



HAL
open science

Étude de paramètres géométriques à partir du code de Freeman

Xavier Trouillot

► **To cite this version:**

Xavier Trouillot. Étude de paramètres géométriques à partir du code de Freeman. Traitement du signal et de l'image [eess.SP]. Ecole Nationale Supérieure des Mines de Saint-Etienne, 2008. Français. NNT : . tel-00496290

HAL Id: tel-00496290

<https://theses.hal.science/tel-00496290v1>

Submitted on 30 Jun 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre : 507 IVS

THÈSE

présentée par

Xavier TROUILLOT

pour obtenir le grade de
Docteur de l'université Jean Monnet de Saint-Étienne
et Docteur de l'École Nationale Supérieure des Mines de Saint-Étienne

Spécialité : Image Vision Signal

Étude de paramètres géométriques à partir du code de Freeman

soutenue à Saint-Étienne, le 12 décembre 2008

Membres du jury

Président :	Pierre GOUTON	Professeur, Université de Bourgogne, Dijon
Rapporteurs :	Pierre BONTON Étienne DECENCIÈRE	Professeur, Université Blaise Pascal, Clermont-Ferrand Maître de recherche, École des Mines de Paris
Examineur(s) :	Jean-Marie BECKER	Enseignant-Chercheur, CPE Lyon
Directeurs de thèse :	Michel JOURLIN Jean-Charles PINOLI	Professeur, Université Jean-Monnet, Saint-Étienne Professeur, ENM-SE

Spécialités doctorales :

SCIENCES ET GENIE DES MATERIAUX
 MECANIQUE ET INGENIERIE
 GENIE DES PROCEDES
 SCIENCES DE LA TERRE
 SCIENCES ET GENIE DE L'ENVIRONNEMENT
 MATHEMATIQUES APPLIQUEES
 INFORMATIQUE
 IMAGE, VISION, SIGNAL
 GENIE INDUSTRIEL
 MICROELECTRONIQUE

Responsables :

J. DRIVER Directeur de recherche – Centre SMS
 A. VAUTRIN Professeur – Centre SMS
 G. THOMAS Professeur – Centre SPIN
 B. GUY Maître de recherche – Centre SPIN
 J. BOURGOIS Professeur – Centre SITE
 E. TOUBOUL Ingénieur – Centre G2I
 O. BOISSIER Professeur – Centre G2I
 JC. PINOLI Professeur – Centre CIS
 P. BURLAT Professeur – Centre G2I
 Ph. COLLOT Professeur – Centre CMP

Enseignants-chercheurs et chercheurs autorisés à diriger des thèses de doctorat (titulaires d'un doctorat d'État ou d'une HDR)

AVRIL	Stéphane	MA	Mécanique & Ingénierie	CIS
BATTON-HUBERT	Mireille	MA	Sciences & Génie de l'Environnement	SITE
BENABEN	Patrick	PR 2	Sciences & Génie des Matériaux	CMP
BERNACHE-ASSOLANT	Didier	PR 1	Génie des Procédés	CIS
BIGOT	Jean-Pierre	MR	Génie des Procédés	SPIN
BILAL	Essaïd	DR	Sciences de la Terre	SPIN
BOISSIER	Olivier	PR 2	Informatique	G2I
BOUCHER	Xavier	MA	Génie Industriel	G2I
BOUDAREL	Marie-Reine	MA	Sciences de l'inform. & com.	DF
BOURGOIS	Jacques	PR 1	Sciences & Génie de l'Environnement	SITE
BRODHAG	Christian	MR	Sciences & Génie de l'Environnement	SITE
BURLAT	Patrick	PR 2	Génie industriel	G2I
CARRARO	Laurent	PR 1	Mathématiques Appliquées	G2I
COLLOT	Philippe	PR 1	Microélectronique	CMP
COURNIL	Michel	PR 1	Génie des Procédés	SPIN
DAUZERE-PERES	Stéphane	PR 1	Génie industriel	CMP
DARRIEULAT	Michel	ICM	Sciences & Génie des Matériaux	SMS
DECHOMETS	Roland	PR 1	Sciences & Génie de l'Environnement	SITE
DESRAYAUD	Christophe	MA	Mécanique & Ingénierie	SMS
DELAFOSSÉ	David	PR 1	Sciences & Génie des Matériaux	SMS
DOLGUI	Alexandre	PR 1	Génie Industriel	G2I
DRAPIER	Sylvain	PR 2	Mécanique & Ingénierie	SMS
DRIVER	Julian	DR	Sciences & Génie des Matériaux	SMS
FOREST	Bernard	PR 1	Sciences & Génie des Matériaux	CIS
FORMISYN	Pascal	PR 1	Sciences & Génie de l'Environnement	SITE
FORTUNIER	Roland	PR 1	Sciences & Génie des Matériaux	CMP
FRACZKIEWICZ	Anna	MR	Sciences & Génie des Matériaux	SMS
GARCIA	Daniel	CR	Génie des Procédés	SPIN
GIRARDOT	Jean-Jacques	MR	Informatique	G2I
GOEURIOT	Dominique	MR	Sciences & Génie des Matériaux	SMS
GOEURIOT	Patrice	MR	Sciences & Génie des Matériaux	SMS
GRAILLOT	Didier	DR	Sciences & Génie de l'Environnement	SITE
GROSSEAU	Philippe	MR	Génie des Procédés	SPIN
GRUY	Frédéric	MR	Génie des Procédés	SPIN
GUILHOT	Bernard	DR	Génie des Procédés	CIS
GUY	Bernard	MR	Sciences de la Terre	SPIN
GUYONNET	René	DR	Génie des Procédés	SPIN
HERRI	Jean-Michel	PR 2	Génie des Procédés	SPIN
KLÖCKER	Helmut	MR	Sciences & Génie des Matériaux	SMS
LAFOREST	Valérie	CR	Sciences & Génie de l'Environnement	SITE
LI	Jean-Michel	EC (CCI MP)	Microélectronique	CMP
LONDICHE	Henry	MR	Sciences & Génie de l'Environnement	SITE
MOLIMARD	Jérôme	MA	Sciences & Génie des Matériaux	SMS
MONTHEILLET	Frank	DR 1 CNRS	Sciences & Génie des Matériaux	SMS
PERIER-CAMBY	Laurent	PR1	Génie des Procédés	SPIN
PIJOLAT	Christophe	PR 1	Génie des Procédés	SPIN
PIJOLAT	Michèle	PR 1	Génie des Procédés	SPIN
PINOLI	Jean-Charles	PR 1	Image, Vision, Signal	CIS
STOLARZ	Jacques	CR	Sciences & Génie des Matériaux	SMS
SZAFNICKI	Konrad	CR	Sciences de la Terre	SITE
THOMAS	Gérard	PR 1	Génie des Procédés	SPIN
VALDIVIESO	François	MA	Sciences & Génie des Matériaux	SMS
VAUTRIN	Alain	PR 1	Mécanique & Ingénierie	SMS
VIRICELLE	Jean-Paul	MR	Génie des procédés	SPIN
WOLSKI	Krzysztof	CR	Sciences & Génie des Matériaux	SMS
XIE	Xiaolan	PR 1	Génie industriel	CIS

Glossaire :

PR 1	Professeur 1 ^{ère} catégorie
PR 2	Professeur 2 ^{ème} catégorie
MA(MDC)	Maître assistant
DR (DR1)	Directeur de recherche
Ing.	Ingénieur
MR(DR2)	Maître de recherche
CR	Chargé de recherche
EC	Enseignant-chercheur
ICM	Ingénieur en chef des mines

Centres :

SMS	Sciences des Matériaux et des Structures
SPIN	Sciences des Processus Industriels et Naturels
SITE	Sciences Information et Technologies pour l'Environnement
G2I	Génie Industriel et Informatique
CMP	Centre de Microélectronique de Provence
CIS	Centre Ingénierie et Santé

à mon épouse et mes enfants

Table des matières

Résumé	ix
Introduction	xi
1 Les différents codages de formes	1
1.1 Quelques codes de contour	2
1.1.1 La tortue de Papert	2
1.1.2 Le code de Bribiesca	3
1.1.2.1 En trois dimensions	3
1.1.2.2 En deux dimensions	5
1.1.3 Le Vertex Chain Code	6
1.1.4 Le code de Liu et Zalic	7
1.1.5 Le code de Zeng et Vasseur	9
1.1.6 Synthèse	9
1.2 Autres codes	10
1.2.1 Run-Length Encoding	10
1.2.2 Le code de Huffman	10
1.2.3 Neighbourhood Code	11
1.3 Le code de Freeman	12
1.3.1 Description et implémentation	12
1.3.1.1 Description	12
1.3.1.2 Implémentation	16
1.3.1.3 Remarques :	16
1.3.2 Utilisations du code de Freeman	20
1.3.2.1 Reconnaissance de caractères chinois	20
1.3.2.2 Correction de caractères arabes manuscrits	20
1.3.2.3 Simplification de polygones	21
1.3.2.4 Reconnaissance de profils de visages humains	22
1.4 Conclusion	23
2 Estimation de paramètres	25
2.1 Paramètres et opérations sur une chaîne	26
2.1.1 Inverse d'une chaîne	26
2.1.2 Longueur d'une chaîne	26
2.1.3 Largeur et hauteur d'une chaîne	26
2.1.4 Intégration par rapport à l'axe des x	27
2.1.5 Moments par rapport à l'axe des abscisses	28
2.1.6 Résidus	28

2.1.6.1	Résidu d'une chaîne	28
2.1.6.2	Résidu d'une paire de codants	29
2.1.7	Distance entre deux points	29
2.1.8	Miroir d'une chaîne	31
2.2	Paramètres géométriques	31
2.2.1	Périmètre	31
2.2.1.1	Méthode Directe	31
2.2.1.2	Périmètre de Crofton	32
2.2.2	Aire	33
2.2.2.1	Utilisation du produit vectoriel	33
2.2.2.2	Méthode de Ballard et Brown	33
2.2.2.3	Amélioration de la méthode de Ballard et Brown	34
2.2.2.4	Correctif pour obtenir l'aire totale	35
2.2.2.5	Autre méthode	36
2.2.3	Centre de masse	38
2.2.4	Rayon maximum, minimum et moyen	38
2.2.5	Diamètre apparent maximal, minimal et moyen	38
2.2.6	Rectangle englobant d'aire minimale	40
2.2.6.1	Méthode de H. Freeman	40
2.2.6.2	Nouvelle Méthode	41
2.2.7	Dimension fractale	42
2.2.7.1	Historique	42
2.2.7.2	A propos de dimension	44
2.2.7.3	Applications	48
2.2.7.4	Conclusion	52
2.3	Paramètres de formes	52
2.3.1	Paramètres de circularité	52
2.3.1.1	Le plus classique	52
2.3.1.2	Paramètre radial	53
2.3.1.3	Utilisation des disques inscrit et circonscrit	53
2.3.2	Paramètres de symétrie	54
2.3.2.1	Coefficient d'asymétrie de Besicovitch	54
2.3.2.2	Coefficient de symétrie de Blaschke	56
2.3.2.3	Coefficient de symétrie de Minkowski	57
2.3.2.4	Coefficient de symétrie de Winternitz	58
2.3.3	Convexité	59
2.3.4	Histogramme d'un code	60
2.3.5	Axes principaux d'inertie	60
2.3.5.1	Par un calcul de valeurs propres	61
2.3.5.2	Par un calcul de moments surfaciques	61
2.4	Conclusion	68
3	Transformations, comparaisons	69
3.1	Opérations sur un objet à partir du codage de Freeman	70
3.1.1	Translation	70
3.1.2	Homothéties	70
3.1.2.1	Expansion	70

3.1.2.2	Contraction	74
3.1.3	Rotation	75
3.1.4	Symétries	75
3.1.4.1	Par rapport à un point	75
3.1.4.2	Par rapport à un axe	76
3.1.5	Appartenance d'un point à un objet	76
3.1.5.1	Étude angulaire	77
3.1.5.2	Géométrie algorithmique	77
3.2	Opération sur deux objets à partir du codage de Freeman	79
3.2.1	Estimation de la distance géométrique entre formes	79
3.2.1.1	Distance d'un point à un objet	81
3.2.1.2	Distance entre deux objets	81
3.2.2	Intersection et union de deux formes	81
3.2.2.1	Intersection de deux formes	81
3.2.2.2	Union de deux formes	86
3.2.3	Comparaison de deux formes	87
3.2.3.1	Rappel sur les métriques	87
3.2.3.2	Différence symétrique	88
3.2.3.3	Distance de Hausdorff	89
3.2.3.4	Distance de Asplünd	89
3.2.4	Corrélation entre deux formes	90
3.3	Conclusion	91
4	Code de Freeman et morphologie mathématique	93
4.1	Morphologie mathématique binaire	94
4.1.1	Addition de Minkowski et dilatation	94
4.1.2	Erosion, ouverture et fermeture	95
4.2	Morphologie mathématique et code de Freeman	97
4.2.1	Première approche de la dilatation	97
4.2.2	Nouvelle approche de la dilatation	99
4.2.2.1	Dilatation par un voisinage V4	99
4.2.2.2	Dilatation par un voisinage V8	101
4.2.2.3	Problèmes liés à cette méthode	103
4.2.3	Approches de l'érosion	105
4.2.3.1	Dilatation du complémentaire	105
4.2.3.2	Intersection de translatés	105
4.2.3.3	Erodé ultime	106
4.3	Conclusion	107
5	Perspectives	109
5.1	Etude sur différents objets types	110
5.2	Code de Freeman en niveaux de gris	110
5.3	Complexité des algorithmes	110
5.4	Etude des propriétés des autres codages	110
5.4.1	Etude du code de Bribiesca	110
5.4.2	Le code de Zeng et Vasseur	111
5.5	Descripteurs topologiques	111

5.6	Floating bodies	111
5.7	Courbure	111
5.8	Aspect multi-échelle	111
5.9	Morphologie mathématique binaire	112
5.10	Supercodage, polygonalisation	112
5.11	Correction de code après rotation	112
5.12	3D	112
5.13	Passage bijectif des différents codes au code de Freeman	112
5.14	Diamètres apparents	112
5.15	Conclusion sur les perspectives	113
6	Conclusion	115
A	Notations	117
B	Union et intersection	119
	Bibliographie	145

Résumé

Cette thèse s'inscrit dans le cadre de la géométrie discrète 2D avec pour principales applications sont l'analyse et la caractérisation de formes.

Nous nous intéressons ici aux différents codages de contour de formes binaires que nous présentons dans un premier temps. Nous présentons ensuite plus en détail le plus ancien d'entre eux : le codage de Freeman, et nous développons plus particulièrement des algorithmes sur ce qu'il est possible de faire à partir de ce code.

Nous étudions donc l'estimation de paramètres géométriques et de paramètres de formes sur une forme binaire comme le périmètre, l'aire, les diamètres apparents, la dimension fractale, et les coefficients de symétrie d'une forme.

Nous voyons ensuite les transformations qu'il est possible d'effectuer sur le code de Freeman sans revenir à la représentation classique de l'image.

Enfin, nous abordons la notion de morphologie mathématique en proposant une méthode d'obtention du code du dilaté et de l'érodé d'une forme connue par son code de Freeman.

Mots clefs : géométrie discrète, codage de contour, Freeman, paramètres géométriques, morphologie mathématique, analyse de formes, compression d'images, objet binaire.

Introduction

L'analyse d'image est une science récente nécessitant l'obtention d'images à travers une chaîne d'acquisition et des outils mathématiques de traitement adaptés. De par sa nature, un ordinateur manipule des données discrètes, nous obligeant à discrétiser les images et à les manipuler à travers les pixels.

La géométrie discrète propose ce contexte mathématique en respectant au mieux les propriétés de la géométrie continue même si le passage du continu au discret demeure délicat pour certaines notions. Son but est d'étudier les formes discrètes formées d'un ensemble fini de pixels, c'est-à-dire les images stockées et exploitables numériquement via un ordinateur.

Dans de nombreux cas, pour extraire une information pertinente de ces images, il est intéressant d'isoler les formes contenues dans l'image par une segmentation. Ces formes sont définies comme des régions ayant des caractéristiques de relative homogénéité, par exemple au sens des niveaux de gris. Une labellisation permet alors d'extraire un objet et de l'étudier.

Par la suite, certains traitements sont appliqués sur ces formes maintenant isolées, dans le but de leur attacher des caractéristiques diverses comme des paramètres de forme. Ces traitements nécessitent la mise en oeuvre d'algorithmes reposant sur des notions diverses importées du domaine continu comme la morphologie mathématique, les descripteurs de Fourier, les splines ou la géométrie différentielle.

Le plus souvent, les images sont représentées en trame carrée et les formes sont attachées à la représentation classique des données dans une image : le codage de la valeur et des coordonnées de chaque pixel. On peut cependant utiliser des représentations comportant une structure de données différente, comme des structures arborescentes si l'on veut travailler sur l'aspect multi-échelle. Ces structures de données masquent les problématiques de réduction d'information (compression avec ou sans perte dans un but de stockage et de transmission) et de simplification pour la perception d'une forme.

Dans ce contexte, Herbert Freeman [Fre74] publie en 1974 un premier article donnant les grands axes d'un moyen original de compression des formes binaires connexes pour lesquelles il suffit de connaître la frontière. Freeman propose donc de ne coder que la frontière de ces

formes présentes dans l'image binaire et non plus en donnant les valeurs des niveaux de gris de chacun des pixels composant l'image. Les coordonnées d'un pixel de la frontière de la forme étudiée constituent alors une information absolue, et l'information relative décrit la position relative des pixels du contour de la forme à travers une chaîne de codants formée de quatre (ou huit) symboles. Cet article comprend également la description et l'obtention de quelques paramètres géométriques à travers des algorithmes simples.

Depuis cette publication, bien que quelques articles mentionnent ce code, peu en ont étudié les propriétés et cherché à en extraire d'autres paramètres. De plus ces cinq dernières années ont vu apparaître de nouveaux codages de contours de forme, peu différents de celui proposé trente ans plus tôt.

Cette thèse s'inscrit dans cette thématique de la compression et du traitement d'images discrètes binaires et vise à développer des aspects algorithmiques décrivant l'obtention de paramètres géométriques sur des formes extraites de ces images. Nous nous sommes plus particulièrement attachés à l'étude du code de Freeman à la fois pour son ancienneté et aussi parce que la plupart des autres codes peuvent s'y ramener.

Il existe un lien important entre la structure de données utilisée et les algorithmes développés pour obtenir des paramètres. De ce fait, l'obtention du code de Freeman du contour d'un objet peut a priori induire une perte de temps mais peut parfois accélérer le traitement et le temps d'obtention d'un paramètre.

Nous nous sommes attachés à présenter les travaux de Freeman et à poursuivre le développement d'algorithmes permettant l'obtention de paramètres à partir de ce code.

Ainsi, dans un premier chapitre, nous verrons les différents codages de contours ayant été publiés depuis 1974 et une description du code de Freeman.

Le second chapitre est consacré à l'estimation de paramètres sur une chaîne de Freeman puis sur une forme à caractériser.

Le troisième chapitre de cette thèse présente les transformations que l'on peut faire subir à une forme, une méthode pour obtenir l'union et l'intersection de deux formes et des mesures de distance entre deux formes.

Le quatrième chapitre présente un développement du code de Freeman appliqué à la morphologie mathématique binaire.

Enfin, le dernier chapitre présente les perspectives à donner à ce travail.

Table des figures

1.1	Alphabet du code de Papert.	2
1.2	Codage d'un objet binaire par la tortue de Papert.	3
1.3	Les cinq changements orthogonaux de directions dans l'espace, alphabet du code de Bribiesca en 3D.	4
1.4	Exemple de représentation du code de Bribiesca en trois dimensions	5
1.5	Alphabet du code de Sanchez-Cruz, projection sur le plan des codes de Bribiesca.	5
1.6	Exemple de représentation du code de Sanchez-Cruz	6
1.7	Alphabet du Vertex Chain Code en trame carrée.	6
1.8	Exemple de codage utilisant le Vertex Chain Code.	7
1.9	Alphabet du code de Liu.	8
1.10	Exemple d'objet représenté grâce à son code de Liu.	8
1.11	Différents codages de contour présentés et leurs propriétés.	10
1.12	Vecteur donnant le nombre de pixels de l'objet dans les quatre directions : $V = (1\ 2\ 4\ 3)$	11
1.13	Voisinages V4 (à gauche) et V8 (à droite) d'un pixel (i,j), i et j représentant respectivement le numéro de ligne et le numéro de colonne.	12
1.14	Code de Freeman à 4 ou 8 directions. X désigne le point courant et les chiffres correspondent aux 4 ou 8 directions possibles pour lesquelles X a un voisin appartenant à la frontière de l'objet.	13
1.15	Exemple d'objet digital.	13
1.16	Exemple d'objet et du code de Freeman à huit directions de son contour.	14
1.17	Remplacement de 32 par 5 lors du passage de 4 à 8 directions, le remplacement de 23 par 5 induirait la perte du pixel grisé.	15
1.18	Passage d'un code à quatre directions à un code à huit directions.	15
1.19	Codage par les centres des pixels (flèches en pointillé) et par les contours des pixels (flèches en gras).	16
1.20	Déplacements en x et en y induits par chaque codant.	16
1.21	Exemple d'obtention du premier point et du premier codant. Le balayage se fait de haut en bas et de gauche à droite.	17
1.22	Exemple d'obtention d'un codant.	18
1.23	Exemple de code impossible avec le sens de rotation trigonométrique choisi au départ : 4524 sera remplacé par 444.	18
1.24	Cas particulier de codage : le pixel de départ est un pixel où le code va passer deux fois.	19
1.25	Exemple de code de Freeman.	19
1.26	Exemple d'objet touchant le bord de l'image. Point initial : $(x_0, y_0) = (10, 1)$. Code : (17 codants) 0 1 1 0 1 2 3 5 4 4 0 0 7 5 5 5 4.	20

1.27	Exemple d'objet discret et sa polygonalisation par Breton.	21
2.1	Calcul de l'aire contenue entre l'axe des x et la chaîne 32301061210.	27
2.2	Détermination du résidu d'une chaîne.	29
2.3	Détermination du résidu de la chaîne 3230106121022.	30
2.4	Détermination du résidu d'une paire de codants.	30
2.5	Notion d'intercepts.	32
2.6	Triangle formé par un point quelconque de l'image et deux pixels de la frontière d'un objet liés par un codant.	33
2.7	Les quatre directions utilisées par Ballard et Brown.	33
2.8	Objet codé en utilisant quatre directions.	34
2.9	Huit directions du code de Freeman.	35
2.10	Contribution de la couronne dans le calcul de l'aire.	36
2.11	Cas du premier point pour le calcul de l'aire.	37
2.12	Calcul des diamètres apparents dans le cas d'un triangle, en fonction de l'angle de projection (en degré).	39
2.13	Enveloppe convexe d'une courbe obtenue par H. Freeman.	40
2.14	Rectangle englobant d'aire minimale obtenu par H. Freeman.	41
2.15	Méthode du compas : $18\epsilon \leq P_\epsilon \leq 19\epsilon$	42
2.16	Méthode des boîtes : $P_\epsilon = 19\epsilon$	43
2.17	Méthode des intercepts : $P_\epsilon = 4 \times 4\epsilon = 16\epsilon$	43
2.18	Saucisse de Minkowski.	44
2.19	Estimation de la dimension de Bouligand-Minkowski.	47
2.20	Boîtes couvrant un objet.	47
2.21	Boules disjointes couvrant un objet.	47
2.22	Deux différents écartements pour mesurer la longueur d'un objet.	48
2.23	Courbe de Von Koch à différentes échelles.	48
2.24	Calcul de la dimension fractale de la courbe de von Koch.	50
2.25	Courbe fractale de Peano à trois différentes échelles.	51
2.26	Représentation des carrés pour la courbe de Peano.	51
2.27	Calcul de la dimension fractale de la courbe de Peano.	51
2.28	Comparaison de $\chi_1(A)$ et $\chi_2(A)$ pour un cercle très bruité.	53
2.29	Disque inscrit dans une forme A pour le calcul de χ_5 et χ_3	54
2.30	Illustration des Floating Bodies.	56
2.31	Fonction support.	57
2.32	Coefficient de Minkowski.	58
2.33	Coefficient de Winternitz.	58
2.34	Etude de la n-quasi-convexité d'une forme (ici n=13).	59
2.35	Objet étudié (à gauche) et l'histogramme de son code de Freeman associé.	60
2.36	Axes principaux d'inertie de deux objets.	61
3.1	Carré minimal (à gauche) et son expansion d'un facteur 2 (à droite).	70
3.2	Forme étudiée (à gauche), son expansion d'un facteur 2 selon [Mai] (au centre) et l'homothétique que l'on aimerait obtenir (à droite).	71
3.3	Forme de départ (à gauche) et son homothétique de rapport 2 (à droite).	71
3.4	Codants (flèches) et points de départs pour les codants suivants (points).	72
3.5	Exemple d'obtention d'un des points d'arrivée des codants de l'homothétique de rapport 2.	72

3.6	Homothétie de rapport deux, cas des premiers codants possibles.	72
3.7	Cas général pour l'homothétie de rapport deux. (X : cas impossible)	73
3.8	Homothétie de rapport deux, cas des derniers codants possibles. - : pas de rajout.	73
3.9	Exemple d'objet homothétisé d'un rapport 2.	74
3.10	Contraction (à droite) d'une chaîne de codants (à gauche).	74
3.11	Rotation (à droite) d'une chaîne (à gauche) de 90° dans le sens direct.	75
3.12	Comparaison des rotations	75
3.13	Exemple d'objet (à gauche) et son symétrisé par rapport à un point de coordonnées (6,6) (à droite).	76
3.14	Symétrie axiale	77
3.15	Test de l'appartenance d'un point à une forme par une étude angulaire (Le point en question se situe à l'intersection des droites en pointillés).	77
3.16	Test de l'appartenance d'un point à une forme par une étude de position relative des pixels appartenant à la même ligne.	78
3.17	Cas particuliers du test d'appartenance d'un point à une forme par une étude de position relative des pixels appartenant à la même ligne.	78
3.18	Les huit cas de croisements possibles de deux courbes sans intersection.	83
3.19	Intersection : 8 cas.	84
3.20	Etude du code de l'intersection de deux formes, cas 0 0 0 X.	84
3.21	Illustration du cas 1 pour l'intersection de deux formes.	85
3.22	Illustration du cas 3 pour l'intersection de deux formes.	85
3.23	Exemple d'intersection de deux formes.	85
3.24	Exemple d'union de deux formes.	87
3.25	Exemple de différence symétrique de deux formes, représentée par l'aire hachurée.	88
3.26	Exemples de distance de Hausdorff entre deux formes.	89
3.27	Différence entre la distance de Hausdorff entre deux formes et la distance de Hausdorff entre les frontières de ces deux formes.	90
3.28	Distance de Asplünd calculée grâce à 2 homothétiques.	90
4.1	Addition de Minkowski.	94
4.2	Dilatation de A par B.	95
4.3	Erosion de A par B.	96
4.4	Zones sortantes et rentrantes de A.	97
4.5	Codant 6 et sa saucisse englobante.	97
4.6	Obtention de la saucisse de dilatation pour les codants 0 et 1. [-] : nombre de codants que l'on enlève au code de Freeman courant ; [+] : codants que l'on ajoute ensuite au code de Freeman courant.	98
4.7	Pixel de départ et premiers codants possibles.	99
4.8	Dilatation par un voisinage V_4 des premiers codants possibles.	100
4.9	Arrêt du codage de la dilatation par un voisinage V_4 pour le codant 6.	100
4.10	Cas général de la dilatation par un voisinage V_4 . X : cas impossible ; - : le code reste inchangé.	101
4.11	Derniers codants à ajouter pour boucler la dilatation par un voisinage V_4	102
4.12	Dilatation par un voisinage V_8 des premiers codants possibles.	102

4.13	Cas général de la dilatation par un voisinage V_8 . X : cas impossible ; - : le code reste inchangé.	102
4.14	Derniers codants à ajouter pour boucler la dilatation par un voisinage V_8 . . .	103
4.15	Représentation de la frontière d'un objet (à gauche) et de la frontière de son dilaté par un voisinage V_4 (à droite).	103
4.16	Représentation de la frontière d'une étoile (à gauche), de la frontière de son dilaté par un voisinage V_4 suivant notre méthode (au centre) et de la frontière du dilaté souhaité (à droite).	103
4.17	Les séquences 6 4 ou 7 4 4 doivent être remplacées par le codant 5.	104
4.18	Séquences impossibles et codants correspondants.	104
4.19	Représentation de la frontière d'un objet (à gauche), de la frontière de son dilaté par un voisinage V_4 suivant notre méthode (au centre) et de la frontière du dilaté souhaité (à droite).	105

Table des algorithmes

1	Obtention du code de Papert	2
2	Obtention du code de Freeman	17
3	Calcul de l'aire par Ballard et Brown	34
4	Calcul de l'aire, généralisation à huit directions	35
5	Calcul de l'aire, nouvelle méthode	37
6	Calcul des diamètres apparents	39
7	Construction du flocon de Von Koch	49
8	Calcul de la dimension fractale par la méthode du compas	49
9	Obtention du coefficient d'asymétrie de Besicovitch	55
10	Obtention du coefficient de symétrie de Blaschke	57
11	Restauration de l'image à partir d'un code de Freeman	80
12	Recherche de pixels d'un contour ayant une abscisse connue	81
13	Calcul de la distance d'un pixel à la frontière d'un objet	81
14	Calcul de la distance entre deux objets	82
15	Calcul du code de l'intersection de deux objets	82
16	Calcul du code de l'union de deux objets	86
17	Calcul de la distance de la différence symétrique entre deux formes	88
18	Correction de code après dilatation	104
19	Calcul du code de l'érodé d'une forme	105
20	Calcul du code de l'érodé ultime d'une forme	106
21	Reconstruction par marqueur	106

Chapitre 1

Les différents codages de formes

Sommaire

1.1 Quelques codes de contour	2
1.1.1 La tortue de Papert	2
1.1.2 Le code de Bribiesca	3
1.1.3 Le Vertex Chain Code	6
1.1.4 Le code de Liu et Zalic	7
1.1.5 Le code de Zeng et Vasseur	9
1.1.6 Synthèse	9
1.2 Autres codes	10
1.2.1 Run-Length Encoding	10
1.2.2 Le code de Huffman	10
1.2.3 Neighbourhood Code	11
1.3 Le code de Freeman	12
1.3.1 Description et implémentation	12
1.3.2 Utilisations du code de Freeman	20
1.4 Conclusion	23

Le code de Freeman [Fre74] a comme but premier la compression des données. En effet, il permet de passer d'une image binaire, dont on mémorise la valeur du niveau de gris (0 ou 1) de chaque pixel, à une chaîne de codants d'une part et aux coordonnées d'un point du contour de l'objet d'autre part. Il est également possible d'utiliser le code de Freeman différentiel compressé grâce au code de Huffman [Liu] pour augmenter encore le taux de compression.

Dans ce chapitre, nous présenterons d'abord les codes de contour les plus connus, puis d'autres codes visant à compresser les images binaires et enfin le code de Freeman en détail.

1.1 Quelques codes de contour

1.1.1 La tortue de Papert

Description :

Une des représentations de contour a été proposée par S. Papert [Pap] en 1973. L'alphabet de ce code utilise uniquement deux symboles illustrés figure 1.1 : le 1, qui correspond à un virage à gauche pour sortir de l'objet étudié et le 0 associé à un virage à droite pour entrer dans l'objet étudié.

Algorithme 1 Obtention du code de Papert

- 1 Localisation d'un premier point de la frontière interne de l'objet (au sens voisinage V_8 , voir figure 1.13 page 12)
 - 2 **si** le pixel courant appartient à l'objet **alors**
 - 3 on tourne à gauche et on avance d'un pixel
 - 4 **sinon**
 - 5 on tourne à droite et on avance d'un pixel
 - 6 **fin si**
-

L'algorithme 1 présente la méthode d'obtention du code d'un contour selon le code de Papert. Dans cet algorithme, à la ligne 1, la localisation du premier point de la frontière interne peut se faire, grâce à la détection du premier point allumé par exemple au sens du balayage électronique (Voir figure 1.21 page 17). Du fait du choix de tourner à gauche lorsqu'on se trouve dans l'objet et à droite lorsqu'on se trouve à l'extérieur, le parcours de la frontière se fait dans le sens horaire.

La figure 1.2 page ci-contre représente le codage d'une forme binaire par cette tortue de Papert.

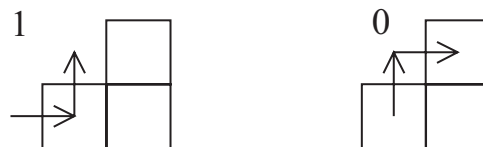


FIGURE 1.1 – Alphabet du code de Papert.

Propriétés :

Le codage par la tortue de Papert possède les propriétés suivantes :

Il est **invariant par rotation** d'un angle multiple de 90° .

Il est **invariant par translation** (il suffit de traduire le point de départ).

En transformant les 1 en 0 et réciproquement, on obtient le code du complémentaire de l'objet.

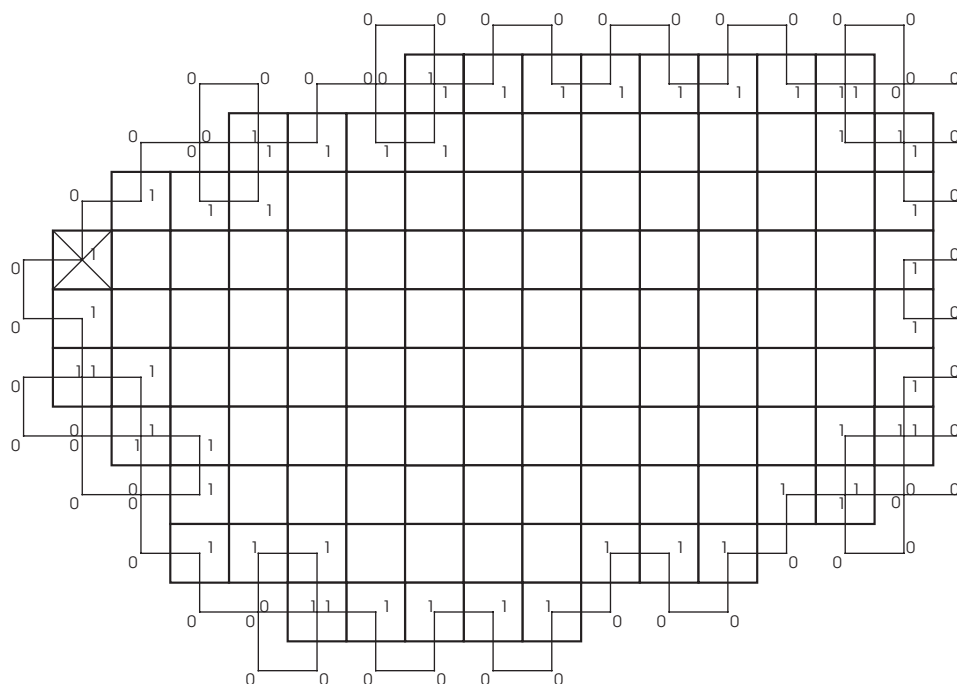


FIGURE 1.2 – Codage d’un objet binaire par la tortue de Papert.

Le code de l’objet obtenu par symétrie centrale (ou axiale) est simplement le même code, mais parcouru en sens inverse.

Le code est **indépendant du point de départ** : deux codages du même objet commençant à deux points différents seront identiques à une permutation circulaire près.

1.1.2 Le code de Bribiesca

1.1.2.1 En trois dimensions

Description :

Le code introduit par E. Bribiesca ([Bri99] et [Bri00]) permet de coder une surface discrète en trois dimensions afin d’en obtenir les caractéristiques. Cette surface ayant été extraite auparavant, on la considère comme une courbe faisant le tour de l’objet de départ (Voir l’illustration figure 1.4 page 5 issue de la publication de [Bri00]). On parle alors de courbe ou de surface par abus de langage.

Cette courbe est composée de segments non épais de taille égale à un côté de pixel. Les frontières d’un objet quelconque en trois dimensions peuvent être représentées par ce genre de chaîne. L’alphabet de cette représentation est composé uniquement de cinq symboles correspondant aux cinq variations possibles de directions orthogonales dans l’espace. Chacun de ces symboles est représenté par un nombre et illustré sur la figure 1.3 page suivante.

Deux segments définissent un changement de direction et deux changements de direction définissent un élément de la chaîne. Les éléments possibles sont les suivants :

- (a) le 0 indique qu’il n’y a pas de changement,

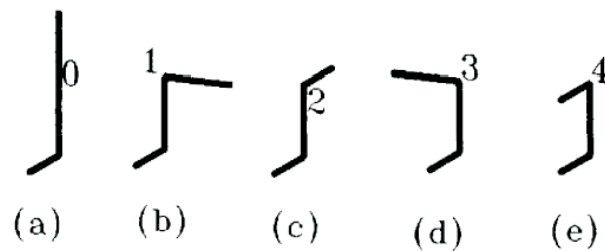


FIGURE 1.3 – Les cinq changements orthogonaux de directions dans l'espace, alphabet du code de Bribiesca en 3D.

- (b) le 1 est un virage vers la droite,
- (c) le 2 est un basculement vers l'avant,
- (d) le 3 symbolise un virage vers la gauche,
- (e) le 4 représente un retour en arrière.

La chaîne représentative d'une courbe est obtenue en calculant les changements relatifs de direction autour de la courbe. On obtient donc une chaîne composée d'un nombre fini d'éléments que l'on pourra coder en base 5. La longueur L de la courbe sera donc $L = (n+2)p$ avec n le nombre de codes dans la chaîne et p la longueur d'un segment (ici égal à un côté de pixel).

Propriétés :

Ce code possède plusieurs propriétés :

Indépendance par rotation : une courbe ainsi codée aura le même code après rotation d'angles multiples de 90° .

L'inverse d'une chaîne (voir § 2.1.1 page 26) peut être facilement calculé simplement en inversant l'ordre des codes. Attention, si des "0" sont présents dans la chaîne, il faudra alors inverser les codes précédant et suivant ces "0".

Indépendance du point de départ pour une courbe fermée.

Invariance après une symétrie par rapport à un plan, excepté l'inversion des 1 et des 3 dans la chaîne.

La comparaison de courbes est alors aisée car deux courbes identiques auront exactement le même code.

Une surface en trois dimensions représentée par un ensemble de voxels peut ainsi être codée par la courbe entourant cette surface. Un exemple issu de la publication de [Bri00] est donné figure 1.4 page ci-contre où l'on voit à gauche l'objet en volume à coder et à droite la succession des arêtes par lesquelles la courbe décrit la surface de cet objet.

Pistes de recherche :

Nous voyons dans ces travaux quelques améliorations possibles :

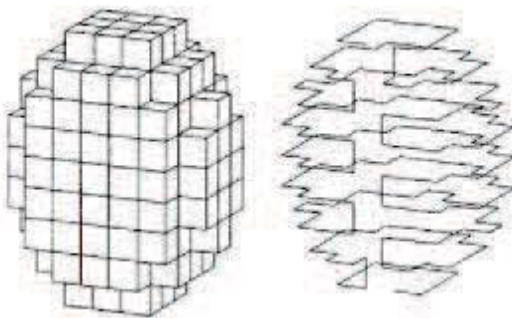


FIGURE 1.4 – Exemple de représentation du code de Bribiesca en trois dimensions

- à partir de représentations de surfaces en trois dimensions codées par ce processus, il semble possible, d'extraire le code du contour d'une coupe de l'objet 3D afin d'en étudier les paramètres géométriques.
- il est également envisageable de compresser encore plus ce code grâce à un code de Huffman ou une compression du nombre de 0 dans la chaîne.

1.1.2.2 En deux dimensions

Description :

L'étude de E. Bribiesca [Bri00] a été poursuivie en 2005 par H. Sanchez-Cruz [San05] en n'utilisant plus que trois des cinq symboles initiaux afin de rester dans le plan. Une représentation graphique de l'alphabet de ce code correspondant à ces trois changements de direction est donnée figure 1.5. Voici ce à quoi ils correspondent :

- Le codant 0 n'induit pas de changement de direction, c'est à dire que l'on continue dans la même direction que le codant précédent.
- Le codant 1 indique un changement de direction en avant par rapport au codant précédent.
- Le codant 2 indique un retour en arrière par rapport au codant précédent.

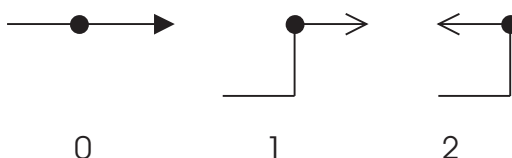


FIGURE 1.5 – Alphabet du code de Sanchez-Cruz, projection sur le plan des codes de Bribiesca.

Propriétés :

Les propriétés géométriques de ce code sont :

- l'**invariance par translation**,
- l'**invariance par rotation**,

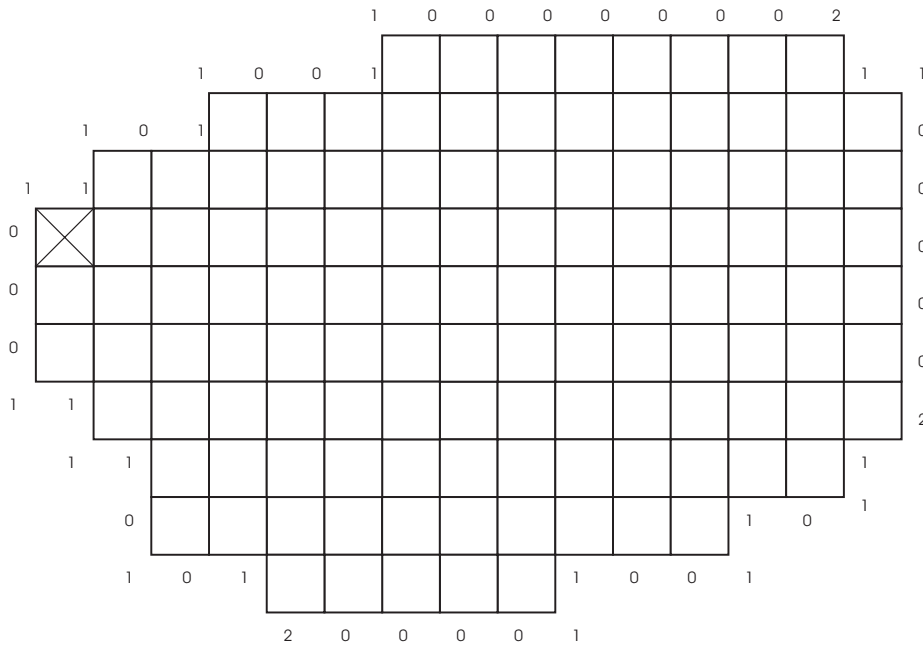


FIGURE 1.6 – Exemple de représentation du code de Sanchez-Cruz

- l'invariance par symétrie par rapport à un point et par rapport à un axe.

La figure 1.6 donne l'exemple d'un objet codé grâce à ce code.

1.1.3 Le Vertex Chain Code

Description :

En 1999, E. Bribiesca [Bri99] a présenté un autre moyen de coder les contours de formes : le Vertex Chain Code ou VCC. Le principe est de compter le nombre de sommets à l'intérieur de l'objet, et touchant le sommet considéré. Pour améliorer la compression, chacun de ces nombres est décrétementé d'une unité.

Dans le cas des pixels d'une image en trame carrée, il existe trois possibilités décrites respectivement par les symboles 0, 1 et 2. L'alphabet de ces symboles, correspondant respectivement à un, deux et trois sommets à l'intérieur de l'objet, est représenté figure 1.7.

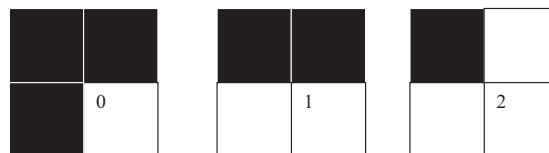


FIGURE 1.7 – Alphabet du Vertex Chain Code en trame carrée.

Propriétés :

Le Vertex Chain Code possède les propriétés suivantes :

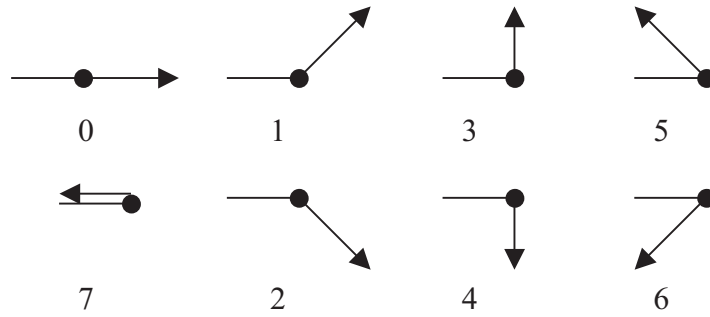


FIGURE 1.9 – Alphabet du code de Liu.

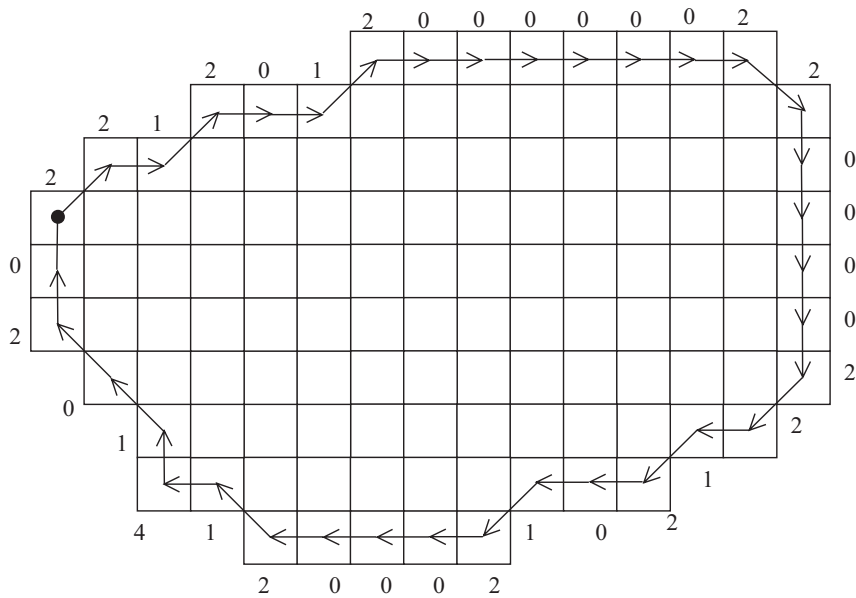


FIGURE 1.10 – Exemple d'objet représenté grâce à son code de Liu.

- **indépendance par rotation** d'angles multiples de 90° ,
- **indépendance par translation**,
- **indépendance du point de départ**.
- la symétrie par rapport à un axe est obtenue en parcourant le code à l'envers et en inversant les 1 avec les 2, les 3 avec les 4, et les 5 avec les 6.

1.1.5 Le code de Zeng et Vasseur

Il existe un autre code développé pour décrire les frontières d'un objet. Il s'agit du code présenté par X. Zeng et C. Vasseur [Zen]. Ce code nécessite des connaissances a priori de l'objet étudié : détection de **segments**, détection de **portions de cercles**, connaissance de la **frontière** de l'objet et **périmètre**.

Le code du contour ne décrit donc que les segments et les portions de cercles. Une portion de cercle est codée par la lettre b alors qu'un segment de droite est codé par la lettre a .

Après détection du contour, les segments de droite de celui-ci sont codés de la façon suivante :

$$(a - m)(c - p)(d - \theta) \quad (1.1)$$

La première partie de ce code : $(a - m)$, signifie que le segment nommé m est une ligne (m est en fait un nombre entier repérant le numéro du segment). La seconde partie $(c - p)$ signifie que la longueur de ce segment représente $p\%$ du périmètre de l'objet (supposé fermé) et la dernière $(d - \theta)$ signifie que l'angle formé par ce segment et l'axe des x est de θ degrés dans le sens indirect.

Les arcs de cercle (AB de centre C) sont codés ainsi :

$$\begin{aligned} (b - n_1)(c - p)(d - \theta_1) \\ (b - n_2)(d - \theta_2) \end{aligned} \quad (1.2)$$

avec $n_2 = n_1 + 1$, $(b - n_1)$ et $(b - n_2)$ représentent les segments AC et BC , p est le pourcentage de la longueur AB dans le périmètre de l'objet, θ_1 et θ_2 sont respectivement les angles formés par l'axe des x et les segments AC et BC .

En plus des connaissances a priori qu'elle requiert, cette méthode est en quelque sorte une *vectorisation* et ne code plus tous les pixels de la frontière de l'objet. Pour ces raisons, l'obtention du code n'est pas aussi simple et rapide que dans les cas vus précédemment et nous avons décidé de ne pas approfondir l'étude de ce code.

1.1.6 Synthèse

Le tableau 1.11 page suivante résume les différents codes et leurs propriétés. Ils sont classés par ordre croissant de l'efficacité calculée par [San07]. Dans ce tableau, les codes donnés pour invariants par symétrie axiale, le sont à un détail près : il faut parcourir le code en sens inverse.

De plus, la longueur du code nous donne une indication sur la taille relative des codes obtenus. Ainsi, le code de Freeman à 4 directions, le VCC et le code introduit par Sanchez-Cruz d'une part et le code de Liu et Zalic et le code de Freeman à 8 directions d'autre part ont exactement la même longueur.

Enfin, les invariances par rotation ne prennent en compte que les rotations multiples de 90° .

Une caractéristique supplémentaire de ces codes est qu'ils sont tous indépendants du point de départ sur la frontière, à une permutation circulaire près du code.

Code	nombre de symboles	longueur du code	invariance par		
			rotation	translation	symétrie
Liu et Zalic	8	le plus court	oui	oui	non
Sanchez-Cruz	3	moyenne	oui	oui	oui
Vertex Chain Code	3	moyenne	oui	oui	oui
Freeman 4 directions	4	moyenne	non	oui	non
Tortue de Papert	2	le plus long	oui	oui	oui
Freeman 8 directions	8	le plus court	non	oui	non

FIGURE 1.11 – Différents codages de contour présentés et leurs propriétés.

1.2 Autres codes

1.2.1 Run-Length Encoding

Le code RLE (Run-Length Encoding) est un algorithme de compression d'images binaires sans perte de données. Il a pour but de coder ligne par ligne le nombre de pixels consécutifs ayant la même couleur. On l'utilise par exemple pour compresser l'image correspondant à une page de texte scannée ou pour l'envoi de fax. Comme il est fréquent sur de telles images d'avoir sur la même ligne plusieurs pixels de la même couleur (noir ou blanc) ce code est bien une compression de l'image de départ.

Prenons pour exemple une partie d'une ligne de fax représentée par des B pour les pixels noirs et des W pour les pixels blancs. On pourra avoir ce genre de séquence :

WWWWWWWWWWBWWWWWWWWBWWWWWWWWWWWWWWWWBWW qui sera alors remplacée par : 11W 3B 7W 1B 11W 3B 1W.

1.2.2 Le code de Huffman

Le code de HUFFMAN (1952) consiste à associer aux différents symboles des codes d'autant plus courts que ces symboles sont fréquents : il minimise la longueur moyenne pour une distribution de probabilité donnée. La perte de performance provient de la quantification des données : le catalogue formé doit être associé à l'objet compressé, il est bien sûr comptabilisé dans la taille totale du fichier.

Une étude ayant été faite dans le but d'utiliser la compression de Huffman, Liu et Zalic [Liu] ont démontré (en utilisant le code présenté sur la figure 1.9 page 8) sur un échantillon de 1000 objets que le codant 0 représente 45% des codants. Viennent ensuite les codants 1 et 2 avec chacun 25%, puis 3 et 4 avec 2%, et enfin 5, 6 et 7 avec moins de 1%. L'utilisation du code de Huffman est donc justifiée et apporte une compression de l'ordre de 40% dans les exemples donnés lors de cette étude.

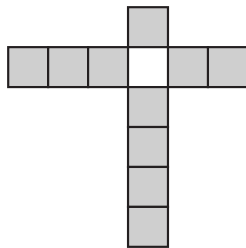


FIGURE 1.12 – Vecteur donnant le nombre de pixels de l’objet dans les quatre directions : $V = (1\ 2\ 4\ 3)$.

1.2.3 Neighbourhood Code

Une autre approche pour décrire les formes (l’étude porte actuellement sur les caractères écrits à la main) est celle présentée par I.J. Tsang [Tsa]. On ne code plus ici uniquement les pixels de la frontière de l’objet, mais tous les pixels de l’objet.

Le but de cet article est de reconnaître des caractères écrits à la main à l’aide d’un code utilisant les relations entre pixels voisins. Pour cela, chaque pixel de l’image est transformé en un vecteur donnant une information sur le nombre de pixels voisins dans les quatre directions (nord, est, sud, ouest). Ces vecteurs de voisinage sont transformés en un code donnant des informations sur la frontière de l’objet. On associe à chaque pixel un vecteur contenant le nombre de voisins dans les quatre directions nord, est, sud et ouest : $V = (n\ e\ s\ o)$. Considérant le pixel blanc de la figure 1.12, nous avons le vecteur $V = (1\ 2\ 4\ 3)$.

L’ensemble de ces vecteurs pour tous les pixels décrit parfaitement l’image.

Ce code présente une propriété d’invariance par translation et par rotation d’angles multiples de 90 degrés. Ces rotations induiront un décalage dans les coordonnées des vecteurs.

On utilise ensuite une fonction Φ de transformation des vecteurs $V = (n\ e\ s\ o)$ en un code $C = \Phi(n\ e\ s\ o)$. Cette transformation dépend de la taille de l’image $N \times N$. La plus grande valeur que puissent prendre n , e , s ou o est $N - 1$. La transformation a alors les conditions limites suivantes :

$$\begin{aligned} n + s &< N \\ e + o &< N \end{aligned} \tag{1.3}$$

Soient α et β les fonctions de transformation pour les paires (n, s) et (e, o) . Sans tenir compte des conditions aux limites, on a :

$$\begin{aligned} \alpha &= nN + s \\ \beta &= eN + o \end{aligned} \tag{1.4}$$

En respectant les conditions aux limites, on obtient :

$$\begin{aligned} \alpha &= nN + s - \frac{n(n-1)}{2} \\ \beta &= eN + o - \frac{e(e-1)}{2} \end{aligned} \tag{1.5}$$

En posant par la suite $\gamma = \alpha K + \beta$ et $K = \beta_{max} + 1 = \frac{N^2 + N}{2}$, on a :

$$\gamma = \alpha \frac{N^2 + N}{2} + \beta \tag{1.6}$$

Nous avons donc ici pour une imagerie 16×16 pixels 18496 codes possibles, cette imagerie étant de taille suffisante pour reconnaître un caractère.

Ce code peut ensuite être réduit pour obtenir la taille d'image désirée, par exemple pour passer d'une image 128×128 à une image 16×16 .

Une fois la bonne taille obtenue, il suffit de comparer le code de l'imagerie avec une base de données et ainsi retrouver le caractère présent dans l'image. Le caractère retenu est celui pour lequel l'erreur (par exemple l'erreur quadratique moyenne) est la plus faible.

1.3 Le code de Freeman

1.3.1 Description et implémentation

1.3.1.1 Description

En dimension 2, sur un maillage carré, on peut étudier les propriétés des objets en utilisant le voisinage à quatre points noté $V_4(i, j)$ ou le voisinage à huit points noté $V_8(i, j)$ d'un pixel (i, j) où i et j représentent respectivement le numéro de ligne et le numéro de colonne du pixel dans l'image. Ces voisinages sont représentés sur la figure 1.13. Ils correspondent respectivement aux quatre et huit pixels les plus proches du pixel (i, j) .



FIGURE 1.13 – Voisinages V_4 (à gauche) et V_8 (à droite) d'un pixel (i, j) , i et j représentant respectivement le numéro de ligne et le numéro de colonne.

En considérant le cas des pixels carrés, le voisinage V_4 a la particularité suivante : tous les points du voisinage sont à une distance égale du point central. Cette distance vaut 1 pixel. On parle alors de 4-connexité. (Un objet pour lequel le passage d'un pixel à l'un de ses voisins ne se fait qu'à la verticale ou l'horizontale est 4-connexe.)

Pour le voisinage V_8 , cette distance vaut également 1 pixel pour les pixels situés horizontalement ou verticalement par rapport au pixel central et $\sqrt{2}$ pixel pour les pixels situés en diagonale du point central. On parle alors de 8-connexité. (Un objet pour lequel le passage d'un pixel à l'un de ses voisins se fait à la verticale, l'horizontale ou en diagonale est 8-connexe.)

On remarque l'inclusion suivante :

$$V_4(i, j) \subset V_8(i, j) \tag{1.7}$$

3	2	1
4	X	0
5	6	7

	1	
2	X	0
	3	

FIGURE 1.14 – Code de Freeman à 4 ou 8 directions. X désigne le point courant et les chiffres correspondent aux 4 ou 8 directions possibles pour lesquelles X a un voisin appartenant à la frontière de l'objet.

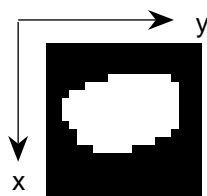


FIGURE 1.15 – Exemple d'objet digital.

Le code de Freeman, introduit la première fois en 1961 dans [Fre61a], a pour but de coder le contour d'un objet par :

- une information *absolue* correspondant aux coordonnées d'un point de départ,
- une chaîne de codants donnant la position *relative* du point suivant du contour de l'objet selon une des représentations présentées figure 1.14 (codage utilisant 4 ou 8 directions). Ainsi, en utilisant 8 directions, le codant 0 signifie que le pixel suivant du contour de l'objet se situe à droite du pixel courant, et le codant 5 désigne un pixel suivant en bas à gauche du pixel courant.

Chacune des 4 et 8 directions peut être codée respectivement sur 2 et 3 bits, induisant une forte compression sans perte de l'image.

Traditionnellement, ces directions sont comptées dans le sens trigonométrique (sens direct). La figure 1.15 présente un objet A blanc sur fond noir. Ainsi, pour le code utilisant 8 directions, les codants pairs lient deux pixels par une distance d'un pixel alors que les codants impairs lient deux pixels par une distance de $\sqrt{2}$ pixel.

On désire obtenir le code de Freeman d'un objet A considéré. Pour cela, on sélectionne un point appartenant à la frontière interne de A (par exemple le premier pixel rencontré par balayage électronique), on mémorise les coordonnées de ce premier point (ici, $(3,8)$) puis on cherche son plus proche voisin appartenant à A (au sens d'un voisinage V_4) selon un sens de rotation donné (ici, le sens direct). On réitère ensuite cette dernière opération jusqu'à revenir au point de départ.

Ainsi, de proche en proche, on reconstitue la forme de l'objet A en donnant le codage de Freeman [Fre74] de la frontière de A .

Cet exemple donne la chaîne 667760700001001012222234444444544545. Un schéma un peu plus détaillé du code obtenu sur cet objet est présenté figure 1.16 page suivante.

On peut également utiliser le codage différentiel qui donne

figure 1.18. Par exemple on pourra remplacer 32 (4 directions) par 5 (8 directions). Pour les autres cas, il suffit d'associer le codant huit directions correspondant au codant quatre directions.

Un tableau résumant tous les cas possibles est présenté dans ce même tableau.

Notons que le remplacement des codants 23 (4 directions) par 5 (8 directions) induirait la perte d'un pixel (voir figure 1.17). Il en est de même par rotation pour les autres cas présentés dans le tableau 1.18. L'ordre des codants dans ce tableau est donc important.

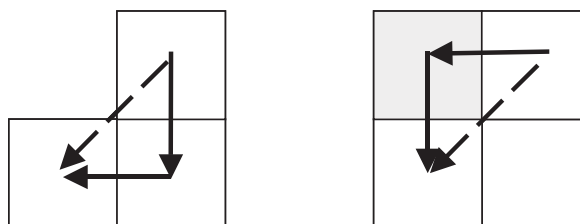


FIGURE 1.17 – Remplacement de 32 par 5 lors du passage de 4 à 8 directions, le remplacement de 23 par 5 induirait la perte du pixel grisé.

En revanche, la transformation d'un code utilisant 8 directions à un code n'utilisant que 4 directions n'est possible que si l'on met en évidence une connexion diagonale ou si l'objet est 4-connexe. Sinon, le passage d'un code de Freeman à 8 directions à un code de Freeman à 4 directions augmentera le nombre d'objets de l'image.

Codant quatre directions		Codant correspondant (huit directions)
paire de codants	codant simple	
3 2		5
0 3		7
1 0		1
2 1		3
	0	0
	1	2
	2	4
	3	6

FIGURE 1.18 – Passage d'un code à quatre directions à un code à huit directions.

A partir du code à huit directions décrivant le polygone passant par le milieu des pixels de la frontière interne d'un objet (flèches en pointillés sur la figure 1.19 page suivante), il est également possible d'obtenir le code à quatre directions décrivant le contour extérieur de ces mêmes pixels (flèches en gras sur la figure 1.19 page suivante).

A chaque codant a_i est associé un vecteur (a_{ix}, a_{iy}) correspondant aux déplacements en x et en y introduits par ce codant. La figure 1.20 page suivante présente ces vecteurs dans le cadre de notre étude. Le cas 1 donne ces déplacements pour les axes orientés en x croissant vers le bas et y croissant vers la droite, le cas 2 pour les axes orientés classiquement en x croissant vers la droite et en y croissant vers le haut.

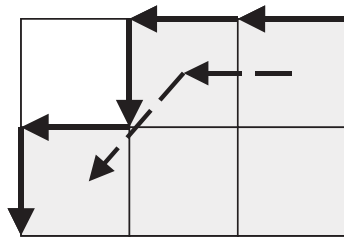


FIGURE 1.19 – Codage par les centres des pixels (flèches en pointillé) et par les contours des pixels (flèches en gras).

	Cas 1		Cas 2	
Codant	a_{ix}	a_{iy}	a_{ix}	a_{iy}
0	0	1	1	0
1	-1	1	1	1
2	-1	0	0	1
3	-1	-1	-1	1
4	0	-1	-1	0
5	1	-1	-1	-1
6	1	0	0	-1
7	1	1	1	-1

FIGURE 1.20 – Déplacements en x et en y induits par chaque codant.

1.3.1.2 Implémentation

Nous avons implémenté le code de Freeman à 4 et à 8 directions. Nous travaillons ici avec un seul objet connexe. Voici le fonctionnement du programme réalisé pour le code à 8 directions :

La fonction a les paramètres suivants :

- en entrée :
 - l'image binaire contenant l'objet dont on désire le code de Freeman
- en sortie :
 - les coordonnées (x, y) du point de départ
 - le code de Freeman de la forme, dans un tableau d'une ligne et n colonnes (n étant le nombre de codants)

L'algorithme développé est l'algorithme [2 page ci-contre](#).

Cet algorithme ne nécessite pas d'extraction de la frontière avant d'être exécuté et a donc l'avantage de travailler sur un objet binaire complet, et non pas uniquement sur la frontière de l'objet.

1.3.1.3 Remarques :

- Chaque codant est noté ici a_i . Un ensemble de codants forme une chaîne, et une chaîne de codants notée $a_1a_2\dots a_i\dots a_n$ donne la représentation du contour d'un objet. n représente le nombre de codants nécessaires pour revenir au point de départ.

Algorithme 2 Obtention du code de Freeman

-
- 1 recherche du premier point courant $P(x_0, y_0)$
 - 2 recherche du premier codant à partir du premier point courant (voir quatrième remarque ci-après)
 - 3 mise en mémoire du premier codant
 - 4 nouveau point courant = point codé
 - 5 **tant que** nouveau point courant $\neq P(x_0, y_0)$ **faire**
 - 6 recherche du codant suivant
 - 7 mise en mémoire du codant
 - 8 **fin tant que**
 - 9 **retourner** le chaîne de codants et $P(x_0, y_0)$
-

- On travaille ici en trame carrée, mais le code de Freeman s'obtient de façon identique si le pas en x est différent du pas en y .
- Les axes des x et des y pour les images étudiées ne sont pas orientés de façon usuelle ; la première coordonnée représente le numéro de la ligne et la seconde le numéro de la colonne de l'image. Ainsi, l'axe des x est vertical et dirigé vers la bas, et l'axe des y est horizontal et dirigé vers la droite, comme représenté sur la figure 1.15 page 13.
- La recherche du premier codant à partir du premier point courant est dissociée du reste du code car elle est un peu particulière. En effet, compte tenu de la méthode d'obtention du premier point de l'objet, par balayage de haut en bas et de gauche à droite, le point suivant ne peut se trouver qu'avec les codants 6, 7, 0 ou 1. Cette particularité est due au fait qu'un pixel est d'abord repéré par son numéro de ligne et ensuite par son numéro de colonne. Certains logiciels comme Matlab utilisent d'ailleurs ce balayage par défaut. Sur l'exemple de la figure 1.21, le premier codant sera donc 6.

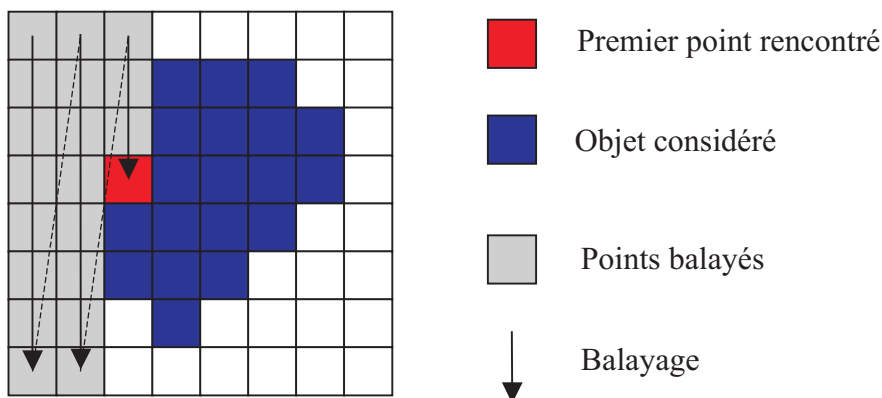


FIGURE 1.21 – Exemple d'obtention du premier point et du premier codant. Le balayage se fait de haut en bas et de gauche à droite.

- La recherche du codant suivant est faite comme suit : comme on désire tourner autour de la forme dans le sens trigonométrique, cela signifie que, dans ce sens de parcours, la forme sera toujours à gauche. Pour cette raison, si le dernier codant trouvé est a , on recherchera le suivant en commençant par $a - 2[8]$ (où $[8]$ signifie modulo 8). Si le point

endroit, il ne faut pas arrêter le codage de la forme. Par conséquent, de retour au point de départ, si l'objet n'a pas été codé complètement, il faut coder le reste. L'exemple d'un tel cas est donné sur la figure 1.24 où le premier pixel au sens du balayage utilisé est marqué d'une croix, la partie de l'objet codée sans prendre de précaution est en blanc et la partie à ajouter est en gris.

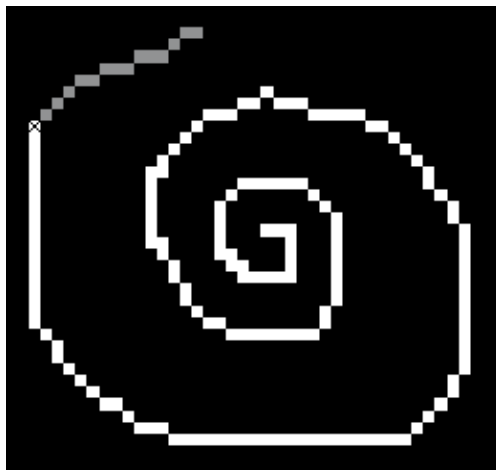


FIGURE 1.24 – Cas particulier de codage : le pixel de départ est un pixel où le code va passer deux fois.

- Voici un exemple de code obtenu sur une roue dentée (voir figure 1.25) :
Point initial : $x_0 = 36, y_0 = 8$
code : (276 codants) 666666600070007666545554767767010010077665665670
7707211121000766676660000000222122210007677761011012322322110070070
121121433343222100010002222224443444322210111032332344545443322122
1234334365556544432223222444444466656665444323332545545676676655443
434456556507770766654445444

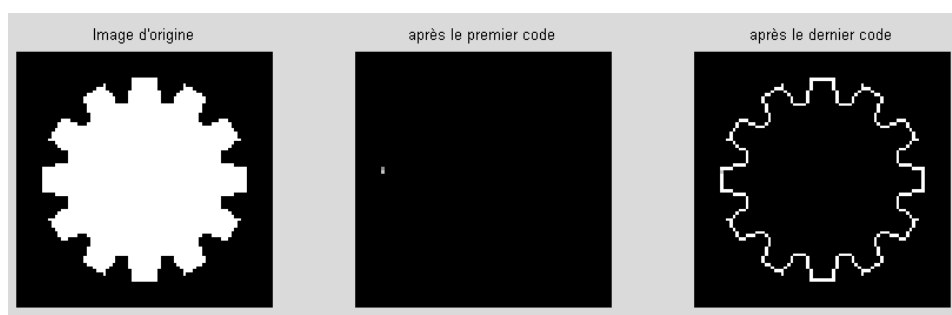


FIGURE 1.25 – Exemple de code de Freeman.

- Ce programme a été amélioré pour prendre en compte les objets touchant le bord de l'image. En effet, la recherche du codant suivant se fait en parcourant les pixels voisins afin de trouver le premier pixel allumé. Si le point courant est un pixel du bord de l'image, la recherche se fera sur un pixel hors de l'image, et le programme se terminera suite à une erreur de dimension. Pour remédier à ce problème, si l'objet touche le

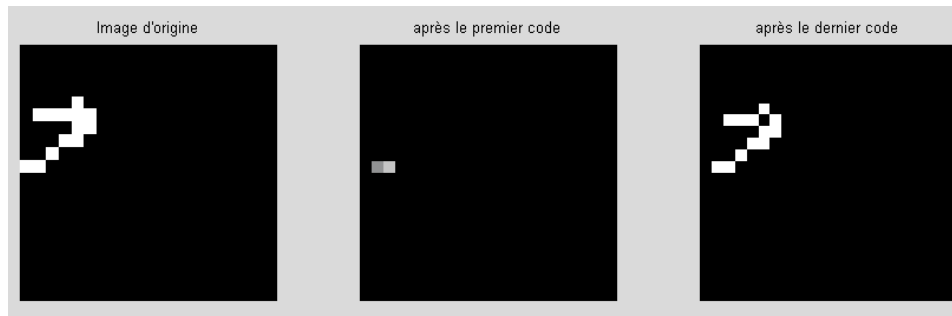


FIGURE 1.26 – Exemple d’objet touchant le bord de l’image. Point initial : $(x_0, y_0) = (10, 1)$. Code : (17 codants) 0 1 1 0 1 2 3 5 4 4 0 0 7 5 5 5 4.

bord, nous ajoutons une bande noire de largeur un pixel, tout autour de l’image, avant d’effectuer le codage. Bien sur, en fin de codage, l’erreur induite par cette méthode sur les coordonnées du premier point est corrigée. Un exemple de résultat obtenu, avec un objet touchant le bord est donné figure 1.26.

Nous sommes alors en mesure d’utiliser le codage de Freeman afin d’estimer certains paramètres de l’objet.

1.3.2 Utilisations du code de Freeman

1.3.2.1 Reconnaissance de caractères chinois

Le code de Freeman est utilisé pour la reconnaissance de caractères chinois écrits à la main par A. Amin [Ami] : après une numérisation de l’écriture, le squelette des caractères est extrait par la méthode des boules maximales(voir[Blu]) et analysé grâce au codage de Freeman. Si nécessaire, ce code de Freeman du contour est simplifié par filtrage. Les caractéristiques sont ensuite extraites et la forme est classifiée.

Ici, A. Amin utilise le squelette; cependant l’opération de squelettisation n’est pas une application continue dans l’espace des formes muni de la distance de Hausdorff. Deux formes voisines peuvent avoir des squelettes très différents et la reconnaissance d’objet par cette méthode n’est donc pas robuste.

Il est cependant théoriquement possible de revenir à l’objet compact de départ à partir du squelette si pour chaque point de celui-ci est conservé en mémoire le rayon de la boule maximale centrée en ce point contenue dans la forme de départ. Ainsi, par union de toutes ces boules, l’objet initial peut être retrouvé.

1.3.2.2 Correction de caractères arabes manuscrits

Une application similaire est réalisée par [Mad] afin d’effectuer une correction géométrique des caractères arabes manuscrits. L’utilisation du code de Freeman est cependant différente. Ici, après détection du contour du caractère et obtention de son code de Freeman, on réalise une transformée de Fourier discrète (FFT). En effet, à partir du code de Freeman obtenu, on génère une fonction temporelle périodique sur laquelle le calcul de la FFT est réalisable.

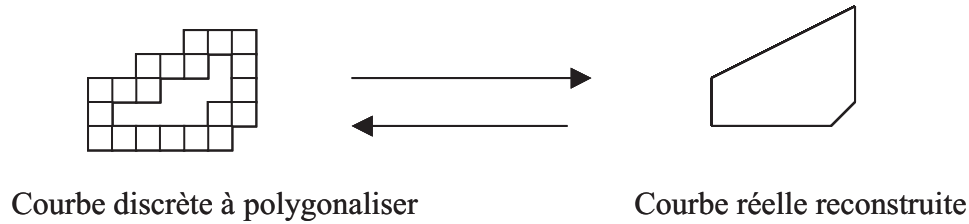


FIGURE 1.27 – Exemple d'objet discret et sa polygonalisation par Breton.

La notion de temps est ici associée à la durée du parcours du contour de sorte que le temps nécessaire pour parcourir les k premiers arcs qui relient les points du contour est :

$$t_k = \sum_{i=1}^k \Delta t_i \quad (1.10)$$

avec $\Delta t_i = 1$ pour un codant pair et $\sqrt{2}$ pour un codant impair. Les distances parcourues à l'instant t_k sont décrites par x_k (projection du contour sur l'axe des X) et y_k (projection du contour sur l'axe des Y) avec :

$$\begin{aligned} x_k &= \sum_{i=1}^k \Delta x_i \\ y_k &= \sum_{i=1}^k \Delta y_i \end{aligned} \quad (1.11)$$

où $\Delta x_i \in \{-1, 0, 1\}$ suivant la direction donnée par le code de Freeman. Ainsi, le contour peut être décrit par la fonction discrète (x_k, y_k) sur laquelle on calculera la transformée de Fourier.

1.3.2.3 Simplification de polygones

Ce code est également utilisé dans le codage et la simplification de polygones. O. Rogalla [Rog] utilise la programmation par la démonstration, afin de programmer des robots. Une des étapes est l'extraction de contours de formes, puis le codage de Freeman des frontières et une simplification afin d'obtenir un polygone pour classifier et utiliser la forme simplifiée obtenue.

L. Dorst [Dor] utilise les propriétés du codage de Freeman pour extraire d'une forme quelconque la (n) -caractérisation, la (n_e, n_o) -caractérisation, la (n_e, n_o, n_c) -caractérisation et la (n, p, q, s) -caractérisation. Celles-ci donnent respectivement le nombre d'éléments de la chaîne (n) , le nombre d'éléments pairs (n_e) et impairs (n_o) de la chaîne, le nombre de "coins" dans la chaîne (n_c) . Les paramètres n, p, q, s ont été introduits en 1984 dans [Dor84].

J. Kim [Kim] présente la notion d'approximation de formes par des polygones. Il utilise le code de Freeman pour coder la frontière des objets avant de réaliser l'approximation.

R. Breton [Bre] travaille également avec des polygones. Il propose un algorithme de vectorisation (reconstruction euclidienne) d'une forme discrète 4-connexe inversible et proche d'un résultat intuitif. Le but de cet algorithme est illustré par la figure 1.27.

La méthode est la suivante : à partir d'un point de la courbe, on recherche le segment auquel ce point appartient. On réitère ensuite le processus jusqu'à obtention de tous les segments constituant l'objet que l'on peut alors reconstruire.

Le code de Freeman est utile ici afin de déterminer les points de rebroussement. Une perspective de ce travail est maintenant d'étudier ces points de rebroussement car l'algorithme échoue en cas de tels points. Une autre perspective consiste en une adaptation vers la 3D.

Amélioration possible : La détection de segments de droites doit pouvoir se faire à partir du code de Freeman du contour de l'objet. En effet, un segment de droite a un code de Freeman périodique. En travaillant sur la recherche de périodes dans le code (Recherche systématique, Fourier...) on peut déterminer les segments de droites composant le polygone que l'on cherche à caractériser.

Nous envisageons ici une polygonalisation d'objets binaires à partir du code de Freeman et des algorithmes de Debled-Renneson et Reveillès (voir [Rev] et [Deb]).

1.3.2.4 Reconnaissance de profils de visages humains

Il existe plusieurs solutions pour reconnaître des visages parmi lesquelles :

- Le codage de Freeman pur du contour du visage, muni d'une distance entre 2 codes,
- La vectorisation du contour du visage muni d'une distance,
- La caractérisation de points particuliers et leur comparaison.

Ce dernier point a été étudié par D. Odin [Odi] en 1995. L'étude se déroule en plusieurs points :

- Cadrage du profil :
Le profil utilisé est le profil gauche afin d'être en harmonie avec les fichiers anthropométriques de la police. Le cadrage doit être relativement serré afin que l'ensemble de la tête soit visible. Le sujet doit être imberbe afin de ne pas fausser les mesures. On doit voir l'oreille dans sa totalité afin de pouvoir caractériser une ellipse qui l'englobe.
- Prétraitement de l'image :
On réalise un filtrage puis un seuillage de l'image afin d'obtenir le contour du visage, puis une reconstruction. On détecte ensuite le contour grâce au code de Freeman. On obtient ainsi le code du profil gauche de l'individu dont on veut extraire les caractéristiques physiques.
- Détection et caractérisation de neuf points de la silhouette du profil :
Ces points sont les suivants : le bout du nez, le bout du menton, le bas du nez, le coin de la bouche, le bas de la bouche, la lèvre supérieure, la lèvre inférieure, le front et le bas du front. Ces points sont recherchés comme des points extrémaux locaux sur le code de Freeman du contour du profil.
- Détermination et caractérisation du coin de l'oeil :
On utilise ici une méthode empirique qui consiste en une suite d'étapes visant à rapprocher un point courant du véritable coin de l'oeil. Elle repose en partie sur la caractérisation des points du profil.
- Détection et caractérisation des grains de beauté éventuels :
Cette détection repose sur le fait que les grains de beauté, supposés assimilables à des cercles, ont une certaine uniformité des niveaux de gris le long d'un cercle donné.

- Détection et caractérisation d'une ellipse d'approximation de l'oreille :
La méthode utilisée a été développée à partir du concept de "templates" déformables.

1.4 Conclusion

Nous avons décrit les codages de contours les plus connus existant actuellement. Notre étude porte maintenant sur l'utilisation du code de Freeman. Nous avons choisi d'approfondir ce codage car il est apparu dans les premiers en 1974 et nous paraît plus adapté à l'étude des paramètres de formes, du fait d'une part de l'information absolue donnant les coordonnées du premier point du contour, et d'autre part de l'information relative donnant les positions des pixels voisins.

Enfin, il est possible de passer du code de la tortue de Papert, ou de celui de Sanchez-Cruz, du VCC ou du code de Liu et Zalic au code de Freeman de façon bijective. Une étude similaire de ces codes n'est donc pas nécessaire.

Chapitre 2

Estimation de paramètres à partir du codage de Freeman

Sommaire

2.1 Paramètres et opérations sur une chaîne	26
2.1.1 Inverse d'une chaîne	26
2.1.2 Longueur d'une chaîne	26
2.1.3 Largeur et hauteur d'une chaîne	26
2.1.4 Intégration par rapport à l'axe des x	27
2.1.5 Moments par rapport à l'axe des abscisses	28
2.1.6 Résidus	28
2.1.7 Distance entre deux points	29
2.1.8 Miroir d'une chaîne	31
2.2 Paramètres géométriques	31
2.2.1 Périmètre	31
2.2.2 Aire	33
2.2.3 Centre de masse	38
2.2.4 Rayon maximum, minimum et moyen	38
2.2.5 Diamètre apparent maximal, minimal et moyen	38
2.2.6 Rectangle englobant d'aire minimale	40
2.2.7 Dimension fractale	42
2.3 Paramètres de formes	52
2.3.1 Paramètres de circularité	52
2.3.2 Paramètres de symétrie	54
2.3.3 Convexité	59
2.3.4 Histogramme d'un code	60
2.3.5 Axes principaux d'inertie	60
2.4 Conclusion	68

Après avoir décrit les différents codages de contours de formes existants, nous allons maintenant voir comment utiliser le code de Freeman pour obtenir des informations sur une chaîne de codants, puis sur le code du contour d'une forme fermée. Mise à part la publication initiale de Freeman, très peu d'études ont été réalisées sur ce thème et nous voyons ici un intérêt pour l'extraction de paramètres de formes.

Avant d'étudier les paramètres sur une forme que l'on considèrera comme un objet, voyons ce qu'il est possible d'étudier sur une chaîne de codants.

2.1 Paramètres et opérations sur une chaîne

2.1.1 Inverse d'une chaîne

L'inverse d'un codant a_i représentant le passage d'un point du contour de l'objet au point suivant est donné par le codant $(a_i)^{-1}$ défini par :

$$(a_i)^{-1} = (a_i + 4)[8] \quad (2.1)$$

[8] étant la notation usuelle de modulo 8. Ainsi, l'inverse du codant 3 est le codant 7 et l'inverse du codant 5 est le codant 1.

L'inverse d'une chaîne de codants $a_1a_2\dots a_i\dots a_n$ allant d'un point a à un point b est la chaîne de codants décrivant le chemin inverse pour aller de b à a de la façon suivante :

$$(a_1a_2\dots a_i\dots a_n)^{-1} = a_n^{-1}\dots a_1^{-1} \quad (2.2)$$

Calculer l'inverse d'une chaîne pour un contour complet équivaut à inverser le sens de parcours, passant ainsi du sens trigonométrique au sens des aiguilles d'une montre.

2.1.2 Longueur d'une chaîne

Freeman présente la longueur d'une chaîne L comme suit :

$$L = (nb_{0,2,4,6} + nb_{1,3,5,7}\sqrt{2}).l \quad (2.3)$$

avec l le pas de la trame, $nb_{0,2,4,6}$ et $nb_{1,3,5,7}$ les nombres de codants pairs et impairs. Cette notion de longueur de chaîne est exploitée pour donner une estimation du périmètre d'un objet connu par le code de Freeman de son contour (voir [2.2.1.1 page 31](#) ci-après) ou la longueur d'un arc de courbe (exemple du squelette ou de l'axe médian) ou d'une distance géodésique entre deux points d'un objet.

Ainsi, la longueur de la chaîne 32301061210 de la figure [2.3 page 30](#) est $L = 6 + 5\sqrt{2} l \approx 13,07 l$.

2.1.3 Largeur et hauteur d'une chaîne

La largeur et la hauteur d'une chaîne, c'est-à-dire les longueurs des projections de cette chaîne sur les axes, peuvent être obtenues de la façon suivante : soient x_i et y_i les positions absolues en x et en y atteintes au i ème codant de la chaîne :

$$x_i = \sum_{j=1}^i a_{jx} + x_0 \quad (2.4)$$

a_i	a_{ix}	y_{i-1}	y_i	a_{iy}	S_i	S
3	-1	0	1	1	-1/2	
2	0	1	2	1	0	
3	-1	2	3	1	-5/2	
0	1	3	3	0	3	
1	1	3	4	1	7/2	
0	1	4	4	0	4	
6	0	4	3	-1	0	
1	1	3	4	1	7/2	
2	0	4	5	1	0	
1	1	5	6	1	11/2	
0	1	6	6	0	6	45/2

FIGURE 2.1 – Calcul de l'aire contenue entre l'axe des x et la chaîne 32301061210.

$$y_i = \sum_{j=1}^i a_{jy} + y_0 \quad (2.5)$$

avec x_0 et y_0 les coordonnées absolues du premier point de la chaîne. Les notations a_{jx} et a_{jy} , présentées dans le tableau de la figure 1.20 page 16 donnent les déplacements relatifs en x et en y induits par les codants a_j .

La largeur w et la hauteur h sont données respectivement par :

$$w = \max_{j \in [1..n]} (x_j) - \min_{j \in [1..n]} (x_j) \quad (2.6)$$

$$h = \max_{j \in [1..n]} (y_j) - \min_{j \in [1..n]} (y_j) \quad (2.7)$$

Par exemple, pour la chaîne 32301061210 (figure 2.3 page 30) dont les coordonnées du point de départ seraient $(0, 0)$, on obtient ainsi $x_i = 6$ et $y_i = 6$, et par suite une largeur de $w = 6$ l et une hauteur de $h = 6$ l , (l étant, rappelons-le, le pas de la trame.)

2.1.4 Intégration par rapport à l'axe des x

Avec les orientations "classiques" des axes (axe des y dirigé vers le haut et axe des x dirigé vers la droite), l'aire S contenue entre une chaîne et l'axe des x est donnée par :

$$S = \sum_{i=1}^n a_{ix} \left(y_{i-1} + \frac{1}{2} a_{iy} \right) \quad (2.8)$$

avec $y_i = y_{i-1} + a_{iy}$ et y_0 l'ordonnée du point de départ de la chaîne. Cette aire est comptée positivement au dessus de l'axe des x (c'est-à-dire pour $y > 0$) et négativement en dessous (pour $y < 0$).

D'après le tableau de la figure 2.1, la chaîne 32301061210 avec $y_0 = 0$ donne une aire $S = 45/2$.

Pour une chaîne fermée décrivant le contour d'un objet, y_0 peut être choisi de manière arbitraire. Cette formule donnera l'aire délimitée par le polygone décrivant le contour de l'objet. Ainsi, un signe négatif donne un déplacement en y décroissant.

2.1.5 Moments par rapport à l'axe des abscisses

Le premier moment par rapport à l'axe des x est donné par :

$$M_{1x} = \sum_{i=1}^n \frac{a_{ix}}{2} [y_{i-1}^2 + a_{iy}(y_{i-1} + \frac{a_{iy}}{3})] \quad (2.9)$$

Le second moment par rapport à l'axe des x (moment d'inertie) est donné par :

$$M_{2x} = \sum_{i=1}^n \frac{a_{ix}}{3} [y_{i-1}^3 + \frac{3}{2}a_{ix}y_{i-1}^2 + a_{iy}^2y_{i-1} + \frac{1}{4}a_{iy}] \quad (2.10)$$

Les formules donnant les moments suivant l'axe des y et suivant les diagonales principales sont similaires et peuvent être déduites de [Fre61b].

Freeman s'est arrêté ici en 1974 sur l'étude des moments. Nous verrons par la suite au § 2.3.5 page 60 qu'il est possible grâce à ces moments d'inertie d'obtenir les axes principaux d'inertie du contour d'une forme à partir de son code de Freeman.

2.1.6 Résidus

2.1.6.1 Résidu d'une chaîne

Le résidu $\mathfrak{R}(A)$ d'une chaîne A est une chaîne de longueur minimale allant du point de départ de A au point final de A et ayant ses codants classés par ordre croissant.

Ce concept est intéressant car il donne le chemin le plus court pour aller du début à la fin de la chaîne. Ce chemin n'est pas unique et les autres chemins les plus courts sont obtenus en changeant l'ordre des codants du résidu.

Pour toute chaîne A , il existe un résidu du fait que celui-ci peut être construit.

Freeman [Fre74] donne un algorithme pour obtenir ce résidu :

- Déplacer le point initial de A à l'origine du système de coordonnées et calculer les coordonnées x' et y' du point d'arrivée dans ce nouveau repère :

$$\begin{aligned} x' &= x_n - x_0 = \sum_{i=1}^n a_{ix} \\ y' &= y_n - y_0 = \sum_{i=1}^n a_{iy} \end{aligned} \quad (2.11)$$

- Déterminer l'octant contenant le dernier point (x', y') .
- Si (x', y') appartient à l'octant défini par les angles des codants $m \times 45^\circ$ et $(m+1) \times 45^\circ$ pour $m \in [0, 7]$, alors $\mathfrak{R}(A)$ sera une concaténation des codants m et $m+1$ classés par ordre croissant. Les valeurs de m et le nombre d'occurrences des codants de valeur m et $m+1$ sont décrits figure 2.2 page ci-contre.

Signe			octant	m	nb de codants de valeur	
x'	y'	$ x' - y' $			m	$m+1$
+	+	+	I	0	$x'-y'$	y'
+	+	-	II	1	x'	$-x'+y'$
+	-	+	VIII	7	$-y'$	$x'+y'$
+	-	-	VII	6	$-x'-y'$	x'
-	+	+	IV	3	y'	$-x'+y'$
-	+	-	III	2	$x'+y'$	$-x'$
-	-	+	V	4	$-x'+y'$	$-y'$
-	-	-	VI	5	$-x'$	$x'-y'$

FIGURE 2.2 – Détermination du résidu d'une chaîne.

La Fig 2.3 page suivante présente l'obtention du résidu de la chaîne 3230106121022 pour laquelle on trouve $x' = 4$ et $y' = 6$. Ces valeurs étant toutes deux positives et $(|x'| - |y'|)$ étant également positif, (x', y') appartient à l'octant II, de plus, $m = 1$; le résidu est donc 111122.

2.1.6.2 Résidu d'une paire de codants

Le résidu de nombreuses paires de codants $a_r a_s, (r, s) \in [1..n]^2, r \neq s$ est un simple codant, une chaîne nulle ou une paire de codants pairs et identiques. Si nous considérons une chaîne et que nous remplaçons toutes les paires de codants non nécessairement adjacents par leur résidu, nous réduisons alors la longueur de la chaîne sans modifier les points de départ et d'arrivée.

Une des applications est la suivante : si toutes les réductions possibles sont effectuées, on obtient, après réarrangement par ordre croissant le résidu de la chaîne complète. De plus, cette réduction par paires judicieusement choisies permettra un lissage de la courbe décrite par une chaîne de codants.

La figure 2.4 page suivante présente l'ensemble des résidus de paires de codants.

2.1.7 Distance entre deux points

La distance entre deux points a et b d'une chaîne A est donnée par l'une des 2 équations suivantes :

$$d(a, b) = \sqrt{\left(\sum_{i=1}^n a_{ix}\right)^2 + \left(\sum_{i=1}^n a_{iy}\right)^2} \quad (2.12)$$

$$d(a, b) = l \times \sqrt{(r_e + r_0)^2 + (r_0)^2} \quad (2.13)$$

avec r_e et r_0 le nombre de codants pairs et impairs de la chaîne résidu $\mathfrak{R}(A)$.

Considérant la chaîne 32301061210, en utilisant l'équation 2.13 pour les points extrémaux a et b , il vient : $d(a, b) = l \times \sqrt{(4+2)^2 + (4)^2} \approx 7,21 l$.

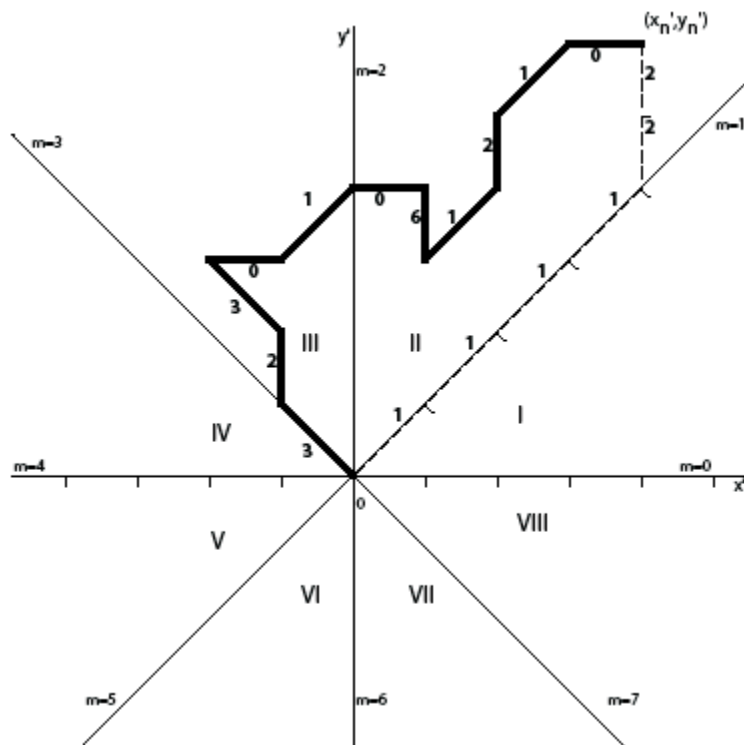


FIGURE 2.3 – Détermination du résidu de la chaîne 3230106121022.

		a_s							
		0	1	2	3	4	5	6	7
a_r	0			1	2	X	6	7	
	1				22	2	X	0	00
	2	1				3	4	X	0
	3	2	22				44	4	X
	4	X	2	3				5	6
	5	6	X	4	44				66
	6	7	0	X	4	5			
	7		00	0	X	6	66		

FIGURE 2.4 – Détermination du résidu d'une paire de codants.

2.1.8 Miroir d'une chaîne

Freeman [Fre74] présente l'obtention du miroir d'un codant a_i par rapport aux quatre axes principaux : l'axe horizontal (h) des x , l'axe vertical (v) des y et les axes diagonaux (d) et antidiagonaux (a).

$$\begin{aligned} a_i^h &= (8 - a_i)[8] \\ a_i^a &= (6 - a_i)[8] \\ a_i^v &= (4 - a_i)[8] \\ a_i^d &= (2 - a_i)[8] \end{aligned} \quad (2.14)$$

Par suite, le miroir A^u d'une chaîne A selon l'axe u (u représentant l'un des quatre axes h, v, d ou a) est donné par :

$$A^u = (a_1 a_2 \dots a_n)^u = a_1^u a_2^u \dots a_n^u \quad (2.15)$$

avec :

$$(A^u)^u = A \quad (2.16)$$

Par exemple, la chaîne 32301061210 a pour miroir selon la droite $y = x$ (dans le repère image présenté figure 1.15 page 13) la chaîne 34365605456.

2.2 Paramètres géométriques

Par paramètre géométrique, nous entendons ici une grandeur mesurable sur une forme et ayant (ou étant associée à) une unité comme une longueur, une aire ou les coordonnées d'un pixel particulier.

2.2.1 Périmètre

2.2.1.1 Méthode Directe

Tout d'abord, le périmètre interne, au sens du voisinage V_4 (voir figure 1.13 page 12), de l'objet A peut être estimé de la façon suivante (voir § 2.1.2 page 26) :

$$P(A) = (nb_{0,2,4,6} + nb_{1,3,5,7}\sqrt{2}).l \quad (2.17)$$

avec $nb_{0,2,4,6}$ et $nb_{1,3,5,7}$ le nombre de codants pairs d'une part et impairs d'autre part dans le code, et l le pas de la trame.

On suppose ici que les pixels sont carrés, c'est-à-dire que le pas en x est le même que le pas en y . Si ce n'est pas le cas, il faut dissocier les codants 0 et 4 (déplacement horizontal) des codants 2 et 6 (déplacement vertical). On obtient alors :

$$P(A) = nb_{0,4}.l_y + n_{2,6}.l_x + n_{1,3,5,7}\sqrt{l_x^2 + l_y^2} \quad (2.18)$$

avec l_x le pas de la trame en x et l_y celui en y .

Ainsi, sur une trame carrée, le périmètre interne de l'objet A présenté figure 1.16 page 14 vaut $P(A) = 26 + 10\sqrt{2} \approx 40,14 l$.

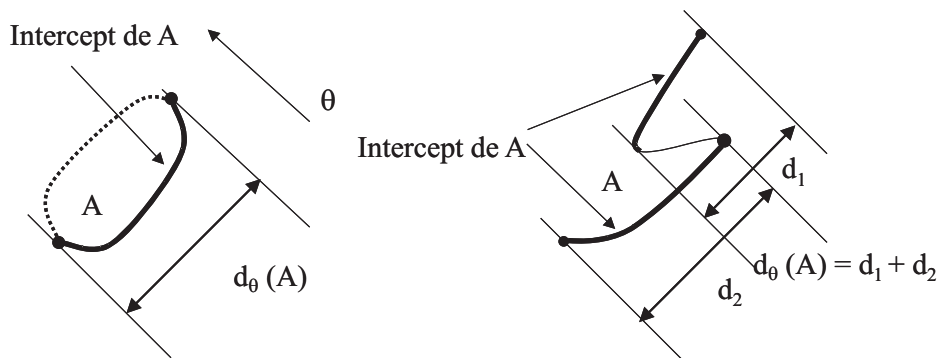


FIGURE 2.5 – Notion d'intercepts.

Il est également possible d'estimer le périmètre externe de l'objet étudié en procédant de même sur le code de son dilaté (voir chapitre 4 page 93).

Possédant les périmètres interne et externe, une moyenne des deux peut donner une estimation du périmètre de l'objet avant numérisation. Cependant, cette idée est à utiliser avec précaution car un périmètre externe peut gommer les zones rentrantes.

2.2.1.2 Périmètre de Crofton

Pour un objet A , convexe ou non convexe, le périmètre de Crofton $P_C(A)$ est défini comme suit :

$$P_C(A) = \pi \times D_{moy}(A) l \quad (2.19)$$

avec $D_{moy}(A)$ le diamètre apparent moyen de A . Le diamètre $D_\theta(A)$ de A est calculé dans chaque direction θ du plan comme la longueur, projetée perpendiculairement à θ , de l'intercept de A (partie de la frontière de A vue dans la direction θ , voir figure 2.5).

Pour les convexes, le périmètre de Crofton peut donc être obtenu grâce à l'estimation des diamètres apparents de l'objet A qui sont alors identiques aux intercepts .

Or, au § 2.2.5 page 38, on obtient l'ensemble des diamètres apparents de A sur l'intervalle $[0, \pi[$. L'estimation du périmètre est donc calculée comme suit en projetant l'objet sur un axe formant un angle θ avec l'axe horizontal :

$$P_C(A) = \frac{\pi}{180} \sum_{\theta=1}^{180} D_\theta(A) l \quad (2.20)$$

Il est bien entendu possible de faire varier θ avec un pas bien différent. Plus le pas sera petit, plus la précision sera importante, mais plus les temps de calcul seront longs.

Si la forme étudiée A n'est pas convexe, on utilisera toujours la formule 2.19, en ne considérant plus les diamètres apparents mais bien les intercepts (voir figure 2.5 et [Cos]).

Avec cette méthode on obtient $P_C(A) \approx 42,88 l$ pour l'objet de la figure 1.16 page 14.

2.2.2 Aire

Le code de Freeman permet également d'estimer l'aire de l'objet que l'on étudie.

2.2.2.1 Utilisation du produit vectoriel

Une première méthode pour obtenir l'aire d'un objet connu par le code de Freeman de son contour utilise le produit vectoriel entre deux vecteurs dont l'origine est un point quelconque P de l'image et les extrémités M et N sont deux pixels consécutifs codés par le code de Freeman. (Un exemple est illustré sur la figure 2.6.)

En effet, le produit vectoriel $\langle \vec{PM}, \vec{PN} \rangle$ de \vec{PM} et \vec{PN} est lié à $\mu(PMN)$ l'aire du triangle PMN par la relation suivante :

$$\langle \vec{PM}, \vec{PN} \rangle = 2 \cdot \mu(PMN) \quad (2.21)$$

Considérons M comme étant le point de départ et N le point d'arrivée d'un codant, en sommant les produits vectoriels pour chacun des codants décrivant le contour d'un objet, on obtient le double de l'aire comprise à l'intérieur du polygone décrivant la frontière interne de l'objet.

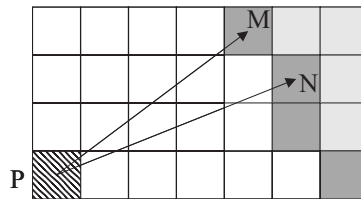


FIGURE 2.6 – Triangle formé par un point quelconque de l'image et deux pixels de la frontière d'un objet liés par un codant.

2.2.2.2 Méthode de Ballard et Brown

Dans le livre de Ballard et Brown [Bal] se trouve un algorithme permettant de calculer l'aire contenue dans une courbe décrite par une chaîne de codes à quatre directions.

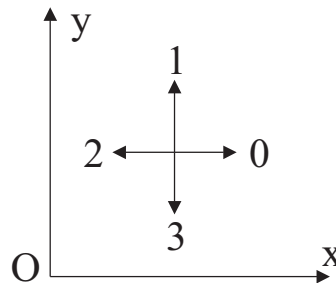


FIGURE 2.7 – Les quatre directions utilisées par Ballard et Brown.

Pour un code utilisant les quatre directions illustrées figure 2.7 (0 : +x, 1 : +y, 2 : -x, 3 : -y) encerclant une région dans le sens horaire, avec comme point de départ (x, y) , l'algorithme 3 page suivante donne l'aire de cette région.

Algorithme 3 Calcul de l'aire par Ballard et Brown

```

1  aire = 0
2  yposition = y
3  pour i = 1 to n faire
4      case direction :
5          [0] aire = aire - yposition
6          [1] yposition = yposition + 1
7          [2] aire = aire + yposition
8          [3] yposition = yposition - 1
9      end case
10 fin pour
11 retourner aire

```

Cet algorithme donne donc l'aire comprise à l'intérieur de la courbe joignant le centre des pixels décrits par le code.

Sur la figure 2.8, le code 4 directions est le suivant : 1110101030333032212232.

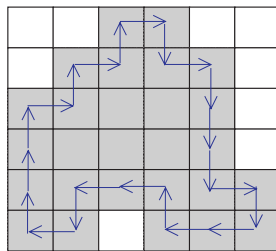


FIGURE 2.8 – Objet codé en utilisant quatre directions.

L'aire donnée par le déroulement de l'algorithme ci-dessus est donc 15 pixels. Ceci correspond bien à l'aire englobée par le polygone décrit par le codage de Freeman à quatre directions du contour de l'objet. On ne compte donc pas les parties des pixels appartenant à l'objet mais étant à l'extérieur de ce polygone.

Cette méthode peut cependant être utilisée pour calculer l'aire totale d'un objet dont on connaît le code de contour codé selon huit directions. Ces huit codants possibles vont du centre d'un pixel au centre du suivant.

Il est possible à partir de ce code d'obtenir le code à quatre directions utilisant des codants allant d'un sommet du pixel à un autre. Ainsi, on ne coderait plus un polygone contenu dans l'objet de départ, mais le plus petit polygone contenant cet objet de départ.

2.2.2.3 Amélioration de la méthode de Ballard et Brown

Nous avons généralisé cet algorithme à un codage de Freeman à huit directions, et nous l'avons adapté à la représentation sous Matlab.

Pour un code utilisant les huit directions représentées figure 2.9 page suivante (0 : +y, 1 : +y -x, 2 : -x, 3 : -y -x, 4 : -y, 5 : +x -y, 6 : +x, 7 : +y +x) et encerclant une région dans

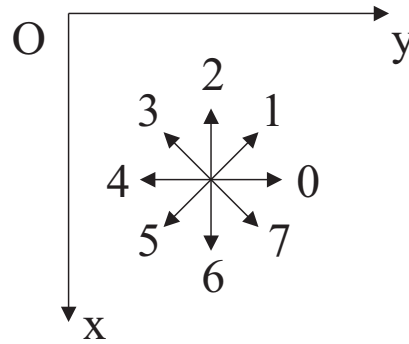


FIGURE 2.9 – Huit directions du code de Freeman.

le sens trigonométrique, avec comme point de départ (x, y) , l'algorithme 4 donne l'aire de cette région.

Algorithme 4 Calcul de l'aire, généralisation à huit directions

```

1:  $aire = 0$ 
2:  $x_{position} = x$ 
3: pour  $i = 1$  to  $n$  faire
4:     case direction :
5:         [0]  $aire = aire + x_{position}$ 
6:         [1]  $aire = aire + x_{position} - 1/2$ 
7:              $x_{position} = x_{position} - 1$ 
8:         [2]  $x_{position} = x_{position} - 1$ 
9:         [3]  $aire = aire - x + 1/2$ 
10:             $x_{position} = x_{position} - 1$ 
11:        [4]  $aire = aire - x_{position}$ 
12:        [5]  $aire = aire - x_{position} - 1/2$ 
13:             $x_{position} = x_{position} + 1$ 
14:        [6]  $x_{position} = x_{position} + 1$ 
15:        [7]  $aire = aire + x_{position} + 1/2$ 
16:             $x_{position} = x_{position} + 1$ 
17:    end case
18: fin pour
19: retourner  $aire$ 

```

Ici encore, cet algorithme fournit l'aire du polygone formé par le code de Freeman de la frontière interne de l'objet.

2.2.2.4 Correctif pour obtenir l'aire totale

Pour obtenir l'aire réelle de l'objet en terme de nombre de pixels allumés, il faut faire une petite rectification (voir figure 2.10) : l'aire obtenue par l'algorithme précédent nous donne en effet l'aire de l'objet comprise à l'intérieur de la courbe décrite par les codants de Freeman. On considère alors que cet objet est un polygone et que l'objet réel est ce polygone objet dilaté par une boule de taille un pixel.

Or, dans le cas continu, pour des polygones convexes, l'aire μ du polygone dilaté est donnée par la formule suivante :

$$\mu(P \oplus B_a) = \mu(P) + p(P) + \mu(B_a) \quad (2.22)$$

où $P \oplus B_a$ représente le dilaté de P par B_a (Voir chapitre 4 page 93).

Dans notre cas, $\mu(P)$ est donné par un des algorithmes précédents, $\mu(B_a)$ vaut un pixel car l'élément structurant de dilatation est ici un pixel, et la contribution du périmètre nous est donnée de la façon suivante : pour chaque codant, on ajoutera $\frac{1}{2}$ pixel.

L'aire réelle de l'objet est donc la suivante :

$$\text{Aire} = \text{aire}(\text{algorithme}) + \frac{Nb_{\text{codants}}}{2} + 1 \quad (2.23)$$

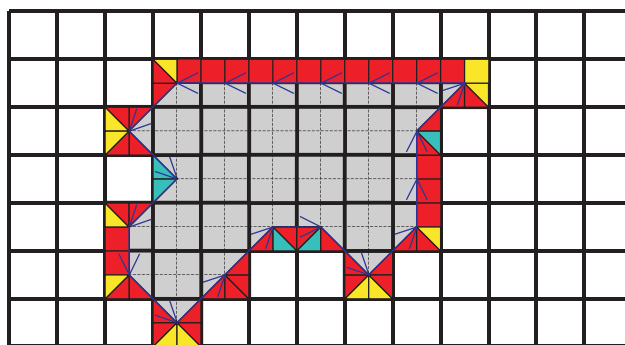


FIGURE 2.10 – Contribution de la couronne dans le calcul de l'aire.

Illustrons et vérifions ceci par l'exemple de la figure 2.10 :

- L'aire interne au polygone nous donne l'aire calculée par l'algorithme ci-dessus.
- L'aire ajoutée par chaque codant apparaît en rouge sur la couronne.
- En bleu sur cette même couronne est matérialisée l'aire que l'on a ajoutée 2 fois.
- L'aire à ajouter est en jaune.

Une petite vérification s'impose : l'aire en jaune moins l'aire en bleu nous donne exactement un pixel correspondant à l'aire de l'élément structurant de dilatation.

La formule (2.22) s'applique ainsi en discret pour des polygones quelconques, c'est-à-dire pas forcément convexes.

On remarquera ici que les zones localement concaves sont en bleu et que les zones localement convexes sont en jaune.

2.2.2.5 Autre méthode

Considérant le pixel $X_0(x_0, y_0)$ comme pixel de départ de la forme, son aire peut être estimée grâce à l'algorithme 5 page suivante que nous avons développé (nous présentons ici l'algorithme simplifié).

Algorithme 5 Calcul de l'aire, nouvelle méthode

```

1:  $aire = 0$ 
2: à partir du point  $X_0(x_0, y_0)$ , retrancher  $y_0$  à l'aire (correspondant au nombre de pixels à gauche de  $X_0$  sur la ligne  $x_0$ )
3: tant que l'on n'est pas revenu au point de départ : faire
4:   passer au point suivant par le code de Freeman du contour de l'objet
5:   cas général
6:     si x augmente, retrancher l'ancien y à l'aire
7:     si x diminue, ajouter le nouvel y à l'aire
8:   cas particulier
9:     si x augmente pour la première fois :
10:      retrancher l'ancien y et ajouter 1 à l'aire
11:      retrancher le nouvel y et ajouter 1 à l'aire
12:     si x diminue pour la première fois :
13:      ajouter l'ancien y à l'aire
14:      ajouter le nouvel y à l'aire
15: fin tant que
16: retourner  $aire$ 

```

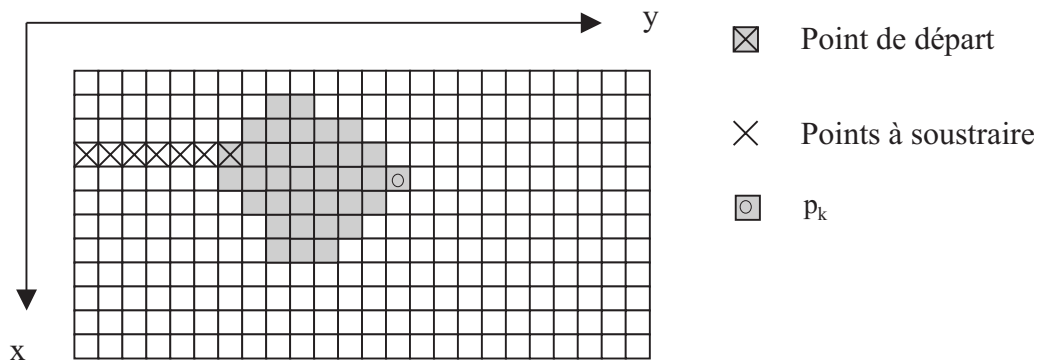


FIGURE 2.11 – Cas du premier point pour le calcul de l'aire.

La figure 2.11 présente le cas du premier point pour cet algorithme : étant sur le pixel de départ du code du contour de l'objet, on soustrait le nombre de pixels se trouvant à gauche : ici 6 pixels.

Lorsque l'algorithme arrive au pixel p_k de la frontière, situé le plus à droite sur la même ligne, il faudra ajouter 13 correspondant aux 13 pixels se situant à gauche de p_k , y compris p_k .

Ainsi, concernant cette ligne nous aurons un bilan positif de 7 correspondant aux 7 pixels allumés.

L'algorithme fait donc ce travail sur tout le contour de l'objet et sur toutes les lignes où l'objet est présent.

Un algorithme plus complexe prenant en compte plusieurs cas particuliers des objets étudiés a été implémenté. Ces cas particuliers sont les suivants :

- retour en arrière (par exemple un codant 4 suivi d'un codant 0, ou inversement, et dans les quatre directions)
- passage par un plateau en x croissant
- passage par un plateau en x décroissant
- cas d'un objet horizontal d'épaisseur 1 pixel

2.2.3 Centre de masse

Les coordonnées du centre de masse de la frontière d'un objet sont facilement calculables à partir du code de Freeman du contour de l'objet. Connaissant x_0 et y_0 les coordonnées du point de départ du code et en considérant l'orientation du plan selon la figure 1.15 page 13, les coordonnées du point joint par le premier codant sont calculables.

Ainsi, de proche en proche sont calculées les coordonnées de tous les points du contour. La moyenne en x et la moyenne en y nous donnent respectivement l'abscisse et l'ordonnée du centre de gravité de la frontière de l'objet.

Il est important de noter ici que dans la plupart des cas, ce point n'est pas le centre de masse de l'objet plein.

Connaissant les coordonnées du centre de masse du contour de l'objet, les rayons maximum, minimum et moyen de cet objet sont alors calculables.

2.2.4 Rayon maximum, minimum et moyen

Disposant des coordonnées (x_G, y_G) de G le centre de masse de la frontière de l'objet et des coordonnées de tous les points de la frontière, en calculant les distances au sens de Pythagore (ou en première approximation grâce à l'algorithme de Borgfors [Bor]) entre G et chaque point du contour $P_i, i \in [1, n]$, il vient :

$$\begin{aligned} r_{max} &= \max_{i \in [1, n]} (d(G, P_i)) \\ r_{min} &= \min_{i \in [1, n]} (d(G, P_i)) \\ r_{moy} &= \frac{1}{N} \sum_{i \in [1, n]} (d(G, P_i)) \end{aligned} \tag{2.24}$$

où r_{max} , r_{min} et r_{moy} désignent respectivement les rayons minimum, maximum et moyen.

L'objet illustré sur la figure 1.15 page 13 donne les résultats suivants : $r_{max}(A) \approx 7,6 l$, $r_{min}(A) \approx 3,6 l$ et $r_{moy}(A) \approx 5,8 l$.

2.2.5 Diamètre apparent maximal, minimal et moyen

Les diamètres apparents maximal $D_{max}(A)$, minimal $D_{min}(A)$ et moyen $D_{moy}(A)$ d'un objet A convexe (voir § 2.3.3 page 59) sont obtenus grâce à l'algorithme 6 page ci-contre.

La valeur du diamètre apparent maximal (respectivement minimal, moyen) est obtenue en trouvant le maximum (le minimum, la moyenne) du tableau $D(A, \theta)$:

Algorithme 6 Calcul des diamètres apparents

```

1: Extraction du code de Freeman du contour d'un objet
2: pour  $\theta$  allant de  $0^\circ$  à  $179^\circ$  par pas de  $1^\circ$  faire
3:   Projection des huit directions du code de Freeman suivant l'orientation  $\theta$ 
4:   Initialisation d'une variable temporaire :  $tmp = 0$ 
5:   pour chaque codant  $j$  faire
6:      $tmp = tmp + projection(codant)$ 
7:   Mise en mémoire de  $tmp(j)$ 
8:   fin pour
9:    $D(A, \theta) = \max(tmp) - \min(tmp) + \sqrt{2} * \cos(45 - mod(i, 90))$ 
10: fin pour

```

$$\begin{aligned}
D_{max}(A) &= \max_{\theta}(D(A, \theta)) \\
D_{min}(A) &= \min_{\theta}(D(A, \theta)) \\
D_{moy}(A) &= \frac{1}{Card(\theta)} \sum_{\theta} D(A, \theta)
\end{aligned}
\tag{2.25}$$

Le segment joignant les points les plus éloignés d'un objet convexe est la corde de longueur maximale. Le diamètre apparent maximal ne fait que mesurer la longueur de ce segment.

L'objet illustré figure 1.15 page 13 donne les résultats suivants : $D_{max}(A) \approx 15,81 l$, $D_{min}(A) \approx 10 l$ et $D_{moy}(A) \approx 13,65 l$.

La méthode utilisée pour obtenir ce paramètre est intéressante car elle permet d'obtenir d'autres données sur l'objet étudié, comme le rectangle englobant d'aire minimale (§ 2.2.6 page suivante), ou le périmètre de Crofton (§ 2.2.1.2 page 32).

La figure 2.12 montre dans le cas d'un triangle droit et isocèle la courbe des diamètres apparents sur les 360° . On remarquera qu'une étude sur $[0, 180^\circ]$ est suffisante.

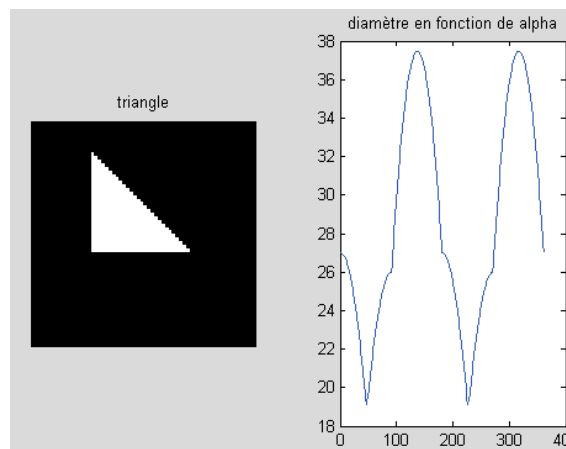


FIGURE 2.12 – Calcul des diamètres apparents dans le cas d'un triangle, en fonction de l'angle de projection (en degré).

2.2.6 Rectangle englobant d'aire minimale

2.2.6.1 Méthode de H. Freeman

H. Freeman [Fre75] propose en 1975 un moyen de déterminer le rectangle d'aire minimale contenant une courbe fermée de type contour d'un objet.

La première étape consiste en la recherche du plus petit rectangle contenant la courbe et dont les côtés sont parallèles aux axes x et y . Pour cela, il faut calculer x_i et y_i suivant les équations 2.4 page 26 et 2.5 page 27 et en extraire les minima et maxima :

$$\begin{aligned} W_{min} &= \min_{i \in [1, n]} (x_i) \\ W_{max} &= \max_{i \in [1, n]} (x_i) \\ H_{min} &= \min_{i \in [1, n]} (y_i) \\ H_{max} &= \max_{i \in [1, n]} (y_i) \end{aligned} \quad (2.26)$$

Le rectangle formé par les droites d'équations $y = H_{min}$, $y = H_{max}$, $x = W_{min}$ et $x = W_{max}$ entoure la courbe, mais en général, l'aire de ce rectangle n'est pas minimale.

L'étape suivante consiste à constituer le polygone P_0 convexe dont les sommets sont ceux se situant sur l'une des quatre droites mentionnées ci-dessus.

Ensuite, chaque segment de ce convexe est subdivisé itérativement pour passer par les points de la courbe les plus éloignés se trouvant à l'extérieur du polygone. On obtient ainsi l'enveloppe convexe de la courbe de départ.

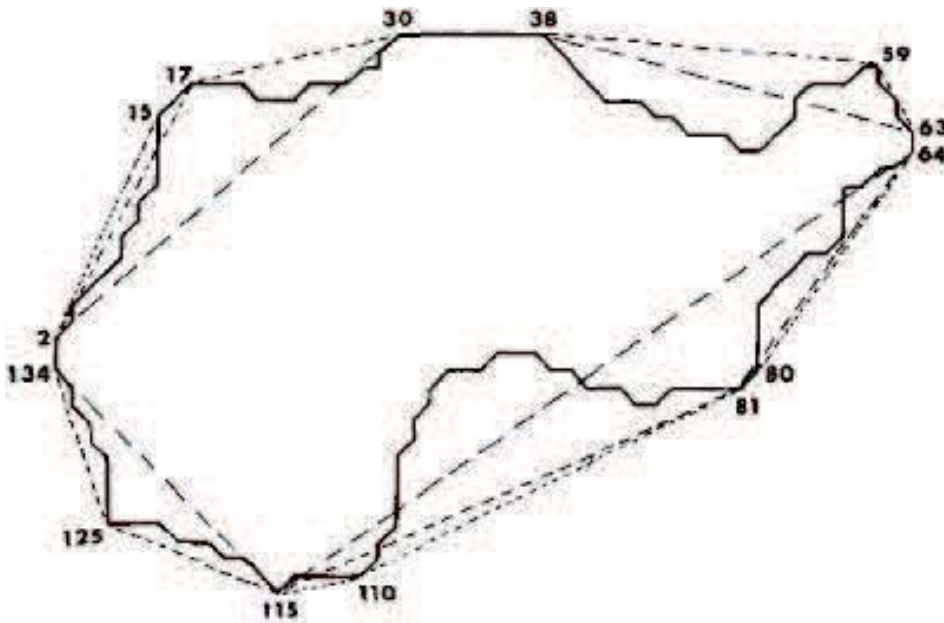


FIGURE 2.13 – Enveloppe convexe d'une courbe obtenue par H. Freeman.

La figure 2.13 montre une forme étudiée par H. Freeman [Fre75] après cette étape. Les pointillés plus longs montrent le polygone initial P_0 , et l'enveloppe convexe est en pointillés.

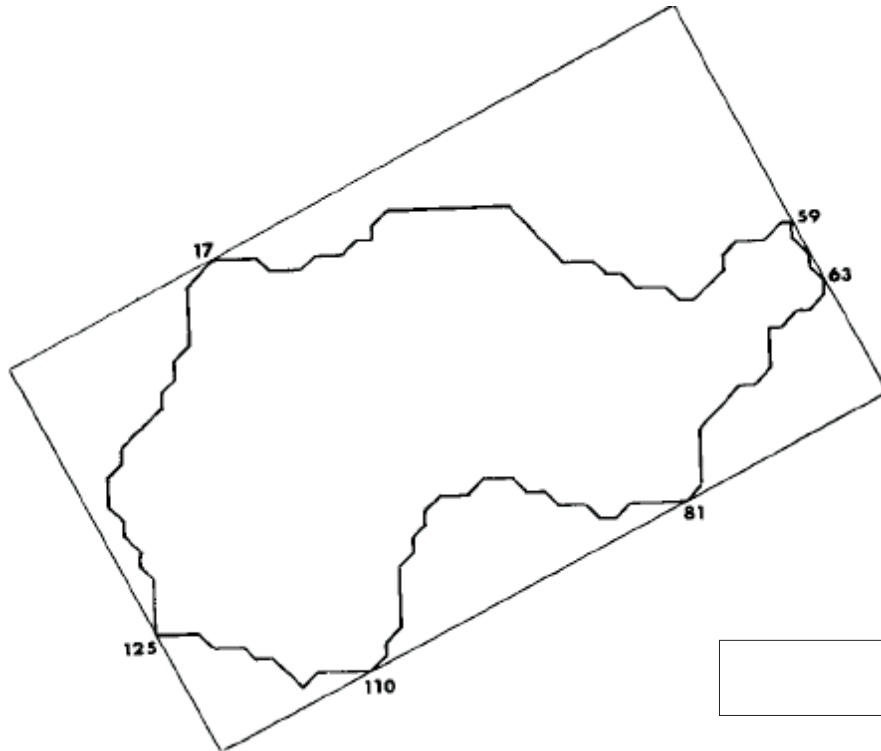


FIGURE 2.14 – Rectangle englobant d’aire minimale obtenu par H. Freeman.

courts. Les différentes tailles de pointillés montrent les étapes intermédiaires : plus les pointillés sont courts, plus il y a eu d’étapes. Les nombres correspondent aux indices des codants de la courbe qui n’est pas codée dans le sens trigonométrique utilisé jusqu’ici, mais cela n’a aucune incidence sur le résultat.

La dernière étape, qui consiste à trouver le rectangle d’aire minimale englobant ce polygone, nécessite trois théorèmes qu’H. Freeman utilise et démontre :

Théorème 1 : Soit un rectangle avec quatre points choisis arbitrairement [dans le plan] et tel qu’aucun côté ne contienne plus d’un [de ces quatre] point[s], il existe un autre rectangle dont chaque côté contient un seul et unique de ces points et tel que l’aire est inférieure à l’aire du rectangle de départ.

Théorème 2 : Le rectangle d’aire minimale entourant un polygone convexe a un côté colinéaire avec un des côtés du polygone.

Théorème 3 : Le rectangle d’aire minimale englobant l’enveloppe convexe d’une chaîne de codants simple fermée est le même que celui englobant la courbe.

A partir de ces trois théorèmes, il suffit de calculer l’aire des rectangles ayant un côté commun avec le polygone donnant l’enveloppe convexe, et de retenir le plus petit en terme de surface. La figure 2.14 présente un exemple de résultat obtenu par H. Freeman.

2.2.6.2 Nouvelle Méthode

Notons $R_{min}(A, \theta)$ le rectangle d’aire minimale englobant un objet A , et dont une des directions est θ .

L'aire de ce rectangle $\mu(R_{min}(A, \theta))$ peut être obtenue de la façon suivante à partir de la liste $D(A, \theta)$ des diamètres apparents obtenue au § 2.2.5 page 38 : il suffit de choisir le rectangle pour lequel le produit de deux diamètres apparents pris à 90° d'intervalle est minimal, selon l'équation :

$$\mu(R_{min}(A, \theta)) = \min_{\theta \in [0, 180[} (D(A, \theta) \times D(A, \theta + 90)) \quad (2.27)$$

L'orientation du rectangle englobant d'aire minimale est donc connue grâce à θ .

Cette méthode semble algorithmiquement plus simple que celle proposée par Freeman, mais elle nécessite au préalable de calculer les diamètres apparents de l'objet.

2.2.7 Dimension fractale

2.2.7.1 Historique

Il n'existe pas de définition unique et acceptée de tous pour la dimension fractale. Plusieurs d'entre elles sont présentées dans [Man]. Une comparaison souvent utilisée pour comprendre ce concept est la mesure de la longueur de la côte de Bretagne par [Rich], qui donne un résultat surprenant : cette longueur dépend de l'unité avec laquelle on la mesure. En effet, plus cette unité de mesure est petite, plus la mesure est grande. La raison est la suivante : plus on regarde de près la côte Bretonne, plus elle apparaît irrégulière et plus on doit mesurer avec une petite règle pour faire le tour des aspérités en les collant le mieux possible.

Quelques-unes des méthodes d'estimation de la longueur d'une courbe sont présentées ici : la méthode du compas, la méthode des boîtes, celle des intercepts et enfin la saucisse de Minkovski.

Méthode du compas

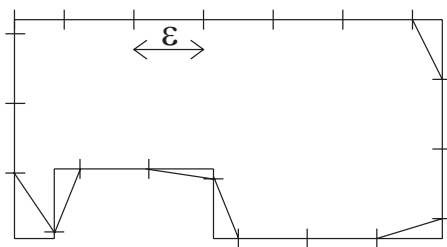


FIGURE 2.15 – Méthode du compas : $18\epsilon \leq P_\epsilon \leq 19\epsilon$

La méthode du compas, présentée par [Tri] consiste à fixer l'ouverture ϵ d'un compas. On appelle cette ouverture le pas et on mesure le nombre de fois où ce pas peut être porté sur la courbe. On obtient ainsi une estimation de la longueur de la courbe :

$$P_\epsilon = \epsilon \times N_\epsilon \quad (2.28)$$

Il faut ensuite considérer un pas deux fois plus petit et refaire la mesure. Dans le cas d'une



FIGURE 2.18 – Saucisse de Minkowski.

$$P_\epsilon = 4 \times N_\epsilon \times \epsilon \quad (2.31)$$

Cette méthode pour calculer le périmètre d'un objet est connue sous le nom de méthode des intercepts (voir [Tri]) et illustrée sur la figure 2.17 page précédente.

Saucisse de Minkowski

La frontière de l'objet est ici épaissie à l'aide d'un pinceau de largeur $2 \times \epsilon$ représentant une dilatation avec une boule de rayon ϵ . L'aire de la frontière épaissie est liée au périmètre et l'on obtient une estimation de celui-ci selon l'équation :

$$P_\epsilon = A_\epsilon / 2\epsilon \quad (2.32)$$

Cette méthode appelée saucisse de Minkowski d'une courbe a été publiée par [Gou] et [Tri].

Après ce rappel des différents moyens d'obtenir la longueur d'une courbe, nous allons présenter quelques considérations sur la notion de dimension.

2.2.7.2 A propos de dimension

Dimension non entière

On pourrait penser que plus ϵ est petit, plus l'estimation du périmètre P_ϵ de l'objet étudié est bonne, mais en réalité, cette estimation diverge. La raison de cette divergence est due aux irrégularités qui apparaissent quand ϵ décroît.

Si l'on désire mesurer la longueur d'une côte, en utilisant une des méthodes décrites plus haut, plus ϵ sera petit, plus le calcul prendra en considération de nouvelles causes de rugosités comme des criques, des rochers, des pierres et des grains de sable, ce qui donnera une croissance infinie de la mesure de la longueur de la côte.

La croissance de l'estimation de la longueur des côtes a été étudiée par [Rich] qui a conclu que la croissance du périmètre était exponentielle et pouvait s'exprimer proportionnellement à ϵ^α où α peut être déterminé par la représentation graphique de P_ϵ en fonction de ϵ en utilisant une échelle $\log - \log$.

Par conséquent, la valeur du périmètre dépend bien de l'échelle. Ce phénomène surélève le problème de l'unité de mesure qui n'est généralement pas adaptée à l'objet mesuré. L'unité de mesure adéquate est celle qui va donner la même estimation quelle que soit l'échelle.

Cela signifie que $N_\epsilon \times \epsilon^{unit}$ diverge ou tend vers zéro.

$$\begin{cases} P_\epsilon = N_\epsilon \times \epsilon \\ P_\epsilon = P_1 \times \epsilon^\alpha \end{cases} \Leftrightarrow \begin{cases} N_\epsilon = P_1 \times \epsilon^{\alpha-1} \\ P_1 = N_\epsilon \times \epsilon^{1-\alpha} \end{cases} \quad (2.33)$$

De l'équation 2.33, on déduit que l'unité du périmètre n'est généralement pas entière et que sa valeur est $1 - \alpha$.

$1 - \alpha$ est la dimension fractale du périmètre de l'objet et est notée DF . Elle donne une idée de la complexité de l'objet observé :

- si $D_f \rightarrow 1$, la surface de l'objet est lisse et comparable à une courbe sans irrégularité de dimension topologique 1.
- si $D_f \rightarrow 2$, la surface de l'objet est très rugueuse et tend à remplir l'espace de dimension topologique 2 : le plan.

Dimension fractale

Les définitions mathématiques de la dimension fractale de Hausdorff-Besicovitch ou Hausdorff ont été développées par [Hau], [Bes33], [Bes34] and [Bes37]. Tout d'abord, le concept de dimension fractale d'une structure ne peut pas être défini sans l'existence d'une distance.

Ensuite, nous devons mentionner ici que dans ces définitions, il y a toujours un processus limite quand $\epsilon \rightarrow 0$. Pour le calcul de la dimension fractale, nous devons utiliser des longueurs finies : la précision du calcul dépend de la longueur ϵ relativement à la taille de l'objet.

Un objet composé de segments a une dimension topologique de 1, un autre objet composé d'éléments de surface a une dimension de 2. Une structure fractale a une dimension fractale strictement supérieure à sa dimension topologique.

Dimension de Hausdorff-Besicovitch

Une première méthode pour calculer la dimension fractale d'un objet est la suivante : l'objet est couvert de boîtes de mesure $\mu = \epsilon^{d(E)}$ où $d(E)$ est la dimension de l'objet. Lorsque l'on cherche $d(E)$, une des solutions consiste à prendre $\mu = \epsilon^\alpha$ comme unité de mesure.

Considérons l'exemple d'un carré de côté L pour lequel $d = 2$, couvert par des boîtes de taille ϵ . Le résultat est donné par $M = N\mu$ avec N le nombre de boîtes utilisées pour couvrir le carré. Par conséquent, nous avons $N = (L/\epsilon)^d$. Il vient alors $M = N\epsilon^\alpha = (L/\epsilon)^d \epsilon^\alpha = L^d \epsilon^{\alpha-d}$.

Pour $\alpha = 1$, $M \rightarrow \infty$ lorsque $\epsilon \rightarrow 0$. Cela signifie qu'un carré a une longueur infinie. Pour $\alpha = 3$, $M \rightarrow 0$ lorsque $\epsilon \rightarrow 0$. Cela signifie qu'un carré n'a pas de volume. Et finalement, pour $\alpha = 2$, $M \rightarrow L^2$ lorsque $\epsilon \rightarrow 0$. Cela signifie qu'un carré a une aire finie de dimension 2.

Il est possible d'utiliser cette méthode pour toutes les dimensions réelles α incluant les dimensions non entières, c'est-à-dire les dimensions fractales.

Cependant, il n'est généralement pas possible de couvrir un objet avec des boîtes identiques de taille ϵ ; c'est pourquoi on utilisera des boules B_i de diamètre $\delta(B_i) \leq \epsilon$. Ceci donne plus de flexibilité mais oblige μ à être la limite inférieure de la somme des mesures élémentaires.

Il faut enfin considérer la mesure de α -recouvrement $m_\alpha(A)$ d'un objet A introduite par [Hau] et [Bes33], et définie comme suit :

$$\begin{aligned} m_\alpha(A) &= \lim_{\epsilon \rightarrow 0} m_\alpha^\epsilon(A) \\ m_\alpha^\epsilon(A) &= \inf(\sum (\delta B_i)^\alpha : \bigcup_i (B_i) \supset A, \delta(B_i) \leq \epsilon) \end{aligned}$$

Ensuite, la dimension de Hausdorff (ou Hausdorff-Besicovitch) $dim_H(A)$ d'un objet A est l'unique valeur telle que :

$$\begin{cases} m_\alpha(A) = \infty & \text{if } 0 \leq \alpha < dim_H(A) \\ m_\alpha(A) = 0 & \text{if } dim_H(A) < \alpha < \infty \end{cases} \quad (2.34)$$

$dim_H(A)$ est donc la valeur limite pour laquelle la mesure de α -recouvrement fait un saut de zéro à l'infini.

Cette définition ne peut clairement pas aboutir à une méthode pratique, c'est pourquoi d'autres moyens de calcul ont été développés.

Dimension de Bouligand-Minkowski

La dimension de Bouligand-Minkowski peut être calculée à partir de la saucisse de Minkowski, de la méthode des boîtes, de la méthode des boules disjointes ou de la méthode du compas ([Gou]).

Méthode de la saucisse de Minkowski

Comme expliqué plus haut, la saucisse de Minkowski d'un objet A est l'ensemble de points situés à une distance inférieure à ϵ de l'objet considéré. Soit μ_ϵ l'aire de cette saucisse, la dimension de Bouligand-Minkowski $dim_B(A)$ est définie par [Bou28], [Bou29] and [Min01] comme :

$$dim_B(A) = \lim_{\epsilon \rightarrow 0} \frac{\log(\frac{\mu_\epsilon}{\epsilon^2})}{\log(\epsilon)} \quad (2.35)$$

La dimension fractale de Bouligand-Minkowski $dim_{BM}(A)$ peut ensuite être déduite de la pente de la représentation graphique de $\log(\frac{\mu_\epsilon}{\epsilon^2})$ en fonction de $\log(\epsilon)$, comme présenté sur la figure 2.19.

Méthode des boîtes

L'objet est couvert par une grille contenant k^2 boîtes carrées contiguës de taille $\epsilon = 1/k$. Soit $N(\epsilon)$ le nombre de boîtes interceptant l'objet. La dimension de la méthode des boîtes $dim_B(A)$ d'un objet A est donnée par :

$$dim_B(A) = \lim_{\epsilon \rightarrow 0} \frac{\log N(\epsilon)}{\log(\epsilon)} \quad (2.36)$$

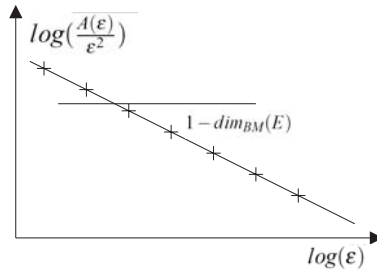


FIGURE 2.19 – Estimation de la dimension de Bouligand-Minkowski.

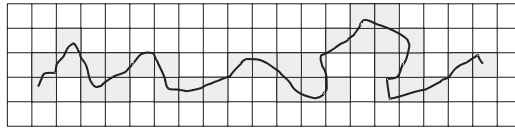


FIGURE 2.20 – Boîtes couvrant un objet.

Méthode des boules disjointes

Soit $N(\epsilon)$ le nombre maximum de boules disjointes de rayon ϵ centrées sur l'objet A . L'estimation de la dimension de Bouligand-Minkowski calculée par la méthode des boules disjointes (voir figure 2.21) est donnée par :

$$\dim_B(A) = \lim_{\epsilon \rightarrow 0} \frac{\log N(\epsilon)}{\log(\epsilon)} \quad (2.37)$$

Méthode du compas

La longueur de l'objet est mesurée avec un compas à différents écartements ϵ . Soit $N(\epsilon)$ le nombre de pas pour couvrir entièrement l'objet, la dimension fractale est donnée par :

$$\dim_B(A) = \lim_{\epsilon \rightarrow 0} \frac{\log N(\epsilon)}{\log(\epsilon)} \quad (2.38)$$

Il a été prouvé par [Fal] que toutes ces méthodes donnent le même résultat pour la dimension fractale des objets étudiés.

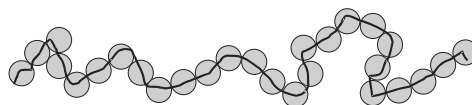


FIGURE 2.21 – Boules disjointes couvrant un objet.

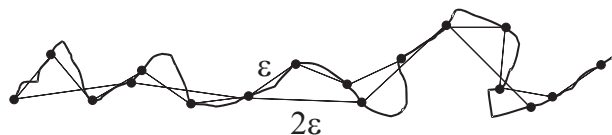


FIGURE 2.22 – Deux différents écartements pour mesurer la longueur d’un objet.

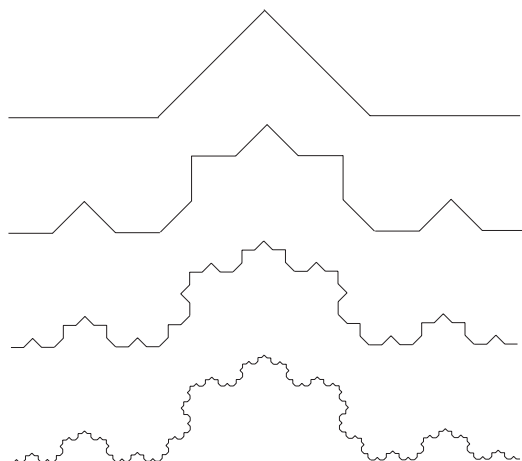


FIGURE 2.23 – Courbe de Von Koch à différentes échelles.

2.2.7.3 Applications

Habituellement, la dimension fractale d’un objet est calculée directement par une mesure sur cet objet. La méthode utilisée ici est inversée dans le sens où l’on construit d’abord le code de Freeman de la courbe et on mesure ensuite sa dimension fractale.

Le code de Freeman est utilisé ici et l’intérêt majeur est que l’on n’a pas besoin de représentation graphique de la courbe pour mesurer sa dimension fractale. Cependant, les courbes de Von Koch et de Peano sont illustrées comme résultat de la construction obtenue grâce au code de Freeman.

Première application : la courbe de Von Koch

Considérons un segment composé de trois parties égales, la courbe de Von Koch (voir figure 2.23) est obtenue en remplaçant la seconde partie par les 2 autres parties d’un triangle équilatéral et en itérant le processus sur chacun des segments obtenus. La courbe obtenue a donc des angles de 60° .

Objectifs :

Grâce au code de Freeman, nous avons étudié le cas de la courbe de Von Koch pour un angle $\alpha = 45^\circ$. La dimension fractale théorique DF_{Koch} est donnée par :

$$DF_{Koch} = \frac{\log(2)}{\log(2 \times \cos(\frac{\alpha}{2}))} \quad (2.39)$$

Avec un angle $\alpha = 45^\circ$, cela donne $DF_{Koch} \approx 1,128$.

Génération de la courbe de Von Koch :

Tout d'abord nous avons généré la courbe de Von Koch dont l'élément de base est codé par la chaîne suivante : 0000000 11111 77777 0000000. En effet, tous les segments doivent avoir la même longueur.

Or les segments codés par 0000000 ont une longueur de $7l$ et ceux codés par 11111 et 77777 ont une longueur de $5\sqrt{2}l$ soit environ $7,07l$.

Le générateur étant créé, il nous reste à remplacer chaque segment par ce générateur et à réitérer le nombre n de fois désiré. L'algorithme 7 donne la construction de la courbe de Von Koch.

Algorithme 7 Construction du flocon de Von Koch

```

1  générateur = [0 0 0 0 0 0 0 1 1 1 1 1 7 7 7 7 7 0 0 0 0 0 0]
2  pour i allant de 1 à n faire
3      concaténation de
4          générateur
5          (générateur + 1) [modulo 8]
6          (générateur - 1) [modulo 8]
7          générateur
8  fin pour

```

Il faut ensuite reprendre chaque segment et modifier sa longueur afin que celle-ci soit de $7l$ pour les segments horizontaux et verticaux, et $5\sqrt{2}l$ pour les segments diagonaux. Ceci se fait simplement en testant le nombre de codants égaux consécutifs et leur parité.

Il nous reste finalement à calculer la dimension fractale de cette courbe. Pour cela nous utilisons la méthode du compas selon l'algorithme 8.

Algorithme 8 Calcul de la dimension fractale par la méthode du compas

```

1  pour p allant de 1 à 64 par puissance de 2 faire
2      pas du compas = p
3       $N(p)$  = mesure de la longueur de la courbe en nombre de pas de compas
4  fin pour
5      Tracé de la courbe  $\log(N(p))$  en fonction de  $\log(1/p)$ 
6      Calcul de l'équation de la meilleure droite passant par les points obtenus, par la
        méthode des moindres carrés
7      La pente de cette droite donne une approximation de la dimension de Hausdorff-
        Besicovitch recherchée

```

La mesure de la longueur de la courbe (seconde étape de l'algorithme) est faite en calculant la distance géométrique entre le premier pixel de la frontière codée et les pixels suivants. Lorsque cette distance dépasse (ou égale) le pas du compas, nous incrémentons N et nous recommençons le processus en considérant comme premier pixel celui pour lequel le pas du compas a été dépassé (ou égalé).

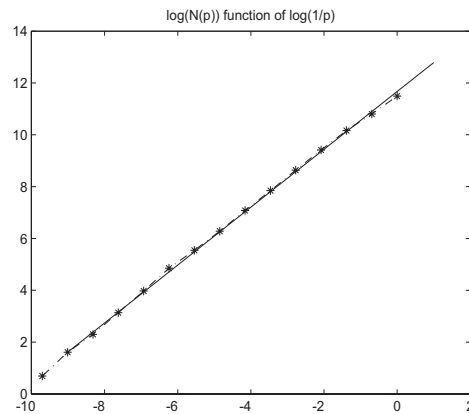


FIGURE 2.24 – Calcul de la dimension fractale de la courbe de von Koch.

Résultat :

La dimension fractale théorique est 1,12 et la dimension obtenue expérimentalement est 1,117. Nous avons donc ici 2% d'erreur. Ceci est principalement dû à l'erreur de départ sur la longueur des segments de base.

Seconde application : la courbe de Peano

Objectifs :

Les objectifs sont ici les mêmes que pour la courbe de Von Koch : établir le code de Freeman de la courbe et mesurer sa dimension fractale directement à partir du code obtenu.

Génération de la courbe de Peano :

La courbe fractale de Peano (see fig. 2.25 page suivante), initialement décrite par [Pea], est une courbe remplissante. Considérons cette courbe en deux dimensions, elle remplit entièrement la surface, sa dimension fractale théorique est donc 2.

Sa construction est la suivante : considérons une image comme neuf carrés disposés comme sur la figure 2.26 page ci-contre et joignons le centre de ces carrés de 1 à 9. Répétons ensuite le processus en divisant chacun de ces carrés en neuf autres carrés. La courbe obtenue est la courbe de Péano.

Résultat :

L'algorithme utilisé ici est le même que pour la courbe de von Koch, et nous avons trouvé une dimension fractale de 1.94. Nous avons ici 3 % d'erreur qui sont principalement dus à l'écartement du compas qui n'est pas constant. En effet, chaque mesure de N est arrondie à la longueur géométrique la plus proche séparant le centre de deux pixels.

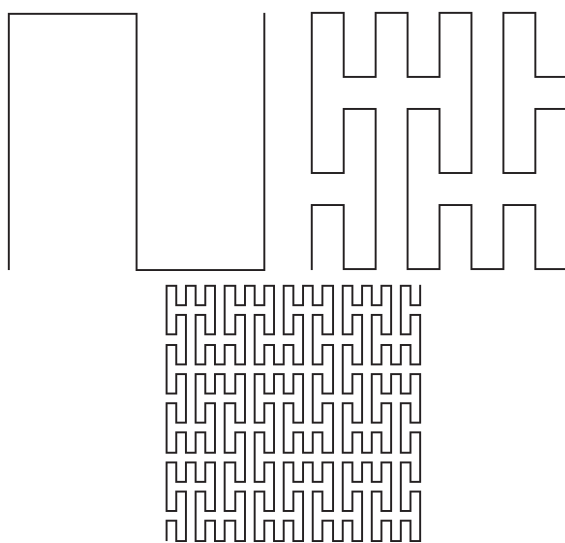


FIGURE 2.25 – Courbe fractale de Peano à trois différentes échelles.

3	4	9
2	5	8
1	6	7

FIGURE 2.26 – Représentation des carrés pour la courbe de Peano.

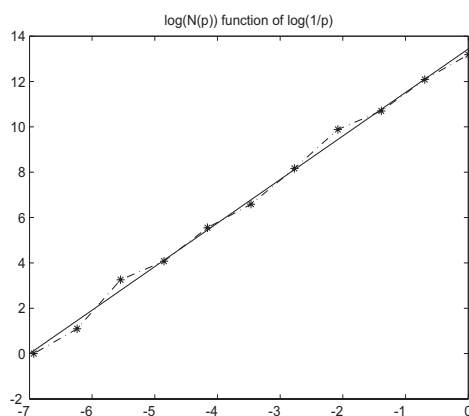


FIGURE 2.27 – Calcul de la dimension fractale de la courbe de Peano.

2.2.7.4 Conclusion

Ce paragraphe a donc introduit une nouvelle méthode de calcul de la dimension fractale d'une courbe, en utilisant le code de Freeman.

2.3 Paramètres de formes

Un paramètre de forme est une grandeur sans dimension qui permet d'évaluer une ou plusieurs caractéristiques d'une forme. En considérant une forme, on obtient ainsi, pour des paramètres de circularité, de symétrie, de convexité... une valeur chiffrée permettant d'effectuer des comparaisons avec d'autres formes.

2.3.1 Paramètres de circularité

Nous présentons ici les paramètres de circularité les plus répandus bien que tous ne soient pas directement calculables grâce au code de Freeman. Notre étude permet pour le moment d'obtenir χ_1 , χ_2 et χ_3 .

Rappelons que les paramètres de forme sont des nombres associés à une forme de façon intrinsèque (c'est-à-dire insensibles au facteur échelle). On peut s'appuyer, à titre d'exemple, sur la notion de circularité : le fait qu'une forme soit plus ou moins circulaire ne doit pas dépendre des opérateurs géométriques tels que translation, rotation et bien sûr homothétie.

2.3.1.1 Le plus classique

Il existe de nombreuses approches de la circularité. Une des plus courantes est la suivante :

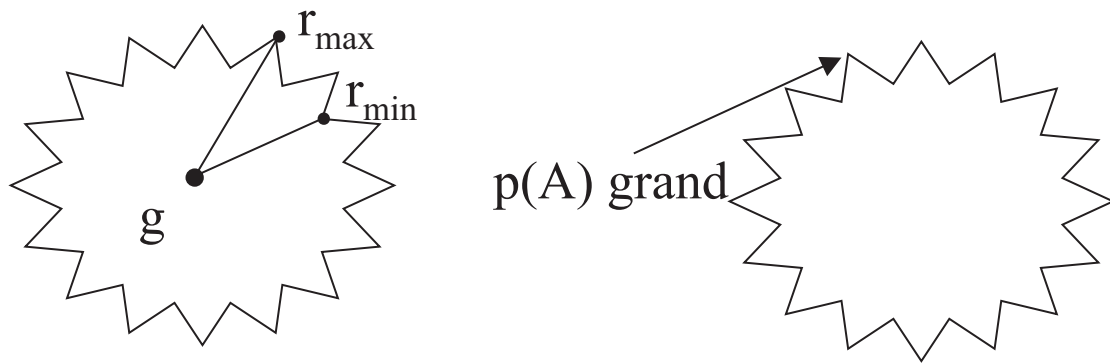
$$\chi_1(A) = \frac{P^2(A)}{\mu(A)} \quad (2.40)$$

Avec $\mu(A)$ l'aire calculée par l'une des méthodes vues précédemment et $P(A)$ le périmètre calculé par la méthode présentée au § 2.2.1 page 31. Ce paramètre, bien entendu sans unité, est issu du théorème des isopérimètres (voir [Lab]). Ce dernier établit qu'avec un lacet de longueur l fixée, c'est-à-dire à périmètre constant, on enfermera une aire maximale à l'intérieur du lacet lorsque la forme dessinée est un cercle et seulement dans ce cas. Autrement dit, ce paramètre vaut $4 \times \pi$ pour un disque et pour une autre forme A , $\chi_1(A) > 4 \times \pi$.

L'utilisation d'un tel paramètre doit se concrétiser en écrivant :

- $\chi_1(A)$ proche de $4 \times \pi \Rightarrow A$ proche d'un disque,
- $\chi_1(A)$ loin de $4 \times \pi \Rightarrow A$ loin de la forme disque.

Remarquons que cela n'est pas très exploitable si $\chi_1(A)$ n'est pas acquis avec une grande fiabilité et que cela sous-entend une continuité de la fonction $\chi_1(A)$ définie sur l'espace des formes et à valeurs dans \mathbb{R} , ce qui n'est pas le cas.

FIGURE 2.28 – Comparaison de $\chi_1(A)$ et $\chi_2(A)$ pour un cercle très bruité.

2.3.1.2 Paramètre radial

Pour toute forme A , on calcule son centre de gravité g , que l'on joint à un point m se déplaçant sur la frontière de A . On retient les valeurs extrêmes de gm comme les rayons maximum et minimum de A : $r_{max}(A)$ et $r_{min}(A)$.

Le paramètre radial $\chi_2(A)$ est alors défini par :

$$\chi_2(A) = \frac{r_{max}(A) - r_{min}(A)}{r_{min}(A)} \quad (2.41)$$

Le fait de disposer de deux paramètres est un avantage si leurs comportements sont différents : cela ouvre des perspectives applicatives plus larges. Nous pouvons illustrer la différence de comportement de $\chi_1(A)$ et $\chi_2(A)$ par la figure 2.28. Sur cette figure, on calcule un $\chi_1(A)$ grand et un $\chi_2(A)$ petit, les résultats de ces deux paramètres semblent donc très différents. Pour des applications concrètes (numériques), on préférera χ_2 qui est beaucoup moins sensible au bruitage de la frontière.

2.3.1.3 Utilisation des disques inscrit et circonscrit

L'accès au disque inscrit dans A (voir figure 2.29 page suivante) est aisé grâce à la notion d'érodé ultime (voir § 4.2.3.3 page 106) et peut être précis grâce à l'algorithme 20 page 106. Si $r(A)$ désigne le rayon du disque inscrit, on définit le taux de remplissage de A par un disque sous la forme :

$$\chi_3(A) = \frac{\pi r^2(A)}{\mu(A)} \quad (2.42)$$

qui tend vers 1 si A s'approche d'un disque.

De la même manière, on peut définir le disque circonscrit à A (unique). Ce problème n'est pas réputé simple et est un classique de géométrie algorithmique (plus petit disque contenant un semis de n points dans le plan).

[Jou83] a apporté une solution avec un algorithme de la fenêtre disquée.

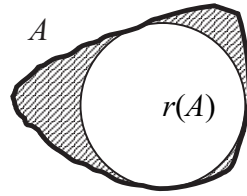


FIGURE 2.29 – Disque inscrit dans une forme A pour le calcul de χ_5 et χ_3 .

Sachant construire le disque circonscrit, on dispose de deux outils d'évaluation de la circularité. En désignant par $R(A)$ le rayon du disque circonscrit à A , on peut poser comme précédemment :

$$\chi_4(A) = \frac{\mu(A)}{\pi R^2(A)} \quad (2.43)$$

qui donne le taux de remplissage du disque circonscrit par la forme A .

Enfin, disposant des disques inscrit et circonscrit, on peut calculer la distance de Asplünd (voir [Grü]) de A à la forme disque :

$$\chi_5(A) = d_{As}(A, \text{disque}) = \log\left(\frac{R(A)}{r(A)}\right) \quad (2.44)$$

La distance de Asplünd est très peu utilisée mais elle demeure intéressante car elle est théoriquement insensible à l'échelle d'observation de chacune des formes.

2.3.2 Paramètres de symétrie

Cette classe de paramètres est peu utilisée en classification des formes, en particulier dans les logiciels du marché. Seuls le coefficient d'asymétrie de Besicovitch et le coefficient de symétrie de Blaschke sont pour le moment directement calculables grâce au code de Freeman.

2.3.2.1 Coefficient d'asymétrie de Besicovitch

Le but du coefficient d'asymétrie de Besicovitch [Bes] est de dire si un point a appartenant à un convexe A est bien ou mal centré dans A . On note $A^-(a)$ le symétrique de A par rapport au point a . (voir § 3.1.4.1 page 75)

L'intersection $A \cap A^-(a)$ de A et de son symétrisé par rapport à a a les propriétés suivantes :

- il est à centre de symétrie a ,
- il est inclus dans A ,
- c'est le plus grand objet à centre de symétrie a inclus dans A .

Avec $\mu(A)$ l'aire de l'objet A , on cherche le point $a_0 \in A$ tel que :

$$\mu(A \cap A^-(a_0)) = \sup_{a \in A} \mu(A \cap A^-(a)) \quad (2.45)$$

On peut alors calculer le coefficient d'asymétrie de Besicovitch du convexe A :

$$a_{BE}(A) = 1 - \frac{\mu(A \cap A^-(a_0))}{\mu(A)} \quad (2.46)$$

Besicovitch a démontré que ce coefficient admet la propriété suivante : $a_{BE} \in [0, 1/3]$, la valeur "0" étant atteinte pour les objets à centre de symétrie et la valeur $1/3$ pour les triangles qui sont donc les convexes les plus dissymétriques du plan.

Nous avons implémenté le calcul de ce coefficient d'asymétrie. Nous travaillons ici avec un seul objet convexe du plan et à chaque étape, nous n'utilisons que le code de Freeman de l'objet (l'intersection utilisée ici est présentée au §3.2.2.1). L'algorithme proposé est l'algorithme 9.

Algorithme 9 Obtention du coefficient d'asymétrie de Besicovitch

- 1 Calcul du code de Freeman du contour de l'objet A
 - 2 Calcul du centre de gravité $g(A)$
 - 3 Calcul de $A^-(g)$
 - 4 Calcul de l'intersection de A et $A^-(g)$
 - 5 Calcul de l'aire de cette intersection
 - 6 Translation de l'objet $A^-(g)$ selon les directions du code de Freeman
 - 7 **si** pour chacune de ces 8 directions $\mu(A \cap A^-(a_0)) < \mu(A \cap A^-(g))$ **alors**
 - 8 g est le point de Besicovitch
 - 9 **sinon si** il existe un ou plusieurs points a_0 tel(s) que $\mu(A \cap A^-(a_0)) \geq \mu(A \cap A^-(g))$ **alors**
 - 10 alors on sélectionne celui dont l'aire de l'intersection est maximale et on réitère la procédure pour ses propres voisins.
 - 11 **fin si**
-

L'algorithme converge vers un point généralement unique en théorie mais la numérisation peut entraîner l'existence de plusieurs points de Besicovitch.

En calculant $\mu(A \cap A^-(a))$ pour tous les points a de l'objet A , on peut définir des courbes de niveau en associant les pixels pour lesquels cette aire est la même. La figure 2.30 page suivante montre deux objets représentant ces courbes de niveaux aussi appelées floating bodies (voir [Buc] et [Rei]).

Un seuillage sur ces courbes de niveaux semble correspondre à une opération de type morphologique (érosion) sur l'objet A où l'on peut notamment voir que "l'érosion" est plus forte dans les zones de forte courbure.

Ceci pourrait être également relié au symétrisé de Minkowski (voir [Min]) et à la morphologie directionnelle des convexes (voir [Jou84] et [Fil]). Il y a ici plusieurs pistes en cours d'étude.

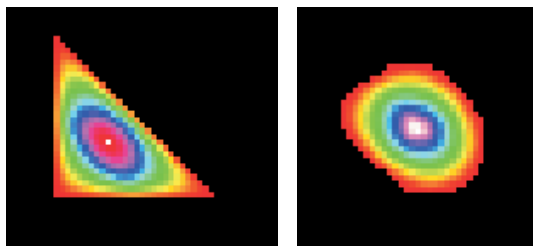


FIGURE 2.30 – Illustration des Floating Bodies.

2.3.2.2 Coefficient de symétrie de Blaschke

Avant d'étudier ce coefficient de symétrie, il nous faut présenter le symétrisé de Minkowski.

Pour obtenir le symétrisé de Minkowski d'une forme A , on effectue la demi-somme de Minkowski de la forme et de son symétrique $A^-(O)$ par symétrie centrale de centre quelconque O selon l'équation 2.47 (le résultat obtenu est à une translation près).

$$S(A) = \frac{1}{2}(A \oplus A^-(O)) \quad (2.47)$$

Le coefficient $1/2$ permet d'avoir un symétrisé d'aire comparable à celle de la forme de départ A . Le théorème de Brunn-Minkowski établit l'inégalité 2.48 pour toute forme A .

$$\mu(S(A)) \geq \mu(A) \quad (2.48)$$

Le principe du coefficient de symétrie de Blaschke [Bla] est assez proche de celui du coefficient de Besicovitch puisqu'il fait intervenir également un rapport d'aire :

$$s_{BL}(A) = \frac{\mu(A)}{\mu(S(A))} \quad (2.49)$$

Ce quotient remplace la notion de "taux de remplissage" de $S(A)$ par A , même si A n'est pas contenu dans $S(A)$.

Le coefficient de symétrie de Blaschke consiste en la comparaison de l'aire de la forme étudiée avec celle de son symétrisé de Minkowski $S(A)$ et présenté dans [Min]. Ce paramètre se distingue du précédent par l'absence de point particulier associé au coefficient car il ne passe pas par l'évaluation du centrage d'un point.

Ce coefficient invariant par transformation affine vaut $2/3$ pour les triangles et 1 pour les convexes à centre de symétrie. La détermination de sa borne inférieure est due à Rademacher [Rad] et Estermann [Est], l'égalité à Rogers et Shephard [Roge]. De plus, calculé sous forme de coefficient d'asymétrie : $1 - s_{BL}(A)$ varie comme $a_{BE}(A)$ dans $[0, 1/3]$.

Une procédure de calcul de ce coefficient est donnée par l'algorithme 10 page suivante.

Algorithme 10 Obtention du coefficient de symétrie de Blaschke

- 1 Calcul du code de Freeman du contour de l'objet A
- 2 Calcul du symétrique A^- de A
- 3 Les coordonnées des premiers pixels de A et A^- sont conservées en p et q
- 4 On translate q de A^- sur tous les pixels de la frontière $Fr(A)$
- 5 On réalise l'union de tous ces translats qui n'est autre que $A \oplus A^-$ soit $2S(A)$ dont l'aire est 4 fois celle de $S(A)$.
- 6 On calcule $s_{BL}(A) = \frac{\mu(A)}{\mu(S(A))}$

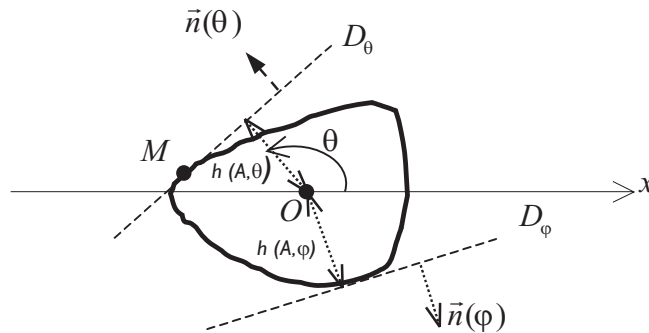


FIGURE 2.31 – Fonction support.

2.3.2.3 Coefficient de symétrie de Minkowski

Le coefficient de symétrie de Minkowski nécessite la définition de la fonction support que l'on notera h . Cette fonction a été mise en évidence par Minkowski lui-même pour décrire les courbes convexes. Le principe est le suivant : considérons un point O d'un objet comme origine et un angle θ auquel on associe un vecteur unitaire $\vec{n}(\theta)$. Enfin, considérons la droite normale au vecteur $\vec{n}(\theta)$ qui est la tangente à la forme en un certain point M de celle-ci.

On décrit alors toute forme par l'angle considéré θ et la distance à l'origine de cette droite. Une illustration est proposée figure 2.31. En d'autres termes, pour tout point M de la tangente la fonction support $h_A(O, \theta)$ en un point O de la forme A pour un angle θ donné est définie comme suit :

$$h_A(O, \theta) = \vec{OM} \bullet \vec{n}(\theta) \quad (2.50)$$

où \bullet désigne le produit scalaire.

On définit la fonction de Minkowski de la manière suivante :

$$m : A \longrightarrow \mathbb{R}^+ \\ x \longrightarrow \inf_{\theta \in [0, 2\pi]} \left(\frac{h(x, \theta)}{h(x, \theta + \pi)} \right) \quad (2.51)$$

De la relation précédente, on tire le coefficient de Minkowski :

$$s_M(A) = \sup_{x \in A} (m(x)) \quad (2.52)$$

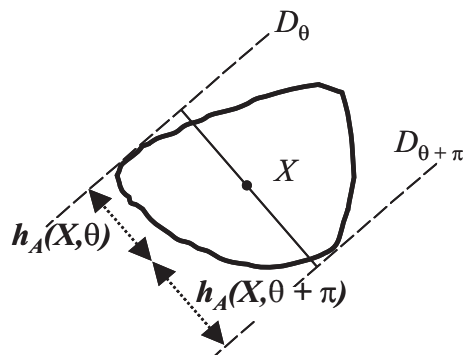


FIGURE 2.32 – Coefficient de Minkowski.

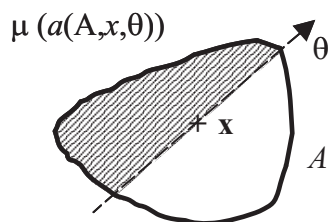


FIGURE 2.33 – Coefficient de Winternitz.

Ce coefficient, défini dans [Min], vise à évaluer le décentrage des points d'un convexe. Il est illustré sur la figure 2.32. Il met en jeu à la fois un angle θ et un point X appartenant à la forme étudiée.

Les calculs théoriques laissent apparaître, pour des convexes à centre de symétrie et pour des triangles, que le point en lequel est atteint le maximum de ce coefficient de Minkowski est confondu avec le centre de gravité. Le coefficient $s_M(A)$ vaut $1/2$ pour les triangles et 1 pour les convexes à centre de symétrie. En plus de la valeur du paramètre en tant que telle, on tire de la relation le point X correspondant au point le mieux centré au sens de ce paramètre, mais qui n'est pas le centre de gravité.

2.3.2.4 Coefficient de symétrie de Winternitz

Nous n'avons pas de référence concernant l'article de Winternitz introduisant ce coefficient, mais seulement l'article de Grünbaum [Grü] le présentant.

Le coefficient de Winternitz est fondé sur des rapports d'aires délimitées par un axe de direction θ passant par un point x donné de A (voir figure 2.33).

On définit tout d'abord la fonction de Winternitz suivante :

$$w : A \longrightarrow \mathbb{R}^+ \\ x \longrightarrow \inf_{\theta \in [0, 2\pi]} \left(\frac{\mu(a(A, x, \theta))}{\mu(A) - \mu(a(A, x, \theta))} \right) \quad (2.53)$$

avec $a(A, x, \theta)$ la partie de A "à gauche" de la corde passant par x et d'orientation θ .

De la relation précédente, on tire le coefficient de symétrie de Winternitz :

$$s_W(A) = \sup_{x \in A} (w(x)) \quad (2.54)$$

Les calculs théoriques montrent que, pour des convexes à centre de symétrie et pour des triangles, le point en lequel est atteint le maximum de ce coefficient de Winternitz est confondu avec le centre de gravité. Le coefficient $s_W(K)$ vaut $4/5$ pour les triangles et 1 pour les convexes à centre de symétrie. En plus de la valeur du paramètre en tant que telle, on tire de la relation le point x correspondant au point le mieux centré au sens de ce paramètre.

2.3.3 Convexité

On sait que la notion de convexité n'a pas de définition naturelle sur une trame. Nous proposons ici une étude originale à partir du code de Freeman. Pour cela, en un point donné du contour, nous considérons les vecteurs ayant comme origine ce point et comme extrémité un autre point du contour, à une distance de n codants dans le sens trigonométrique d'une part et dans le sens horaire d'autre part. Ainsi, le signe du produit vectoriel de ces deux vecteurs nous donne une indication sur la convexité locale de l'objet.

- Si ce produit est nul (cas des pixels verts sur la figure 2.34), les deux vecteurs sont alignés, et le point considéré se trouve sur une zone plate, non courbée.
- Si le produit est positif (pixel rouge), le point se trouve sur une zone "convexe" de l'objet.
- Si le produit est négatif (pixel bleu), le point se trouve sur une zone "concave" de l'objet.

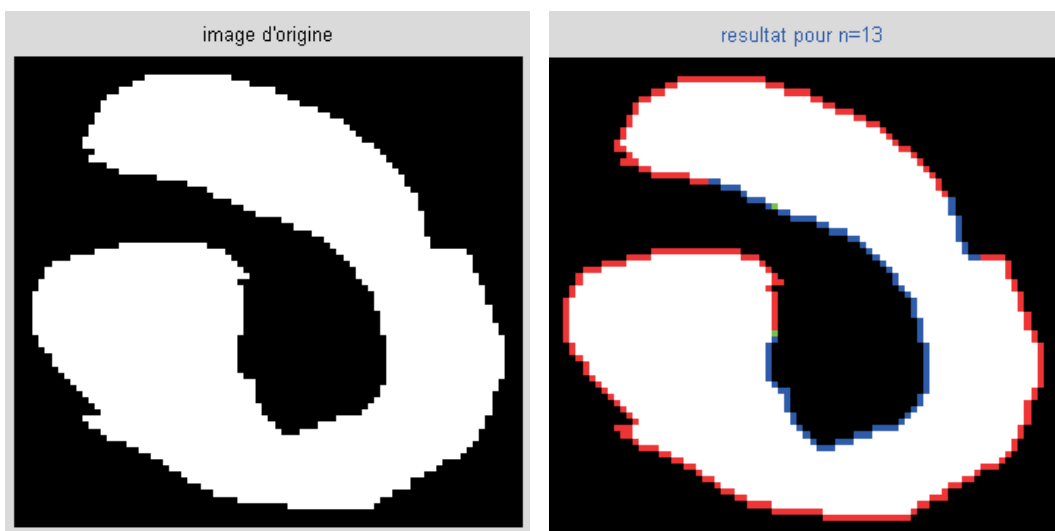


FIGURE 2.34 – Étude de la n -quasi-convexité d'une forme (ici $n=13$).

Le signe de ce produit vectoriel nous donne donc une information sur la courbure *en un point* du contour de la forme étudiée et pour un n donné (n étant le nombre de codants à utiliser) : si le produit vectoriel étudié ci-dessus est positif, on dira que l'objet est n -convexe en ce point.

Il est par suite possible, pour ce même n donné, de dire si la forme complète est n -convexe : une forme est n -convexe si elle est n -convexe en tout point de son contour.

Notons que le critère n choisi ici désigne bien le nombre de codants considérés pour effectuer le calcul.

2.3.4 Histogramme d'un code

L'histogramme du code de Freeman du contour d'un objet η a été présenté par Iivarinen et al [Iiv97]. De manière classique, il compte le nombre d'occurrences de chaque codant n_k , rapporté au nombre total n de codants contenus dans la description de l'objet étudié.

$$\eta(k) = \frac{n_k}{n}, k \in [0, 7] \quad (2.55)$$

Un exemple d'histogramme est présenté sur la figure 2.35. Le code de l'objet présenté est : 666676777770777700000000101121222222233233333434444444455454555.

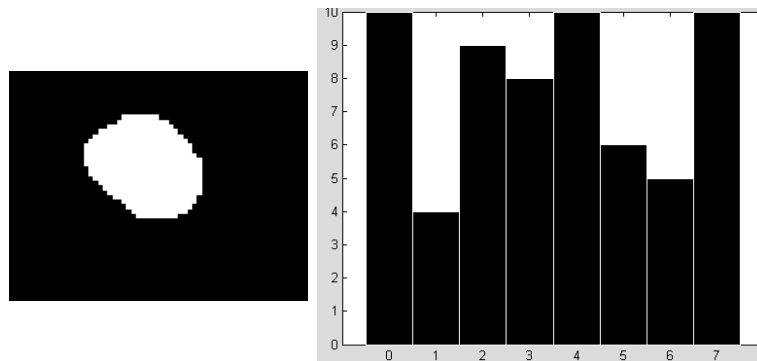


FIGURE 2.35 – Objet étudié (à gauche) et l'histogramme de son code de Freeman associé.

Ce paramètre est simple et rapide à obtenir et possède les propriétés suivantes :

- invariance par translation,
- invariance après un changement d'échelle.
- invariance par rotation multiple de 90° .

Pour obtenir une meilleure invariance par rotation pour des angles différents de 90° , on peut utiliser l'histogramme normalisé des chaînes de codants présenté par Jukka Iivarinen [Iiv96].

2.3.5 Axes principaux d'inertie

Les axes principaux d'inertie d'une forme donnent la direction dans laquelle cet objet est le plus allongé et la direction orthogonale à celle-ci, le premier étant l'axe majeur et le second l'axe mineur. Seules l'orientation et la longueur de ces axes sont importantes, le sens quant à lui peut être choisi arbitrairement.

Nous présentons ici trois différentes méthodes d'obtention de ces axes : la première est basée sur la notion de matrice de covariance des points composant la frontière de l'objet et utilise le calcul de vecteurs propres et de valeurs propres, et les deux autres méthodes utilisent les moments surfaciques.

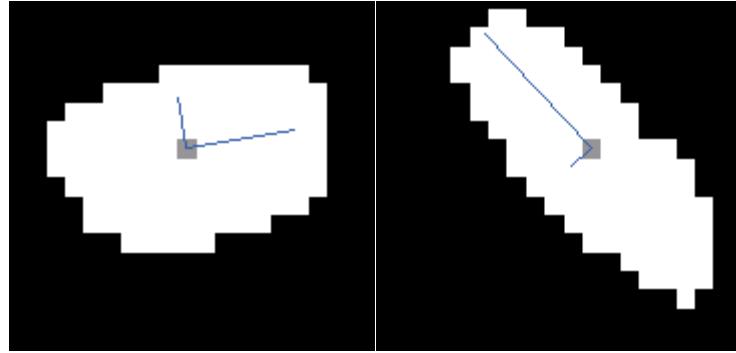


FIGURE 2.36 – Axes principaux d’inertie de deux objets.

2.3.5.1 Par un calcul de valeurs propres

Cette méthode a été présentée brièvement par Costa et Cesar [Costa], sans justifications.

Pour obtenir l’orientation et la longueur des axes principaux d’inertie, il faut considérer les coordonnées absolues des points formant le contour de l’objet :

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n) \quad (2.56)$$

En supposant que ces coordonnées jouent le rôle de vecteurs aléatoires, soit C la matrice de covariance de tous ces vecteurs. Les vecteurs propres de C correspondent aux directions des axes d’inertie du contour de l’objet. Le vecteur propre associé à la valeur propre principale donne la direction de l’axe majeur d’inertie du contour de l’objet, alors que la seconde valeur propre est rattachée à la direction de l’axe mineur d’inertie.

Ceci provient du fait que la matrice de covariance est calculée sur un nuage de points centré. Ce que cette méthode ne précise pas est qu’il faut au préalable considérer le centre de masse de la forme comme origine de l’image et donc effectuer un décalage des coordonnées des points considérés.

En considérant une forme A codée par le code de Freeman de son contour, il est alors possible d’obtenir les axes d’inertie du contour en revenant aux coordonnées absolues des points le décrivant.

Deux exemples de résultats sont illustrés sur la figure 2.36. En gris au centre est représenté le centre de masse du contour, calculé selon la méthode présentée au § 2.2.3 page 38. Les traits fins indiquent l’axe majeur et l’axe mineur d’inertie, l’axe majeur étant représenté par le trait le plus long.

2.3.5.2 Par un calcul de moments surfaciques

Dans cette partie, on considère un objet P codé au sens de Freeman utilisant quatre directions, les codants allant du sommet d’un pixel au sommet suivant, et non pas du centre d’un pixel au centre du pixel suivant. La frontière d’un tel objet fermé a un nombre pair $(2n)$ de côtés et ses sommets sont au même nombre. On les notera $S_i, i \in [1..2n]$ avec la convention cyclique $S_1 = S_{2n+1}$.

La convention suivante est utilisée pour la numérotation des sommets :

- L'initialisation se fait en prenant comme sommet de départ un de ceux auxquels on arrive par le haut de l'image et dont on repart par la droite.
- La numérotation se fait dans le sens direct.
- On attribue à un sommet :
 - d'indice pair S_{2k} les coordonnées (x_k, y_k) ,
 - d'indice impair S_{2k+1} les coordonnées (x_{k+1}, y_k) .

En effet, dans ce type de polygone, une seule des coordonnées change lors du passage d'un sommet du contour au suivant. De plus, ces changements se font par alternance entre l'abscisse et l'ordonnée. On aura donc comme numérotation :

$$\{S_0(x_0, y_0), S_1(x_1, y_0), S_2(x_1, y_1) \dots S_{2n-1}(x_n, y_{n-1})\} \quad (2.57)$$

On remarquera que bien que l'on ait $2n$ points, on n'a que $2n$ (et non $4n$) coordonnées : x_0, x_1, \dots, x_{n-1} , et y_0, y_1, \dots, y_{n-1} .

Les moments surfaciques dans le domaine discret sont donnés par :

$$I_{pq} = \begin{cases} \frac{1}{(p+1)(q+1)} \sum_{k=0}^{n-1} (x_k^{p+1} - x_{k+1}^{p+1}) y_k^{q+1} \\ \text{ou} \\ \frac{1}{(p+1)(q+1)} \sum_{k=0}^{n-1} x_{k+1}^{p+1} (y_{k+1}^{q+1} - y_k^{q+1}). \end{cases} \quad (2.58)$$

De plus, toute droite (D) du plan est caractérisée par les coordonnées polaires $[\theta, p]$ du point H , projection de l'origine O sur (D).

L'angle θ est donc défini modulo 2π et $p \geq 0$. En particulier, la droite (D) a pour vecteur normal $\overrightarrow{OH}(p \cos \theta, p \sin \theta)$.

On notera $(D) = (D_{\theta, p})$. En effet, cette représentation est unique (pour être rigoureux, il faut traiter à part les droites passant par l'origine : pour celles-ci, on a clairement $p = 0$ mais θ doit être directement obtenu en tant qu'angle polaire d'un vecteur normal, et dans ce cas, cet angle est défini modulo π).

En conséquence de ce qui précède, $(D_{\theta, p})$ a pour équation :

$$x \cos \theta + y \sin \theta - p = 0 \text{ (avec } 0 \leq \theta < 2\pi \text{ et } p \geq 0). \quad (2.59)$$

Enfin, l'ensemble des points situés à une distance $\delta > 0$ de cette droite $(D_{\theta, p})$ est constitué par deux droites parallèles. Celle de ces deux droites qui est la plus éloignée de l'origine est nécessairement $(D_{\theta, p+\delta})$. Considérons par exemple que $M_0(x_0, y_0)$ appartient à cette droite. Il vérifiera son équation, c'est-à-dire que l'on aura :

$$x_0 \cos \theta + y_0 \sin \theta - (p + \delta) = 0 \text{ d'où } \delta = x_0 \cos \theta + y_0 \sin \theta - p.$$

On peut tenir le même type de raisonnement dans le cas où $M_0(x_0, y_0)$ appartient à l'autre droite mais cette fois en distinguant les cas $p \geq \delta$ ou $p < \delta$ c'est-à-dire les droites $(D_{\theta, p-\delta})$

ou $(D_{\theta+\pi, \delta-p})$. On obtiendra alors $-\delta = x_0 \cos \theta + y_0 \sin \theta - p$, ce qui explique la présence de la valeur absolue dans la formule ci-dessous qui donne la distance δ d'un point $M_0(x_0, y_0)$ à la droite $(D_{\theta, p})$:

$$\text{dist}(M_0, (D_{\theta, p})) = \delta = |x_0 \cos \theta + y_0 \sin \theta - p| \quad (2.60)$$

A partir d'ici, nous allons voir deux méthodes, l'une que l'on peut qualifier de trigonométrique, et l'autre, purement analytique, qui semble préférable à plusieurs points de vue.

Méthode trigonométrique :

Suite au résultat de l'équation 2.60, la recherche de l'axe vis-à-vis duquel l'inertie de la forme est minimale revient à la recherche de minimisation (par rapport aux deux variables θ et p) de l'intégrale :

$$J = J_{\theta, p} = \iint_{(x, y) \in D} (x \cos \theta + y \sin \theta - p)^2 dx dy \quad (2.61)$$

(où D est l'intérieur de la forme). Une condition nécessaire et suffisante ici est que les deux dérivées partielles suivantes s'annulent simultanément :

$$\frac{\partial J}{\partial p} = 0 \text{ et } \frac{\partial J}{\partial \theta} = 0 \quad (2.62)$$

Formons les deux équations résultantes :

a) **Paramètre p :**

Si l'on dérive J sous le signe somme par rapport à p , étant donné qu'il s'y trouve une expression de la forme v^2 , on obtiendra l'intégrale de sa dérivée $2vv'$ ou plus exactement $2u \frac{\partial u}{\partial p}$:

$$\frac{\partial J}{\partial p} = \int \int_{(x, y) \in D} 2(x \cos \theta + y \sin \theta - p)(-1) dx dy = 0 \quad (2.63)$$

Soit, en simplifiant par (-2) et en développant en trois intégrales :

$$\cos \theta \int \int_{(x, y) \in D} x dx dy + \sin \theta \int \int_{(x, y) \in D} y dx dy - p \int \int_{(x, y) \in D} dx dy = 0 \quad (2.64)$$

Soit encore

$$I_{10} \cos \theta + I_{01} \sin \theta - p I_{00} = 0 \quad (2.65)$$

d'où l'on tire la valeur de $p = p_{\min}$ réalisant le minimum :

$$p_{\min} = \frac{1}{I_{00}} (I_{10} \cos \theta + I_{01} \sin \theta) \quad (2.66)$$

associée évidemment, à la valeur $\theta = \theta_{\min}$.

b) **Paramètre θ :**

Si l'on dérive J sous le signe somme par rapport à θ , pour les mêmes raisons que celles qui ont été données plus haut :

$$\frac{\partial J}{\partial \theta} = \int \int_{(x,y) \in D} 2(x \cos \theta + y \sin \theta - p)(-x \sin \theta + x \cos \theta) dx dy = 0 \quad (2.67)$$

Soit, après simplification par 2, en séparant en deux membres :

$$\int \int_{(x,y) \in D} (x \cos \theta + y \sin \theta)(-x \sin \theta + y \cos \theta) dx dy = p \int \int_{(x,y) \in D} (-x \sin \theta + y \cos \theta) dx dy \quad (2.68)$$

ce qui, après divers développements, donne :

$$((\cos \theta)^2 - (\sin \theta)^2)I_{11} + \cos \theta \sin \theta (I_{02} - I_{20}) = p(-I_{10} \sin \theta + I_{01} \cos \theta) \quad (2.69)$$

Soit, d'après la formule obtenue pour p :

$$\cos(2\theta)I_{11} + \frac{1}{2} \sin(2\theta) (I_{02} - I_{20}) = \frac{1}{I_{00}} (I_{10} \cos \theta + I_{01} \sin \theta) (-I_{10} \sin \theta + I_{01} \cos \theta) \quad (2.70)$$

ou encore, après multiplication par I_{00}

$$\left[\cos(2\theta)I_{11} + \frac{1}{2} \sin(2\theta) (I_{02} - I_{20}) \right] I_{00} = ((I_{01})^2 - (I_{10})^2) \cos \theta \sin \theta + (I_{10}I_{01}) \quad (2.71)$$

$$\cos(2\theta)I_{11}I_{00} + \frac{1}{2} \sin(2\theta) [(I_{02} - I_{20})I_{00} + (I_{10})^2 - (I_{01})^2] = I_{10}I_{01} \quad (2.72)$$

C'est une équation trigonométrique classique, du type

$$A \cos \alpha + B \sin \alpha = C \quad (2.73)$$

qui se résoud de la façon suivante : on recherche un angle β , s'il existe, tel que :

$$\cos \beta = \frac{A}{\sqrt{A^2 + B^2}} \text{ et } \sin \beta = \frac{B}{\sqrt{A^2 + B^2}} \quad (2.74)$$

autrement dit, en formant le quotient, tel que $\tan \beta = \frac{B}{A}$

$$\text{On obtiendra deux solutions : } \beta = \text{Arctan} \frac{B}{A} \text{ ou } \beta = \pi + \text{Arctan} \frac{B}{A} \quad (2.75)$$

(attention à prévoir les deux cas selon le quadrant dans lequel se situe le point (A, B) : on pourra pour cela utiliser la fonction spéciale $\text{Arctan} 2$ qui existe dans la plupart des bibliothèques).

Alors, l'équation (2.73) devient :

$$\sqrt{A^2 + B^2}(\cos \beta \cos \alpha + \sin \beta \sin \alpha) = C \quad (2.76)$$

Soit $\cos(\alpha - \beta) = \frac{C}{\sqrt{A^2 + B^2}}$ d'où les deux solutions :

$$\alpha = \beta \pm \text{Arc cos} \frac{C}{\sqrt{A^2 + B^2}} \quad (2.77)$$

(ceci suppose que $\frac{C}{\sqrt{A^2 + B^2}}$ est une quantité comprise entre -1 et $+1$; ce sera le cas dans notre problème).

On peut ajouter ici un conseil pour la programmation : vérifier que dans le logiciel de programmation utilisé, la fonction *ArcCos* désigne bien la fonction classique. En effet, la programmation des fonctions trigonométriques réciproques peut donner lieu à de véritables casse-têtes, dûs essentiellement à de mauvaises prises en compte des domaines de définition.

Il reste à remplacer A, B et C par leurs expressions en fonction des différents I_{pq} (voir (2.72)) pour obtenir θ_{\min} . On reporte alors cette valeur dans (2.66) pour en déduire p_{\min} .

Méthode analytique :

On peut écrire la minimisation (2.61) sous la forme suivante, en rangeant les différents I_{pq} dans une matrice A :

$$\text{Minimiser}_{\theta,p} X^T A X \quad (2.78)$$

avec

$$X^T A X = \begin{bmatrix} \cos \theta & \sin \theta & -p \end{bmatrix} \begin{bmatrix} I_{20} & I_{11} & I_{10} \\ I_{11} & I_{02} & I_{01} \\ I_{10} & I_{01} & I_{00} \end{bmatrix} \begin{bmatrix} \cos \theta \\ \sin \theta \\ -p \end{bmatrix}. \quad (2.79)$$

Remarque : puisque le résultat est un carré quel que soit X , la matrice A est donc symétrique définie positive.

Le problème posé comporte une contrainte cachée que nous allons faire émerger par une reformulation. Pour cela, introduisons deux matrices :

$$\text{l'adjointe de } A : A^{ad} = \begin{bmatrix} a & b & d \\ b & c & e \\ d & e & f \end{bmatrix} \text{ et la matrice } J = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

on rappelle que l'adjointe est la transposée de la matrice des cofacteurs, on a donc $A^{ad} = \det(A)A^{-1}$.

Transformons le problème initial en le suivant :

$$\text{Minimiser } X^T A X \text{ sous la contrainte : } X = \begin{bmatrix} c \\ s \\ -p \end{bmatrix} \text{ avec } c^2 + s^2 = 1 \quad (2.80)$$

ou encore :

$$\text{Minimiser } X^T A X \text{ sous la contrainte : } \|JX\| = 1 \quad (2.81)$$

La méthode des multiplicateurs de Lagrange nous permet de remplacer cette minimisation par la recherche d'une solution au problème suivant : $AX = \lambda JX$ soit $\frac{1}{\lambda}X = A^{-1}JX$, ou encore (par définition de l'adjointe) $\mu X = A^{ad}JX$, pour un certain μ . Ainsi X apparaît comme un vecteur propre de la matrice

$$B = A^{ad}J = \begin{bmatrix} a & b & 0 \\ b & c & 0 \\ d & e & 0 \end{bmatrix}. \quad (2.82)$$

Déterminons ces vecteurs propres qui fourniront **directement** les deux triplets de coefficients (u, v, w) des équations $ux + vy + w = 0$ des **axes d'inertie (principal et secondaire)** de la forme (en effet, cette méthode trouve a priori les "extrema sous contraintes" et non pas seulement les minima).

Nous devons d'abord rechercher les valeurs propres de B : mise à part la valeur propre 0, ce seront les valeurs propres de la matrice $C = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$ qui est elle aussi définie positive : ces valeurs propres sont donc en particulier réelles.

Soit ρ l'une d'elles ; elle vérifie l'équation :

$\det(C - \rho I) = 0$ soit $(a - \rho)(c - \rho) - b^2 = 0$ ou encore

$$(\rho - a)\rho = c\rho + b^2 - ac \quad (2.83)$$

Par définition, les composantes (u, v, w) d'un vecteur propre associé à ρ sont solutions du système :

$$\begin{bmatrix} a - \rho & b & 0 \\ b & c - \rho & 0 \\ d & e & -\rho \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (2.84)$$

Les deux premières équations étant proportionnelles, il suffira de prendre une solution de la première. En ce qui concerne la troisième composante w , elle doit vérifier $w = \frac{du + ev}{\rho}$.

Par suite, on peut prendre :

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} b \\ \rho - a \\ (db + e(\rho - a)) / \rho \end{bmatrix} = \begin{bmatrix} b \\ c \\ e \end{bmatrix} + \frac{1}{\rho} \begin{bmatrix} 0 \\ b^2 - ac \\ bd - ae \end{bmatrix} \quad (2.85)$$

que l'on peut encore écrire sous la forme $B_2 + \frac{1}{\rho}(bB_1 - aB_2)$ en notant B_k les colonnes de B . Il s'agit donc d'une combinaison linéaire des deux premières colonnes de B (donc de I^{ad}) $k_1B_1 + k_2B_2$. (c'est ce qui est implémenté dans l'algorithme ci-dessous).

Or ceci n'a rien d'étonnant puisque les coefficients de la colonne B_1 , tout comme ceux de la colonne B_2 , fournissent des équations de droites passant par le centre de gravité ; en effet, si nous regardons les zéros entre parenthèses de l'identité ci-dessous :

$$\begin{bmatrix} I_{20} & I_{11} & I_{10} \\ I_{11} & I_{02} & I_{01} \\ I_{10} & I_{01} & I_{00} \end{bmatrix} \begin{bmatrix} a & b & d \\ b & c & e \\ d & e & f \end{bmatrix} = \begin{bmatrix} * & 0 & 0 \\ 0 & * & 0 \\ (0) & (0) & * \end{bmatrix} \quad (2.86)$$

ils expriment que $aI_{10} + bI_{01} + dI_{00} = 0$ et que $bI_{10} + cI_{01} + eI_{00} = 0$ c'est-à-dire que les équations $ax + by + d = 0$ et $bx + cy + e = 0$ sont vérifiées par $x = I_{10}/I_{00}$ et $y = I_{01}/I_{00}$ qui sont les coordonnées du centre de gravité (surfaceutique). Il est donc normal que les droites recherchées soient combinaisons linéaires de ces deux droites.

L'un des gros avantages de cette méthode est qu'elle évite de passer par la trigonométrie. De plus, elle ne nécessite que peu de calculs :

Algorithme développé :

$$a = I_{02} * I_{00} - I_{01}^2$$

$$b = I_{10} * I_{01} - I_{11} * I_{00}$$

$$c = I_{20} * I_{00} - I_{10}^2$$

$$d = I_{11} * I_{01} - I_{02} * I_{10}$$

$$e = I_{11} * I_{10} - I_{20} * I_{01}$$

$$s = c - a$$

$$dg = \sqrt{s^2 + 4b^2}$$

$$\mu_1 = s + dg$$

$$\mu_2 = s - dg$$

Les deux triplets (u, v, w) recherchés sont :

$$(u, v, w) = 2b * (a, b, d) + \mu_1 * (b, c, e) \text{ et}$$

$$(u, v, w) = 2b * (a, b, d) + \mu_2 * (b, c, e)$$

Ce qui fait au total très peu d'opérations et surtout aucun test.

2.4 Conclusion

Nous avons décrit ici les paramètres et opérations qu'il est possible d'obtenir à partir d'une chaîne de codants, puis les paramètres géométriques d'une forme correspondant à une chaîne fermée, et enfin quelques paramètres de forme pouvant être calculés grâce au code de Freeman du contour d'un objet.

Nous allons maintenant voir comment obtenir le code de la transformée d'une forme et le code de l'intersection et de l'union de deux formes, puis comment comparer deux formes grâce à leur code de Freeman.

Chapitre 3

Transformations, comparaisons de formes à partir du code de Freeman

Sommaire

3.1 Opérations sur un objet à partir du codage de Freeman	70
3.1.1 Translation	70
3.1.2 Homothéties	70
3.1.3 Rotation	75
3.1.4 Symétries	75
3.1.5 Appartenance d'un point à un objet	76
3.2 Opération sur deux objets à partir du codage de Freeman . . .	79
3.2.1 Estimation de la distance géométrique entre formes	79
3.2.2 Intersection et union de deux formes	81
3.2.3 Comparaison de deux formes	87
3.2.4 Corrélacion entre deux formes	90
3.3 Conclusion	91

Ce chapitre a pour but de présenter dans un premier temps les transformations possibles sur des formes connues par le code de Freeman de leurs contours, puis de comparer deux formes entre elles.

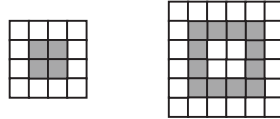


FIGURE 3.1 – Carré minimal (à gauche) et son expansion d'un facteur 2 (à droite).

3.1 Opérations sur un objet à partir du codage de Freeman

Cette section présente la translation, l'homothétie, la rotation, la symétrie d'une forme connue par son code de Freeman, mais aussi un test d'appartenance d'un pixel à un objet connu par son code de Freeman.

3.1.1 Translation

Le translaté d'un objet connu par le code de Freeman de son contour s'obtient de façon évidente : il suffit de translater le point de départ, le code restant strictement identique. En effet, l'information relative au point de départ n'est pas modifiée.

3.1.2 Homothéties

3.1.2.1 Expansion

D'après [Mai], ce qui est appelé dilatation, mais qui correspondrait à l'expansion d'un contour d'un facteur entier n , se ferait simplement en répétant chaque entier du code de Freeman n fois. Par exemple, l'homothétie de rapport trois, du carré minimal représenté figure 3.1 (à gauche) et codé par 6024, donnera le carré décrit par le code 666000222444 (à droite sur cette même figure).

Regardons ce qui se passe réellement lors de ce processus. Pour cela, considérons le voisinage V_4 comme un objet dont on cherche l'homothétie de rapport 2. Cet objet, codé par 7135, est représenté à gauche sur la figure 3.2 page suivante. Au centre de cette figure est présenté ce que l'on obtient par la méthode décrite ci-dessus.

Cette méthode revient en fait à considérer la forme avec une échelle deux fois plus petite que l'échelle initiale. Cependant, elle présente un inconvénient : si l'on reporte sur l'objet homothétique les codants obtenus, ceux-ci ont déformé l'objet de départ.

Principe développé :

Voici le principe développé ici pour obtenir cette homothétie de rapport deux : (des détails seront donnés par la suite)

- calcul des coordonnées du premier pixel de l'homothétique visé ;
- cas particulier du premier codant ;
- pour chaque codant
 - ajout au code de l'homothétique d'un ou plusieurs codants dépendant du codant précédent ;
- cas particulier du dernier codant.

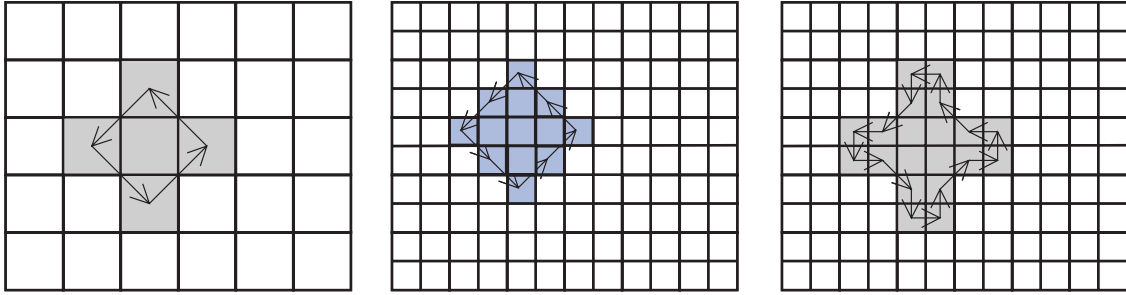


FIGURE 3.2 – Forme étudiée (à gauche), son expansion d'un facteur 2 selon [Mai] (au centre) et l'homothétique que l'on aimerait obtenir (à droite).

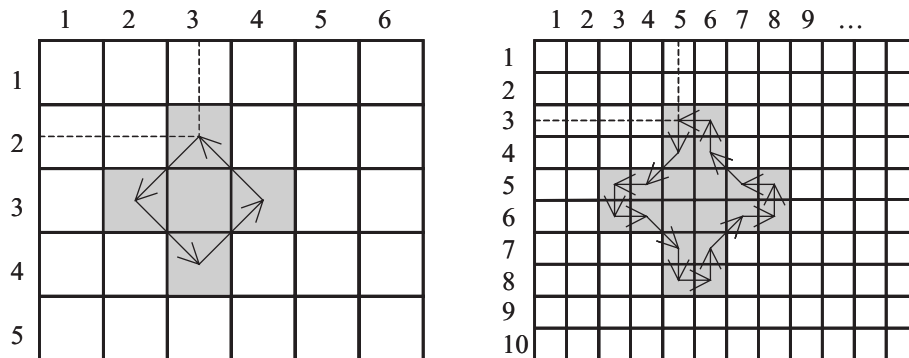


FIGURE 3.3 – Forme de départ (à gauche) et son homothétique de rapport 2 (à droite).

Coordonnées du premier pixel :

Tout d'abord, il faut étudier le cas du premier pixel de la forme dont on désire l'homothétique de rapport deux : si les coordonnées du premier pixel de cette forme sont (x, y) , les coordonnées du premier pixel de son homothétique seront $(2x - 1, 2y - 1)$.

En effet, le passage du codage d'une forme à son homothétique de rapport deux s'apparente à un changement d'échelle du même rapport, et les pixels initiaux sont donc divisés en quatre (du point de vue de l'aire) dans l'homothétique, la distance inter-pixel étant elle-même divisée par deux.

Ainsi, en reprenant l'exemple précédent (voir Figure 3.3) on obtient, pour une forme de départ ayant comme coordonnées pour le premier pixel $(2, 3)$, les coordonnées du premier pixel de l'homothétique : $(3, 5)$.

Convention choisie :

Ensuite, il faut fixer la convention suivante : pour pouvoir étudier le cas du prochain codant, il est nécessaire que le codant précédent arrive toujours au même endroit pour un codant donné. Les points choisis sont illustrés sur la figure 3.4 page suivante. Ces points doivent se retrouver par rotations d'angles multiples de 90° sans distorsion. De plus ils doivent se trouver vers l'extérieur de la forme de départ étudiée pour couvrir l'ensemble des codants suivants possibles. Ils doivent enfin appartenir au codage possible de l'homothétique

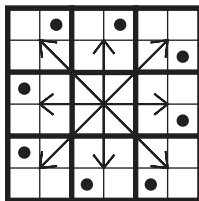


FIGURE 3.4 – Codants (flèches) et points de départs pour les codants suivants (points).

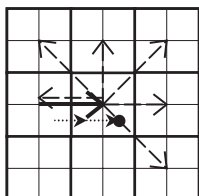


FIGURE 3.5 – Exemple d’obtention d’un des points d’arrivée des codants de l’homothétique de rapport 2.

dans tous les cas possibles de codant suivant. Il sont représentés ici par des points au centre des pixels à l’échelle de l’homothétique. Les différents codants sont représentés par les flèches.

Un exemple d’obtention d’un de ces points est présenté figure 3.5 pour le codant courant 0. Le codant courant est en gras, les codants suivants possibles sont en pointillés longs, et les pointillés courts représentent le chemin et surtout l’arrivée de l’homothétique de rapport deux. (Le codant suivant 4 est décalé par rapport à sa véritable position, afin d’être visible et de ne pas être masqué par le codant courant 0.)

Cas du premier codant :

Le cas du premier codant est à considérer à part. En effet, il faut considérer les quatre cas suivant le premier codant décrivant le contour de la forme à étudier. Ces cas sont présentés dans le tableau de la figure 3.6 : la première colonne donne les premiers codants possibles et la seconde le début du code de la représentation de l’homothétique. Il s’agit en fait pour le code d’aller du pixel de coordonnées $(2x - 1, 2y - 1)$ au point représenté figure 3.4 correspondant au premier codant.

Par exemple, si le premier codant est 0, le code de l’homothétique devra aller du point de coordonnées $(2x - 1, 2y - 1)$ au point de coordonnées $(2x + 1, 2y + 3)$ par le chemin 6000.

Cas général :

Codant de départ	Codants de l’homothétique
6	6 6 6
7	6 0 7 6
0	6 0 0 0
1	6 0 2 1 0

FIGURE 3.6 – Homothétie de rapport deux, cas des premiers codants possibles.

Codant courant	Nouveau codant							
	0	1	2	3	4	5	6	7
0	00	210	222	2432	2444	X	X	76
1	00	210	222	2432	2444	24654	X	76
2	X	10	22	432	444	4654	4666	X
3	X	10	22	432	444	4654	4666	46076
4	6000	X	X	32	44	6545	666	6076
5	6000	60210	X	32	44	654	666	6076
6	000	0210	0222	X	X	54	66	076
7	000	0210	0222	02432	X	54	66	076

FIGURE 3.7 – Cas général pour l’homothétie de rapport deux. (X : cas impossible)

Dernier codant	Codant de l’homothétique
2	4
3	4
4	-
5	-

FIGURE 3.8 – Homothétie de rapport deux, cas des derniers codants possibles. - : pas de rajout.

Le cas général est présenté dans le tableau de la figure 3.7. Il présente les codants à ajouter au code de l’homothétique pour un nouveau codant arrivant après un codant courant. Le principe est de relier le point de la figure 3.4 page ci-contre correspondant au codant courant au point de cette même figure, correspondant au codant suivant. Sur cette figure, les cas impossibles proviennent du choix de départ du sens de rotation trigonométrique pour le codage (voir figure 1.23 page 18).

Cas du dernier codant :

Enfin, après le dernier codant, il est nécessaire de fermer le contour de l’objet selon la règle du tableau de la figure 3.8. En effet, il faut relier le dernier point obtenu selon la figure 3.4 page précédente au point de coordonnées $(2x - 1, 2y - 1)$.

Par exemple, si le dernier codant est un 2, la chaîne de codants arrivera en $(2x - 1, 2y)$ et il faudra ajouter le codant 4 en fin de chaîne pour rejoindre le point de coordonnées $(2x - 1, 2y - 1)$.

Exemple :

La figure 3.9 page suivante présente un exemple de forme codée par le code de Freeman (objet le plus à gauche), puis le contour de l’objet reconstruit à partir du code de son contour, vient ensuite le contour de son homothétique de rapport deux reconstruit, et enfin le contour du reconstruit de l’objet obtenu en doublant chaque codant.

Remarques :

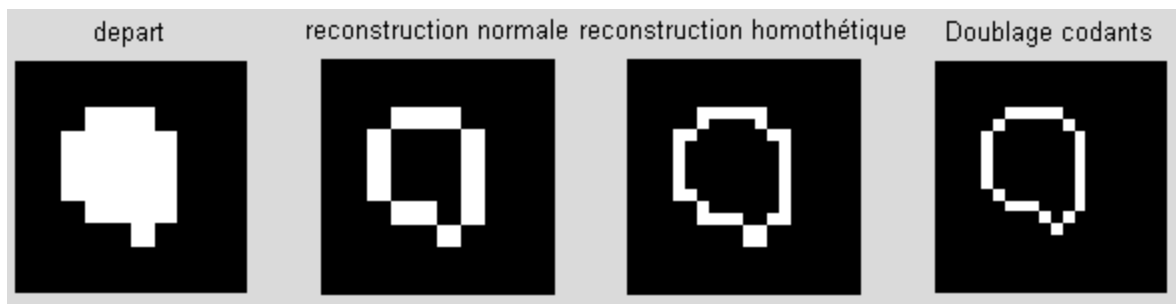


FIGURE 3.9 – Exemple d’objet homothétisé d’un rapport 2.

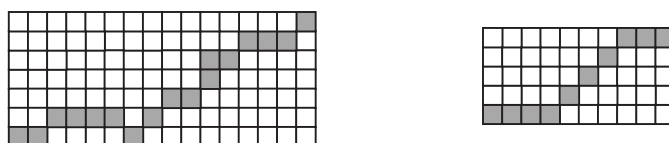


FIGURE 3.10 – Contraction (à droite) d’une chaîne de codants (à gauche).

- Deux des propriétés des homothéties de rapport k sont de multiplier respectivement les distances et les aires par k et k^2 . Le travail que nous avons effectué ici respecte ces deux propriétés alors que l’expansion présentée par [Mai] ne les respecte pas.
- Nous avons également étudié le cas d’une homothétie de rapport trois sur le même principe. Il est envisageable de le faire pour les rapports 5 et 7, nous donnant ainsi la possibilité d’avoir les homothétiques de rapport entier compris entre 2 et 10 (et leurs multiples), par compositions.
- Le cas des homothéties de rapport $1/k$, avec k entier donnera des résultats avec perte d’information et ne sera pas étudié ici. Seule une piste pour $k = 2$ est donnée dans le paragraphe suivant.

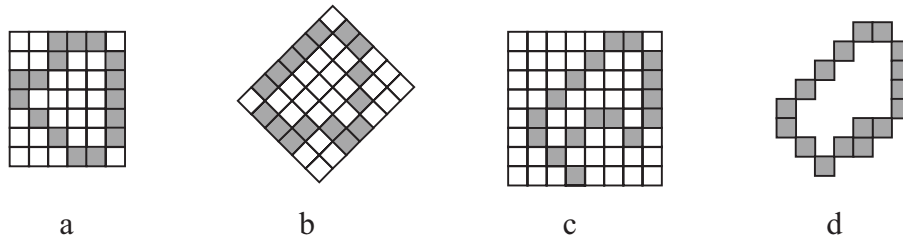
3.1.2.2 Contraction

Toujours d’après [Mai], la contraction est un peu plus complexe et entraîne souvent une perte d’information. En effet, la contraction vise à donner une représentation plus courte et donc plus simple du contour.

Si les directions données par le codage de Freeman peuvent être regroupées par paires, il n’y a pas de perte d’information et la contraction est simple. Dans le cas contraire, on réalise une moyenne des deux directions. Si on ne trouve pas un nombre entier, il faut alors tenir compte de la direction précédente (ou de la suivante).

Après contraction d’une chaîne fermée décrivant le contour d’une forme, il faut bien entendu vérifier que la chaîne contractée est toujours fermée.

L’exemple de la figure 3.10, pour la chaîne 0100071101201001 donnera 00011100 après contraction. La chaîne contractée a été lissée et il y a donc bien eu perte d’information.

FIGURE 3.11 – Rotation (à droite) d'une chaîne (à gauche) de 90° dans le sens direct.FIGURE 3.12 – Comparaison des rotations. a : objet d'origine ; b : rotation réelle de 135° ; c : rotation sans correction ; d : rotation après correction

3.1.3 Rotation

Le codage de Freeman permet également des rotations de la forme étudiée. Si l'on considère le voisinage V_4 , on peut uniquement réaliser des rotations multiples de 90° ($n \times 90^\circ$). Il suffit pour cela d'ajouter n (modulo 4) à chaque élément de la chaîne.

Exemple : considérant un code à quatre directions, la chaîne 32332300 illustrée sur la figure 3.11 donnera la chaîne 03003011 pour une rotation de 90° dans le sens direct.

Le voisinage V_8 permet des rotations d'angles multiples de 45° ($p \times 45^\circ$). Il suffit pour cela d'ajouter p (modulo 8) à chaque composante de la chaîne décrivant le contour de l'image. Si p est pair, l'angle de rotation est alors multiple de 90° et la rotation se fait sans distorsion. En revanche, si p est impair, la rotation engendrera des distorsions provenant du fait que les distances ne sont pas égales entre le point central et les directions paires d'une part et entre le point central et les directions impaires d'autre part. En particulier, on n'est pas du tout sûr de revenir au point de départ.

Sur l'exemple de la figure 3.12, on aperçoit très nettement les distorsions. On pourrait les corriger en partie en évaluant la longueur de chaque segment et en la reportant dans la bonne direction après rotation. Il y a là une piste de travail : correction après polygonalisation.

En poursuivant ce principe, il est également envisageable d'effectuer d'autres rotations avec des angles (par exemple) multiples de 15° . Il suffit pour cela d'ajouter un à un codant sur trois. Par exemple, la droite horizontale codée par 0000000 donnera après rotation de 15° dans le sens direct le code 001001001001.

3.1.4 Symétries

3.1.4.1 Par rapport à un point

Le symétrique $S(X_S, Y_S)$ par rapport à un point $P(X, Y)$ d'un objet dont on connaît le code de Freeman du contour peut être obtenu de façon très simple en deux étapes :

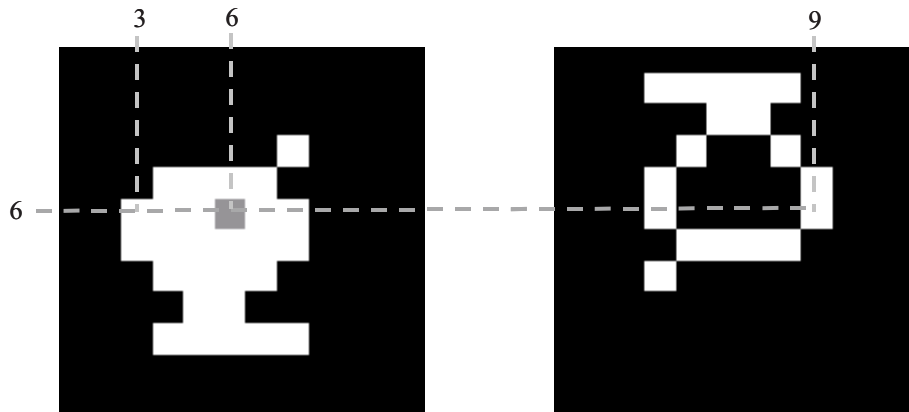


FIGURE 3.13 – Exemple d’objet (à gauche) et son symétrisé par rapport à un point de coordonnées (6,6) (à droite).

- Calcul du point symétrique du point de départ $P_0(X_0, Y_0)$ du code de Freeman

$$\begin{cases} X_S = 2X - X_0 \\ Y_S = 2Y - Y_0 \end{cases} \quad (3.1)$$

- Calcul du symétrique de l’objet en calculant le codant symétrique $C_{i,S}$ de chaque codant de départ C_i :

$$C_{i,S} = (C_i + 4) [8] \quad (3.2)$$

La figure 3.13 présente à gauche un objet commençant au pixel de coordonnées (6, 3) et le recomposé de son symétrisé par rapport au pixel de coordonnées (6, 6) à droite. Le premier pixel de la forme symétrisée a donc comme coordonnées (6, 9). Le code de l’objet de départ est : 6 7 7 5 0 0 0 0 4 3 1 1 2 3 1 5 4 4 4 5 et celui de l’objet symétrisé est : 2 3 3 1 4 4 4 4 0 7 5 5 6 7 5 1 0 0 0 1.

3.1.4.2 Par rapport à un axe

Le symétrique d’un objet par rapport à un axe θ peut être obtenu de façon simple et sans distorsion pour des angles multiples de 45° . Il suffit de remplacer chaque codant par le codant lui correspondant en se reportant au tableau suivant :

Par exemple, le code symétrisé de 6 6 6 0 0 0 3 3 3 par rapport à la droite polaire d’angle 45° est 4 4 4 2 2 2 7 7 7.

Ce paramètre a été présenté par Freeman [Fre74] sous le nom de miroir d’une chaîne et résumé au § 2.1.8 page 31 en quatre équations (voir l’équation 2.14 page 31) qui peuvent finalement se lire en une seule :

$$a_i^m = (8 - 2m - a_i)[8] \quad (3.3)$$

avec m le numéro d’axe utilisé figure 2.3 page 30.

3.1.5 Appartenance d’un point à un objet

Pour savoir si un point X se trouve à l’intérieur ou à l’extérieur de l’objet étudié, il faut faire une vérification. Nous proposons ici plusieurs méthodes.

codant de départ	après symétrie de			
	0°	45°	90°	135°
0	0	2	4	6
1	7	1	3	5
2	6	0	2	4
3	5	7	1	3
4	4	6	0	2
5	3	5	7	1
6	2	4	6	0
7	1	3	5	7

FIGURE 3.14 – Symétrie axiale

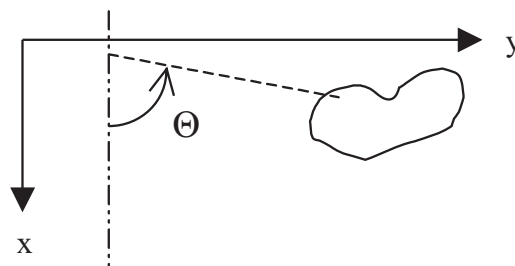


FIGURE 3.15 – Test de l'appartenance d'un point à une forme par une étude angulaire (Le point en question se situe à l'intersection des droites en pointillés).

3.1.5.1 Étude angulaire

Cette première méthode consiste à regarder l'aspect de la courbe donnant l'angle θ entre l'axe des x (droite verticale) d'une part et la droite passant par le point X et le point courant d'autre part. En effet, si cette courbe ne parcourt pas les 360° , il ne se trouve pas à l'intérieur de l'objet. De plus, si cette courbe parcourt l'ensemble des 360° , le point a de fortes chances de se trouver à l'intérieur de l'objet, mais ce n'est pas forcément le cas. En effet, un point se trouvant dans le creux d'un objet ayant la forme d'un C par exemple (ou un O pas tout à fait fermé) donnera une courbe d'angle balayant les 360° .

Une illustration donnant le calcul d'un angle pour un point donné de l'image et un point du contour de la forme étudiée est donnée figure 3.15.

3.1.5.2 Géométrie algorithmique

Une seconde méthode, plus classique en géométrie algorithmique, consiste à considérer le point dans une ligne, à rechercher les points de l'objet se trouvant sur cette ligne et à étudier la position relative du point considéré.

En effet, un point se situant dans l'objet aura à sa gauche, sur la ligne de l'image où il se trouve, un point appartenant à la frontière. De plus, ce point de la frontière sera atteint par un codant venant du haut de l'image, et le codant partant de ce point ira vers le bas de l'image. Ceci est illustré sur la figure 3.16 page suivante.

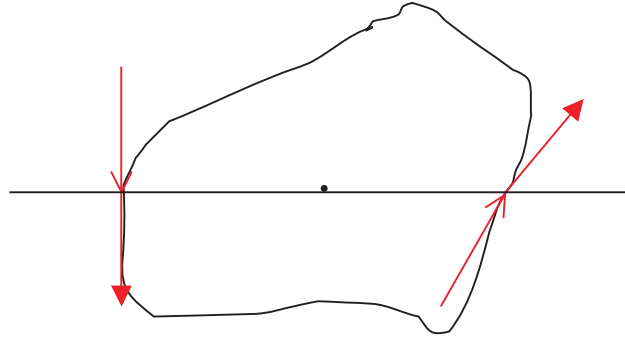


FIGURE 3.16 – Test de l'appartenance d'un point à une forme par une étude de position relative des pixels appartenant à la même ligne.

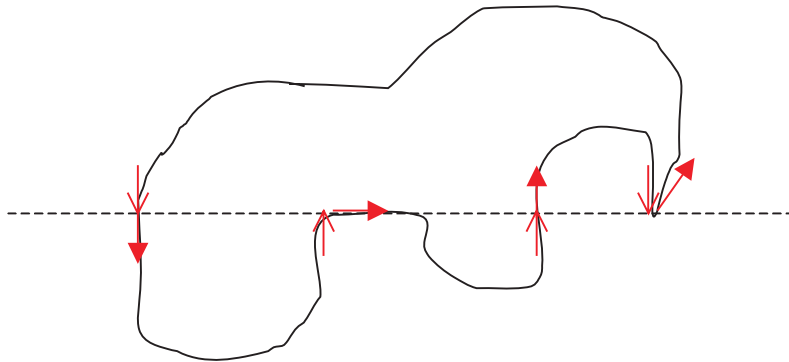


FIGURE 3.17 – Cas particuliers du test d'appartenance d'un point à une forme par une étude de position relative des pixels appartenant à la même ligne.

Remarquons que le point considéré (dont on cherche l'appartenance à la forme) aura à sa droite un point appartenant à la frontière atteint par un codant allant vers le haut et dont le codant partant ira également vers le haut. Cette méthode présente cependant quelques cas particuliers illustrés par la figure 3.17 :

Un point se trouvant sur la ligne en pointillé se trouve à l'extérieur de la forme :

- S'il n'a pas de point de la frontière sur sa ligne (ce qui n'est pas le cas sur la figure 3.17),
- Ou si tous les points de frontière de sa ligne sont à sa droite,
- Ou si tous les points de frontière de sa ligne sont à sa gauche,
- Ou si le point de la frontière le plus proche à sa gauche est traversé par les codants en allant vers le haut.

De plus, un point se trouvant sur la ligne en pointillé est à l'intérieur de l'objet :

- S'il se trouve sur la frontière ou
 - Si le point de la frontière le plus proche à sa gauche est traversé par les codants en allant vers le bas
 - Et si le point de la frontière le plus proche à sa droite est traversé par les codants en allant vers le haut.

Enfin, la dernière condition est redondante par rapport à la précédente. En effet, si le point de la frontière le plus proche à sa gauche est traversé par les codants en allant vers le

bas, alors le point de la frontière le plus proche à sa droite est traversé par les codants en allant vers le haut.

En géométrie algorithmique, il suffit de considérer la ligne à laquelle appartient le pixel X dont on désire tester l'appartenance à la forme étudiée, et de compter le nombre de fois où cette ligne *traverse* la frontière de cette forme avant d'arriver en X . Si ce nombre de fois est impair, alors X appartient à l'objet, sinon, X n'appartient pas à l'objet.

Toutes ces remarques sont valables pour un seul pixel. Nous avons élargi l'usage de ce principe à une ligne, nous permettant ainsi de recréer l'image de la forme de départ à partir du code de Freeman de son contour.

Compte tenu de toutes ces conditions, nous avons développé l'algorithme [11 page suivante](#).

(1) La recherche des points d'un contour de forme dont l'abscisse est la même que celle d'un point connu X se fait de la manière suivante :

En entrée, on a les paramètres suivants :

- le code de Freeman du contour de l'objet
- X_0 : l'abscisse du premier point du contour
- Y_0 : l'ordonnée du premier point du contour
- x : l'abscisse du point connu X
- y : l'ordonnée du point connu X

En sortie, on obtient un tableau de 5 lignes et n colonnes. Chaque colonne concerne un point du contour de l'objet se trouvant sur la même ligne que X . Les cinq informations sont :

- Abscisse du point du contour,
- Ordonnée du point du contour,
- Indice du code partant du point du contour (correspond au quantième pixel codé sur la frontière),
- Codant partant du point X ,
- Codant arrivant au point X .

L' algorithme développé est l'algorithme [12 page 81](#).

3.2 Opération sur deux objets à partir du codage de Freeman

3.2.1 Estimation de la distance géométrique entre formes

L'information absolue donnant le point initial, ainsi que le code de Freeman du contour d'une forme nous permettent de remonter aux coordonnées de tous les points de la frontière interne de cette forme. Ainsi, pour un pixel quelconque de l'image, on peut calculer sa distance géométrique à un objet (au sens de Pythagore). Pour savoir si ce pixel se trouve à l'intérieur ou à l'extérieur de l'objet, voir le § [3.1.5 page 76](#).

Algorithme 11 Restauration de l'image à partir d'un code de Freeman

```

1  pour chaque ligne de l'image faire
2    Recherche des points du contour de l'objet ayant une abscisse  $x$  connue (1), on garde
    alors en mémoire les abscisses, ordonnées du point, les codants arrivant au point et
    partant de lui.
3    Tri par ordre croissant des ordonnées des points d'abscisse commune obtenus.
4    si il y a des points d'abscisse commune alors
5      pour chacun d'eux : faire
6        si le codant partant du point courant est horizontal (0 ou 4) alors
7          propagation du code jusqu'à obtenir un codant non horizontal
8        fin si
9        si le codant arrivant au point courant est horizontal (0 ou 4) alors
10         rétro-propagation du code jusqu'à obtenir un codant non horizontal
11       fin si
12       Marquage de tous les points de la ligne situés à gauche du premier point d'abscisse
       commune : ces points sont hors de l'objet
13       si la courbe traverse la ligne au point courant en allant vers le bas alors
14         marquage des points de l'objet entre le point courant et le point suivant : ces
       points sont dans l'objet.
15         si le point est répété à la colonne précédente et la courbe traverse la ligne au
       point précédent en montant alors
16           marquage des points entre le point courant et le point suivant : ces points
       sont hors de l'objet.
17         fin si
18       fin si
19       si la courbe traverse la ligne au point courant en montant alors
20         marquage des points de l'objet entre le point courant et le point suivant : ces
       points sont en dehors de l'objet.
21         si le point est répété à la colonne précédente et la courbe traverse la ligne au
       point précédent en descendant alors
22           marquage des points entre le point courant et le point suivant : ces points
       sont hors de l'objet
23         fin si
24       fin si
25       si le point courant est un point de rebroussement alors
26         si on est à l'intérieur de l'objet alors
27           marquage des points entre le point courant et le point suivant : ces points
       sont dans l'objet.
28         sinon
29           marquage des points entre le point courant et le point suivant : ces points
       sont hors de l'objet.
30         fin si
31       fin si
32     fin pour
33     sinon si il n'y a pas d'abscisse commune alors
34       marquage de tous les points de la ligne : ces points sont hors de l'objet
35     fin si
36 fin pour

```

Algorithme 12 Recherche de pixels d'un contour ayant une abscisse connue

```

1  $X_1 = X_0$ 
2  $Y_1 = Y_0$ 
3 pour chaque codant faire
4   si  $X_1 = x$  et  $Y_1 = y$  alors
5     Le point est sur la frontière
6   sinon si  $X_1 = x$  alors
7     Sauvegarde des informations du point du contour (abscisse, ordonnée, quantième
      codant, codant partant du point, codant arrivant au point)
8   fin si
9   passage au point suivant : actualisation des coordonnées de  $X_1$  et  $Y_1$  en fonction du
      codant
10 fin pour

```

3.2.1.1 Distance d'un point à un objet

Pour connaître la distance d'un point à un objet, l'algorithme implémenté est l'algorithme 13.

Algorithme 13 Calcul de la distance d'un pixel à la frontière d'un objet

```

1 Calcul de la distance  $d$  entre le point initial du contour et le point  $X$ 
2 Calcul des coordonnées du point suivant du contour
3 tant que le point courant du contour est différent du point initial faire
4   Calcul de la distance  $d'$  entre le point courant du contour et le point  $X$ 
5   Si  $d' < d$  alors  $d = d'$ 
6   Calcul des coordonnées du point suivant du contour
7 fin tant que
8 retourner  $d$ 

```

3.2.1.2 Distance entre deux objets

Pour connaître la distance géométrique séparant deux objets A et B , il suffit de calculer la distance de chaque point de la frontière interne de A à l'objet B (comme précédemment) et de retenir la distance la plus petite, selon l'algorithme 14 page suivante.

3.2.2 Intersection et union de deux formes

Ces deux opérations sont étudiées dans le cas de formes générales. Elles fonctionnent très bien dans le cas de formes convexes, mais méritent d'être approfondies pour des formes plus particulières.

3.2.2.1 Intersection de deux formes

Connaissant deux objets par un point de leur contour et le code de Freeman de leur contour, peut-on connaître le code de l'intersection de ces deux objets ?

Algorithme 14 Calcul de la distance entre deux objets

```

1 Calcul des coordonnées  $(X_1, Y_1)$  et  $(X_2, Y_2)$  des premiers points des deux contours
2  $d = d((X_1, Y_1), (X_2, Y_2))$ 
3 tant que le point courant du second contour est différent du point initial faire
4   tant que le point courant du premier contour est différent du point initial faire
5     Caclul de la distance  $d'$  entre les points courants des deux contours
6     Si  $d' < d$  alors  $d = d'$ 
7     Calcul des coordonnées du point suivant du premier contour
8   fin tant que
9   Calcul des coordonnées du point suivant du second contour
10 fin tant que
11 retourner d

```

Algorithme 15 Calcul du code de l'intersection de deux objets

```

1 Recherche des points communs des frontières de deux objets et recherche des points de
  croisement des courbes, sans intersection.
2 si aucun pixel commun de la frontière n'est trouvé alors
3   si un pixel de la forme 1 est à l'intérieur de la forme 2 alors
4     l'intersection des deux objets correspond au code de la forme 1.
5     Break (Fin de l'algorithme)
6   sinon si un pixel de la forme 2 est à l'intérieur de la forme 1 alors
7     l'intersection correspond au code de la forme 2
8     Break (Fin de l'algorithme)
9   fin si
10 fin si
11 Pour chaque point obtenu, mise en mémoire, pour les 2 courbes de l'abscisse, l'ordonnée,
  l'indice du codant partant, le codant arrivant, le codant partant. (On obtient ainsi un
  tableau à 10 lignes et n colonnes, n étant le nombre de points où les courbes se croisent.
  Ces points peuvent être doubles et plus rarement triples.)
12 pour chaque point du tableau faire
13   recherche du cas en fonction des codants arrivant et partant des 2 courbes
14   si les courbes ont un point commun alors
15     Cas 0 : les 2 codants partant du point de croisement sont les mêmes. On retient
      alors uniquement ce codant.
16     Cas 1 : les codants retenus sont ceux de l'objet 1 à partir de ce point et jusqu'au
      point de croisement suivant.
17     Cas 2 : les codants retenus sont ceux de l'objet 2 à partir de ce point et jusqu'au
      point de croisement suivant.
18     Cas 3 : cette configuration nous donnera pour l'intersection un point isolé que nous
      stockerons.
19   sinon si les courbes n'ont pas de point commun alors
20     suivant le cas de croisement, un codant est ajouté pour raccorder les deux courbes.
      Il existe exactement huit cas représentés figure 3.18 page ci-contre.
21   fin si
22 fin pour

```

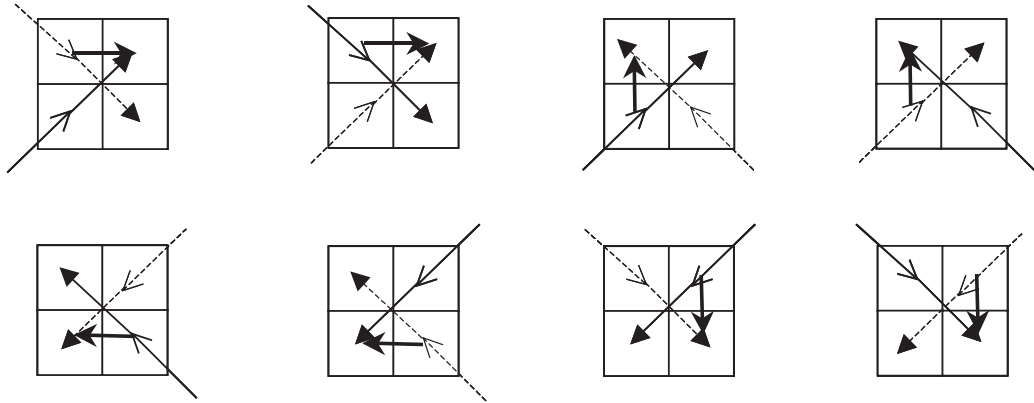


FIGURE 3.18 – Les huit cas de croisements possibles de deux courbes sans intersection.

Pour répondre à cette question, nous avons développé et implémenté l'algorithme [15 page précédente](#).

Cas de croisement :

Les cas possibles de croisement sont au nombre de huit exactement. L'illustration proposée figure [3.18](#) les présente. Pour cette figure :

- les flèches en pointillé représentent les codants de l'objet 2,
- les flèches en trait fin représentent les codants de l'objet 1,
- les flèches en gras représentent les codants à ajouter,
- les extrémités des flèches symbolisent les codants arrivant (pointe de flèche fine), et partant (pointe de flèche en gras) du point considéré.

Sur cette figure, de gauche à droite et de haut en bas :

- les codants à ajouter pour poursuivre l'intersection sont 0, 0, 2, 2, 4, 4, 6 et 6,
- on arrivera alternativement par les codants des formes 2 et 1 (forme 2 en premier),
- on poursuivra le codage de l'intersection alternativement par les codants de la forme 1 et 2 (forme 1 en premier).

Cas généraux :

Les quatre cas (numérotés de 0 à 3) dans l'algorithme dépendent bien entendu des codants arrivant et partant du point de croisement des deux formes dont on désire l'intersection. Disposant de huit possibilités pour chacun des codants, il y a donc 8^4 soit 4096 possibilités de croisement que nous avons étudiées afin de finaliser cette méthode.

L'étude est considérablement simplifiée et se ramène à 1024 cas si l'on prend en compte les rotations. Les 1024 cas correspondant aux codants arrivant 0 et 1 pour la forme 1 sont présentés à l'annexe [B page 119](#). Pour obtenir un cas non mentionné dans ce tableau (par exemple 5476, forme 1 codant arrivant : 5, codant partant : 4, forme 2 codant arrivant : 7, codant partant : 6) il suffit de retrancher le premier codant (ici 5) et d'ajouter 1 aux quatre codants (ce qui donnera ici le cas présenté : 1034 ou on obtiendra le cas 2).

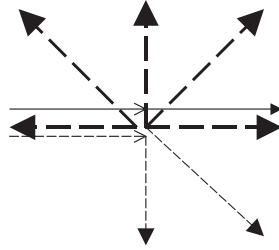


FIGURE 3.19 – Intersection : 8 cas.

Forme 1, codant		Forme 2, codant		Cas
arrivant	partant	arrivant	partant	
0	0	0	0	0
0	0	0	1	2
0	0	0	2	2
0	0	0	3	2
0	0	0	4	2
0	0	0	5	8
0	0	0	6	1
0	0	0	7	1

FIGURE 3.20 – Etude du code de l'intersection de deux formes, cas 0 0 0 X.

Nous présentons ici la méthode globale d'obtention du cas à considérer. Prenons le cas simple des codants arrivant (pointe de flèche fine) et partant (pointe de flèche en gras) 0 et 0 pour la forme 1 (en trait plein sur la figure 3.19) et le codant arrivant 0 pour la forme 2 (en pointillés). Il nous reste à donner le cas obtenu en fonction du codant partant de la forme 2. Nous avons donc ici huit cas à étudier.

Le tableau de la figure 3.20 présente l'objet à considérer en sortie pour cet exemple :

- 0 correspond au cas indifférent (l'intersection est donnée par le codant lui même)
- 1 correspond à la forme 1 (l'intersection est donnée par les codants de la forme 1 jusqu'au prochain point de croisement)
- 2 correspond à la forme 2 (l'intersection est donnée par les codants de la forme 2 jusqu'au prochain point de croisement)
- 3 donnera un point isolé dont nous stockerons les coordonnées.
- 8 est un cas impossible

Les figures 3.21 page suivante et 3.22 page ci-contre présentent respectivement les cas 1 et 3 plus en détail. Sur ces figures, la forme 1 est représentée en gris foncé, la forme 2 en gris clair et les pixels communs aux deux formes sont hachurés. Les codants de la forme 1 sont en trait plein et ceux de la forme 2 sont en pointillés.

Exemple :

La figure 3.23 page suivante présente un exemple d'intersection de deux objets.

3.2.2.2 Union de deux formes

Nous avons ici développé et implémenté un algorithme très semblable à celui employé pour l'intersection : l'algorithme 16.

Algorithme 16 Calcul du code de l'union de deux objets

```

1 Recherche des points communs des frontières de deux objets et recherche des points de
  croisement des courbes, sans intersection.
2 si aucun pixel commun de la frontière n'est trouvé alors
3   si un pixel de la forme 1 est à l'intérieur de la forme 2 alors
4     l'union des deux objets correspond au code de la forme 2.
5     Break (Fin de l'algorithme)
6   sinon si un pixel de la forme 2 est à l'intérieur de la forme 1 alors
7     l'union correspond au code de la forme 1
8     Break (Fin de l'algorithme)
9   fin si
10 fin si
11 Pour chaque point obtenu, mise en mémoire, pour les 2 courbes de l'abscisse, l'ordonnée,
  l'indice du codant partant, le codant arrivant, le codant partant. (On obtient ainsi un
  tableau à 10 lignes et n colonnes, n étant le nombre de points où les courbes se croisent.
  Ces points peuvent être doubles et plus rarement triples.)
12 pour chaque point du tableau faire
13   recherche du cas en fonction des codants arrivant et partant des 2 courbes
14   si les courbes ont un point commun alors
15     Cas 0 : les 2 codants partant du point de croisement sont les mêmes. On retient
      alors uniquement ce codant.
16     Cas 1 : les codants retenus sont ceux de l'objet 1 à partir de ce point et jusqu'au
      point de croisement suivant.
17     Cas 2 : les codants retenus sont ceux de l'objet 2 à partir de ce point et jusqu'au
      point de croisement suivant.
18     Cas 3 : cette configuration nous donnera pour l'union un point isolé que nous sto-
      ckerons.
19   sinon si les courbes n'ont pas de point commun alors
20     suivant le cas de croisement, un codant est ajouté pour raccorder les 2 courbes.
      Il existe exactement 8 cas. Ces 8 cas sont les mêmes que pour l'intersection (voir
      figure 3.18 page 83), mais le raccordement se fait différemment, à savoir qu'il ne
      va plus garder les codants étant à l'intérieur des deux formes, mais ceux étant à
      l'intérieur d'au moins une forme.
21   fin si
22 fin pour

```

Le cas 3 est un cas bien particulier qui mérite une étude approfondie, en particulier quand le codant partant de la forme 2 est l'opposé du codant arrivant ou partant de la forme 1. Il faut alors considérer les codants précédents des codants arrivant des formes 1 et 2, ce qui ajoute un test dans l'algorithme et multiplie par 16 le nombre de cas à étudier. Cette étude n'a pas été complétée et fait donc partie des perspectives à donner à ce travail.

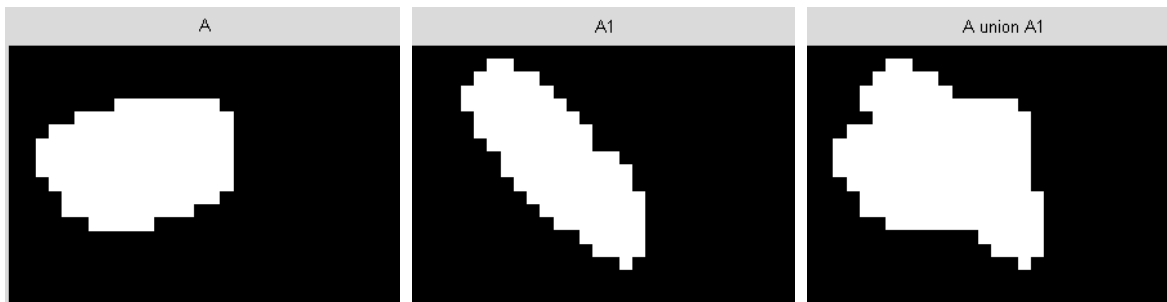


FIGURE 3.24 – Exemple d’union de deux formes.

3.2.3 Comparaison de deux formes

3.2.3.1 Rappel sur les métriques

E étant un ensemble quelconque et d une application de $E \times E$ dans \mathbb{R} , on dit que d est une distance (ou une métrique) sur E si et seulement si elle vérifie les propriétés suivantes :

- $\forall x \in E, \forall y \in E, d(x, y) \geq 0$ (positivité)
- $\forall x \in E, \forall y \in E, d(x, y) = d(y, x)$ (symétrie)
- $\forall x \in E, \forall y \in E, d(x, y) = 0 \Leftrightarrow x = y$ (séparation)
- $\forall (x, y, z) \in E^3, d(x, z) \leq d(x, y) + d(y, z)$ (inégalité triangulaire)

Voyons ici deux exemples de métriques : d_1 et d_∞ .

Notons $\mathbb{F}_I([a, b], \mathbb{R})$ l’ensemble des fonctions à valeurs réelles, définies et intégrables sur un intervalle $[a, b]$ de \mathbb{R} . La fonction d_1 définie pour tout couple (f, g) de \mathbb{F}_I par :

$$d_1(f, g) = \int_a^b |f(x) - g(x)| dx \quad (3.4)$$

est une métrique sur \mathbb{F}_I et représente l’aire comprise entre les deux représentations graphiques de f et g . Cette expression de d_1 se généralise aux images, c’est-à-dire à des fonctions définies sur une même partie D de \mathbb{R}^2 et à valeurs dans $[0, M[$.

La fonction d_∞ est définie comme suit :

$$d_\infty(f, g) = \sup_{x \in a, b} (|f(x) - g(x)|) \quad (3.5)$$

Cette métrique dérive de la norme de la convergence uniforme dans l’espace L^∞ (L^p étant un espace constitué de fonctions dont la puissance p ième est intégrable).

Cette métrique, appliquée en imagerie, est parfaitement adaptée à la détection de petits défauts, même réduits à un pixel.

Définissons maintenant des métriques classiques (métrique de la différence symétrique, métrique de Hausdorff) et beaucoup moins connues (Asplünd) sur l’espace des formes binaires.

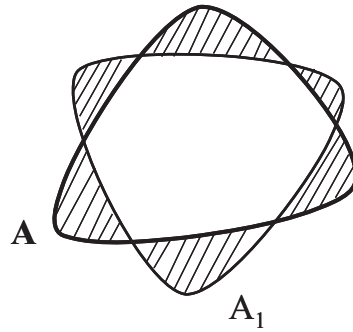


FIGURE 3.25 – Exemple de différence symétrique de deux formes, représentée par l'aire hachurée.

3.2.3.2 Différence symétrique

Ainsi, considérant deux formes A et A_1 leur différence symétrique est donnée par la formule suivante :

$$A \Delta A_1 = (A \cup A_1) \setminus (A \cap A_1) = (A \setminus A_1) \cup (A_1 \setminus A) \quad (3.6)$$

La lettre μ désignant l'aire, la distance de la différence symétrique de A à A_1 est alors définie par :

$$d_{\Delta}(A, A_1) = \mu(A \Delta A_1) \quad (3.7)$$

L'exemple de la figure 3.25 illustre la distance de la différence symétrique entre deux formes.

En utilisant le code de Freeman du contour de deux formes étudiées, l'obtention de ce paramètre se fait selon l'algorithme 17.

Algorithme 17 Calcul de la distance de la différence symétrique entre deux formes

- 1 Obtention du code de Freeman du contour de la forme A
 - 2 Obtention du code de Freeman du contour de la forme A_1
 - 3 Calcul du code de $A \cup A_1$
 - 4 Calcul du code de $A \cap A_1$
 - 5 Calcul de l'aire de $A \cup A_1$
 - 6 Calcul de l'aire de $A \cap A_1$
 - 7 $d_{\Delta} = \mu(A \cup A_1) - \mu(A \cap A_1)$
 - 8 **retourner** d_{Δ}
-

L'union et l'intersection de deux formes par le code de Freeman ont été présentées respectivement aux § 3.2.2.2 page 86 et § 3.2.2.1 page 81.

L'aire d'un objet est calculée également à partir du code de Freeman du contour de la forme étudiée, selon l'une des méthodes présentées au § 2.2.2 page 33.

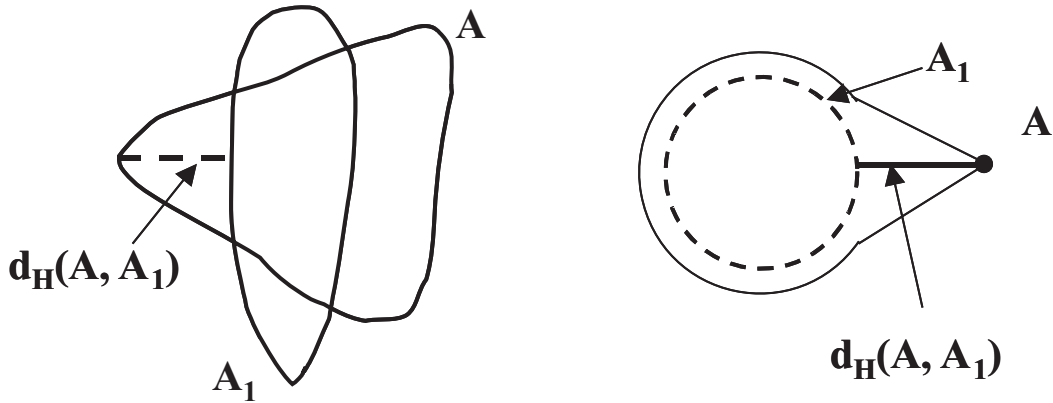


FIGURE 3.26 – Exemples de distance de Hausdorff entre deux formes.

3.2.3.3 Distance de Hausdorff

Pour deux formes A et A_1 du plan, la distance de Hausdorff est donnée par la formule suivante :

$$d_H(A, A_1) = \text{Max} \{ \text{Sup}_{a \in A} (\text{Inf}_{a_1 \in A_1} (d(a, a_1))), \text{Sup}_{a_1 \in A_1} (\text{Inf}_{a \in A} (d(a, a_1))) \} \quad (3.8)$$

La figure 3.26 illustre que cette notion de distance fait ressortir le point (de A ou A_1) le plus éloigné de l'autre forme. Par conséquent, cette mesure de distance est très sensible au bruit.

Ce qui nous intéresse ici est de constater qu'il n'est malheureusement pas possible de calculer la distance de Hausdorff à partir du codage de Freeman du contour de deux objets. En effet, la figure 3.27 page suivante illustre que :

$$d_H(A, A_1) \neq d_H(\text{Fr}(A), \text{Fr}(A_1)) \quad (3.9)$$

où $\text{Fr}(A)$ désigne la frontière de A et donc ce que représente le code de Freeman du contour de la forme A . En effet, sur cette illustration, la distance de Hausdorff entre A et A_1 vaut $d_H(A, A_1)$ = la longueur du segment $[bh]$ alors que la distance de Hausdorff entre la frontière de A et la frontière de A_1 vaut $d_H(\text{Fr}(A), \text{Fr}(A_1))$ = la longueur du segment $[ab]$.

3.2.3.4 Distance de Asplünd

Considérons les formes A et A_1 . L'une d'entre elles (A_1 dans notre exemple) est choisie comme palpeur. On cherche alors le plus grand homothétique de A_1 contenu dans A de rapport d'homothétie λ puis le plus petit homothétique de A_1 de rapport μ et contenant A .

La distance de Asplünd (voir [Asp] et [Jou92]) est alors définie par :

$$d_{As}(A, A_1) = \log\left(\frac{\mu}{\lambda}\right) \quad (3.10)$$

De par les définitions de μ et λ , d_{As} est toujours un nombre positif et est bien une distance.

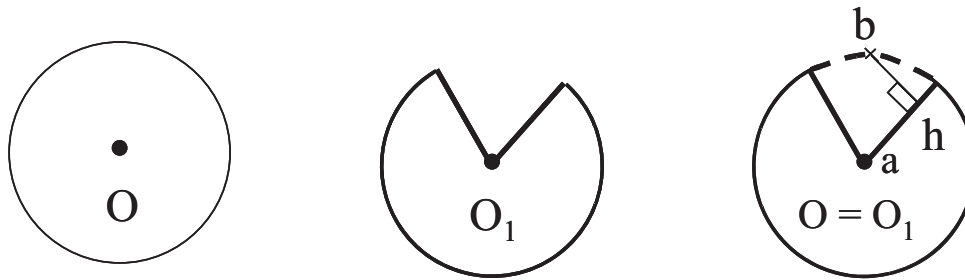


FIGURE 3.27 – Différence entre la distance de Hausdorff entre deux formes et la distance de Hausdorff entre les frontières de ces deux formes.

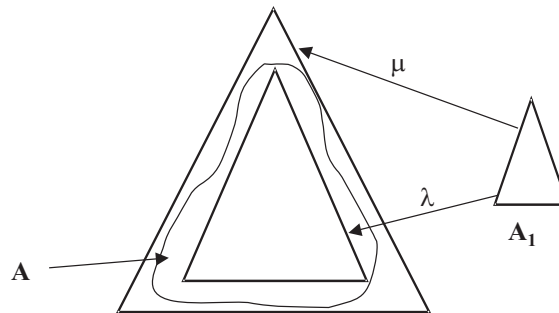


FIGURE 3.28 – Distance de Asplünd calculée grâce à 2 homothétiques.

La figure 3.28 présente le calcul des deux rapports μ et λ .

La méthode suivante permet de calculer la distance de Asplünd entre deux formes A et A_1 :

- Obtention du code de Freeman du contour de la forme A
- Obtention du code de Freeman du contour de la forme A_1
- Choix de A_1 comme palpeur
- Recherche du plus grand homothétique (rapport λ) de A_1 contenu dans A
- Recherche du plus petit homothétique (rapport μ) de A_1 contenant dans A
- Calcul de $d_{As}(A, A_1) = \log(\frac{\mu}{\lambda})$

Le § 3.1.5.2 page 77 nous aidera à savoir si une forme est contenue dans une autre.

La méthode proposée au § 3.1.2 page 70 nous aidera à obtenir le code des homothétiques, mais pas directement le plus grand homothétique contenu dans A ou le plus petit homothétique contenant A .

3.2.4 Corrélation entre deux formes

Freeman [Fre74] propose de calculer le degré de similarité entre deux formes représentées par le code de leur contour en utilisant une fonction de corrélation de chaîne $\Phi_{ab}(j)$. Cette fonction peut être utilisée avec des chaînes décrivant des contours ouverts ou fermés. En considérant deux chaînes $A = a_1a_2\dots a_n$ et $B = b_1b_2\dots b_m$, avec $n \leq m$ le calcul est le suivant :

$$\Phi_{ab}(j) = \frac{1}{n} \sum_{i=1}^n \cos((a_i - b_{i+j})\pi/4) \quad (3.11)$$

Φ donne une indication sur le degré de ressemblance pour différents déplacements de B relativement à A . Elle possède les propriétés suivantes :

$$\forall j \in [1, m] |\Phi_{ab}(j)| \leq 1 \quad (3.12)$$

$$n = m \Rightarrow \Phi_{ab}(j) = \Phi_{ba}(-j) \quad (3.13)$$

Si $A = B$, on obtient $\Phi_{aa}(j)$, fonction d'autocorrélation qui caractérise complètement la chaîne A et peut être utilisée pour classifier les contours étudiés.

3.3 Conclusion

Le code de Freeman du contour d'une forme permet donc, sans repasser à la représentation de l'image complète, de réaliser des transformations simples en obtenant le code du transformé.

Quelques aspects de géométrie algorithmique permettent également d'avoir connaissance de l'appartenance d'un pixel à une forme.

Enfin, considérant deux formes distinctes, il est également possible d'obtenir le code de leur intersection et le code de leur union, nous donnant ainsi accès à des mesures de similarité et de distances entre ces formes.

Chapitre 4

Code de Freeman et morphologie mathématique

Sommaire

4.1 Morphologie mathématique binaire	94
4.1.1 Addition de Minkowski et dilatation	94
4.1.2 Erosion, ouverture et fermeture	95
4.2 Morphologie mathématique et code de Freeman	97
4.2.1 Première approche de la dilatation	97
4.2.2 Nouvelle approche de la dilatation	99
4.2.3 Approches de l'érosion	105
4.3 Conclusion	107

La théorie de la morphologie mathématique binaire est basée sur les opérations ensemblistes de Minkowski, et a été introduite dans les années 1960 par Matheron [Mat] et Serra [Ser82] et [Ser88].

Nous présentons ici un résumé de cet aspect du traitement d'image qui peut être complété par l'ouvrage de Coster et Chermant [Cos].

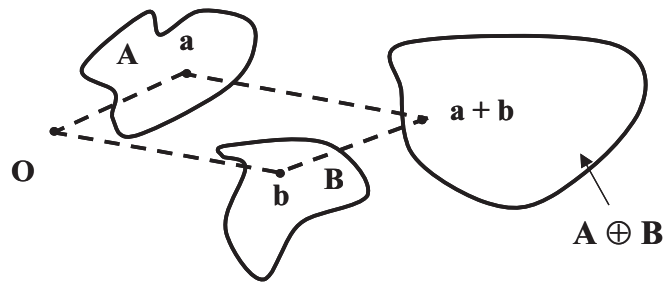


FIGURE 4.1 – Addition de Minkowski.

4.1 Morphologie mathématique binaire

4.1.1 Addition de Minkowski et dilatation

La base mathématique de la morphologie mathématique se trouve dans l'addition de Minkowski (1897) :

soient :

- deux parties quelconques A et B de \mathbb{R}^2
- l'origine O de \mathbb{R}^2

Au couple (A, B) et à O , on associe $A \oplus B = \bigcup(a + b)$, $a \in A, b \in B$, où $a + b$ désigne l'extrémité du vecteur $\overrightarrow{Oa} + \overrightarrow{Ob}$. Une illustration de ce principe est présentée figure 4.1.

Cette addition n'utilise donc que l'addition vectorielle et permet d'ajouter différents ensembles entre eux, sans qu'ils aient a priori de points communs, que ce soit au niveau de la forme ou de la taille.

Prenons maintenant O' comme origine. Pour tout $a \in A$ et $b \in B$, on a alors :

$$\overrightarrow{O'a} + \overrightarrow{O'b} = \overrightarrow{O'O} + \overrightarrow{Oa} + \overrightarrow{O'O} + \overrightarrow{Ob} = \overrightarrow{Oa} + \overrightarrow{Ob} + 2\overrightarrow{O'O}.$$

Ainsi, le résultat $A \oplus B$ est indépendant du choix de l'origine, à une translation près : $2\overrightarrow{O'O}$. De plus, si la partie B est réduite à un seul point b , on obtient A_b le translaté de A par b :

$$A \oplus B = \bigcup_{a \in A} (a + b) = (\bigcup_{a \in A} a) + b = A + b = A_b.$$

La morphologie mathématique binaire n'utilise qu'un cas particulier de cette addition de Minkowski en faisant jouer un rôle différent à A et B :

- A est l'objet à étudier.
- B est appelé l'élément structurant choisi par l'utilisateur.

Dans le cadre de notre étude d'objet grâce au code de Freeman, A est un objet blanc sur fond noir, issu de la binarisation et étudié avec le codage de Freeman ; B est ici un disque centré en O . Nous ne verrons plus loin que le cas des 2 disques élémentaires principaux, à savoir le voisinage V_4 et le voisinage V_8 dans le cadre discret.

En considérant ce cas particulier, $A \oplus B$ est appelée la dilatation de A par B et est illustrée sur la figure 4.2 page suivante. Sur cette figure, a_1 , a_2 et a_3 jouent chacun un rôle différent :

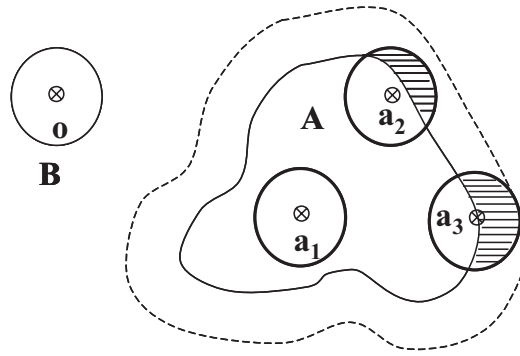


FIGURE 4.2 – Dilatation de A par B.

- $a_1 \in A$
- a_2 ajoute à $A \oplus B$ une petite partie de A^C
- a_3 est situé sur la frontière de A et apporte à $A \oplus B$ une portion de disque de rayon r_B .

Ceci s'explique simplement en écrivant :

$$A \oplus B = \bigcup_{a \in A} (\bigcup_{b \in B} (a + b)) = \bigcup_{a \in A} (B_a)$$

c'est-à-dire que le dilaté de A par B n'est autre que l'union des translatés de B par tous les $a \in A$.

De plus, les points a de type a_3 (voir figure 4.2) appartenant à la frontière de A , notée $Fr(A)$, jouent le rôle le plus important dans la dilatation de A par B . On peut donc écrire :

$$A \oplus B = A \cup (\bigcup_{a \in Fr(A)} (B_a)).$$

En imaginant maintenant l'élément structurant B roulant sur la frontière de l'objet à étudier A , la trajectoire du centre de B donne la frontière du dilaté de A .

On a donc : $Fr(A \oplus B) = \{x \in \mathbb{R}^2, d(x, A) = r_B\}$ et $A \oplus B = \{x \in \mathbb{R}^2, d(x, A) \leq r_B\}$.

Ainsi, des dilatations successives propageront à partir de A une distance par pas de r_B .

La dilatation :

- est commutative : $A \oplus B = B \oplus A$,
- ne conserve pas la connexité : le dilaté de deux objets proches "géographiquement" peut ne former qu'un seul objet,
- est invariante par translation : $A_x \oplus B = (A \oplus B)_x$,
- est compatible avec la multiplication par un réel λ , symbolisant une homothétie de rapport λ : $\lambda(A \oplus B) = \lambda A \oplus \lambda B$,
- est croissante : $A \subseteq A' \Rightarrow A \oplus B \subseteq A' \oplus B$,
- est extensive : $A \subseteq (A \oplus B)$.

4.1.2 Erosion, ouverture et fermeture

L'érodé de A par B , noté $A \ominus B$ et représenté figure 4.3 page suivante, est défini par :

$$A \ominus B = (A^C \oplus B)^C$$

De même que pour le dilaté, l'érodé peut être obtenu simplement en considérant la trajectoire du centre de l'élément structurant B roulant à l'intérieur de l'objet étudié A .

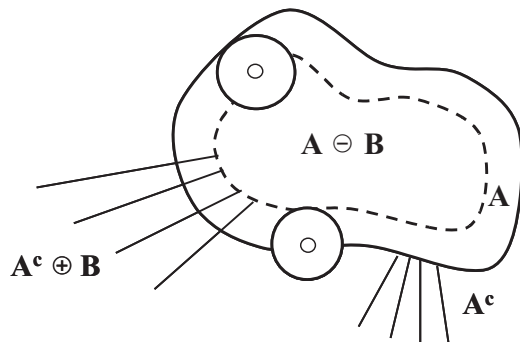


FIGURE 4.3 – Erosion de A par B.

L'érosion :

- est commutative : $A \ominus B = B \ominus A$,
- ne préserve pas la connexité,
- est invariante par translation : $A_x \ominus B = (A \ominus B)_x$,
- est compatible avec la multiplication par un réel λ , symbolisant une homothétie de rapport λ : $\lambda(A \ominus B) = \lambda A \ominus \lambda B$,
- est croissante : $A \subseteq A' \Rightarrow A \ominus B \subseteq A' \ominus B$,
- est anti-extensive : $(A \ominus B) \subseteq A$.

L'érosion et la dilatation peuvent être composées de deux façons différentes :

- L'ouverture de A par B est le dilaté de l'érodé de A. On la note $O(A, B) = (A \ominus B) \oplus B$.

L'ouverture :

- ne préserve pas la connexité,
- est une opération croissante : $A \subseteq A' \Rightarrow O(A, B) \subseteq O(A', B)$,
- est anti-extensive : $O(A, B) \subseteq A$,
- est idempotente : $O(O(A, B), B) = O(A, B)$.

- La fermeture de A par B est l'érodé du dilaté de A. On la note $F(A, B) = (A \oplus B) \ominus B$.

La fermeture :

- ne préserve pas la connexité,
- est une opération croissante : $A \subseteq A' \Rightarrow F(A, B) \subseteq F(A', B)$,
- est extensive : $A \subseteq F(A, B)$,
- est idempotente : $F(F(A, B), B) = F(A, B)$.

La particularité de ces deux opérateurs se trouve dans les différences : $A \setminus [(A \ominus B) \oplus B]$ et $[(A \oplus B) \ominus B] \setminus A$ qui mettent en évidence respectivement les zones sortantes et rentrantes de A, schématisées sur la figure 4.4 page ci-contre. Ces résultats sont dépendants de la taille et de la forme de l'élément structurant.

Pour un élément structurant donné, les opérateurs de morphologie mathématique permettent de compter le nombre de zones rentrantes et sortantes d'un objet, et ainsi déterminer par exemple le nombre d'objets présents dans un amas ou de trier des objets par le nombre de zones sortantes qu'ils possèdent.

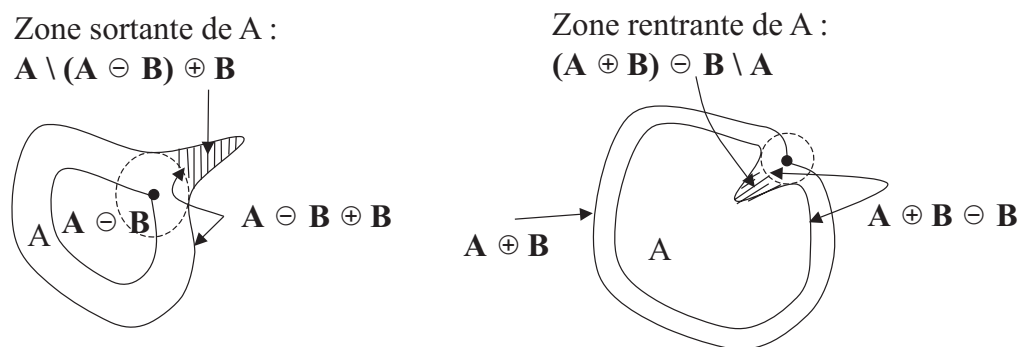


FIGURE 4.4 – Zones sortantes et rentrantes de A.



FIGURE 4.5 – Codant 6 et sa saucisse englobante.

4.2 Morphologie mathématique et code de Freeman

4.2.1 Première approche de la dilatation

Le but est ici d'obtenir, à partir du code de Freeman d'un objet, les codes de Freeman de son érodé et de son dilaté au sens de la morphologie mathématique. Par la suite, nous pourrons utiliser toutes les méthodes de traitement d'objets au sens morphomathématique, directement appliquées sur le code de Freeman.

Comme pour l'ensemble de ce manuscrit, nous étudions ici les objets sans trou car le codage de Freeman simple ne détecte pas ces derniers.

L'idée de départ de David Odin [Odi] pour rechercher le dilaté par un voisinage V_8 est de coder la "saucisse" englobante de chaque codant. C'est-à-dire qu'à un codant, on associe la chaîne de codants faisant le tour du codant de départ. Ainsi, l'exemple de la figure 4.5 montre le codant 6 et son dilaté : 6660022244. Les cas des codants pairs et impairs étant très différents, la saucisse englobante du codant 1 est 5660012244.

Il n'est cependant pas possible d'associer à chaque codant cette saucisse englobante car il faut tenir compte du code précédent dans le cas général et des codes précédents dans quelques cas particuliers.

Pour obtenir cette "saucisse", on se propose de l'initialiser au premier codant comme dans l'exemple ci-dessus. Par la suite, on propage ce point de départ à la suite des codants de l'objet. Pour finir, on réajustera la fin et le début du code de la "saucisse" englobante suivant des règles bien particulières.

Les cas dépendent des huit directions du code de Freeman, mais on peut réduire ce nombre de cas par simple rotation d'un angle multiple de 45° . (Rappelons ici qu'une rotation du code

de Freeman d'un angle de $k \times 45^\circ$ correspond à ajouter $k[8]$ à chaque codant du code de Freeman, voir § 3.1.3 page 75.)

On se base alors sur les codants 0 et 1 qui ont des voisinages V_8 associés très différents. On analyse ensuite le codant suivant et le codant précédent. (Voir le tableau de la figure 4.6)

Codant	précédent	début[-]	début[+]	fin[-]	fin[+]
0-6		-	022	-3	60
0-7		-	224	-2	600
0-0		-1	24	-1	02
0-1		-2	446	-	022
0-2	6	-3	46660222	-3	002224
0-2	$\neq 6$	-3	46	-	224
0-3	7	-5	24244	-	-
0-3	$\neq 7$	-4	46	-	2424
0-4	-	-	60002	-	24446
0-5	3	-5	00202	-	-
0-5	$\neq 3$	-4	00202	-	6
1-7		-	2242	-4	00
1-0		-	24	-2	02
1-1		-1	446	-1	022
1-2		-2	46	-	24
1-3		-4	66	-	4244
1-4		-	-	-	-
1-5		-3	00224	-3	24466
1-6		-	-	-	-

FIGURE 4.6 – Obtention de la saucisse de dilatation pour les codants 0 et 1. [-] : nombre de codants que l'on enlève au code de Freeman courant ; [+] : codants que l'on ajoute ensuite au code de Freeman courant.

Après le déroulement de l'algorithme, on obtient le code de Freeman de la "sauce" du contour. Celle-ci est bien fermée, mais il y a superposition de la "sauce" sur elle-même au niveau du premier et du dernier codant du code étudié.

Cette méthode possède donc ses limites, notamment dans la déduction de l'érodé et du dilaté. Il semble possible d'adopter une autre approche qui consisterait à ne plus utiliser une "sauce" dès le premier codant, mais à suivre le contour (voir § 4.2.2 page ci-contre).

De plus, nous verrons les cas de dilatation et d'érosion en voisinage V_4 et en voisinage V_8 . Ici, David Odin n'a travaillé qu'en voisinage V_8 .

Nous avons vu que cette méthode présente des limites, c'est pourquoi nous nous attacherons à trouver une autre méthode permettant d'obtenir le dilaté et l'érodé d'un objet en travaillant uniquement sur le code de Freeman de son contour. Nous étudierons également la notion d'érodé ultime à partir de cette méthode.

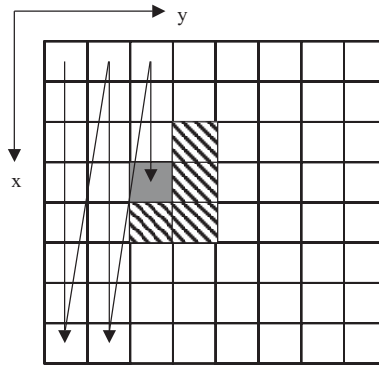


FIGURE 4.7 – Pixel de départ et premiers codants possibles.

4.2.2 Nouvelle approche de la dilatation

4.2.2.1 Dilatation par un voisinage V_4

Le but ici est d'aborder la notion de morphologie mathématique à partir du code de Freeman du contour d'un objet, selon une méthode différente. On ne considère donc pas l'objet complet mais seulement la couronne représentant le code de Freeman de la frontière interne de l'objet. Le dilaté par un voisinage V_4 de l'objet correspond donc à la couronne entourant l'objet de départ. Connaissant le code de Freeman de la première couronne, il est possible d'obtenir le code de la seconde, en travaillant uniquement sur la chaîne de codants.

Cas du premier point :

Rappelons que le premier point du contour de l'objet est obtenu par balayage lexicographique selon la figure 4.7 : le pixel foncé est le premier pixel allumé trouvé, et les pixels hachurés correspondent aux pixels voisins possibles. En effet, dans ces conditions de balayage représenté par les flèches, le premier codant du code de Freeman du contour d'un objet ne peut être que 6, 7, 0 ou 1.

Les coordonnées du premier point du code de Freeman du contour de l'objet dont on désire le code du dilaté sont (x, y) . Prenons comme premier point du dilaté le point de coordonnées $(x, y - 1)$. Ce point nous paraît judicieux car avec les mêmes conventions de balayage, il correspond au point de départ réel du dilaté de l'objet par un voisinage V_4 avec une méthode classique de dilatation. Il s'agit donc du pixel situé immédiatement à gauche du pixel foncé sur la figure 4.7.

Cas du premier codant :

Sur la figure 4.8 page suivante, le premier codant du code à dilater est représenté par des pixels, celui de départ étant hachuré et celui d'arrivée étant vide. Le codage du dilaté est représenté par des flèches.

Il faut s'arrêter ici car les codants pouvant suivre le codant 6 (en trait plein sur la figure 4.9 page suivante) sont 5, 6, 7, 0, 1 et 2 (en pointillé). Le code du dilaté (en gras) doit s'arrêter avant les pixels pouvant être atteints par le dilaté du codant suivant. Les autres cas sont identiques et obtenus par rotation de ce cas donné en exemple.

Illustration Codant de départ Codant du dilaté





	6	6
	7	7
	0	70
	1	71

FIGURE 4.8 – Dilatation par un voisinage V_4 des premiers codants possibles.

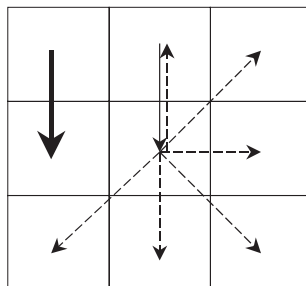


FIGURE 4.9 – Arrêt du codage de la dilatation par un voisinage V_4 pour le codant 6.

Cas général :

Le code du dilaté dépend bien entendu du codant actuel, mais également du codant précédent. Pour chaque cas, il suffit d'ajouter au code du dilaté les codants selon le tableau de la figure 4.10.

Ancien codant	Nouveau codant							
	0	1	2	3	4	5	6	7
0	0	1	12	13	134	X	X	-
1	0	1	12	13	134	135	X	-
2	X	-	2	3	34	35	356	X
3	X	-	2	3	34	35	356	357
4	570	X	X	-	4	5	56	57
5	570	571	X	-	4	5	56	57
6	70	71	712	X	X	-	6	7
7	70	71	712	713	X	-	6	7

FIGURE 4.10 – Cas général de la dilatation par un voisinage V_4 . X : cas impossible; - : le code reste inchangé.

Cas du dernier codant :

Après avoir appliqué le cas général au dernier codant, il faut également lui appliquer le correctif représenté sur la figure 4.11 page suivante, afin de revenir au point de départ de l'objet dilaté choisi plus haut. Le dernier codant à dilater est représenté par des pixels, le départ étant un pixel vide et l'arrivée un pixel hachuré. Le dilaté est représenté par des flèches.

Ainsi, en déroulant le code du contour de l'objet, et en respectant les tableaux ci-dessus, on obtiendra le code du dilaté par un voisinage V_4 de la forme considérée.

4.2.2.2 Dilatation par un voisinage V_8

Le principe appliqué ici est exactement le même que celui de la dilatation par un voisinage V_4 . Il suffit simplement de se référer aux tableaux suivants : 4.12 page suivante pour le cas du premier codant, 4.13 page suivante pour le cas général et 4.14 page 103 pour les codants à ajouter après le dernier cas général.

Il est amusant dans le tableau de la figure 4.13 page suivante de remarquer que pour passer de la dilatation par un voisinage V_4 à la dilatation par un voisinage V_8 , on ne fait que remplacer un code à huit directions par le code à quatre directions le plus proche lui correspondant. En effet, par exemple, le codant à huit directions 13 (correspondant à la dilatation du codant 3 précédé du codant 0) aurait été codé par 0224 si l'on n'avait utilisé que les directions paires.





Illustration	Dernier codant	Derniers codants du dilaté
	2	2 3 5
	3	3 5
	4	4 5
	5	5

FIGURE 4.11 – Derniers codants à ajouter pour boucler la dilatation par un voisinage V_4 .

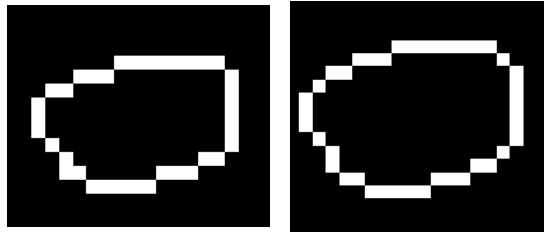
Codant de départ	Codant(s) du dilaté
6	6 6
7	6 6 0
0	6 6 0 0
1	6 6 0 0 2

FIGURE 4.12 – Dilatation par un voisinage V_8 des premiers codants possibles.

Ancien codant	Nouveau codant							
	0	1	2	3	4	5	6	7
0	0	02	022	0224	02244	X	X	-
1	0	02	022	0224	02244	022446	X	-
2	X	-	2	24	244	2446	24466	X
3	X	-	2	24	244	2446	24466	244660
4	46600	X	X	-	4	46	466	4660
5	46600	466002	X	-	4	46	466	4660
6	600	6002	60022	X	X	-	6	60
7	600	6002	60022	600224	X	-	6	60

FIGURE 4.13 – Cas général de la dilatation par un voisinage V_8 . X : cas impossible ; - : le code reste inchangé.

Codant de départ	Codant(s) du dilaté
2	244
3	244
4	4
5	4

FIGURE 4.14 – Derniers codants à ajouter pour boucler la dilatation par un voisinage V_8 .FIGURE 4.15 – Représentation de la frontière d'un objet (à gauche) et de la frontière de son dilaté par un voisinage V_4 (à droite).

4.2.2.3 Problèmes liés à cette méthode

Premier problème :

Regardons d'un peu plus près ce que nous obtenons : dans le cas d'un objet convexe, la dilatation, qu'elle soit faite avec un voisinage V_4 ou un voisinage V_8 , ne pose aucun problème. Un exemple de résultat correct d'un objet dilaté par un voisinage V_4 est présenté sur la figure 4.15.

Le contour de l'objet de gauche a pour code 667760700001001012222234444444544545, et le dilaté de ce contour est codé par : 6677670700001001011222223344444445445455.

En revanche, pour certains objets un peu plus complexes, tels que l'étoile représentée figure 4.16, il y a quelques imperfections à corriger. En effet, on s'aperçoit qu'en appliquant la méthode vue précédemment, on obtient le résultat présenté sur cette même figure.

FIGURE 4.16 – Représentation de la frontière d'une étoile (à gauche), de la frontière de son dilaté par un voisinage V_4 suivant notre méthode (au centre) et de la frontière du dilaté souhaité (à droite).

Ceci provient du fait que la méthode proposée ne prend pas en compte les concavités. Une solution pour y remédier est de corriger le code de l'objet dilaté afin de supprimer ce

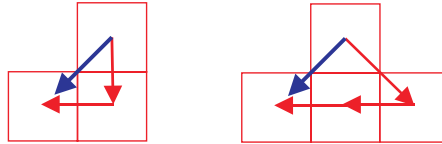


FIGURE 4.17 – Les séquences 6 4 ou 7 4 4 doivent être remplacées par le codant 5.

Séquences impossibles	Codant correspondant
05	6
06	7
16	0
27	0
20	1
30	2
41	2
42	3
52	4
63	4
64	5
74	6

FIGURE 4.18 – Séquences impossibles et codants correspondants.

défaut. Pour cela, il suffit de parcourir le code en recherchant les suites de codants impossibles compte tenu du sens de rotation choisi au départ. Ces séquences correspondent à celles déjà évoquées figure 1.23 page 18. Nous allons ici les compléter dans le tableau de la figure 4.18 et par l'exemple 4.17.

Il existe alors plusieurs possibilités pour corriger le code, mais la plus intéressante consiste en l'algorithme 18.

Algorithme 18 Correction de code après dilatation

```

1 ancien code = code à corriger
2 nouveau code = [] (code vide)
3 tant que nouveau code  $\neq$  ancien code faire
4   nouveau code = ancien code
5   Recherche des séquences impossibles dans ancien code
6   si séquence impossible trouvée alors
7     remplacement par le code correspondant dans le nouveau code
8   fin si
9 fin tant que
10 retourner nouveau code

```

Second problème :

Il existe un second problème lié à cette méthode de dilatation. Nous l'illustrons figure 4.19 page suivante.



FIGURE 4.19 – Représentation de la frontière d’un objet (à gauche), de la frontière de son dilaté par un voisinage V_4 suivant notre méthode (au centre) et de la frontière du dilaté souhaité (à droite).

Le dilaté de l’objet étudié présente la particularité de passer deux fois au même endroit : la dilatation d’une partie de l’objet se superpose avec la dilatation d’une autre partie de l’objet. La solution pour corriger ce problème consiste à suivre la chaîne de codants et à vérifier si elle passe plusieurs fois au même endroit. Si c’est le cas, il faut raccorder correctement les 2 séquences de la chaîne et considérer la chaîne du dilaté en 2 parties. Nous obtenons donc un objet avec un trou, nécessitant une structure de code particulière que nous n’étudions pas ici.

Il est également possible d’étudier la dilatation conditionnelle avec cette méthode : pour cela, il suffit de considérer la dilatation comme nous venons de la voir et de réaliser l’intersection avec la région d’intérêt grâce à la méthode d’intersection vue au § 3.2.2.1 page 81.

4.2.3 Approches de l’érosion

4.2.3.1 Dilatation du complémentaire

Nous avons vu que l’érosion d’un objet est définie comme le complémentaire du dilaté du complémentaire de cet objet (§ 4.1.2 page 95). De la même façon, nous obtenons le code de Freeman de l’érodé d’un objet selon l’algorithme 19.

Algorithme 19 Calcul du code de l’érodé d’une forme

- 1 obtention du complémentaire de A ,
 - 2 dilatation de ce complémentaire par la méthode vue au § 4.2.2 page 99,
 - 3 calcul du complémentaire de ce dilaté.
-

L’obtention du complémentaire d’un objet grâce au code de Freeman s’obtient simplement en prenant la chaîne inverse (voir § 2.1.1 page 26). En effet, d’après notre convention, lorsque l’on suit la chaîne de Freeman du contour d’un objet, on laisse celui-ci sur la gauche. En tournant dans le sens inverse, c’est-à-dire en prenant la chaîne inverse, on a bien l’objet de départ à droite et le complémentaire à gauche.

4.2.3.2 Intersection de translats

Le dilaté d’un objet A par un voisinage V_4 peut se faire en prenant l’union des translats de A dans chacune des 4 directions données par ce voisinage. De même, l’érodé s’obtient en prenant l’intersection de ces translats.

La translation d’un objet décrit par le code de Freeman de son contour (§ 3.1.1 page 70) et les algorithmes d’intersection et d’union (§ 3.2.2.1 page 81 et § 3.2.2.2 page 86) étant

connus, les codes de Freeman de l'érodé et du dilaté peuvent être obtenus de cette façon grâce au code de Freeman du contour de l'objet initial.

4.2.3.3 Erodé ultime

L'algorithme 20 donne l'obtention de l'érodé ultime d'un objet A par un élément structurant B .

Algorithme 20 Calcul du code de l'érodé ultime d'une forme

```

1  Acquisition de  $A$ 
2   $C = A$ 
3   $D = \{\emptyset\}$ 
4  tant que aire( $C$ ) > 0 faire
5     $A = R(C, A)$ 
6     $A = C \setminus A$ 
7     $D = D \cup A$ 
8     $C = C \ominus B$ 
9     $A = C$ 
10 fin tant que
11 retourner  $D$ 

```

C et D sont des mémoires temporaires nécessaires à l'exécution de cet algorithme. L'érodé ultime se trouve au final dans D . Il faut noter ici que $R(C, A)$ désigne la reconstruction par marqueur de C par A . Cette reconstruction par marqueur nécessite les algorithmes de dilatation et d'intersection tels que nous les avons étudiés.

L'algorithme 21 présente la reconstruction par marqueur.

Algorithme 21 Reconstruction par marqueur

```

1  Acquisition de  $A$  (image binaire à reconstruire)
2  Acquisition de  $M$  (masque de reconstruction)
3   $M = M \cap A$  (on conserve uniquement les marqueurs utiles)
4   $r = \text{aire}(M)$ 
5   $s = 0$ 
6  tant que  $r \neq s$  faire
7     $s = r$ 
8     $M = A \cap (M \oplus B)$ 
9     $r = \text{aire}(M)$ 
10 fin tant que
11 retourner  $M$ 

```

Lors de l'exécution de ces algorithmes, il faut faire particulièrement attention à la structure de données utilisée pour stocker le code de Freeman des objets et bien considérer les points isolés introduits par exemple par l'algorithme de calcul de l'intersection de deux objets.

4.3 Conclusion

Le code de Freeman permet donc de réaliser les opérations de base de la morphologie mathématique binaire. Les méthodes étudiées ici sont assez rapides pour des objets convexes et s'avèrent nettement plus lentes pour des objets complexes nécessitant une correction du code obtenu.

Chapitre 5

Perspectives

Sommaire

5.1 Etude sur différents objets types	110
5.2 Code de Freeman en niveaux de gris	110
5.3 Complexité des algorithmes	110
5.4 Etude des propriétés des autres codages	110
5.4.1 Etude du code de Bribiesca	110
5.4.2 Le code de Zeng et Vasseur	111
5.5 Descripteurs topologiques	111
5.6 Floating bodies	111
5.7 Courbure	111
5.8 Aspect multi-échelle	111
5.9 Morphologie mathématique binaire	112
5.10 Supercodage, polygonalisation	112
5.11 Correction de code après rotation	112
5.12 3D	112
5.13 Passage bijectif des différents codes au code de Freeman	112
5.14 Diamètres apparents	112
5.15 Conclusion sur les perspectives	113

5.1 Etude sur différents objets types

Les algorithmes développés dans ce manuscrit ont été appliqués à un ensemble réduit d'objets créés spécialement pour mettre à l'épreuve leur robustesse au bruit et aux cas particuliers des objets discrets, notamment les objets quasi-squelettiques.

Nous aimerions poursuivre cette étude en utilisant un ensemble de formes classiques comme le triangle de Reuleaux, un carré, des triangles remarquables, un losange, des formes convexes ou concaves et des objets réels obtenus après segmentation, par exemple des images rétiniennes ou des empreintes digitales (voir § 5.10 page 112).

5.2 Code de Freeman en niveaux de gris

Le code de Freeman que nous avons utilisé ici n'utilise que les images binaires issues par exemple d'une segmentation. Il serait intéressant d'étudier un codage par plateaux qui nous permettrait de coder complètement une image à niveaux de gris grâce au code de Freeman qui suivrait des lignes de niveau dans l'image.

Cette piste de travail en cours d'étude nécessite une structure de données permettant l'imbrication des formes les unes dans les autres. On peut par exemple utiliser une liste chaînée d'objets contenant le code de Freeman d'un contour d'une forme, associé à son niveau de gris et un (ou plusieurs) liens vers le (ou les) formes contenues dans cette forme.

5.3 Complexité des algorithmes

La complexité des algorithmes n'a pas été suffisamment étudiée dans ce travail et mérite que l'on s'y attarde un peu plus. En effet, certains algorithmes (comme le calcul du coefficient d'asymétrie de Besicovitch) paraissent pouvoir concurrencer des méthodes plus classiques de traitement d'image (et ce malgré l'extraction du code de Freeman du contour de la forme étudiée) alors que d'autres sont beaucoup plus longs et n'apportent pas de gain de temps (comme la morphologie mathématique binaire), malgré le faible nombre de données.

5.4 Etude des propriétés des autres codages

Les autres codages de contour de formes cités au chapitre 1 ne sont pas utilisés dans la littérature. Il est envisageable d'étudier leur propriétés et l'utilisation possible que l'on peut en faire, de la même façon que nous avons étudié le codage de Freeman. Nous serions alors en mesure de comparer ces différents codages.

5.4.1 Etude du code de Bribiesca

Le codage de Bribiesca mérite d'être approfondi et étudié en trame hexagonale. Nous aimerions également étudier les pistes mentionnées au chapitre 1.

5.4.2 Le code de Zeng et Vasseur

Comme nous l'avons vu au § 1.1.5 page 9, ce code nécessite la connaissance a priori du contour de l'objet à étudier et en particulier les segments et les éventuels arcs de cercles qu'il contient et son périmètre. Pour cette raison, nous n'avons pas étudié ce code, mais son étude semble être intéressante lorsque nous aurons développé une polygonalisation des contours grâce au code de Freeman ou grâce à une autre méthode comme celle proposée par Debled et Reveillès (voir [Deb] et [Rev])

5.5 Descripteurs topologiques

Les descripteurs topologiques tels que le nombre d'Euler en 2D (nombre d'Euler = nombre de composants connexes - nombre de trous d'une forme) méritent d'être étudiés grâce au code de Freeman. La structure de données évoquée pour le codage des images à niveaux de gris appliquée aux images binaires donnerait un résultat immédiat.

En créant une autre structure de données, il est également envisageable d'utiliser le codage de Freeman actuel avec d'autres paramètres nous décrivant la présence éventuelle de trous dans la forme. Un trou serait alors codé de la même façon qu'un objet plein, toujours en laissant l'objet à gauche dans le sens de parcours, ce qui ferait "tourner" le code dans le sens inverse du sens utilisé jusqu'ici, c'est-à-dire dans le sens des aiguilles d'une montre.

5.6 Floating bodies

Les lignes de niveaux évoquées au § 2.3.2.1 page 54 sont appelées Floating Bodies dans la littérature. On pourra se référer à [Buc] et [Rei] pour une première approche.

5.7 Courbure

La notion de courbure discrète n'a pas été abordée dans notre étude. Cependant, la norme du produit vectoriel étudié au § 2.3.3 page 59 peut donner une information sur la courbure locale. De plus, il est possible d'avoir accès à cette donnée grâce au code de Freeman et à la signature du contour de la forme étudiée.

5.8 Aspect multi-échelle

Nous avons vu au § 3.1.2 page 70 l'homothétie de rapport k avec $k = 2$. Il est également possible d'étudier la notion d'homothéties de rapport $1/k$, avec k entier ou pour $k > 2$. Dans le premier cas, cette transformation induira une perte de donnée sur le contour de la forme étudiée.

5.9 Morphologie mathématique binaire

Notre travail sur la morphologie mathématique binaire peut se poursuivre par l'étude de l'ouverture et de la fermeture qui sont deux opérations ne nécessitant pas forcément de calculer l'érodé du dilaté ou le dilaté de l'érodé, mais qui peuvent se faire selon un algorithme du même type que ceux présentés aux § 4.2.2 page 99 et 4.2.3 page 105.

5.10 Supercodage, polygonalisation

Le supercodage que nous aurions aimé développer à partir du code de Freeman nous aurait conduit à une représentation vectorielle du contour d'une forme, et donc à une simplification du codage obtenu. Nous aurions alors pu considérer la forme comme une approximation polygonale de l'objet continu numérisé, selon une certaine tolérance.

5.11 Correction de code après rotation

Au § 3.1.3 page 75, nous précisons qu'il est possible de faire des rotations d'angles non multiples de 45° mais avec une certaine perte de données. Nous aimerions approfondir ce résultat et donner un algorithme donnant une correction du code obtenu afin que la forme issue de la rotation corresponde au mieux à la forme de départ, ce qui revient à minimiser l'effet de distorsion induit par la rotation.

Une autre piste consisterait en une correction du code après une étude de la longueur de chaque segment puis rotation d'un angle donné en respectant la longueur et le codage d'un segment de droite. Ceci s'appuyerait sur les algorithmes déjà mentionnés de Reveillès et Debled-Rennesson.

5.12 3D

Le code de Freeman travaille en deux dimensions. Le code introduit par Bribiesca ([Bri99]) permet de donner le codage du contour d'un volume. Ce code permettrait, après une étude plus approfondie, de donner la topographie de la surface codée.

5.13 Passage bijectif des différents codes au code de Freeman

Le passage du code de la tortue de Papert, ou de celui de Sanchez-Cruz, du VCC ou du code de Liu et Zalic au code de Freeman nous semble possible et de façon bijective. Il nous reste à confirmer les algorithmes développés.

5.14 Diamètres apparents

Le § 2.2.5 page 38 n'est applicable qu'à des objets convexes. Pour les objets non convexes, il faut étudier la courbe donnée par la variable *tmp* de l'algorithme 6 page 39. En effet, en

orientant la droite de projection et en cumulant les progressions positives sur cet axe, nous obtenons l'intercept pour la direction θ utilisée.

L'algorithme de cette méthode reste donc à développer.

5.15 Conclusion sur les perspectives

Les perspectives à donner à ce travail sont très nombreuses et il serait intéressant de développer des applications utilisant ces travaux. Nous avons pensé par exemple à la biométrie : une empreinte digitale "super-codée" par le code de Freeman serait-elle différentiable d'une autre ?

Chapitre 6

Conclusion

Dans ce travail de thèse, nous nous sommes intéressés à la géométrie discrète et particulièrement à la représentation des formes binaires par leurs contours. Nous avons présenté différents codes pour réaliser ce codage et nous avons particulièrement étudié le premier d'entre eux, celui de H. Freeman.

Ce code permet d'abord une compression importante, et sans perte de l'information contenue dans les images étudiées. Nous nous sommes attachés à l'extraction de caractéristiques géométriques des formes contenues dans ces images. Ainsi, nous avons présenté différents algorithmes pour obtenir le périmètre, l'aire, les coordonnées du centre de gravité, les rayons et diamètres (minimum, maximum et moyen) apparents, la dimension fractale et le rectangle d'aire minimale englobant une forme.

Nous avons ensuite présenté des paramètres de forme calculables grâce au code de Freeman, comme les paramètres de circularité, de symétrie, les axes principaux d'inertie et un test de convexité de la forme.

Nous avons également traité des algorithmes de transformations que peut subir une forme comme la translation, l'homothétie de rapport entier, la rotation et les symétries.

Nous avons proposé une méthode pour dire si un pixel appartient à une forme connue par son code de Freeman, et donner la distance de ce point au contour de l'objet.

Le code de Freeman permet également de calculer l'intersection et l'union de deux formes et de calculer certaines distances entre les formes étudiées.

Enfin, certains algorithmes de morphologie mathématique sont envisageables grâce à ce codage, comme le calcul du dilaté et de l'érodé d'une forme.

Le code de Freeman est un code parmi différents codes de contour de formes. C'est un outil puissant de compression des images binaires qui permet certains calculs sur des formes, sans pour autant être, en général, plus performant en termes de complexité que les méthodes classiques. En effet, cette méthode de codage induit une perte de temps au départ pour obtenir le code du contour des formes étudiées. Par la suite, tous les paramètres ne sont pas calculables directement avec le code, mais certains sont obtenus rapidement.

Mes apports se situent :

- au second chapitre,
 - dans la section des paramètres géométriques, avec de nouvelles approches pour calculer l'aire d'une forme et les coordonnées de son centre de gravité, les différents rayons et diamètres, une proposition pour obtenir le rectangle englobant d'aire minimale d'une forme, et enfin une approche totalement nouvelle du calcul de la dimension fractale.
 - dans la section paramètres de forme, avec les méthodes de calcul des différents paramètres de circularité et de symétrie, une nouvelle méthode de test de convexité d'une forme, et enfin deux méthodes d'obtention des axes principaux d'inertie.
- au troisième chapitre,
 - dans la section opérations sur un objet, avec les algorithmes d'obtention d'un homothétique, la généralisation des symétries par rapport aux quatre axes principaux (l'axe vertical, l'axe horizontal et les deux diagonales), et le test d'appartenance d'un pixel à une forme.
 - dans la section opération sur deux objets, avec les algorithmes donnant l'intersection et l'union de deux formes.
- au quatrième chapitre, avec les nouvelles approches de la dilatation et de l'érosion d'une forme.

Les perspectives de travail sur ce sujet restent encore nombreuses. Nous nous intéressons particulièrement à l'aspect multi-échelle que pourrait exploiter ce codage.

Annexe A

Notations

Généralités

A, A'	objets ou formes à étudier
a, b	pixels d'un objet
x_i, y_i	abscisse et ordonnée atteintes par le i^{eme} codant d'une chaîne
$a_i, (a_i)^{-1}$	codant numéro i et son inverse
a_{ix}, a_{iy}	déplacements en x et en y induits par le codant a_i
l	pas de la trame
h	hauteur d'une chaîne
w	largeur d'une chaîne
L	longueur d'une chaîne
r_e, r_o	nombre de codants pairs et impairs d'une chaîne
$[8]$	notation usuelle de modulo 8
$V_4(i, j)$ et $V_8(i, j)$	voisinage d'un pixel (i, j)
$FI(A)$ et $FE(A)$	frontière interne et externe de A

Paramètres

$A^-(A)$	symétrisé de A par rapport à a
$a_{BE}(A)$	coefficient d'assymétrie de Besicovitch
$d(a, b)$	distance entre les pixels a et b
$dim_B(A)$	dimension de Bouligand-Minkowski
$DF(A)$	dimension fractale de l'objet A
$D_{min}(A), D_{max}(A), D_{moy}(A)$	diamètres apparents de A
$h_A(O, \theta)$	fonction support
M_{1x}, M_{2x}	premier et second moment par rapoport à l'axe des x
$m_\alpha(A)$	mesure de α -recouvrement
$P(A)$	périmètre de l'objet A
$P_C(A)$	périmètre de Crofton de l'objet A
$r_{min}(A), r_{max}(A), r_{moy}(A)$	rayon apparent minimum, maximum et moyen de A
$r(A)$	rayon du disque inscrit dans A
$R(A)$	rayon du disque circonscrit à A
S	aire contenue entre une chaîne et l'axe des x
$S(A)$	symétrisé de Minkowski de la forme A
$s_B L(A)$	coefficient de symétrie de Blaschke
$s_M(A)$	coefficient de symétrie de Minkowski
$s_W(A)$	coefficient de symétrie de Winternitz
w	fonction de Winternitz
\Re	résidu d'une chaîne
M_{ix}	i^{eme} moment d'une chaîne par rapport à l'axe des x
η	histogramme du code de Freeman d'une forme
$\chi_i(A)$	paramètre de circularité de A
$\mu(A)$	aire de la forme A

Morphologie Mathématique

A^c	complémentaire de l'objet A
B	élément structurant
$Fr(A)$	frontière de A
$A \oplus B$	dilatation de A par B
$A \ominus B$	érosion de A par B
$O(A, B)$	ouverture de A par B
$F(A, B)$	fermeture de A par B

Annexe B

Union et intersection

Ce tableau a en entrée les codants entrants et partants pour les formes 1 et 2 dont on cherche l'union et l'intersection. En sortie, il faut lire :

- 0 : prendre les codants de la forme 1 ou 2 (identiques) jusqu'au point de croisement suivant
- 1 : prendre les codants de la forme 1 jusqu'au point de croisement suivant
- 2 : prendre les codants de la forme 2 jusqu'au point de croisement suivant
- 3 : cas particulier : stocker les coordonnées du pixel considéré
- 8 : cas impossible

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	0	0	0	0	0
0	0	0	1	2	1
0	0	0	2	2	1
0	0	0	3	2	1
0	0	0	4	2	1
0	0	0	5	8	8
0	0	0	6	8	8
0	0	0	7	1	2
0	0	1	0	0	0
0	0	1	1	2	1
0	0	1	2	2	1
0	0	1	3	2	1
0	0	1	4	2	1
0	0	1	5	3	3
0	0	1	6	8	8
0	0	1	7	1	2
0	0	2	0	8	8
0	0	2	1	2	1
0	0	2	2	2	1
0	0	2	3	2	1
0	0	2	4	2	1
0	0	2	5	3	3

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	0	2	6	3	3
0	0	2	7	8	8
0	0	3	0	8	8
0	0	3	1	2	1
0	0	3	2	2	1
0	0	3	3	2	1
0	0	3	4	2	1
0	0	3	5	3	3
0	0	3	6	3	3
0	0	3	7	3	3
0	0	4	0	0	0
0	0	4	1	8	8
0	0	4	2	8	8
0	0	4	3	2	1
0	0	4	4	2	1
0	0	4	5	3	2
0	0	4	6	3	2
0	0	4	7	3	2
0	0	5	0	0	0
0	0	5	1	2	1
0	0	5	2	8	8
0	0	5	3	2	1
0	0	5	4	2	1
0	0	5	5	1	2
0	0	5	6	1	2
0	0	5	7	1	2
0	0	6	0	0	0
0	0	6	1	2	1
0	0	6	2	2	1
0	0	6	3	8	8
0	0	6	4	8	8
0	0	6	5	1	2
0	0	6	6	1	2
0	0	6	7	1	2
0	0	7	0	0	0
0	0	7	1	2	1
0	0	7	2	2	1
0	0	7	3	2	1
0	0	7	4	8	8
0	0	7	5	1	2
0	0	7	6	1	2
0	0	7	7	1	2

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	1	0	0	1	2
0	1	0	1	0	0
0	1	0	2	2	1
0	1	0	3	2	1
0	1	0	4	2	1
0	1	0	5	8	8
0	1	0	6	8	8
0	1	0	7	1	2
0	1	1	0	1	2
0	1	1	1	0	0
0	1	1	2	2	1
0	1	1	3	2	1
0	1	1	4	2	1
0	1	1	5	3	3
0	1	1	6	8	8
0	1	1	7	1	2
0	1	2	0	8	8
0	1	2	1	0	0
0	1	2	2	2	1
0	1	2	3	2	1
0	1	2	4	2	1
0	1	2	5	3	3
0	1	2	6	3	3
0	1	2	7	8	8
0	1	3	0	8	8
0	1	3	1	0	0
0	1	3	2	2	1
0	1	3	3	2	1
0	1	3	4	2	1
0	1	3	5	3	3
0	1	3	6	3	3
0	1	3	7	3	3
0	1	4	0	3	3
0	1	4	1	8	8
0	1	4	2	8	8
0	1	4	3	2	1
0	1	4	4	2	1
0	1	4	5	3	3
0	1	4	6	3	3
0	1	4	7	3	3
0	1	5	0	3	2
0	1	5	1	0	0
0	1	5	2	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	1	5	3	2	1
0	1	5	4	2	1
0	1	5	5	3	2
0	1	5	6	3	2
0	1	5	7	3	2
0	1	6	0	3	2
0	1	6	1	0	0
0	1	6	2	2	1
0	1	6	3	8	8
0	1	6	4	8	8
0	1	6	5	3	2
0	1	6	6	3	2
0	1	6	7	3	2
0	1	7	0	1	2
0	1	7	1	0	0
0	1	7	2	2	1
0	1	7	3	2	1
0	1	7	4	8	8
0	1	7	5	1	2
0	1	7	6	1	2
0	1	7	7	1	2
0	2	0	0	1	2
0	2	0	1	1	2
0	2	0	2	0	0
0	2	0	3	2	1
0	2	0	4	2	1
0	2	0	5	8	8
0	2	0	6	8	8
0	2	0	7	1	2
0	2	1	0	1	2
0	2	1	1	1	2
0	2	1	2	0	0
0	2	1	3	2	1
0	2	1	4	2	1
0	2	1	5	3	3
0	2	1	6	8	8
0	2	1	7	1	2
0	2	2	0	8	8
0	2	2	1	1	2
0	2	2	2	0	0
0	2	2	3	2	1
0	2	2	4	2	1
0	2	2	5	3	3

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	2	2	6	3	3
0	2	2	7	8	8
0	2	3	0	8	8
0	2	3	1	1	2
0	2	3	2	0	0
0	2	3	3	2	1
0	2	3	4	2	1
0	2	3	5	3	3
0	2	3	6	3	3
0	2	3	7	3	3
0	2	4	0	3	3
0	2	4	1	8	8
0	2	4	2	8	8
0	2	4	3	2	1
0	2	4	4	2	1
0	2	4	5	3	3
0	2	4	6	3	3
0	2	4	7	3	3
0	2	5	0	3	3
0	2	5	1	3	3
0	2	5	2	8	8
0	2	5	3	2	1
0	2	5	4	2	1
0	2	5	5	3	3
0	2	5	6	3	3
0	2	5	7	3	3
0	2	6	0	3	3
0	2	6	1	3	3
0	2	6	2	0	0
0	2	6	3	8	8
0	2	6	4	8	8
0	2	6	5	3	3
0	2	6	6	3	3
0	2	6	7	3	3
0	2	7	0	1	2
0	2	7	1	1	2
0	2	7	2	0	0
0	2	7	3	2	1
0	2	7	4	8	8
0	2	7	5	1	2
0	2	7	6	1	2
0	2	7	7	1	2

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	3	0	0	1	2
0	3	0	1	1	2
0	3	0	2	1	2
0	3	0	3	0	0
0	3	0	4	2	1
0	3	0	5	8	8
0	3	0	6	8	8
0	3	0	7	1	2
0	3	1	0	1	2
0	3	1	1	1	2
0	3	1	2	1	2
0	3	1	3	0	0
0	3	1	4	2	1
0	3	1	5	3	3
0	3	1	6	8	8
0	3	1	7	1	2
0	3	2	0	8	8
0	3	2	1	1	2
0	3	2	2	1	2
0	3	2	3	0	0
0	3	2	4	2	1
0	3	2	5	3	3
0	3	2	6	3	3
0	3	2	7	8	8
0	3	3	0	8	8
0	3	3	1	1	2
0	3	3	2	1	2
0	3	3	3	0	0
0	3	3	4	2	1
0	3	3	5	3	3
0	3	3	6	3	3
0	3	3	7	3	3
0	3	4	0	3	3
0	3	4	1	8	8
0	3	4	2	8	8
0	3	4	3	0	0
0	3	4	4	2	1
0	3	4	5	3	2
0	3	4	6	3	2
0	3	4	7	3	2
0	3	5	0	3	3
0	3	5	1	3	3
0	3	5	2	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	3	5	3	0	0
0	3	5	4	2	1
0	3	5	5	3	3
0	3	5	6	3	3
0	3	5	7	3	3
0	3	6	0	3	3
0	3	6	1	3	3
0	3	6	2	3	3
0	3	6	3	8	8
0	3	6	4	8	8
0	3	6	5	3	3
0	3	6	6	3	3
0	3	6	7	3	3
0	3	7	0	3	3
0	3	7	1	3	3
0	3	7	2	3	3
0	3	7	3	0	0
0	3	7	4	8	8
0	3	7	5	3	3
0	3	7	6	3	3
0	3	7	7	3	3
0	4	0	0	1	2
0	4	0	1	1	2
0	4	0	2	1	2
0	4	0	3	1	2
0	4	0	4	0	0
0	4	0	5	8	8
0	4	0	6	8	8
0	4	0	7	1	2
0	4	1	0	1	2
0	4	1	1	1	2
0	4	1	2	1	2
0	4	1	3	1	2
0	4	1	4	0	0
0	4	1	5	3	3
0	4	1	6	8	8
0	4	1	7	1	2
0	4	2	0	8	8
0	4	2	1	1	2
0	4	2	2	1	2
0	4	2	3	1	2
0	4	2	4	0	0
0	4	2	5	3	3

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	4	2	6	3	3
0	4	2	7	8	8
0	4	3	0	8	8
0	4	3	1	1	2
0	4	3	2	1	2
0	4	3	3	1	2
0	4	3	4	0	0
0	4	3	5	3	3
0	4	3	6	3	3
0	4	3	7	3	3
0	4	4	0	3	3
0	4	4	1	8	8
0	4	4	2	8	8
0	4	4	3	1	2
0	4	4	4	0	0
0	4	4	5	3	3
0	4	4	6	3	3
0	4	4	7	3	3
0	4	5	0	3	3
0	4	5	1	3	3
0	4	5	2	8	8
0	4	5	3	1	2
0	4	5	4	0	0
0	4	5	5	3	3
0	4	5	6	3	3
0	4	5	7	3	3
0	4	6	0	3	3
0	4	6	1	3	3
0	4	6	2	3	3
0	4	6	3	8	8
0	4	6	4	8	8
0	4	6	5	3	3
0	4	6	6	3	3
0	4	6	7	3	3
0	4	7	0	3	3
0	4	7	1	3	3
0	4	7	2	3	3
0	4	7	3	3	3
0	4	7	4	8	8
0	4	7	5	3	3
0	4	7	6	3	3
0	4	7	7	3	3

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	5	0	0	8	8
0	5	0	1	8	8
0	5	0	2	8	8
0	5	0	3	8	8
0	5	0	4	8	8
0	5	0	5	8	8
0	5	0	6	8	8
0	5	0	7	8	8
0	5	1	0	8	8
0	5	1	1	8	8
0	5	1	2	8	8
0	5	1	3	8	8
0	5	1	4	8	8
0	5	1	5	8	8
0	5	1	6	8	8
0	5	1	7	8	8
0	5	2	0	8	8
0	5	2	1	8	8
0	5	2	2	8	8
0	5	2	3	8	8
0	5	2	4	8	8
0	5	2	5	8	8
0	5	2	6	8	8
0	5	2	7	8	8
0	5	3	0	8	8
0	5	3	1	8	8
0	5	3	2	8	8
0	5	3	3	8	8
0	5	3	4	8	8
0	5	3	5	8	8
0	5	3	6	8	8
0	5	3	7	8	8
0	5	4	0	8	8
0	5	4	1	8	8
0	5	4	2	8	8
0	5	4	3	8	8
0	5	4	4	8	8
0	5	4	5	8	8
0	5	4	6	8	8
0	5	4	7	8	8
0	5	5	0	8	8
0	5	5	1	8	8
0	5	5	2	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	5	5	3	8	8
0	5	5	4	8	8
0	5	5	5	8	8
0	5	5	6	8	8
0	5	5	7	8	8
0	5	6	0	8	8
0	5	6	1	8	8
0	5	6	2	8	8
0	5	6	3	8	8
0	5	6	4	8	8
0	5	6	5	8	8
0	5	6	6	8	8
0	5	6	7	8	8
0	5	7	0	8	8
0	5	7	1	8	8
0	5	7	2	8	8
0	5	7	3	8	8
0	5	7	4	8	8
0	5	7	5	8	8
0	5	7	6	8	8
0	5	7	7	8	8
0	6	0	0	8	8
0	6	0	1	8	8
0	6	0	2	8	8
0	6	0	3	8	8
0	6	0	4	8	8
0	6	0	5	8	8
0	6	0	6	8	8
0	6	0	7	8	8
0	6	1	0	8	8
0	6	1	1	8	8
0	6	1	2	8	8
0	6	1	3	8	8
0	6	1	4	8	8
0	6	1	5	8	8
0	6	1	6	8	8
0	6	1	7	8	8
0	6	2	0	8	8
0	6	2	1	8	8
0	6	2	2	8	8
0	6	2	3	8	8
0	6	2	4	8	8
0	6	2	5	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	6	2	6	8	8
0	6	2	7	8	8
0	6	3	0	8	8
0	6	3	1	8	8
0	6	3	2	8	8
0	6	3	3	8	8
0	6	3	4	8	8
0	6	3	5	8	8
0	6	3	6	8	8
0	6	3	7	8	8
0	6	4	0	8	8
0	6	4	1	8	8
0	6	4	2	8	8
0	6	4	3	8	8
0	6	4	4	8	8
0	6	4	5	8	8
0	6	4	6	8	8
0	6	4	7	8	8
0	6	5	0	8	8
0	6	5	1	8	8
0	6	5	2	8	8
0	6	5	3	8	8
0	6	5	4	8	8
0	6	5	5	8	8
0	6	5	6	8	8
0	6	5	7	8	8
0	6	6	0	8	8
0	6	6	1	8	8
0	6	6	2	8	8
0	6	6	3	8	8
0	6	6	4	8	8
0	6	6	5	8	8
0	6	6	6	8	8
0	6	6	7	8	8
0	6	7	0	8	8
0	6	7	1	8	8
0	6	7	2	8	8
0	6	7	3	8	8
0	6	7	4	8	8
0	6	7	5	8	8
0	6	7	6	8	8
0	6	7	7	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	7	0	0	2	1
0	7	0	1	2	1
0	7	0	2	2	1
0	7	0	3	2	1
0	7	0	4	2	1
0	7	0	5	8	8
0	7	0	6	8	8
0	7	0	7	0	0
0	7	1	0	2	1
0	7	1	1	2	1
0	7	1	2	2	1
0	7	1	3	2	1
0	7	1	4	2	1
0	7	1	5	3	
0	7	1	6	8	8
0	7	1	7	0	0
0	7	2	0	8	8
0	7	2	1	2	1
0	7	2	2	2	1
0	7	2	3	2	1
0	7	2	4	2	1
0	7	2	5	3	3
0	7	2	6	3	3
0	7	2	7	8	8
0	7	3	0	8	8
0	7	3	1	2	1
0	7	3	2	2	1
0	7	3	3	2	1
0	7	3	4	2	1
0	7	3	5	1	2
0	7	3	6	1	2
0	7	3	7	0	0
0	7	4	0	2	1
0	7	4	1	8	8
0	7	4	2	8	8
0	7	4	3	2	1
0	7	4	4	2	1
0	7	4	5	1	2
0	7	4	6	1	2
0	7	4	7	0	0
0	7	5	0	2	1
0	7	5	1	2	1
0	7	5	2	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
0	7	5	3	2	1
0	7	5	4	2	1
0	7	5	5	1	2
0	7	5	6	1	2
0	7	5	7	0	0
0	7	6	0	2	1
0	7	6	1	2	1
0	7	6	2	2	1
0	7	6	3	8	8
0	7	6	4	8	8
0	7	6	5	1	2
0	7	6	6	1	2
0	7	6	7	0	0
0	7	7	0	2	1
0	7	7	1	2	1
0	7	7	2	2	1
0	7	7	3	2	1
0	7	7	4	8	8
0	7	7	5	1	2
0	7	7	6	1	2
0	7	7	7	0	0
1	0	0	0	0	0
1	0	0	1	2	1
1	0	0	2	2	1
1	0	0	3	2	1
1	0	0	4	2	1
1	0	0	5	8	8
1	0	0	6	8	8
1	0	0	7	1	2
1	0	1	0	0	0
1	0	1	1	2	1
1	0	1	2	2	1
1	0	1	3	2	1
1	0	1	4	2	1
1	0	1	5	2	1
1	0	1	6	8	8
1	0	1	7	1	2
1	0	2	0	8	8
1	0	2	1	2	1
1	0	2	2	2	1
1	0	2	3	2	1
1	0	2	4	2	1
1	0	2	5	2	1

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	0	2	6	3	3
1	0	2	7	8	8
1	0	3	0	8	8
1	0	3	1	2	1
1	0	3	2	2	1
1	0	3	3	2	1
1	0	3	4	2	1
1	0	3	5	2	1
1	0	3	6	3	3
1	0	3	7	3	3
1	0	4	0	0	0
1	0	4	1	8	8
1	0	4	2	8	8
1	0	4	3	2	1
1	0	4	4	2	1
1	0	4	5	2	1
1	0	4	6	1	2
1	0	4	7	1	2
1	0	5	0	0	0
1	0	5	1	2	1
1	0	5	2	8	8
1	0	5	3	2	1
1	0	5	4	2	1
1	0	5	5	2	1
1	0	5	6	1	2
1	0	5	7	1	2
1	0	6	0	0	0
1	0	6	1	2	1
1	0	6	2	2	1
1	0	6	3	8	8
1	0	6	4	8	8
1	0	6	5	2	1
1	0	6	6	1	2
1	0	6	7	1	2
1	0	7	0	0	0
1	0	7	1	2	1
1	0	7	2	2	1
1	0	7	3	2	1
1	0	7	4	8	8
1	0	7	5	2	1
1	0	7	6	1	2
1	0	7	7	1	2

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	1	0	0	1	2
1	1	0	1	0	0
1	1	0	2	2	1
1	1	0	3	2	1
1	1	0	4	2	1
1	1	0	5	8	8
1	1	0	6	8	8
1	1	0	7	1	2
1	1	1	0	1	2
1	1	1	1	0	0
1	1	1	2	2	1
1	1	1	3	2	1
1	1	1	4	2	1
1	1	1	5	2	1
1	1	1	6	8	8
1	1	1	7	1	2
1	1	2	0	8	8
1	1	2	1	0	0
1	1	2	2	2	1
1	1	2	3	2	1
1	1	2	4	2	1
1	1	2	5	2	1
1	1	2	6	3	3
1	1	2	7	8	8
1	1	3	0	8	8
1	1	3	1	0	0
1	1	3	2	2	1
1	1	3	3	2	1
1	1	3	4	2	1
1	1	3	5	2	1
1	1	3	6	3	3
1	1	3	7	3	3
1	1	4	0	3	
1	1	4	1	8	8
1	1	4	2	8	8
1	1	4	3	2	1
1	1	4	4	2	1
1	1	4	5	2	1
1	1	4	6	3	3
1	1	4	7	3	3
1	1	5	0	3	3
1	1	5	1	0	0
1	1	5	2	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	1	5	3	2	1
1	1	5	4	2	1
1	1	5	5	2	1
1	1	5	6	3	3
1	1	5	7	3	3
1	1	6	0	1	2
1	1	6	1	0	0
1	1	6	2	2	1
1	1	6	3	8	8
1	1	6	4	8	8
1	1	6	5	2	1
1	1	6	6	1	2
1	1	6	7	1	2
1	1	7	0	1	2
1	1	7	1	0	0
1	1	7	2	2	1
1	1	7	3	2	1
1	1	7	4	8	8
1	1	7	5	2	1
1	1	7	6	1	2
1	1	7	7	1	2
1	2	0	0	1	2
1	2	0	1	1	2
1	2	0	2	0	0
1	2	0	3	2	1
1	2	0	4	2	1
1	2	0	5	8	8
1	2	0	6	8	8
1	2	0	7	1	2
1	2	1	0	1	2
1	2	1	1	1	2
1	2	1	2	0	0
1	2	1	3	2	1
1	2	1	4	2	1
1	2	1	5	2	1
1	2	1	6	8	8
1	2	1	7	1	2
1	2	2	0	8	8
1	2	2	1	1	2
1	2	2	2	0	0
1	2	2	3	2	1
1	2	2	4	2	1
1	2	2	5	2	1

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	2	2	6	3	3
1	2	2	7	8	8
1	2	3	0	8	8
1	2	3	1	1	2
1	2	3	2	0	0
1	2	3	3	2	1
1	2	3	4	2	1
1	2	3	5	2	1
1	2	3	6	3	3
1	2	3	7	3	3
1	2	4	0	3	3
1	2	4	1	8	8
1	2	4	2	8	8
1	2	4	3	2	1
1	2	4	4	2	1
1	2	4	5	2	1
1	2	4	6	3	3
1	2	4	7	3	3
1	2	5	0	3	3
1	2	5	1	3	3
1	2	5	2	8	8
1	2	5	3	2	1
1	2	5	4	2	1
1	2	5	5	2	1
1	2	5	6	3	3
1	2	5	7	3	3
1	2	6	0	3	3
1	2	6	1	3	3
1	2	6	2	0	0
1	2	6	3	8	8
1	2	6	4	8	8
1	2	6	5	2	1
1	2	6	6	3	3
1	2	6	7	3	3
1	2	7	0	3	3
1	2	7	1	3	3
1	2	7	2	0	0
1	2	7	3	2	1
1	2	7	4	8	8
1	2	7	5	2	1
1	2	7	6	3	3
1	2	7	7	3	3

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	3	0	0	1	2
1	3	0	1	1	2
1	3	0	2	1	2
1	3	0	3	0	0
1	3	0	4	2	1
1	3	0	5	8	8
1	3	0	6	8	8
1	3	0	7	1	2
1	3	1	0	1	2
1	3	1	1	1	2
1	3	1	2	1	2
1	3	1	3	0	0
1	3	1	4	2	1
1	3	1	5	2	1
1	3	1	6	8	8
1	3	1	7	1	2
1	3	2	0	8	8
1	3	2	1	1	2
1	3	2	2	1	2
1	3	2	3	0	0
1	3	2	4	2	1
1	3	2	5	2	1
1	3	2	6	3	3
1	3	2	7	8	8
1	3	3	0	8	8
1	3	3	1	1	2
1	3	3	2	1	2
1	3	3	3	0	0
1	3	3	4	2	1
1	3	3	5	2	1
1	3	3	6	3	3
1	3	3	7	3	3
1	3	4	0	3	3
1	3	4	1	8	8
1	3	4	2	8	8
1	3	4	3	0	0
1	3	4	4	2	1
1	3	4	5	2	1
1	3	4	6	3	3
1	3	4	7	3	3
1	3	5	0	3	3
1	3	5	1	3	3
1	3	5	2	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	3	5	3	0	0
1	3	5	4	2	1
1	3	5	5	2	1
1	3	5	6	3	3
1	3	5	7	3	3
1	3	6	0	3	3
1	3	6	1	3	3
1	3	6	2	3	3
1	3	6	3	8	8
1	3	6	4	8	8
1	3	6	5	2	1
1	3	6	6	3	3
1	3	6	7	3	3
1	3	7	0	3	3
1	3	7	1	3	3
1	3	7	2	3	3
1	3	7	3	0	0
1	3	7	4	8	8
1	3	7	5	2	1
1	3	7	6	3	3
1	3	7	7	3	3
1	4	0	0	3	3
1	4	0	1	3	3
1	4	0	2	3	3
1	4	0	3	3	3
1	4	0	4	0	0
1	4	0	5	8	8
1	4	0	6	8	8
1	4	0	7	3	3
1	4	1	0	1	2
1	4	1	1	1	2
1	4	1	2	1	2
1	4	1	3	1	2
1	4	1	4	0	0
1	4	1	5	2	1
1	4	1	6	8	8
1	4	1	7	1	2
1	4	2	0	8	8
1	4	2	1	1	2
1	4	2	2	1	2
1	4	2	3	1	2
1	4	2	4	0	0
1	4	2	5	2	1

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	4	2	6	3	3
1	4	2	7	8	8
1	4	3	0	8	8
1	4	3	1	1	2
1	4	3	2	1	2
1	4	3	3	1	2
1	4	3	4	0	0
1	4	3	5	2	1
1	4	3	6	3	3
1	4	3	7	3	3
1	4	4	0	3	3
1	4	4	1	8	8
1	4	4	2	8	8
1	4	4	3	1	2
1	4	4	4	0	0
1	4	4	5	2	1
1	4	4	6	3	3
1	4	4	7	3	3
1	4	5	0	3	3
1	4	5	1	3	3
1	4	5	2	8	8
1	4	5	3	1	2
1	4	5	4	0	0
1	4	5	5	2	1
1	4	5	6	3	3
1	4	5	7	3	3
1	4	6	0	3	3
1	4	6	1	3	3
1	4	6	2	3	3
1	4	6	3	8	8
1	4	6	4	8	8
1	4	6	5	2	1
1	4	6	6	3	3
1	4	6	7	3	3
1	4	7	0	3	3
1	4	7	1	3	3
1	4	7	2	3	3
1	4	7	3	3	3
1	4	7	4	8	8
1	4	7	5	2	1
1	4	7	6	3	3
1	4	7	7	3	3

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	5	0	0	3	3
1	5	0	1	3	3
1	5	0	2	3	3
1	5	0	3	3	3
1	5	0	4	3	3
1	5	0	5	8	8
1	5	0	6	8	8
1	5	0	7	3	3
1	5	1	0	1	2
1	5	1	1	1	2
1	5	1	2	1	2
1	5	1	3	1	2
1	5	1	4	1	2
1	5	1	5	0	0
1	5	1	6	8	8
1	5	1	7	1	2
1	5	2	0	8	8
1	5	2	1	1	2
1	5	2	2	1	2
1	5	2	3	1	2
1	5	2	4	1	2
1	5	2	5	0	0
1	5	2	6	3	3
1	5	2	7	8	8
1	5	3	0	8	8
1	5	3	1	1	2
1	5	3	2	1	2
1	5	3	3	1	2
1	5	3	4	1	2
1	5	3	5	0	0
1	5	3	6	3	3
1	5	3	7	3	3
1	5	4	0	3	3
1	5	4	1	8	8
1	5	4	2	8	8
1	5	4	3	1	2
1	5	4	4	1	2
1	5	4	5	0	0
1	5	4	6	3	3
1	5	4	7	3	3
1	5	5	0	3	3
1	5	5	1	3	3
1	5	5	2	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	5	5	3	1	2
1	5	5	4	1	2
1	5	5	5	0	0
1	5	5	6	3	3
1	5	5	7	3	3
1	5	6	0	3	3
1	5	6	1	3	3
1	5	6	2	3	3
1	5	6	3	8	8
1	5	6	4	8	8
1	5	6	5	0	0
1	5	6	6	3	3
1	5	6	7	3	3
1	5	7	0	3	3
1	5	7	1	3	3
1	5	7	2	3	3
1	5	7	3	3	3
1	5	7	4	8	8
1	5	7	5	0	0
1	5	7	6	3	3
1	5	7	7	3	3
1	6	0	0	8	8
1	6	0	1	8	8
1	6	0	2	8	8
1	6	0	3	8	8
1	6	0	4	8	8
1	6	0	5	8	8
1	6	0	6	8	8
1	6	0	7	8	8
1	6	1	0	8	8
1	6	1	1	8	8
1	6	1	2	8	8
1	6	1	3	8	8
1	6	1	4	8	8
1	6	1	5	8	8
1	6	1	6	8	8
1	6	1	7	8	8
1	6	2	0	8	8
1	6	2	1	8	8
1	6	2	2	8	8
1	6	2	3	8	8
1	6	2	4	8	8
1	6	2	5	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	6	2	6	8	8
1	6	2	7	8	8
1	6	3	0	8	8
1	6	3	1	8	8
1	6	3	2	8	8
1	6	3	3	8	8
1	6	3	4	8	8
1	6	3	5	8	8
1	6	3	6	8	8
1	6	3	7	8	8
1	6	4	0	8	8
1	6	4	1	8	8
1	6	4	2	8	8
1	6	4	3	8	8
1	6	4	4	8	8
1	6	4	5	8	8
1	6	4	6	8	8
1	6	4	7	8	8
1	6	5	0	8	8
1	6	5	1	8	8
1	6	5	2	8	8
1	6	5	3	8	8
1	6	5	4	8	8
1	6	5	5	8	8
1	6	5	6	8	8
1	6	5	7	8	8
1	6	6	0	8	8
1	6	6	1	8	8
1	6	6	2	8	8
1	6	6	3	8	8
1	6	6	4	8	8
1	6	6	5	8	8
1	6	6	6	8	8
1	6	6	7	8	8
1	6	7	0	8	8
1	6	7	1	8	8
1	6	7	2	8	8
1	6	7	3	8	8
1	6	7	4	8	8
1	6	7	5	8	8
1	6	7	6	8	8
1	6	7	7	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	7	0	0	2	1
1	7	0	1	2	1
1	7	0	2	2	1
1	7	0	3	2	1
1	7	0	4	2	1
1	7	0	5	8	8
1	7	0	6	8	8
1	7	0	7	0	0
1	7	1	0	2	1
1	7	1	1	2	1
1	7	1	2	2	1
1	7	1	3	2	1
1	7	1	4	2	1
1	7	1	5	2	1
1	7	1	6	8	8
1	7	1	7	0	0
1	7	2	0	8	8
1	7	2	1	2	1
1	7	2	2	2	1
1	7	2	3	2	1
1	7	2	4	2	1
1	7	2	5	2	1
1	7	2	6	3	3
1	7	2	7	8	8
1	7	3	0	8	8
1	7	3	1	2	1
1	7	3	2	2	1
1	7	3	3	2	1
1	7	3	4	2	1
1	7	3	5	2	1
1	7	3	6	1	2
1	7	3	7	0	0
1	7	4	0	2	1
1	7	4	1	8	8
1	7	4	2	8	8
1	7	4	3	2	1
1	7	4	4	2	1
1	7	4	5	2	1
1	7	4	6	1	2
1	7	4	7	0	0
1	7	5	0	2	1
1	7	5	1	2	1
1	7	5	2	8	8

Forme 1, codant		Forme 2, codant		Sortie	
arrivant	partant	arrivant	partant	Intersection	Union
1	7	5	3	2	1
1	7	5	4	2	1
1	7	5	5	2	1
1	7	5	6	1	2
1	7	5	7	0	0
1	7	6	0	2	1
1	7	6	1	2	1
1	7	6	2	2	1
1	7	6	3	8	8
1	7	6	4	8	8
1	7	6	5	2	1
1	7	6	6	1	2
1	7	6	7	0	0
1	7	7	0	2	1
1	7	7	1	2	1
1	7	7	2	2	1
1	7	7	3	2	1
1	7	7	4	8	8
1	7	7	5	2	1
1	7	7	6	1	2
1	7	7	7	0	0

Bibliographie

- [Ami] AMIN A., SINGH S. : *Hand-printed Chinese Character Recognition*, 1996.
- [Asp] ASPLÜND E. : *Comparison between plane symmetric convex bodies and parallelograms*, Math. Scand. 8 , pp 171-180, 1960.
- [Bal] BALLARD D.H., BROWN C.M. : *Computer Vision*, Prentice-Hall, 1982.
- [Ber] DE BERG M., VAN KREVELD M., OVERMARS M., SCHWARZKOPF O. : *Computational Geometry*, Springer, 1997.
- [Bes33] BESICOVITCH A.S. : *On the Sets of Fractional Dimensions (II) : On the Sum of Digits of Real Numbers Represented in The Dyadic System*, Mathematische Annalen, Vol. 110, pp 321-329, 1933.
- [Bes34] BESICOVITCH A.S. : *On the Sets of Fractional Dimensions (IV) : On a Rational Approximation to Real Number*, Journal of the London Mathematical Society, Vol 9, pp 126-137, 1934.
- [Bes37] BESICOVITCH A.S. : *On the Sets of Fractional Dimensions (V) : On Dimensional Numbers of Some Continuous Curves*, Journal of the London Mathematical Society, Vol 12, pp 18-25, 1937.
- [Bes] BESICOVITCH A.S. : *Measure of asymmetry of convex curves*, J. London Math. Soc. 23, 237-240, 1948.
- [Bla] BLASCHKE W. : *Aufgabe 573*, Arch Math Phys (3), 23, pp74, 1920.
- [Blu] BLUM H. : *An associative machine for dealing with the visual fields and some of its biological implications*, Biol.Prot. and Synth. syst.I, pp.244-260, 1962.
- [Bor] Borgefors G. : *Distance transformations in digital images*, Comput. Vis. Graph. Image Process. 34 (1986), pp. 344-371.
- [Bou28] BOULIGAND G. : *Ensembles impropres et nombres dimensionnels*, Bulletin des sciences mathématiques, Vol II, No 52, pp 320-344, 1928.

- [Bou29] BOULIGAND G. : *Sur la notion d'ordre de mesure d'un ensemble fermé*, Bulletin des sciences mathématiques, Vol II, No 52, pp 185-192, 1929.
- [Bre] BRETON R., ANDRES E. : *Vectorisation d'une courbe discrète standard 2D*, 15èmes journées de l'AFIG, décembre 2002.
- [Bri99] BRIBIESCA E. : *A new chain code*, Pattern Recognition 32, 235-251, 1999.
- [Bri00] BRIBIESCA E. : *A chain code for representing 3D curves*, Pattern Recognition, vol 33 pp 755-765, 2000.
- [Buc] BUCHTA C., REITZNER M. : *Equiaffine inner parallel curves of a plane convex body and the convex hulls of randomly chosen points*, Probability Theory and Related Fields, 108, 385-415, Springer, 1997.
- [Cor] CORMEN T., LEISERSON C., RIVEST R. : *Foundations of computer Science*, MIT Press, Cambridge, Massachusetts, 1990.
- [Costa] COSTA L. DA F., CESAR R.M. : *Shape analysis and Classification. Theory and Practice*, CRC Press, 2001.
- [Cos] COSTER M., CHERMANT J.-L. : *Précis d'analyse d'images*, Ed. du CNRS, 1989.
- [Deb] DEBLED-RENNESON I., REVEILLES J.-P. : *Un algorithme linéaire de polygonalisation de courbes discrètes*, 2th Discrete Geometry for Computer Imagery, Grenoble, Septembre 1992.
- [Dor84] DORST L., SMEULDERS A.W.M. : *Discrete Representation of Straight Lines*, IEEE trans on Pattern Analysis and Machine Intelligence, vol. PAMI-6, pp. 450-463, 1984.
- [Dor] DORST L., SMEULDERS A.W.M. : *Discrete Straight Line Segments : Parameters, Primitives and Properties*
- [Est] ESTERMANN T. : *Zwei neue Beweise eines Satzes von Blaschke und Rachenarner*, JBer Deutsch Math Verein, 36, pp197-200, 1927.
- [Fal] FALCONER K.J. : *Fractal Geometry : Mathematical Foundations and Applications*, John Wiley, New York, 1990.
- [Fil] FILERE I. : *Outils mathématiques pour la reconnaissance des formes - Propriétés et applications*, Thèse de Doctorat, Université Jean Monnet (Saint-Etienne), 1995.
- [Fre61a] FREEMAN H. : *On the encoding of arbitrary geometric configurations*, IRE Trans. Electron. Comput. EC-10, 260-268, 1961.

- [Fre61b] FREEMAN H. : *Techniques for the digital computer analysis of chain-encoded arbitrary plane curves*, Proc. Natl Elect. Conf. 17, pp 421-432, Oct. 1961.
- [Fre74] FREEMAN H. : *Computer Processing of Line-Drawing Images*, Computing surveys, Vol 6, No 1, March 1974.
- [Fre75] FREEMAN H. : *Determining the minimum-area encasing rectangle for an arbitrary closed curve*, Communications of the ACM, Volume 18, Number 7, July 1975.
- [Gou] GOUYET JF. : *Physique et Structures Fractales*, Masson, 1992.
- [Grü] GRÜNBAUM B. : *Measure of symmetry for convex sets.*, Proc. Symp. Pure Math. (AMS), 7, pp933-1270, 1963.
- [Hau] HAUSDORFF F. : *Dimension Und Äusseres Mass*, Mathematische Annalen, Vol 79, pp 157-179, 1919.
- [Iiv96] IIVARINEN J. VISA A. : *Shape recognition of irregular objects*. David P. Casasent, editor, Intelligent Robots and Computer Vision XV : Algorithms, Techniques, Active Vision, and Materials Handling, Proc. SPIE 2904, pages 25-32, 1996.
- [Iiv97] IIVARINEN J., PEURA M., SÄRELÄ J. AND VISA A. : *Comparison of Combined Shape Descriptors for Irregular Objects*. In Proceedings of the Eighth British Machine Vision Conference, vol. 2, pp. 430-439, University of Essex, UK, September 8-11, 1997.
- [Jou83] JOURLIN M., LAGET B. : *Un algorithme de fenêtre disquée*. 1983.
- [Jou84] JOURLIN M., LAGET B. : *Contribution de la morphologie directionnelle à la reconnaissance de formes*. Thèse d'état de l'université de Saint Etienne, 7 déc 1984, n° d'ordre 12 et 36.
- [Jou92] JOURLIN M. : *Métriques de formes et applications*, Congrès International sur l'Analyse en Distance, Distancia '92, pp. 279-281, 1992.
- [Kim] KIM J-I., EVANS B.L. : *Predictive shape coding using generic polygon approximation*, Proc IEEE Int Sym on Circuits and Systems, May 31-June 3, vol 5, pp277-280, 1998.
- [Lab] LABOURÉ MJ., JOURLIN M., FILLÈRE I., BECKER JM., FRÉNÉA F. : *Isoperimetric inequalities and shape parameters*, Acta Stereologica, vol 15, n°1, pp65-70, 1996.
- [Liu] LIU Y. K., ZALIK B. : *An efficient chain code with Huffman coding*, Pattern Recognition, Vol 38, pp553-557, 2005.
- [Mad] MADDOURI S.S., AMIRI H., BELAIR A., NOURI N. : *Caractérisation elliptique par coefficients de Fourier du contour du script Arabe en vue de la normalisation des caractères*, Colloque International Francophone sur l'Ecrit et le Document - CIFEd'00, (2000) [inria-00099145 - version 1].

- [Mai] MAÎTRE H. (SOUS LA DIRECTION DE) : *Le traitement des images*, Hermes Lavoisier IC2, 2003.
- [Man] MANDELBROT B. : *Les objets fractals*, Flammarion, Paris, 3ème ed., 1989.
- [Mat] MATHERON G. : *Random Sets and Integral Geometry*, Wiley, 1975.
- [Min01] MINKOWSKI H. : *Über die Begriffe Länge, Oberfläche und Volumen*, Jahr Deut. Math., Vol 9, pp 115-121, 1901.
- [Min] MINKOVSKY H. : *Theorie der konvexer Körper insbesondere Begründung ihres Oberflächenbegriffs*. Ges. Abhandl., Leipzig Berlin, 2, 131-229, 1911.
- [Odi] ODIN D. : *Vers une reconnaissance automatique des profils de visages humains*, Rapport de stage de DEA avec Michel Jourlin et Jean-Marie Becker, CPE, 1995.
- [Pap] PAPERT S. : *Uses of technology to enhance education*, Technical Report 298, AI lab, MIT, 1973.
- [Pea] PEANO G. : *Sur une courbe, qui remplit toute une aire plane*, Math. Ann. , Vol 36, pp 157-160, 1890.
- [Rad] RADEMACHER H. : *Ueber den Vektorenbereich eines konvexen ebenen Bereiches*, Jber Deutsch Math Verein, 34, pp64-79, 1925.
- [Rei] REITZNER M. : *The Floating Body and the Equiaffine Inner Parallel Curve of a Plane Convex Body*, Geometriae Dedicata 84 : 151-167, 2001.
- [Rev] REVEILLÈS J-P. : *Géométrie discrète, Calcul en nombres entiers et algorithmique*. PhD thesis, Université Louis Pasteur, Strasbourg, France, 1991.
- [Rich] RICHARDSON LF. : *The problem of Continuity : an appendix of statistics of deadly quarrels*, Gen. Sys. Year., Vol 6, pp 139, 1961.
- [Rog] ROGALLA O., EHRENMANN M., DILLMAN R. : *A sensor Fusion Approach for PbD*.
- [Roge] ROGERS C.A., SHEPHARD G.C. : *Convex bodies associated with a given convex Body*, J London Math. Soc., 33, pp270-281, 1958.
- [San05] SANCHEZ-CRUZ H., RODRIGUEZ-DAGNINO R.M. : *Compression bi-level images by means of a 3-bit chain code*, SPIE Opt. Eng. 44, 1-8, 097004, 2005.
- [San07] SANCHEZ-CRUZ H., BRIBIESCA E., RODRIGUEZ-DAGNINO R.M. : *Efficiency of chain codes to represent binary objects*, Pattern Recognition 40, 1660-1674, 2007.

- [Ser82] SERRA J. : *Image Analysis and Mathematical Morphology, Volume I*, Academic Press, 1982.
- [Ser88] SERRA J. : *Image Analysis and Mathematical Morphology : theoretical advances, Volume II*, Academic Press, 1988.
- [Tri] TRICOT C., QUINIOU JF., ROQUES-CARMES C., DUBUC B. : *Evaluation de la dimension fractale d'un graphe*, Rev. Phys. Appl., Vol 23, pp 111-124, 1988.
- [Tro] TROUILLOT X., JOURLIN M., PINOLI J-C. *Geometric parameters computation with Freeman code*, 12th International Congress for Stereology, St Etienne, France, septembre 2007.
- [Tsa] TSANG I.J., TSANG I.R., VAN DYCK D. *Image coding using neighbourhood relations*, Pattern recognition Letters (1999) 1279- 1286.
- [Zen] ZENG X., VASSEUR C. : *A syntactic method for object recognition from a color image*, Systems, Man and Cybernetics, 1993. 'Systems Engineering in the Service of Humans', Conference Proceedings, 1993.

École Nationale Supérieure des Mines
de Saint-Étienne

N° d'ordre : 507 IVS

Xavier TROUILLOT

GEOMETRIC PARAMETERS COMPUTATION WITH FREEMAN CODE

Speciality : Image

Keywords : discrete geometry, Freeman, chain code, shape parameters, mathematical morphology, shape analysis, image compression, binary shape.

Abstract :

This thesis is a part of 2D discrete geometry whose main applications are shape analysis and characterization.

We begin by presenting the different boundary encodings of binary shapes and we then present in detail the oldest one : the Freeman coding scheme, and in particular we develop algorithms based on this coding scheme.

Among these algorithms, we study the estimation of geometric parameters and shape parameters on a binary object as the perimeter, area, apparent diameter, fractal dimension, and the symmetry coefficients of a shape.

Then we study some shape transformations that can be fully operated within this coding framework.

Finally, we consider the mathematical morphology basic operations by providing a method to compute the Freeman code of the dilated and of the eroded of a known shape.

École Nationale Supérieure des Mines
de Saint-Étienne

N° d'ordre : 507 IVS

Xavier TROUILLOT

ETUDE DE PARAMETRES GEOMETRIQUES A PARTIR DU CODE DE
FREEMAN

Spécialité: Image

Mots clefs : géométrie discrète, codage de contour, Freeman, paramètres géométriques, morphologie mathématique, analyse de formes, compression d'images, objet binaire.

Résumé :

Cette thèse s'inscrit dans le cadre de la géométrie discrète 2D avec pour principales applications l'analyse et la caractérisation de formes.

Nous nous intéressons ici aux différents codages de contour de formes binaires que nous présentons dans un premier temps. Nous présentons ensuite plus en détail le plus ancien d'entre eux : le codage de Freeman, et nous développons plus particulièrement des algorithmes sur ce qu'il est possible de faire à partir de ce code.

Nous étudions donc l'estimation de paramètres géométriques et de paramètres de formes sur une forme binaire comme le périmètre, l'aire, les diamètres apparents, la dimension fractale, et les coefficients de symétrie d'une forme.

Nous voyons ensuite les transformations qu'il est possible d'effectuer sur le code de Freeman sans revenir à la représentation classique de la scène.

Enfin, nous abordons la notion de morphologie mathématique en proposant une méthode d'obtention du code du dilaté et de l'érodé d'une forme connue par son code de Freeman.