Transformations, comparaisons de formes à partir du code de Freeman

Sommaire

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

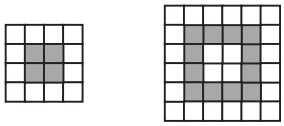

FIGURE 3.1 – Carré minimal (à gauche) et son expansion d'un facteur 2 (à droite).

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'homothétique 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'homothétique 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.

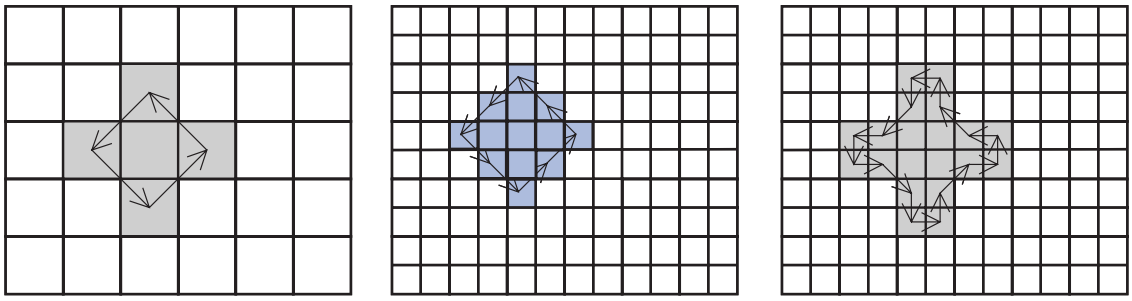

FIGURE 3.2 – Forme étudiée (à gauche), son expansion d'un facteur 2 selon [Mai] (au centre) et l'homothétique que l'on aimerait obtenir (à droite).

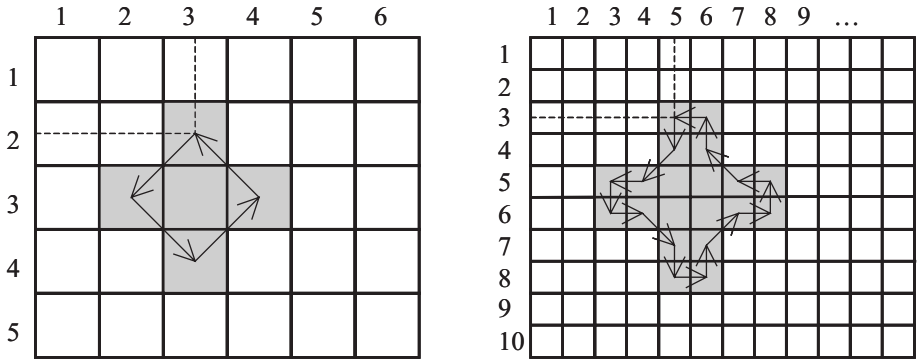

FIGURE 3.3 – Forme de départ (à gauche) et son homothétique de rapport 2 (à droite).

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 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

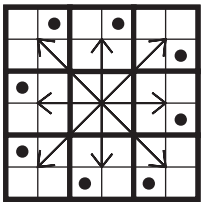

FIGURE 3.4 – Codants (flèches) et points de départs pour les codants suivants (points).

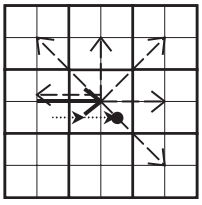

FIGURE 3.5 – Exemple d'obtention d'un des points d'arrivée des codants de l'homothétique de rapport 2.

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	**Nouveau codant**							
courant	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 :

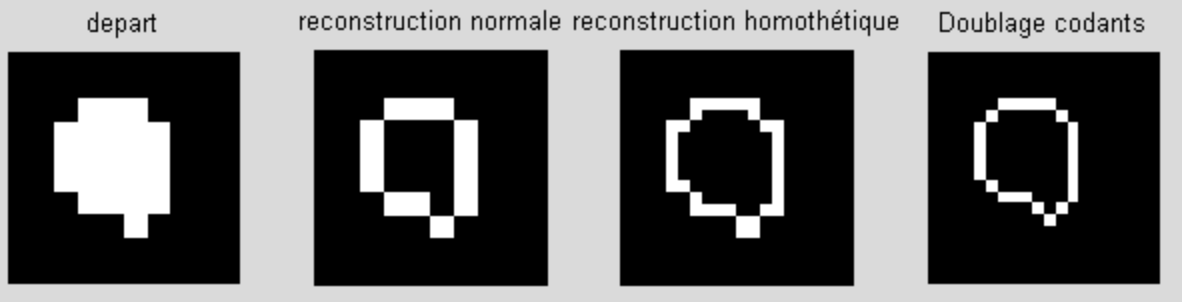

FIGURE 3.9 – Exemple d'objet homothétisé d'un rapport 2.

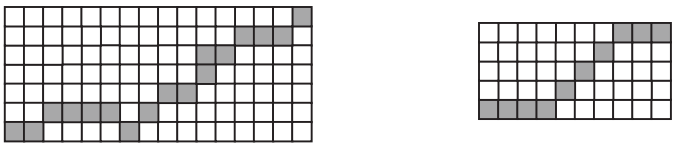

FIGURE 3.10 – Contraction (à droite) d'une chaîne de codants (à gauche).

- 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.

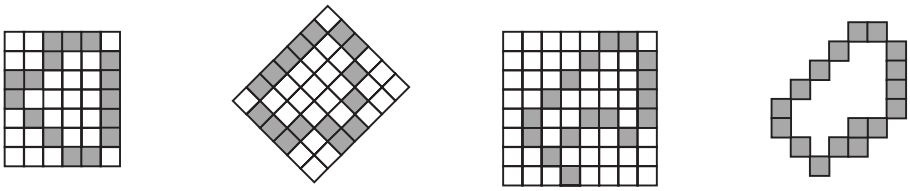

FIGURE 3.12 – Comparaison des rotations. a : objet d'origine ; b : rotation réelle de 135° ; c : rotation sans correction ; d : rotation après correction

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° ($n \times 90°$). 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° ($p \times 45°$). 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 connait le code de Freeman du contour peut être obtenu de façon très simple en deux étapes :

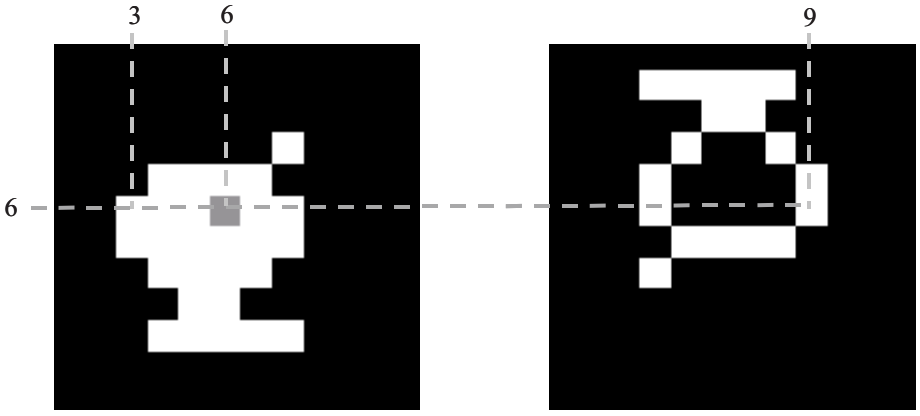

FIGURE 3.13 – Exemple d'objet (à gauche) et son symétrisé par rapport à un point de coordonnées (6,6) (à droite).

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

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

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

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

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 :

$$a_i^m = (8 - 2m - a_i)[8] \tag{3.3}$$

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.

	après symétrie de			
codant de départ	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

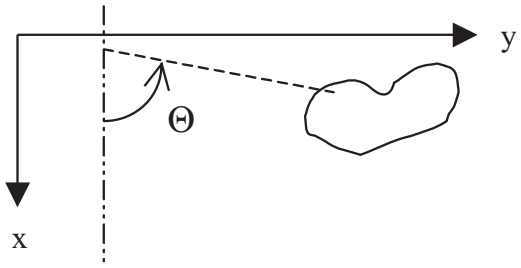

FIGURE 3.15 – Test de l'appartenance d'un point à une forme par une étude angulaire (Le point en question se situe à l'intersection des droites en pointillés).

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.

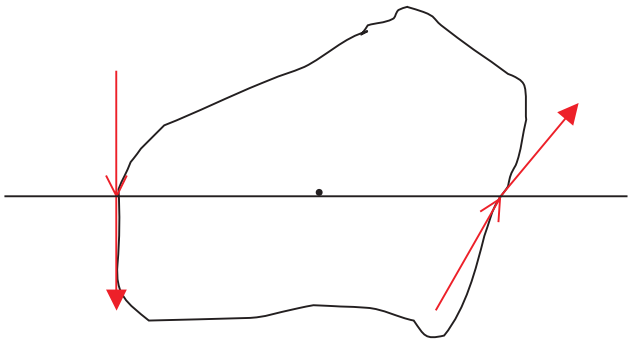

FIGURE 3.16 – Test de l'appartenance d'un point à une forme par une étude de position relative des pixels appartenant à la même ligne.

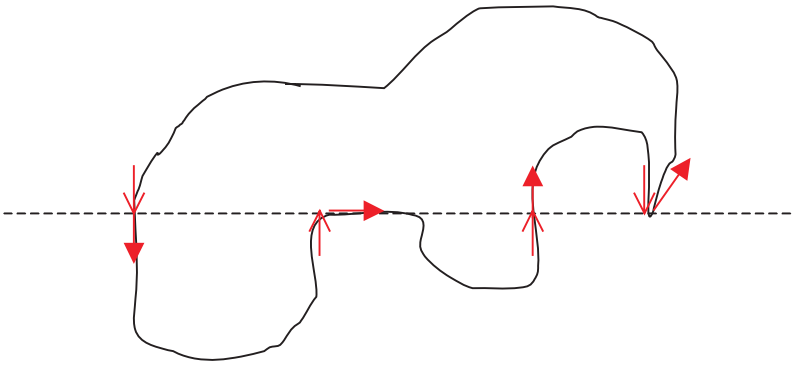

FIGURE 3.17 – Cas particuliers du test d'appartenance d'un point à une forme par une étude de position relative des pixels appartenant à la même ligne.

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 algorithimque, 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

Calcul des coordonnées (X_1, Y_1) et (X_2, Y_2) des premiers points des deux contours
$d = d((X_1, Y_1), (X_2, Y_2))$
tant que le point courant du ***second*** contour est différent du point initial **faire**
  tant que le point courant du ***premier*** contour est différent du point initial **faire**
    Caclul de la distance d' entre les points courants des deux contours
    Si $d' < d$ alors $d = d'$
    Calcul des coordonnées du point suivant du premier contour
  fin tant que
  Calcul des coordonnées du point suivant du second contour
fin tant que
retourner d

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

Recherche des points communs des frontières de deux objets et recherche des points de croisement des courbes, sans intersection.
si aucun pixel commun de la frontière n'est trouvé **alors**
  si un pixel de la forme 1 est à l'intérieur de la forme 2 **alors**
    l'intersection des deux objets correspond au code de la forme 1.
    Break (Fin de l'algorithme)
  sinon si un pixel de la forme 2 est à l'intérieur de la forme 1 **alors**
    l'intersection correspond au code de la forme 2
    Break (Fin de l'algorithme)
  fin si
fin si
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.)
pour chaque point du tableau **faire**
  recherche du cas en fonction des codants arrivant et partant des 2 courbes
  si les courbes ont un point commun **alors**
    Cas 0 : les 2 codants partant du point de croisement sont les mêmes. On retient alors uniquement ce codant.
    Cas 1 : les codants retenus sont ceux de l'objet 1 à partir de ce point et jusqu'au point de croisement suivant.
    Cas 2 : les codants retenus sont ceux de l'objet 2 à partir de ce point et jusqu'au point de croisement suivant.
    Cas 3 : cette configuration nous donnera pour l'intersection un point isolé que nous stockerons.
  sinon si les courbes n'ont pas de point commun **alors**
    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.
  fin si
fin pour

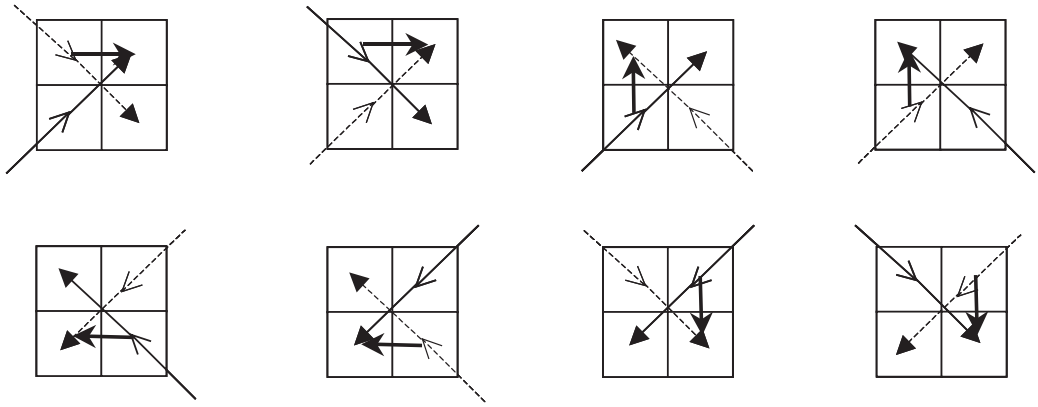

FIGURE 3.18 – Les huit cas de croisements possibles de deux courbes sans intersection.

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).

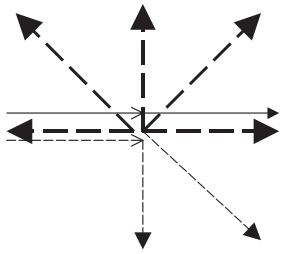

FIGURE 3.19 – Intersection : 8 cas.

Forme 1, codant		Forme 2, codant		
arrivant	partant	arrivant	partant	Cas
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 1 (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.

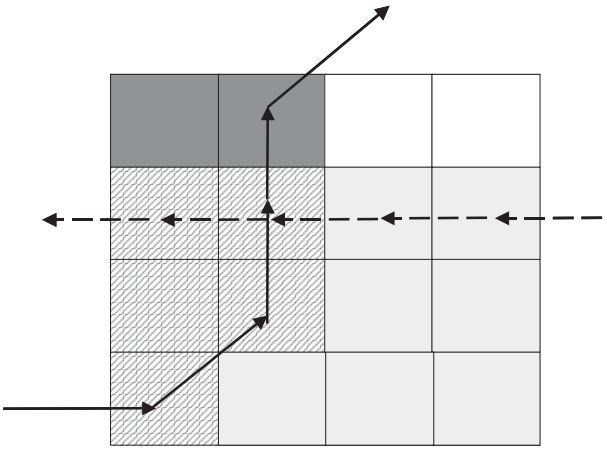

FIGURE 3.21 – Illustration du cas 1 pour l'intersection de deux formes.

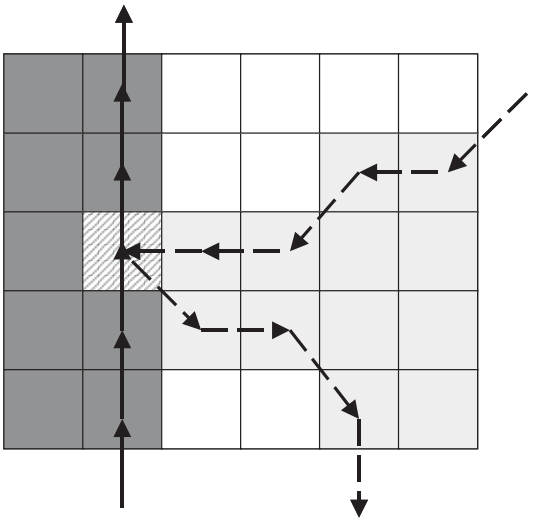

FIGURE 3.22 – Illustration du cas 3 pour l'intersection de deux formes.

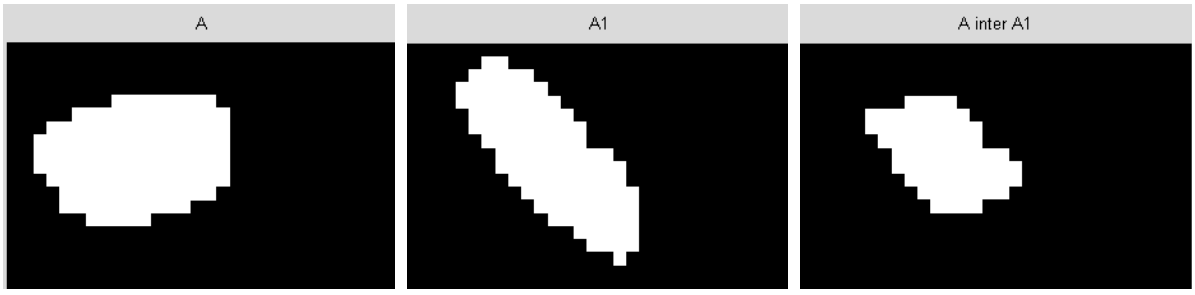

FIGURE 3.23 – Exemple d'intersection de deux formes.

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

Recherche des points communs des frontières de deux objets et recherche des points de croisement des courbes, sans intersection.
si aucun pixel commun de la frontière n'est trouvé **alors**
  si un pixel de la forme 1 est à l'intérieur de la forme 2 **alors**
    l'union des deux objets correspond au code de la forme 2.
    Break (Fin de l'algorithme)
  sinon si un pixel de la forme 2 est à l'intérieur de la forme 1 **alors**
    l'union correspond au code de la forme 1
    Break (Fin de l'algorithme)
  fin si
fin si
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.)
pour chaque point du tableau **faire**
  recherche du cas en fonction des codants arrivant et partant des 2 courbes
  si les courbes ont un point commun **alors**
    Cas 0 : les 2 codants partant du point de croisement sont les mêmes. On retient alors uniquement ce codant.
    Cas 1 : les codants retenus sont ceux de l'objet 1 à partir de ce point et jusqu'au point de croisement suivant.
    Cas 2 : les codants retenus sont ceux de l'objet 2 à partir de ce point et jusqu'au point de croisement suivant.
    Cas 3 : cette configuration nous donnera pour l'union un point isolé que nous stockerons.
  sinon si les courbes n'ont pas de point commun **alors**
    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.
  fin si
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.

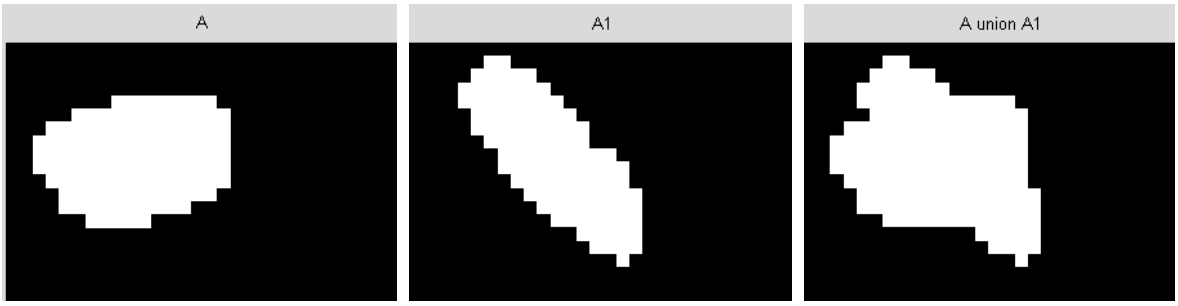

FIGURE 3.24 – Exemple d'union de deux formes.

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 :

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

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

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 :

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

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.

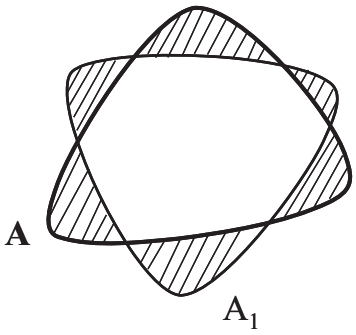

FIGURE 3.25 – Exemple de différence symétrique de deux formes, représentée par l'aire hachurée.

3.2.3.2 Différence symétrique

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

$$A\Delta A_1 = (A \cup A_1)\backslash(A \cap A_1) = (A\backslash A_1) \cup (A_1\backslash A) \tag{3.6}$$

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

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

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.

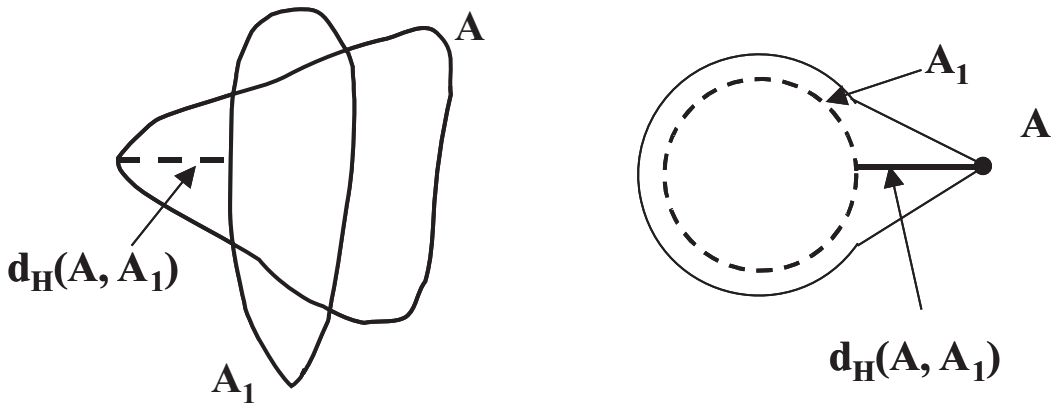

FIGURE 3.26 – Exemples de distance de Hausdorff entre deux formes.

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 :

$$d_H(A, A_1) = Max\left\{Sup_{a\in A}\left(Inf_{a_1\in A_1}(d(a, a_1))\right), Sup_{a_1\in A_1}\left(Inf_{a\in A}(d(a, a_1))\right)\right\} \tag{3.8}$$

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 :

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

où $Fr(A)$ 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 $A1$ 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 $A1$ vaut $d_H(Fr(A), Fr(A_1))$ =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 :

$$d_{As}(A, A1) = \log(\frac{\mu}{\lambda}) \tag{3.10}$$

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

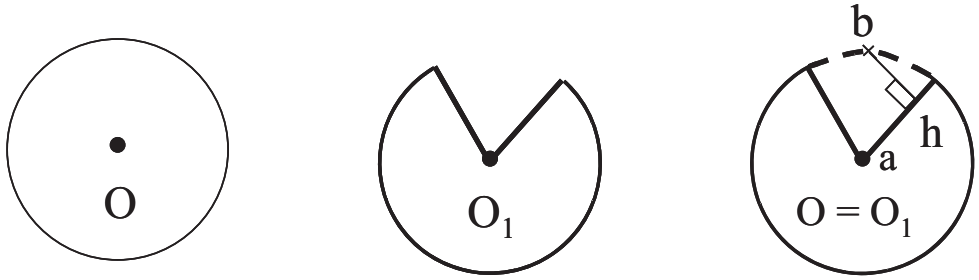

FIGURE 3.27 – Différence entre la distance de Hausdorff entre deux formes et la distance de Hausdorff entre les frontières de ces deux formes.

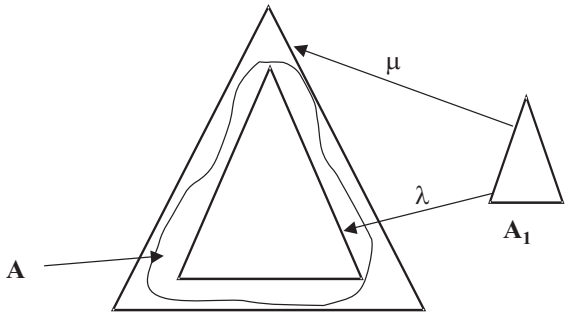

FIGURE 3.28 – Distance de Asplünd calculée grâce à 2 homothétiques.

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 $A1$:

- 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 $d_{As}(A, A1) = \log(\frac{\mu}{\lambda})$

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 $A = a_1a_2...a_n$ et $B = b_1b_2...b_m$, avec $n \leq m$ le calcul est le suivant :

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

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

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

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

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.