultat ne dépend que de l'état initial. Le problème est que la convergence se trouve piégée dans un minimum local. L'algorithme peut être décrit de la manière suivante :

Algorithme : ICM

1. Choisir une configuration initiale x de Ω aussi proche que possible de la configuration optimale.

2. Pour tous les sites s de S :

   (a) Calculer l'ensemble des probabilités conditionnelles déduites de P du site s considéré.

   (b) Sélectionner l'état $\alpha \in \Lambda$) le plus probable de $s(x_s \leftarrow \alpha)$.

3. Répéter 2 pour un nombre déterminé d'itérations.

En pratique, l'algorithme ICM converge après une dizaine d'itérations.

Dans l'étape 2(a), la probabilité conditionnelle peut s'écrire de la manière suivante :

$$P(X_s = x_s | X_r = x_r, r \neq s) = \exp[-\sum_{c \in C, s \in c} V_c(x)] \times \frac{1}{Z_{s,x}} \tag{22}$$

où on a :

$$Z_{s,x} = \sum_{u \in \Lambda} \exp[-\sum_{c \in C, s \in c} V_c(x|_{x_s = u})]$$

où :

$$x|_{x_s=u} \text{ représente la configuration } x \text{ avec } x_s = u.$$

Cette propriété classique des champs de Markov est démontrée dans [109]. On voit bien, à partir de l'équation 22, que l'état le plus probable d'un site est obtenu de manière locale. De plus, pour choisir la probabilité conditionnelle d'état maximum, il suffit de ne considérer que le numérateur. Pour comparer toutes les probabilités quand u varie, il suffit de minimiser le terme $\sum_{c \in C, s \in c} V_c(x)$, et donc de minimiser l'énergie U.

Pratiquement, l'algorithme converge très vite, mais dans un minimum local. En effet, comme dans l'algorithme de Hopfield (chapitre 5, section 5.3), à chaque itération de l'ICM, l'énergie décroît et se trouve donc piégée dans un minimum local. Effectivement, quand on sélectionne l'état le plus probable (étape 2(a)), on obtient :

$$P(X = x|_{x_s = o_p}) \geq P(X = x)$$

donc d'après le théorème 6.1,

$$\Pi(x|_{x_s = o_p}) \geq \Pi(x)$$

et donc $\Delta U \leq 0$Γce qui montre que l'énergie décroît.

La convergence vers un minimum local montre que l'ICM peut être utilisé si on connaît une solution assez proche de l'optimum (étape 1)Γde manière à converger dans un minimum global. Cet état proche de l'optimum dépend de connaissances a priori. L'ICM est donc bien adapté aux processus pyramidaux par exemple (chapitre 5Γsection 5.2) [105]. Quand on descend la pyramideΓchaque niveau peut être initialisé par le niveau précédent.

6.4 Application à la segmentation de textures

Les algorithmes d'optimisation présentés dans la section 6.3 sont souvent très coûteux. Nous proposons donc ici de réduire au préalable la taille du problème (i.e. la taille de Λ^S) en réduisant le nombre de sites de S.

Une solution est d'associer les champs de Markov à une structure pyramidale [105]. Nous les associons ici aux diagrammes de Voronoï. Nous précisons d'abord la notion de voisinage et d'ordre du champ de Markov dans la section 6.4.1. Puis dans la section 6.4.2Γnous définissons le critère de similarité proposé dans [60] et donc l'énergie associée à notre champ de Markov.

6.4.1 Voisinage et ordre

Au lieu de considérer l'ensemble de tous les pixels d'une imageΓnous ne prendrons en compte que les sites associés aux polygones de Voronoï. L'intérêt de l'utilisation des diagrammes de Voronoï est double :

- On obtient moins d'éléments dans S : le nombre de sites de Voronoï est plus faible que le nombre de pixels contenus dans l'image.
- Les connexions sont moins nombreuses mais plus informatives dans le graphe de Delaunay que dans le graphe des quatre ou huit voisins utilisé classiquement.

De plusΓavec un tel graphe il ne se pose plus la question de l'ordre du champ de Markov (i.e. jusqu'à quelle distance on considère un pixel voisin d'un autre). Cette question peut être formulée de la manière suivante : *quel est l'ordre du champ de Markov représenté ou contenu dans l'image étudiée ?* [110].

Nous avons vu dans les chapitres précédents que la structure du graphe de Delaunay permet de décrire les propriétés locales **adaptées** à l'image étudiée. Par conséquentΓnous pensons que la question de l'ordre du champ de Markov dans le graphe de Delaunay n'a plus lieu d'êtreΓce qui est un avantage non négligeableΓ puisque plus l'ordre du champ de Markov augmenteΓplus les temps de calcul sont longs.

Une dernière motivation nous ayant poussé à utiliser les diagrammes de Voronoï est de pouvoir se passer des *configurations interdites* définies dans [60]. Ceci tient au fait que le contenu de chaque région de Voronoï est significatif et adapté aux images.

6.4.2 Segmentation des textures

Notre objectif est d'affecter à chaque région de VoronoïΓle label le plus en conformité avec les données. Comme nous l'avons remarqué dans l'introduction de ce chapitreΓcet objectif sera atteint si on choisit une *bonne* énergie liée à la distribution de Gibbs associée au système de voisinage engendré par le graphe de Delaunay.

Dans le modèle de texturesΓnous n'introduisons pas de connaissance a priori etΓ en ce sensΓla segmentation des textures sera dite *non supervisée.*

Avant de présenter l'énergie correspondant à notre problème de classification des texturesΓnous allons définir quelques notions de base.

En effetΓla segmentation des textures consiste à analyser chaque région de Voronoï et à définir un critère de similarité. Ce critère sera donné par la distance de Kolmogorov-Smirnov.

Définition 6.8 (Distance de Kolmogorov-Smirnov) *Soit $Vor_S(p)$ et $Vor_S(q)$ deux régions de Voronoï. On pose $F_{Vor_S(p)}$, (resp. $F_{Vor_S(q)}$), la fonction de répartition empirique des niveaux de gris des pixels contenus dans $Vor_S(p)$, (resp. $Vor_S(q)$). La distance de Kolmogorov-Smirnov est alors :*

$$d_k(Vor_S(p), Vor_S(q)) = d_\infty(F_{Vor_S(p)}, F_{Vor_S(q)}) = \max_{x\in\{0,\ldots,255\}} |F_{Vor_S(p)}(x) - F_{Vor_S(q)}(x)|$$

Une propriété remarquable de la distance de Kolmogorov-Smirnov est son invariance par toute transformation strictement monotone des données [60].

Nous pouvons alors définir le critère de similarité entre deux régions de Voronoï voisines.

Définition 6.9 (Critère de similarité) *Soit $Vor_S(p)$ et $Vor_S(q)$ deux régions voisines de Voronoï. Ces deux régions sont semblables si, et seulement si :*

$$\Phi(p,q) = -1$$

où Φ est la fonction de disparité entre $Vor_S(p)$ et $Vor_S(q)$ (Définition 6.10).

La fonction de disparité peut s'écrire de la manière suivante :

Définition 6.10 (Fonction de disparité) *La disparité entre deux régions de Voronoï, $Vor_S(p)$ et $Vor_S(q)$, est calculée de la manière suivante :*

$$\Phi(p,q) = 2\delta\{d_k(Vor_S(p), Vor_S(q)) < seuil\} - 1$$

Le seuil est un paramètre de l'algorithme qui sera à régler. Il est important de noter que la relation induite par Φ n'est pas transitive. Sinon l'algorithme de recherche de composantes connexes (chapitre 3 section 3.3.3) sous la contrainte Φ serait suffisant pour la segmentation des textures.

Le modèle d'énergie que nous présentons ici a été proposé dans [60]. Soit x un élément de $\Omega = \Lambda^S$ où Λ représente un ensemble de p étiquettes et S les germes de Delaunay. L'énergie du système peut alors être donnée de la manière suivante :

$$U(x) = \sum_{c \in C} \Psi_c(x)\Phi_c \tag{23}$$

où $\Phi = \{\Phi_c / c \in C\}$ est une famille de disparités sur S généralisée à des cliques d'ordre quelconque et $\Psi = \{\Psi_c / c \in C\}$ une famille de fonctions de contrôles dépendant des labels affectés aux sites des cliques $c \in C$.

Nous nous limiterons ici aux cliques d'ordre 2. L'énergie 23 prend alors la forme suivante :

$$U(x) = \sum_{(s,t) \in DEL(S)} \Psi_{(s,t)}(x)\Phi(s,t). \tag{24}$$

Dans le modèle d'étiquetage qui nous intéresse nous avons :

$$\Psi_{(s,t)}(x) = \delta\{x_s, x_t\}$$

et Φ est la fonction décrite dans la définition 6.10. La variation de l'énergie prend donc l'expression suivante :

$$\Delta U = \sum_{s \in N_S(p)} \Psi_{(s,p)}(x')\Phi(s,p) - \sum_{s \in N_S(p)} \Psi_{(s,p)}(x)\Phi(s,p).$$

Nous pouvons alors décrire brièvement notre algorithme de détection des textures de la manière suivante :

Algorithme : Segmentation des textures

1. Calculer une partition de Voronoï adaptée à l'image avec l'algorithme "split and merge" (Chapitre 3Γsection 3.3).

2. Minimiser l'énergie U (équation 23) avec l'algorithme de recuit simulé (section 6.3.1).

Dans l'algorithme de recuit simulé (section 6.3.1)Γ le changement d'état opéré dans l'étape 3(a) pour un siteΓrevient à changer la valeur de l'étiquette de ce site.

Une amélioration certaine du résultat de la segmentation de textures serait de calculer le partitionnement adaptatif avec un autre critère que la variance dans l'algorithme "split and merge". Ce critère pourrait être une fonction de l'énergie U que nous chercherions à minimiser.

6.5 Application à une partition optimale

6.5.1 Position du problème

Le problème ici est de vouloir optimiser le partitionnement d'une image en régions de Voronoï. L'optimisation est liée à la minimisation d'une énergie U associée à la *position* des germes dans l'image.

Nous avons donc un ensemble fini S de N germes. L'espace des étatsΓnoté ΛΓestΓ pour un germe p donnéΓtoutes les positions possibles de p sur une grille discrète I_D ($N << Card(I_D)$). L'ensemble Ω de toutes les configurations de S est donc toutes les positions possibles des N germes de S sur I_DΓdeux germes ne pouvant occuper la même position. Nous avons donc :

$$Card(\Omega) = \begin{pmatrix} Card(I_D) \\ N \end{pmatrix}.$$

L'énergie que nous voulons minimiser est la suivante :

$$U = \sum_{p \in S} \sigma(Vor_S(p)). \tag{25}$$

Nous avons choisi une telle énergie car dans l'algorithme "split and merge" présenté chapitre 3Γsection 3.3.1Γla division des régions de Voronoï est faite selon un critère d'écart type. Nous n'assurions alors nullement la minimisation de ce critère.

Une telle minimisation peut se faire avec un algorithme de type recuit simulé. Avant de présenter des solutions pratiquesΓnous allons nous attacher à résoudre quelques problèmes théoriques.

6.5.2 Problèmes théoriques

Le problèmeΓiciΓest queΓsi nous considérons tout l'espace ΩΓnous ne sommes plus dans un contexte markovien. En effetΓil existe deux configurations possible ω_1 et ω_2 telles que $DEL(\omega_1)$ et $DEL(\omega_2)$ ne soient pas topologiquement équivalents. Nous ne pouvons donc pas garantir la stabilité topologique du champ de Markov puisqu'elle est liée d'après la définition 6.4 de la section 6.2Γaux relations de voisinage et donc au graphe de Delaunay.

Une solution théorique est de forcer le système à rester dans un contexte markovien en maintenant les relations de voisinage et donc en conservant la stabilité topologique du graphe de Delaunay. Ceci nécessite de restreindre l'espace Ω en considérant la proposition suivante :

Proposition 6.2 (Stabilité topologique) *Soit S un ensemble de points et $p \in S$. Soit q un point n'appartenant pas à S.*

$$N_S(p) = N_{(S-p)\cup q}(q)$$

si, et seulement si,

$$q \in [(\cup B_S(N_S(p)))^c] \cap [\cap B_{S-p}(N_S(p))]$$

où :

- *$B_S(N_S(p))$ désigne l'intérieur d'un cercle circonscrit à un triangle de Delaunay de $DEL(S)$ défini par deux points de $N_S(p)$ et un point de $S - (N_S(p) \cup p)$ et $(\cup B_S(N_S(p)))^c$ la surface complémentaire de $(\cup B_S(N_S(p)))$ dans $PE(p)$ (Figure 68 c),*
- *$B_{S-p}(N_S(p))$ désigne l'intérieur d'un cercle circonscrit à un triangle de Delaunay de $DEL(S-p)$ défini par trois points de $N_S(p)$ (Figure 68 b).*

Démonstration : Pour que q soit voisin de tous les points de $N_S(p)$Γil est nécessaire que $q \in \cap B_{S-p}(N_S(p))$ d'après la propriété du cercle vide (définition 2.5Γchapitre 2Γ section 2.2). Nous allons monter que $q \in (\cup B_S(N_S(p)))^c$ par l'absurde. S'il existe un cercle de Delaunay contenant p et n'appartenant pas à $\cup B_S(N_S(p))$Γalors il existe au moins un point x de $S - (N_S(p) \cup p)$ qui est voisin de q. Or toutes les arêtes frontières de $PE(p)$ sont dans $DEL((S-p) \cup q)$ puisque $q \in (\cup B_S(N_S(p)))^c$. Pour relier q à x il faut donc couper une arête frontières de $PE(p)$ car x est à l'extérieur de $PE(p)$. Ce qui est en contradiction avec le fait que la triangulation de Delaunay est planaire. ■

La figure 68 tirée de [85] est une illustration de la proposition précédente.

La proposition 6.2 montre qu'il est possible de rester avec le même champ de Markov durant l'évolution du système que nous cherchons à rendre optimal.

6.5.3 Solutions pratiques

Nous allons présenter ici une solution au problème posé en restreignant l'espace des configurarations Ω à l'espace Ω'Γoù Ω' est l'espace de toutes les configurations possibles ω' des N germes sur I_D garantissant la stabilité topologique du graphe de Delaunay. Cette solution permet de garder un modèle de Markov.

La variation de l'énergie s'écrit comme suit en supposant que le site p est remplacé par q dans le domaine de stabilité topologique donné par la proposition 6.2 :

$$\Delta U = \sum_{x \in N_{(S-p)\cup\{q\}}(q)\cup\{q\}} (Vor_{(S-p)\cup\{q\}}(x)) - \sum_{x \in N_S(p)\cup\{p\}} (Vor_S(x)).$$

Il nous suffit alors d'utiliser l'algorithme de recuit simulé proposé section 6.3.1 pour minimiser l'énergie U de l'équation 25.

ToutefoisΓun problème théorique important se pose. Nous ne pouvons garantir de pouvoir atteindre toutes les configurations de Ω'Γà partir d'une configuration $\omega' \in \Omega'$Γle changement d'état étant fait de manière séquentielle. Par exempleΓsi nous prenons un système carréΓnous ne pourrons jamais atteindre une configuration rectangulaireΓpuisque le domaine de stabilité topologique d'un germe situé sur une grille carrée est réduit à l'ensemble vide.

Une alternative est alors de minimiser U dans l'espace Ω et donc dans un contexte markovien. Malgré toutΓl'écriture de la variation d'énergie ΔU est encore locale. En

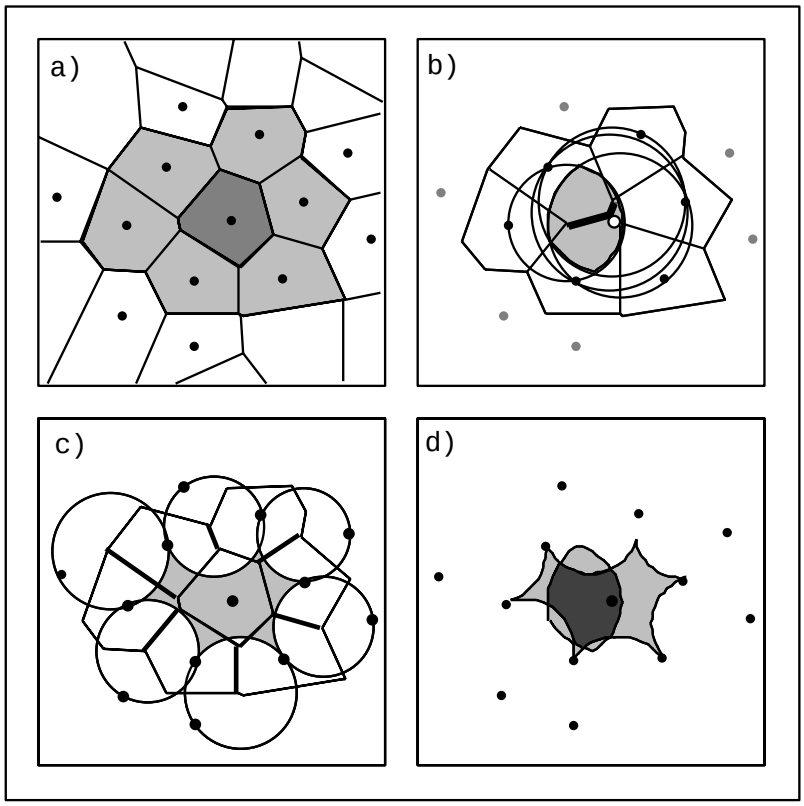


FIG. 68 - Recherche de la surface de stabilité topologique d'un germe au sein du graphe de Delaunay. Quelle que soit la position du germe à l'intérieur de cette surface Γ les relations de voisinage restent inchangées. a- Diagramme de Voronoï. b- La surface grisée est la surface de stabilité des relations de premier voisinage du point considéré. c- La surface grisée est la surface de stabilité du reste du diagramme. d- La surface en gris foncé représente la surface de stabilité du point considéré.

effet Γ nous restons dans un contexte de graphe de Delaunay. L'expression de la variation de l'énergie est plus complexe que précédemment car il faut tenir compte du changement possible de topologie :

$$
\begin{aligned}
U' &= \sum_{x \in N_S(p) \cup \{p\}} \sigma(Vor_S(x)) + \sum_{x \in N_{(S-p)\cup\{q\}}(q)} \sigma(Vor_S(x)). \\
U &= \sum_{x \in N_S(p)} \sigma(Vor_{(S-p)\cup\{q\}}(x)) + \sum_{x \in N_{(S-p)\cup\{q\}}(q)\cup\{q\}} \sigma(Vor_{(S-p)\cup\{q\}}(x)). \\
\Delta U &= U' - U.
\end{aligned}
$$

Expérimentalement, l'algorithme de recuit simulé dans un contexte non markovien semble converger.

6.6 Conclusion

Nous avons montré dans la section 6.4 comment, sur les structures conjointes des graphes de Voronoï et de Delaunay, on peut associer un modèle de Markov. Cette approche est de plus en plus de plus en plus exploitée et permet d'associer l'approche markovienne à une structure de données plus souple que les grilles rigides classiques.

Une perspective possible pour améliorer l'algorithme de segmentation des textures serait de réaliser l'algorithme "split and merge" (Chapitre 3, section 3.3.1) avec un critère plus adapté à l'application considérée.

Une autre possibilité serait d'introduire des cliques d'ordre 3 dans le modèle énergétique. On pourrait aussi introduire des transformations strictement croissantes sur la distance de Kolmogorov-Smirnov [60], ou bien tester des critères plus fins que cette distance.

L'algorithme de classification pourrait être intégré dans un processus pyramidal comme dans le chapitre 5. Dans ce type de pyramide nous perdons la notion de noeuds survivants et non-survivants, mais nous gardons la notion de champ récepteur. Pour nous, un champ récepteur sera simplement, à un niveau donné de la pyramide, l'ensemble des composantes connexes ayant la même étiquette.

Dans la section 6.5, nous avons proposé un algorithme pour obtenir une partition optimale par régions de Voronoï. Si nous utilisons la stabilité topologique du graphe de Delaunay, nous restons dans un cadre markovien et la convergence du système est assurée. Cette approche pourrait constituer une nouvelle façon de concevoir l'algorithme "split and merge" décrit dans la section 3.3 du chapitre 3.

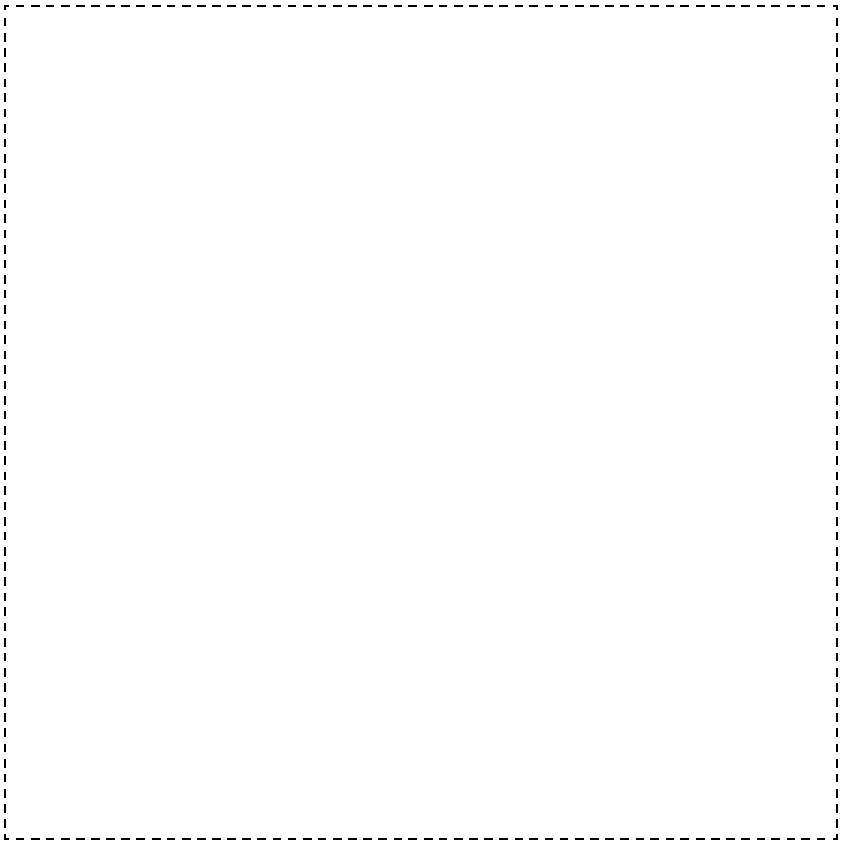
Toutefois, si nous ne respectons pas la stabilité topologique du graphe de Delaunay, le système semble converger dans un recuit simulé. Nous pouvons donc nous demander s'il est bien utile de rester dans un contexte markovien. En effet, si nous pouvions nous en passer, nous pourrions non seulement minimiser une énergie U, mais aussi le nombre de régions de Voronoï, en nous autorisant des suppressions et des insertions de germes dans la structure de Voronoï au cours de l'optimisation du système par l'algorithme de recuit simulé.

Quoi qu'il en soit, l'approche markovienne demeure un outil puissant pour traiter

des problèmes liés à l'analyse d'images. C'est pourquoi les techniques qui en découlent sont de plus en plus développéesΓdans l'objectif de trouver une alternative aux approches déterministes parfois impuissantes devant certains problèmes.

Chapitre 7

Conclusion



"Quand je pense que cela ne fait que trois ans, jour pour jour, qu'un cube d'apparence anodine a été déposé sur cette table. Tant de choses se sont passées depuis, je ne sais pas ce que l'avenir nous réserve."

"La Fièvre d'Urbicande" (Schuiten et Peeters).

7.1 Conclusion

Dans ce mémoire de thèse nous avons voulu montrer comment nous pouvions exploiter des outils issus de la géométrie algorithmique au profit de l'analyse d'images.

Dans le chapitre 2Γ nous avons présenté de manière approfondie le diagramme de Voronoï ponctuel 3D. De l'étude des différents algorithmes de construction du diagramme de Voronoï ponctuel 3DΓil est ressorti que l'*approche incrémentale* était la mieux adaptée par son côté *dynamique* aux problèmes d'analyse d'images que nous nous sommes proposé d'étudier. De plusΓil est apparu que dans la plupart des casΓl'approche incrémentale était *optimale*Γavantage non négligeable puisque nous avions à traiter des problèmes de grande taille. Nous avons aussi présenté un algorithme de remise à jour par *suppression* de germesΓalgorithme qui nous a été utile dans le chapitre 6Γconsacré aux champs de Markov.

Le chapitre 3 a été consacré à l'étude de différents modèles géométriques de partitionnement de volumes de données dans un but de représentation. Nous avons mis l'accent sur les partitionnements *adaptatifs*Γet plus particulièrement sur les partitionnements en régions de Voronoï. L'adaptation des régions de Voronoï aux données a été réalisée dans un environnement *"split and merge"*. Nous avons constaté que les partitionnements en polyèdres de Voronoï et en tétraèdres de Delaunay sont comparables au niveau du nombre d'éléments de volume composant la partition. Nous avons aussi constaté que les partitionnements en octrees demandent moins de temps de calcul mais plus d'éléments cubiques composant la partition et une structure moins riche.

Dans le chapitre 4 nous avons montré comment on pouvait lier les diagrammes de Voronoï généralisés à un ensemble de polyèdres aux *squelettes, exosquelettes et squelettes par zones d'influence*. Nous avons donné un *théorème de caractérisation* des équations des séparateurs. Nous avons ensuite proposé un algorithme pour calculer une approximation du diagramme de Voronoï 3D. L'idée était de procéder à une discrétisation des objets polyédriquesΓce qui nous donne un ensemble S de points 3D. Il suffit ensuite de calculer le diagramme de Voronoï ponctuel des points de S et de supprimer les arêtes dites inutiles. La validité de cet algorithme est prouvée dans un *théorème de convergence*.

Le but du chapitre 5 était d'exposer une exploitation possible en *segmentation* de la représentation des images en régions de Voronoï. Nous avons donc lié l'algorithme

"split and merge" à un algorithme *pyramidal* contrôlé par un *réseau de Hopfield* dans le but de fusionner les régions de Voronoï. L'avantage de cette méthode est de pouvoir traiter des images de grande tailleΓde réduire le nombre de niveaux dans la pyramide et de diminuer les temps de calcul.

Nous avons envisagé quelques perspectives dans le chapitre 6. Nous avons voulu lier la théorie des *champs de Markov* aux diagrammes de Voronoï et de Delaunay dans deux buts bien distincts. Le premier était de segmenter des images texturées. L'idée consistait à avoir un graphe adapté aux données et des régions avec un très bon contenu informatif. L'avantage est qu'il suffit de prendre des cliques d'ordre 1 pour caractériser le champ de Markov. Le deuxième but était de définir un algorithme permettant de minimiser la somme des écarts types de chaque polygone associé à une image donnée. Les résultats obtenus sont décevants. En effetΓil nous semble queΓpour la segmentation des images texturéesΓaucun problème d'ordre théorique ne peut expliquer la mauvaise qualité des résultats.

7.2 Perspectives

De nombreuses perspectives sont à envisager de près. Nous en avons donné quelques-unes au cours de ce mémoire de thèse.

On peut sans doute retenir la coopération pyramide-diagramme de Voronoï contrôlée par un réseau de Hopfield avec des *fonctions continues*. Celle-ci permettra de rendre plus souple la décimation dans la pyramide au cours du processus. La coopération région-contour sera sans doute aussi un élément déterminant pour l'amélioration de la qualité des résultats.

Pour ce qui est des champs de MarkovΓtous les algorithmes sont implémentés. Pour les images texturés il est nécessaire de trouver une énergie plus fine pour obtenir de meilleurs résultats. Nous croyons fermement que c'est une voie à exploiter et qui doit donner de bons résultats. Pour ce qui est du partitionnement optimal du plan il nous apparaît que des problèmes d'ordre théorique sont encore à résoudre. Une question est de savoir si on doit rester dans un contexte markovien ou non pour obtenir un meilleur partitionnement. Nous sommes encore loin d'avoir apporté une solution satisfaisante à ce problème d'optimisationΓmais nous pensons que son intérêt est tel qu'il mérite qu'on cherche d'autres solutions théoriques et pratiques.

7.3 Autres travaux

D'autres applications n'ont pas été abordées dans ce mémoire. Elles ont été réalisées en collaboration avec d'autres personnes et nous allons très brièvement les exposer ici.

Parmi celles-ci figure la compression d'images par fractals qui repose sur la théorie fractale des LIFS (Local Iterated Function System) [75]. L'idée est d'utiliser un partitionnement de l'image en triangles de Delaunay pour diminuer le nombre de transformations affines utiles au codage de l'image. Les résultats obtenus sont d'ores et déjà convaincants et constituent une nouvelle démonstration de l'utilité de partitionnements adaptés aux imagesΓmoins rigides que des partitionnements de type quadtree [44Γ37].

Nous avons aussi participé à une recherche relative au problème de la reconstruction d'une surface à partir d'un modèle numérique de terrain (M.N.T.) irrégulier. L'idée est de calculer la triangulation de Delaunay des points du M.N.T. projetés dans le plan d'équation $\{z = 0\}$. Une fois la triangulation calculéeΓla surface est obtenue en remontant les triangles dans l'espace tout en respectant l'altitude des points du M.N.T. La méthode est justifiée par l'article [111] qui montre que cette manière de procéder lisse la surface en minimisant la semi-norme de Sobolev. A partir de cette surface ainsi triangulée nous avons construit un algorithme qui calcule les courbes de niveau. Nous appliquons ensuite un lissage par des fonctions splines d'interpolationΓpour obtenir une meilleure qualité visuelle. De nombreux travaux sont encore à réaliser pour améliorer les temps de calcul et avoir ainsi un logiciel opérationnel [6].

Des travaux auxquels nous avons encore participé ont été réalisés en 2D et en 3D sur la sociologie cellulaire [85]. Il apparaît que la conjugaison du diagramme de Voronoï et du graphe de Delaunay permet de connaître de manière assez précise et robuste le désordre interne à une population cellulaireΓchaque germe étant le centre de gravité d'un noyau cellulaire. Cette connaissance peut être obtenue aussi bien de manière globale que de manière locale. Des perspectives relatives à ce travail se trouvent du côté de la modélisation dans un contexte de recuit simulé. L'idée serait de modéliser le passage d'un tissu sain à un tissu cancéreuxΓen trouvant une énergie adéquate [86].

De plusΓnous avons réalisé une étude à propos de la percolation en 2D et en 3D

sur le graphe de Delaunay. Les germes sont distribués uniformément dans un carré en 2D et dans un cube en 3D. L'idée consiste à supposer que chaque germe possède une marque à 0 ou à 1. Le but est alors de pouvoir passer d'une arête donnée à une arête opposée du carréΓou d'une face donnée à une face opposée du cube par les arêtes de Delaunay reliant deux germes ayant une marque à 1. Nous avons obtenu de manière statistique un processus de percolation. Pour avoir un système qui percole en 2DΓil faut que 50% des germes aient une marque à 1Γalors qu'en 3D 21% suffisent. Une interprétation biologique d'un tel résultat devrait être possible.

Nous espérons que d'autres applications suivront pour que les diagrammes de Voronoï et de Delaunay deviennent un jour des outils incontournables dans de multiples disciplines dont l'analyse d'images.

Bibliographie

[1] N. AhujaΓB. AnΓand B. Schachter. Image representation using Voronoi tessellation. *Computer Vision Graphics and Image processing*Γ29:286–295Γ1985.

[2] E. Andres. Le plan discret. In *Proc. du colloque de géométrie discrète en imagerie*Γpages 45–61ΓSeptembre 1993.

[3] C. Arcelli and G. Sanniti di Baja. A with-independent fast thinning algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*Γ7(4):463–474Γ1985.

[4] D. Attali and A. Montanvert. Semi-continuous skeletons of 2d and 3d shapes. In L.P. CordellaΓG. Saniti di Baja C. ArcelliΓeditorΓ*Visual Form, Analysis and Recognition*Γpage SoumisΓNew YorkΓ1994. Plenum.

[5] F. Aurenhammer. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Surveys*Γ33(3):345–405Γ1991.

[6] F. BenoitΓE. BertinΓM. MémierΓand J.-M. Chassery. Génération de courbes de niveau à partir de données contraintes. In *Proc. du 9ème Congrès RFIA, AFCET-INRIA ed., Paris*Γpage In pressΓJanvier 1994.

[7] E. Bertin. Diagramme de Voronoï 3D. Rapport de DEA de mathématiquesΓ Université Joseph FourierΓGrenobleΓJuin 1990.

[8] E. Bertin and H. Bischof. Voronoi pyramids controlled by hopfield neural networks. Technical Report PRIP-TR-24ΓPRIPΓTU WienΓ1993.

[9] E. Bertin and J.-M. Chassery. Diagramme de Voronoï 3D: Construction et applications en imagerie 3D. In *Proc. du 8ème Congrès RFIA, AFCET-INRIA ed., Lyon-Villeurbanne*Γpages 803–808ΓNovember 1991.

[10] E. Bertin and J.-M. Chassery. 3D generalized Voronoi diagram. In *Proc. of the 2nd Congrès Curves and Surfaces, Chamonix-Mont-Blanc*Γ June 1993.

[11] E. BertinΓR. MarcelpoilΓand J.-M. Chassery. Morphological algorithms based on Voronoi and Delaunay graphs: Microscopic and medical applications. In *Proc. of the Image Algebra and Morphological, Image Processing III, SPIE ed., San Diego*Γpages 356–367Γ July 1992.

[12] E. BertinΓF. ParazzaΓand J.-M. Chassery. Segmentation and measurement based on Voronoi diagram: Application to confocal microscopy. *Computerized Medical Imaging and Graphics*Γ17(3):175–182Γ1993.

[13] P. Bertolino. Structure pyramidale irrégulière pour la segmentation d'images en niveaux de gris. Mémoire C.N.A.M.ΓCUEFAΓGrenobleΓDécembre 1992.

[14] P. Bertolino and A. Montanvert. Edge detection for biomedical image: a self-adaptive and randomized operator. In *Proc. of the 14th Annual International Conference of the IEEE Engineering in Medecine and Biology Society, IEEE Comp. Soc. Press, Paris*Γpages 129–133Γ November 1992.

[15] J. Besag. On the statistical analysis of dirty pictures. *Journal Roy. Stat., Serie B*Γ48:259–302Γ1986.

[16] H. Bischof and W.G. Kropatsch. Hopfield networks for irregular decimation. Accepted at 17th ÖAGMΓGrazΓ1993.

[17] H. Bischof and W.G. Kropatsch. Neural Networks versus Image Pyramids. Technical Report PRIP-TR-7ΓDept. for Pattern Recognition and Image ProcessingΓTU WienΓ1993.

[18] M. BisterΓJ. CornelisΓand A. Rosenfeld. A critical view of pyramid segmentation algorithms. *Pattern Recognition Letters*Γ Vol. 11(No. 9):pp. 605–617Γ September 1990.

[19] A. Blake and A. Zisserman. *Visual Reconstruction.* MIT PressΓ Cambridge MAΓ1987.

[20] I. Bloch-BoulangerΓF. SchmittΓand H. Maître. Calcul de l'enveloppe convexe d'un objet tridimensionnel. In *Journées de géométrie algorithmique, INRIA Sophia Antipolis*Γpages 791–800Γ Juin 1990.

[21] H. Blum. A transformation for extracting new descriptors of shape. In *Proc of the symp. on models for perception of speech and visual form. Boston, MIT press*Γpages 362–380Γ1964.

[22] J.-D. Boissonnat. Representation of objects by triangulating points in 3D space. In *Proc. of the ICPR 82, IEEE, Silver Springs, Munich*Γ pages 830–832Γ1982.

[23] J.-D. Boissonnat. Shape reconstruction from planar cross-sections. Technical Report 546ΓINRIAΓSophia AntipolisΓ1986.

[24] J.-D. Boissonnat and M. Devillers-Teillaud. On the randomized construction of the Delaunay tree. Technical Report 1140ΓINRIAΓSophia AntipolisΓ1989.

[25] J.-D. BoissonnatΓ O. D. FaugerasΓ and E. Le Bras-Mehlman. Representing stereo data with the Delaunay triangulation. Technical Report 788ΓINRIAΓ Sophia AntipolisΓ1988.

[26] J.-D. Boissonnat and B. Geiger. Three dimensional reconstruction of complex shapes based on the Delaunay triangulation. Technical Report 1697ΓINRIAΓ Sophia AntipolisΓ1992.

[27] J.-D. Boissonnat and M. Yvinec. *Géométrie Algorithmique*. A paraîtreΓ1994.

[28] G. Borgefors. Distance transforms in arbitrary dimensions. *Computer Vision, Graphics and Image Processing*Γ27:321–345Γ1984.

[29] C. Bouman and B. Liu. Multiple resolution of textured images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*Γ13(2)Γ1991.

[30] A. Bowyer. Computing Dirichlet tessellation. *Computer Journal*Γ24:162–166Γ 1981.

[31] J. W. Brandt and V. R. Algazi. Continuous skeleton computation by Voronoi diagram. *CVGIP, Image Understanding*Γ55:329–338Γ1992.

[32] W. BrostowΓJ. P. DussaultΓand B. L. Fox. Construction of Voronoï polyedra. *Journal of Computational Physics*Γ29:81–92Γ1978.

[33] K.Q. Brown. Voronoi diagrams from convex hulls. *Inf. Process. Lett.*Γ9:223–228Γ1979.

[34] E. BruzzoneΓG. GaribottoΓand F. Mangili. Three-dimensional surface reconstruction using delaunay triangulation in the image plane. In C. ArcelliΓeditorΓ *Visual Form*Γpages 99–108ΓNew YorkΓ1992. Plenum.

[35] C.E. Buckley. Split-and-merge image segmentation based on Delaunay triangulation. In *Lectures notes in computer science, 333, Internationnal Workshop Computationnal Geometry, Warzburg*ΓMarch 1988.

[36] B. Chalmond. Image restoration using an estimated markov model. *Signal Processing*Γ15:115–129Γ1988.

[37] J.-M. ChasseryΓF. DavoineΓand E. Bertin. Compression fractale par partitionnement de Delaunay. In *Proc. of the 14ème Colloque GRETSI, Juan-les-Pins*Γ volume 2Γpages 819–922ΓSeptember 1993.

[38] J.-M. Chassery and M. Melkemi. Diagramme de Voronoï appliqué la segmentation d'images et à la détection d'événements en imagerie multi-source. *Traitement du Signal*Γ8:155–164Γ1991.

[39] J.-M. Chassery and A. Montanvert. *Géométrie Discrète en Analyse d'Images.* Edition HermèsΓParisΓ1991.

[40] X. Chen and F. Schmitt. Split-and-merge image segmentation based on Delaunay triangulation. In *Proc. of the 7th Scandinavian Conf. on Image Analysis, Aalborg, Denmark*Γpages 910–917ΓAugust 1991.

[41] M. Coster and J.-L. Chermant. *Précis d'analyse d'images.* Editions du CNRSΓ 1985.

[42] E.E. David and C.W. David. Voronoi polyhedra as a tool for studying solvation. *J. Chem. Phys.*Γ76(9):4611–4614Γ1982.

[43] F. DavoineΓE. BertinΓand J.-M. Chassery. Fractal image coding based on Delaunay tessellation. In *Proc. of the 17th OAGM - Meeting, Graz, Austria*Γ June 1993.

[44] F. Davoine, E. Bertin, and J.-M. Chassery. From rigidity to adaptive tesselations for fractal image compression: comparative studies. In *Proc. of the 8ème Workshop on Image Multidimensional Signal Processing, IEEE Signal Processing Society, Cannes*, pages 56–57, September 1993.

[45] H. Derin and H. Helliot. Modeling and segmentation of noisy and textured images using gibbs random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):39–55, 1987.

[46] O. Devillers. Randomization yields simple $O(nlogn)$ algorithms for difficult $W(n)$ problems. *International Journal of Computational Geometry and Applications*, 2:97–111, 1992.

[47] R.L. Dobrusin. The description of a random field by means of conditional probabilities and conditions of its regularity. *Theory of Probability and its applications*, 13(2):197–224, 1968.

[48] R.A. Dwyer. Average-case analysis of algorithms for convex hulls and Voronoi diagrams. Technical Report CMU-CS-88-132, Departement of Computer Science, Carnegie Mellon University, 1988.

[49] H. Edelsbrunner. *Algorithms in combinatorial geometry.* Texts and Monographs in Computer Science, Springer Verlag, NewYork, 1988.

[50] H. Edelsbrunner and W. Shi. An $o(n \log^2 h)$ time algorithm for the three-dimensionnal convex hull problem. *Siam J. Comput.*, 20(2):259–269, 1979.

[51] M. Elbaz. *Sur les diagrammes de Voronoï et de Delaunay dans le plan et dans l'espace.* PhD thesis, Université de Haute Alsace, 1992.

[52] D.A. Field. Implementing watson's algorithm in three dimensions. In *Proc. of the 2nd ACM Symposium in Computational Geometry*, pages 246–259, 1986.

[53] J.L. Finney. A procedure for the construction of Voronoi polyhedra. *Journal of computational physics*, 32:137–143, 1979.

[54] Y. Fischer, E.W. Jacobs, and R.D. Boss. Fractal image compression using iterated transform. In J.A. Storer, editor, *Image and Text Compression*, pages 35–61, Boston, Dordrecht, London, 1992. Kluwer Academic Publishers.

[55] A. Gagalowictz and O. Monga. Un algorithme de segmentation hiérarchique. In *Proc. du 5ème Congrés RFIA, AFCET-INRIA ed., Grenoble*Γpages 163–177ΓNovembre 1985.

[56] A.E. Galashev and V.P. Skripov. Stability of Lennars-Jones crystal structures in the molecular dynamics model. *Sov. J. Low Temp. Phys.*Γ6Γ1985.

[57] D. Geiger and F. Girosi. Parallel and deterministic algorithms from mrf's: Surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*Γ12(7)Γ1990.

[58] S. Geman and D. Geman. Stochastic relaxationΓgibbs distributionsΓand the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*Γ6(6):721–741Γ1984.

[59] S. Geman and D. Geman. Relaxation and annealing with constraints. Technical reportΓDivision Appl. Math.ΓBrown UniversityΓ1992.

[60] S. GemanΓD. GemanΓC. GraffigneΓand P. Dong. Boundary detection and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*Γ12(7):609–628Γ1990.

[61] I.G. GowdaΓD.G. KirkpatrickΓD. Tsai LeeΓand A. Naamad. Dynamic Voronoi diagrams. *IEEE Transactions on Information Theory*Γ29(5):724–731Γ1983.

[62] P. J. Green and R. Sibson. Computing Dirichlet tessellation in the plane. *Computer Journal*Γ21:168–173Γ1978.

[63] W.I. Grosky and R. Jain. A pyramid based approach to segmentation applied to region matching. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*Γ8(5):639 –650Γ1986.

[64] L.J. Guibas and J. Stolfi. Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. *ACM Transaction on Graphics*Γ4:74–123Γ1985.

[65] X. Guyon. *Champs aléatoire sur un réseau.* MassonΓParisΓ1992.

[66] R. M. Haralick and L.G. Shapiro. Glossary of computer vision terms. *Pattern Recognition*Γ24:pp.69–93Γ1991.

[67] R. Hecht-Nielsen. *Neurocomputing.* Addisson-Wesley editor,Reading,Massachusets,1990.

[68] L. Herault. *Reseaux de neurones recursifs pour l'optimisation combinatoire. Application à la theorie des graphes et à la vision par ordinateur.* PhD thesis, INPG,Grenoble,1991.

[69] H. Honda. Description of cellular patterns by Dirichlet domains: the two dimensionnal case. *J. Theor. Biol.*,75:523–543,1978.

[70] J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *PNAS*,79:2554–2558,1982.

[71] J.J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *PNAS*,82:3088–3092,1984.

[72] J.J. Hopfield and D.W. Tank. Neural computation of decisions in optimization problems. *Biological Cybernetics*,52:141–152,1985.

[73] S.L. Horowitz and Pavlidis T. Picture segmentation by directed split-and-merge procedure. In *Proc. of the 2. ICPR*,pages 424 – 433,1974.

[74] H.T. Hu. *Diagramme de Voronoï Généralisé pour un ensemble de Polygones.* PhD thesis,Université Joseph Fourier,Grenoble,1991.

[75] A.E. Jacquin. Image coding based on a fractal theory of iterated contractive image transformations. *IEEE transactions on image processing*,1(1):18–30,1992.

[76] J.-M. Jolion and A. Montanvert. The adaptive pyramid,a framework for 2D image analysis. *Computer Vision, Graphics, and Image Processing: Image Processing*,Vol. 55(No. 3):pp.339–348,May 1992.

[77] B. Kamgar-Parsi. Hopfield model and optimization problems. In Wechsler H.,editor,*Neural Networks for Perception*,volume 2,pages 94 – 110. Academic Press,Inc.,1992.

[78] R. Klein. Concrete and abstract Voronoi diagrams. *Lecture Notes in Computer Science, Springer Verlag ed.*,1989.

[79] T. Kohonen. The self-organizing map. *Proc. of the IEEE*Γ78(9):1464–1480Γ 1990.

[80] W.G. Kropatsch and A. Montanvert. Irregular pyramids. Technical Report PRIP-TR-5Γ Dept. f. Pattern Recognition and Image processingΓ TU WienΓ 1992.

[81] W.G. KropatschΓC. ReitherΓD. WillersinnΓand G. Wlaschitz. The dual irregular pyramid. In D. Chetverikov and W. K. KropatschΓeditorsΓ*Proc. of the 5th International Conference CAIP'93, Lecture Notes in Computer Science 719*Γpages 31–40ΓBudapestΓHungaryΓSeptember 1993. Springer-Verlag.

[82] C. LacoteΓJ. MailfertΓand J.P. Uhry. Parallel annealing by partitioning of configurations: an application to optimal 3D triangulation. In Robert AzencottΓeditorΓ*Simulated Annealing, Parallelisation Techniques*Γpages 187–218Γ 1992.

[83] D.T. Lee. Concrete and abstract Voronoi diagrams. *IEEE Trans. PAMI*Γ 4(4):363–369Γ1982.

[84] M.E. Lewis. La segmentation d'images avec la morphologie mathématique et les diagrammes de Voronoï. RapportΓ Université Joseph FourierΓ GrenobleΓ Juin 1991.

[85] R. Marcelpoil. *Méthodologie pour l'étude de la sociologie cellulaire : application à l'étude du tissu prostatique normal et pathologique.* PhD thesisΓUniversité Joseph FourierΓ1993.

[86] R. Marcelpoil and E. Bertin. Cellules en société. *Science et Vie*Γ184:68–74Γ 1993.

[87] R. Marcelpoil and Y. Usson. Methods for the study of cellular sociology: Voronoi diagrams and parametrization of the spatial relationships. *J. Theor. Biol.*Γ154:359–369Γ1992.

[88] P. Martin. *Réseaux de Neurones Artificiels : Application à la Reconnaissance Optique de Partitions Musicales.* PhD thesisΓUniversité Joseph FourierΓ1992.

[89] A. Mauss. Delaunay triangulation and the convex hull of n points in expected linear time. *BIT*Γ24Γ1983.

[90] R.J. McEliece Γ E.C. Posner Γ and E.R. Rodemich. The capacity of the hopfield associative memory. *IEEE Trans. on Information Theory*Γ33:461–482Γ1987.

[91] P. McMullen. The maximum numbers of faces of a convex polytope. *Mathematika*Γ17Γ1970.

[92] N.N. Medvedev and Yu. I. Naberukhin. Delaunay simplexes of a simple liquid and amorphous substances. *Sov. Phys. Dokl.*Γ31(6):465–466Γ1985.

[93] P. Meer. Stochastic image pyramids. *Computer Vision, Graphics, and Image Processing*ΓVol. 45(No. 3):pp.269–294ΓMarch 1989.

[94] M. Melkemi. *Approches géométriques par modèles de Voronoï en segmentation d'images.* PhD thesisΓUniversité Joseph FourierΓGrenobleΓ1992.

[95] N. MetropolisΓA.W. RosenbluthΓM.N. RosenbluthΓA.H. TellerΓand E. Teller. Equations of the state calculations by fast computing machines. *J. Chem. Phys.*Γ21:1087–1091Γ1953.

[96] D. Milhet. Visualisation des polyèdres de Voronoï. Rapport de stage de 3ème annéeΓEcole Normale Supérieure des TélécommunicationsΓParisΓJuin 1992.

[97] A. Montanvert. Obtention d'une ligne médiane par connexion de l'axe médian. In *Proc. du 5ème Congrés RFIA, AFCET-INRIA ed., Grenoble*Γpages 777–785Γ1985.

[98] A. Montanvert. Graph environment from medial axis for shape manipulation. In *proc. of the 4th ICPR, IEEE Comp. Soc. Press, Celafu (Italie)*Γpages 197–203ΓSeptember 1987.

[99] A. Montanvert and P. Bertolino. Irregular pyramids for parallel image segmentation. In H. Bischof and W.G. KropatschΓeditorsΓ*Pattern Recognition 1992*Γvolume OCG-Schriftenreihe 62Γpages 13–34. OldenbourgΓ1992.

[100] P. Moreau. Partition de l'espace et tétraédrisation de Delaunay sur machines parallèles. Rapport de DEA de mathématiques appliquéesΓUniversité Joseph FourierΓGrenobleΓSeptembre 1991.

[101] OhyaΓIriΓand Muroto. A fast Voronoi diagram algorithm with quaternary tree bucketing. *Information processing letters*Γ18Γ1984.

[102] O. Palacio-Velez and B. Cuevas-Renaud. A dynamic hierarchical subdivision algorithm for computing triangulations and others closest-point problems. *ACM Transaction on Mathematical Software*Γ16Γ1990.

[103] F. ParazzaΓC. HumbertΓand Y. Usson. Methods for 3D topographical analysis of intra-nuclear fluorescence distribution. *Computerized Medical Imaging and Graphics*Γ17(3)Γ1993.

[104] J.B. Pawley. *Handbook of Biological Confocal Microscopy.* Plenum PressΓ NewYorkΓ1990.

[105] P. Perez and F. Heitz. Restriction d'un champ markovien sur un graphe. Application à l'analyse d'images multirésolution. Technical Report 713ΓINRIAΓ RennesΓ1993.

[106] I. Pitas and A.N. Venetsanopoulos. Morphological shape decomposition. *IEEE transaction on Pattern Analysis and Machine Intelligence*Γ12(1):38–45Γ1990.

[107] F.P. Preparata and M.I.S. Shamos. *Computational Geometry, an Introduction.* Springer VerlagΓNewYorkΓ1988.

[108] K. Preston and R. Siderits. New techniques for three-dimensional data analysis in histopatology. *Analytical and Quantitative Cytology and Histology*Γ 14(5):398–406Γ1992.

[109] F. Préteux. L'approche markovienne en analyse d'images et en reconnaissance de formes. Cours de troisième annéeΓTELECOM ParisΓDépartement ImageΓ ParisΓAvril 1991.

[110] A. Rida. Etude d'une méthode de partitionnement d'images par l'approche markovienne. Rapport de DEA de mathématiques appliquéesΓUniversité Joseph FourierΓGrenobleΓJuin 1993.

[111] S. Rippa. Minimal roughness property of the Delaunay triangulation. *Computer Geometric Design*Γ7:489–497Γ1990.

[112] F. Rolland, A. Montanvert, and J.-M. Chassery. 3D medial axis and 3D skeletons. In *Proc. of the 7th Scandinavian Conf. on Image Analysis, Aalborg, Denmark*, pages 395–402, August 1991.

[113] H. Rom and S. Peleg. Image representation using Voronoi tessellation: adaptive and secure. In *Proc. of Computer Vision and Pattern Recognition, Ann Arbon (Michigan)*, pages 282–285, June 1988.

[114] A. Rosenfeld. Arc colorings, partial path groups, and parallel graph contractions. Technical Report TR-1524, University of Maryland, Computer Science Center, July 1985.

[115] A. Rosenfeld and A.C Kak. *Digital image processing.* Academic Press, New York, 1982.

[116] J.-C. Roux. Approximation des surfaces et triangulation sphérique. In *Proc. du XXIVème congrés national d'analyse numérique, Vittel*, Mai 1992.

[117] M. Sakarovitch. *Optimisation combinatoire, méthodes mathématiques et algorithmiques.* Hermann, Enseignement des sciences, Paris, 1984.

[118] H. Samet. Region representation: quadtrees from binary arrays. *CVGIP*, 13:88–93, 1980.

[119] H. Samet. *The design and analysis of spatial data structures.* Addison Wesley, 1990.

[120] F. Schmitt and H. Borouchaki. Algorithme rapide de maillage de Delaunay dans Rd. In *Proc. des journées de géométrie algorithmique, INRIA Sophia-Antipolis*, pages 131–133. Springer, Juin 1990.

[121] M. Schmitt. Some examples of algorithms analysis in computational geometry by means of mathematical morphology techniques. In J.D. Boissonnat and J.P. Laumond, editors, *Lecture Note in Computer Science, Geometry and Robotics*, pages 225–246, Berlin, September 1989. Springer Verlag.

[122] J. Serra. *Image Analysis and Mathematical Morphology.* Academic Press, London, 1982.

[123] J. Serra. *Image Analysis and Mathematical Morphology, Part II: Theoretical Advances.* Academic Press, London, 1988.

[124] Y. Shrivastava, S. Dasgupta, and S.M. Reddy. Guaranteed convergence in a class of hopfield networks. *IEEE Trans. on Neural Networks*, 3(6):951–961, 1992.

[125] J.C. Spehner. Merging in maps and in pavings. *Theoretical Computer Science*, 86:205–232, 1991.

[126] D.F. Stevens. *Patterns in Nature.* Little Brown, Boston, 1978.

[127] K. Sugihara and M. Iri. Construction of the Voronoi diagram for one million generators in single-precision arithmetic. In *Proc. of the First Canadian Conference on Computational Geometry, Montréal*, pages 1–31, August 1989.

[128] J.M. Talon. *Génération et amélioration de maillage pour les éléments finis en deux et trois dimensions.* PhD thesis, Université Joseph Fourier, Grenoble, 1989.

[129] H.L. Tan, S.B. Gelfand, and E.J. Delp. A cost minimization approach to edge detection using simulating annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1):3–18, 1991.

[130] M. Tanemura, T. Ogawa, and N. Ogita. A new algorithm for three-dimensional Voronoi tessellation. *Journal of Computational Physics*, 51:191–207, 1983.

[131] S.L. Tanimoto. Paradigms for pyramid machine algorithms. In S. Levialdi and V. Cantoni, editors, *Pyramidal Systems for Image Processing and Computer Vision*, volume F25 of *NATO ASI Series*, pages 173–194. Springer-Verlag Berlin, Heidelberg, 1986.

[132] M. Teillaud. *Vers des algorithmes dynamiques randomisés en géométrie algorithmique.* PhD thesis, Université de Paris XI, Orsay, 1992.

[133] E. Thiel and A. Montanvert. Etude et amélioration des distances du chanfrein pour l'analyse d'images. *TSI*, 1992.

[134] J.C. Tilton. Image segmentation by iterative parallel region growing with applications to data compression and image analysis. In *Frontiers of Massively Parallel Computation*Γpages 357 – 361Γ1988.

[135] G.T. Toussaint. Pattern recognition and geometrical complexity. In *Proc. of the 5th ICPR, IEEE, Miami Beach*Γpages 1324–1347Γ1989.

[136] M. Tuceryan and A.K Jain. Texture segmentation using Voronoi polygons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*Γ12:211–216Γ 1990.

[137] M.Q. Vahidi-asl. First-passage percolation on the Voronoi tesselation and the Delaunay triangulation. In PoepplΓeditorΓ*Mustererkennung 1993*ΓLübeckΓ September 1993. Springer.

[138] L. Vincent. Mathematical morphology on graphs. *Signal Processing*Γ 16(4):365–388Γ1989.

[139] L. Vincent. *Algorithmes morphologiques à base de files d'attente et de lacets. Extension aux graphes.* PhD thesisΓEcole Normale Supérieure des mines de ParisΓ1990.

[140] D.F. Watson. Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes. *Computer Journal*Γ24:167–172Γ1981.

[141] J. Zerubia and F. Ployette. Détection de contours et lissage par deux algorithmes déterministes de relaxation: mise en oeuvre sur la machine à connexions CM2. *Traitement du Signal*Γ8(3):165–175Γ1991.

[142] S.W. Zucker. Region growing: childhood and adolescence. *Computer Graphics and Image Processing*Γ5:382 – 399Γ1976.