

UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

SEGMENTATION D'OBJETS : APPROCHES PAR ANALYSE DE FORME ET
APPRENTISSAGE PROBABILISTE

MÉMOIRE
PRÉSENTÉ
COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN INFORMATIQUE

PAR
GUILLAUME LARIVIÈRE

AVRIL 2012

UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

Département d'informatique et d'ingénierie

Ce mémoire intitulé :

SEGMENTATION D'OBJETS : APPROCHES PAR ANALYSE DE FORME ET
APPRENTISSAGE PROBABILISTE

présenté par
Guillaume Larivière

pour l'obtention du grade de maître ès sciences (M.Sc.)

a été évalué par un jury composé des personnes suivantes :

Dr. Mohand Saïd Allili Directeur de recherche
Dr. Marek B. Zaremba Président du jury
Dr. Nadia Baaziz Membre du jury

Mémoire accepté le : 16 avril 2012

Résumé

Dans ce travail, nous proposons une méthode par apprentissage pour la segmentation d'objets. Notre méthode corrige la segmentation partielle d'une image qui contient un objet d'une classe donnée, et ce, à l'aide d'information a priori concernant la forme de l'objet. Nous décrivons dans un premier temps l'acquisition de cette information a priori effectuée grâce à un apprentissage automatique. Cette information consiste en des fragments qui sont extraits d'images d'objets de la classe d'objet concernée. L'ensemble de ces fragments constitue le modèle que l'on associe à cette classe. Dans un deuxième temps, nous décrivons comment utiliser ce modèle pour effectuer la correction de toute segmentation partielle d'une image, et ainsi délimiter les frontières d'un objet.

Pour ce faire, deux approches ont été explorées. Dans la première approche, nous utilisons l'analyse par corrélation pour segmenter un objet en connaissant sa classe. Dans la deuxième approche, aucune information sur la classe de l'objet n'est fournie. Un modèle probabiliste sera alors utilisé pour identifier la classe d'un objet et segmenter ce dernier. L'application de notre méthode à des images naturelles montre de très bons résultats, que nous quantifions par rapport à la vérité terrain de la segmentation figure-fond de ces images. Notre méthode permet ultimement d'attribuer un niveau de confiance à la présence d'objets de classes connues dans une image (reconnaissance d'objets), en plus de segmenter ces derniers.

Abstract

In this work, we present an image segmentation method that uses automatic learning of object shape. Our method corrects the segmentation of an image containing an object of a given class using a priori knowledge about that class. We first describe how a priori knowledge is modeled. This knowledge, in our method, takes the form of image fragments that are extracted from training images containing objects of any given class. The set of fragments thus extracted is the shape model we associate to that class. Once a class' model is defined, we describe how we use the information contained therein to correct an image's segmentation, thereby delimiting the contour of the object it contains. Applying our method to a natural image yields very promising results, the quality of which we're able to quantify in regard to the ground truth segmentation of the image.

Remerciements

À Mohand Saïd Allili pour sa grande disponibilité, sa patience et son incommensurable aide dans le cadre de ce travail de recherche et de rédaction.

À Nadia Baaziz et Marek Zaremba pour avoir accepté de lire et d'analyser ce mémoire.

À ma conjointe Myriam Legault-Beauregard pour son soutien et pour la révision du texte de ce mémoire.

À mes parents pour m'avoir toujours encouragé à cultiver mon intérêt pour la connaissance et, en particulier, pour les sciences.

À Stéphane Bouchard du Laboratoire de cyberpsychologie de l'UQO pour la flexibilité d'horaire de travail qu'il m'a accordée.

À Al-Khawarizmi, Leonhard Euler et Alan Turing pour l'ensemble de leur œuvre.

À Miles Davis, John Coltrane et Dave Brubeck pour avoir souvent accompagné la rédaction de ce mémoire.

« Les savants des temps passés et des nations révolues n'ont cessé de composer des livres. Ils l'ont fait pour léguer leur savoir à ceux qui les suivent. Ainsi demeurera vive la quête de la vérité. »

Al-Khawarizmi

« Bien que la compréhension des plus profonds mystères de la nature et, par conséquent, la connaissance des véritables causes des phénomènes qui nous entourent ne nous soient pas accessibles, il est néanmoins possible qu'à l'avenir une hypothèse fictive suffise pour expliquer plusieurs phénomènes. »

Leonhard Euler

« Je serai désormais moins sujet à la distraction. »

Leonhard Euler, après avoir perdu l'usage de son œil droit

« Les tentatives de création de machines pensantes nous seront d'une grande aide pour découvrir comment nous pensons nous-mêmes. »

Alan Turing

Table des matières

Résumé	i
Abstract	ii
Remerciements	iii
Table des matières	v
Liste des tableaux	ix
Liste des figures	x
Liste des algorithmes	xiii
Chapitre 1 — Introduction	1
1.1 Généralités	1
1.2 Segmentation d'images	2
1.3 Reconnaissance d'objets	4

1.4	Problématique	6
1.5	Structure du mémoire	8
Chapitre 2 — État des connaissances		9
2.1	Segmentation d'images : une définition classique	9
2.2	Approche ascendante	11
2.2.1	Méthode du seuillage	11
2.2.2	Méthode du watershed	13
2.2.3	Méthode de croissance de régions	17
2.2.4	Méthode de division et fusion	18
2.2.5	Méthode du mean-shift	20
2.3	Approche descendante	24
2.3.1	Segmentation par contours actifs	25
2.3.2	Méthode des <i>K-moyennes</i>	29
2.3.3	Méthode par coupures de graphes	30
2.3.4	Méthodes utilisant des <i>patches</i>	32
2.4	Comparaison des méthodes présentées	34
Chapitre 3 — Problématique		38
3.1	Aperçu de la problématique	38
3.2	Modèle : formation et utilisation	39

3.2.1	Extraction de la connaissance a priori	40
3.2.2	Segmentation utilisant les fragments	46
3.3	Expérimentations	50
3.4	Discussion	54
Chapitre 4 — Contribution		56
4.1	Extension aux cas de multiples classes d'objets	56
4.2	Facteurs de correspondance fragment / image	59
4.2.1	Score algébrique	60
4.2.2	Score géométrique	60
4.2.3	Distance de Hausdorff	62
4.2.4	Chevauchement	64
4.2.5	Position d'un fragment	65
4.3	Facteurs de correspondance fragment / modèle	66
4.3.1	Validité	66
4.3.2	Pertinence	66
4.4	Méthodologie	67
4.4.1	Formation des modèles de classes	68
4.4.2	Détection d'objets	69
4.4.3	Calcul de la probabilité de chaque classe	71
4.4.4	Correction de la segmentation	74

4.4.5	Algorithmes	74
4.5	Expérimentations	74
4.6	Interprétation des résultats	76
4.7	Complexité algorithmique	81
4.7.1	Extraction de la connaissance a priori	82
4.7.2	Correction de la segmentation	82
4.8	Discussion	82
4.9	Extensions de ce travail	84
4.9.1	Échelle	85
4.9.2	Perspective	85
4.9.3	Orientation	87
4.9.4	Direction	87
	Conclusion	89
	Bibliographie	90

Liste des tableaux

2.1	Méthodes de segmentation d'images : avantages et inconvénients	37
3.1	Tableau de contingence d'un fragment	45
3.2	Moyennes et écarts-types des <i>précisions</i> et <i>rappels</i> calculés	53

Liste des figures

1.1	Exemples de segmentations	3
2.1	Segmentation par seuillage	13
2.2	Représentations 2D et 3D d'une image à niveaux de gris	14
2.3	Emplacements des points dans une image	15
2.4	Segmentation selon la méthode du watershed	15
2.5	Segmentation de l'image d'un ventricule cérébral grâce à la méthode de croissance de régions	19
2.6	Quadrants (régions homogènes) et leur arbre associé	20
2.7	<i>Division et fusion</i> : différentes tailles minimales de quadrants	21
2.8	Distribution de points, les points x_t <i>initial</i> et x_t <i>final</i> et les vecteurs <i>mean-shift</i> \vec{m}_1 et \vec{m}_2	23
2.9	Segmentation <i>mean-shift</i> d'une image	24
2.10	<i>Snake</i> décrit à différentes itérations de la segmentation d'une image	27
2.11	<i>Méthode des ensembles de niveaux</i> à trois moments dans le temps	28

2.12	Ensembles de niveaux décrits à différentes itérations de la segmentation d'une image	29
2.13	Image segmentée selon la méthode des <i>K-moyennes</i> à l'aide de différents <i>K</i>	31
2.14	Illustration de la méthode des coupures de graphes	35
2.15	Méthode des coupures de graphes appliquée à une image naturelle	36
3.1	Exemples de fragments de la classe <i>cheval</i>	41
3.2	Combinaison de segments	48
3.3	Organigrammes	49
3.4	<i>Exemples positifs</i> de la classe <i>cheval</i>	50
3.5	<i>Exemples négatifs</i> de la classe <i>cheval</i>	51
3.6	Fragments extraits pour la classe <i>cheval</i>	52
3.7	Expérimentation sur cinq images naturelles	55
4.1	Exemple d'extraction du meilleur cadre	59
4.2	Fragment, cadre et calcul de leurs scores <i>algébrique</i> et <i>géométrique</i>	62
4.3	<i>Distance de Hausdorff</i> entre les polygones p_1 (<i>bleu</i>) et p_2 (<i>vert</i>)	63
4.4	Exemples de <i>chevauchements</i>	64
4.5	Exemples de segments significatifs et non significatifs d'une image naturelle	70
4.6	Cas de correspondance fragment / image pénalisés	73
4.7	<i>Exemples positifs</i> utilisés dans nos expérimentations	77
4.8	Résultats de nos expérimentations pour la classe <i>autruche</i>	78

4.9	Résultats de nos expérimentations pour la classe <i>cheval</i>	79
4.10	Résultats de nos expérimentations pour la classe <i>cygne</i>	80
4.11	Exemples de formes variées chez des objets de la classe <i>cheval</i>	84
4.12	Transformations géométriques appliquées à une image naturelle	88

Liste des algorithmes

4.1	Algorithme pour l'extraction d'un modèle de classe	75
4.2	Algorithme pour la segmentation de l'image	75

Chapitre 1

Introduction

1.1 Généralités

Les images numériques jouent un rôle de plus en plus important dans la société. Leur omniprésence dans les médias électroniques et l'usage qu'on en fait dans plusieurs domaines technologiques de pointe (imagerie médicale, reconnaissance des visages et des empreintes digitales, Internet, télédétection, vision artificielle, etc.) témoignent de leur importance.

L'un des principaux avantages que l'on retire de la numérisation des images consiste en la possibilité d'automatiser leur traitement informatique. Ainsi, de nombreux algorithmes ont été mis au point pour, par exemple, corriger certains défauts présents dans des images numériques ou rehausser la qualité de ces dernières [16]. Le bruit numérique peut être grandement réduit ou même éliminé grâce à des filtres moyens, gaussiens ou médians [16]. Les contrastes des couleurs ou des niveaux de gris peuvent quant à eux être accentués grâce, entre autres, à une transformation gamma ou linéaire [21], ou à une égalisation d'histogramme [36]. Les contours peuvent, par exemple, être accentués grâce au Laplacien

de l'image [16][36]. Ces opérations sont nécessaires pour accomplir des traitements de haut niveau plus précis dans l'image, tels que l'extraction de régions ou de contours [16], la reconnaissance d'objets, etc. Crowley et al. [12] précisent que la qualité des images utilisées influera fortement sur les résultats de la plupart de ces techniques de vision artificielle.

La nature d'une image varie selon le domaine de la vision artificielle où elle est utilisée. L'image naturelle sera utilisée dans des domaines comme la sécurité routière et la robotique, ou encore en gestion et indexation d'images ; il s'agit du type d'images dont il sera question dans ce rapport. Les images en rayons X sont entre autres utilisées en imagerie médicale, en sécurité et dans de nombreux procédés industriels. Les images topographiques sont utilisées dans le domaine de la télédétection, en hydrographie ainsi qu'en géodésie. Les images infrarouges sont quant à elles utilisées dans le domaine de la sécurité, du génie du bâtiment, de la vision nocturne de même qu'en astronomie. D'autres modalités d'images existent, en particulier dans le domaine de l'imagerie médicale [13].

1.2 Segmentation d'images

La segmentation d'images consiste d'abord et avant tout en la division d'une image en parties (segments) qui montrent une forte corrélation avec des objets ou des zones de la scène contenus dans cette même image [36]. Selon la méthode utilisée, la segmentation d'images peut se baser sur les contours observés dans une image, sur des groupes de pixels observés dans une image ou sur d'autres caractéristiques de l'image (par exemple, l'histogramme de l'image, ses points d'intérêt, sa transformée dans le domaine de Fourier, etc.) Indépendamment du domaine où elle est réalisée, la segmentation d'une image peut être :

complète : elle compose un ensemble de segments disjoints où chaque segment correspond à un objet de l'image pourvu d'une sémantique (voir Figure 1.1.a),

partielle : auquel cas les segments ne correspondent pas nécessairement à des objets pourvus d'une sémantique, mais sont quand même homogènes du point de vue de leur couleur, de leur texture [10] ou d'une autre de leurs caractéristiques (voir Figure 1.1.b).

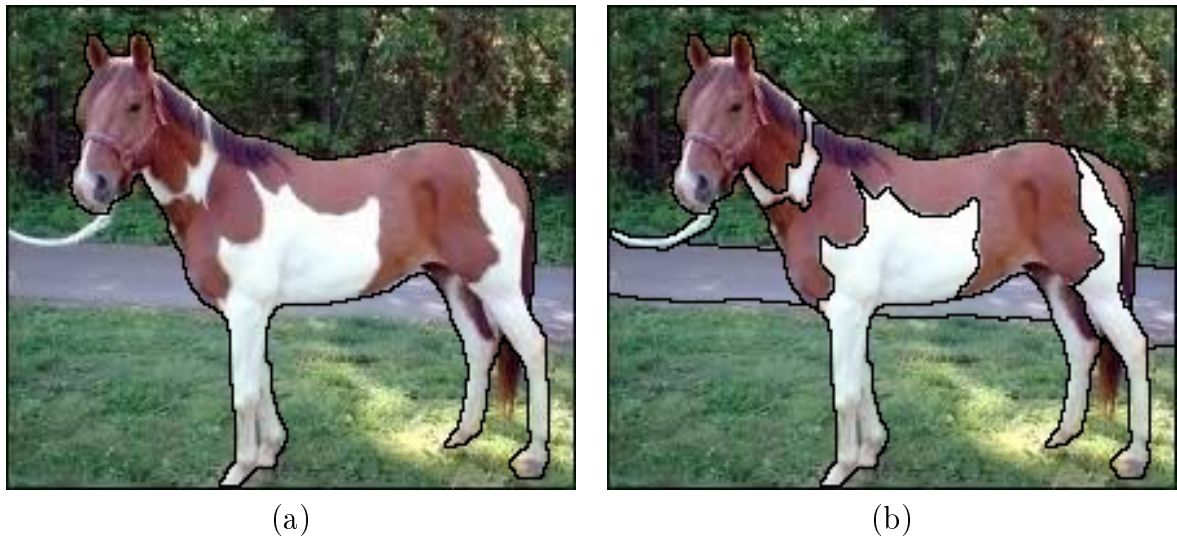


Figure 1.1 – Exemples de segmentations

- a) : segmentation complète*
- b) : segmentation partielle*

Les méthodes utilisées pour effectuer la segmentation d'images peuvent être classées en deux approches, suivant la façon selon laquelle l'information des images à segmenter est utilisée :

- *L'approche ascendante (bottom-up)* cherche à définir des segments dans une image en y faisant ressortir des régions homogènes, c'est-à-dire des régions dont les pixels sont semblables selon certains critères. Dans ce cas, la segmentation est complètement guidée par les données [36].

- L'*approche descendante (top-down)* définit quant à elle des segments dans une image conformément à un modèle connu au préalable. Autrement dit, il s'agit de trouver dans l'image les régions qui correspondent au concept dont on dispose [36].

L'être humain effectue naturellement les segmentations complète et partielle dans une scène ; ceci lui permet, de façon implicite, d'isoler visuellement les objets les uns des autres. Cela dit, bien qu'il existe une multitude de méthodes qui permettent d'effectuer la segmentation partielle en vision artificielle, la segmentation complète, quant à elle, est plus difficile à accomplir adéquatement. Cette difficulté découle de la contrainte sémantique, à laquelle la segmentation partielle n'est pas soumise. Dans une optique où l'on désire reproduire la segmentation dont la vision humaine est capable, plus précisément pour segmenter correctement les objets dans une image, il importe de proposer de nouvelles approches automatiques et efficaces. Ce mémoire propose une telle approche et décrit les paramètres de son fonctionnement.

1.3 Reconnaissance d'objets

L'un des principaux intérêts d'une segmentation complète dans un contexte de vision artificielle est la reconnaissance des objets segmentés. Au sens large, la reconnaissance consiste à pouvoir identifier un objet comme faisant partie d'une catégorie connue a priori. La reconnaissance d'objets requiert alors que l'on détienne de l'information concernant les objets recherchés [36].

La reconnaissance d'objets se base sur divers critères auxquels satisfont ou non les segments extraits de l'image. Ces critères peuvent concerner des caractéristiques d'un segment, comme sa silhouette (la forme que décrit son contour), sa texture ou encore sa couleur. Ainsi, en comparant les caractéristiques observées dans un segment avec celles

qu'il est attendu de retrouver dans un objet donné, on peut déterminer si ce segment fait partie ou non de ce même objet.

Contrairement à la segmentation complète, la segmentation partielle est insuffisante pour accomplir la reconnaissance d'objets, étant donné que les segments qui y sont trouvés ne sont pas nécessairement pourvus d'une sémantique. La reconnaissance des objets contenus dans une image est donc liée à une segmentation complète et fidèle de cette image. Dans la suite de ce rapport, on utilisera l'expression « reconnaissance d'objets » [14] pour désigner le fait de reconnaître la catégorie de l'objet. Au sens plus large, la reconnaissance d'objets, appliquée par exemple à des images de personnes, peut être interprétée comme la reconnaissance du sexe, de l'identité ou même de l'expression du visage de ces personnes ; ce rapport n'explorera pas ces dernières interprétations.

Pour accomplir la reconnaissance d'objets, il faut ultimement un certain modèle de l'objet que l'on cherche à reconnaître dans l'image. La reconnaissance de formes primitives (lignes, ellipses, rectangles, etc.) est plus simple à réaliser, par exemple, à l'aide de la transformée de Hough, où le modèle qu'on leur associe consiste en une équation paramétrique [36]. Cependant, lorsque l'objet à reconnaître (par exemple, un cheval, une voiture, un piéton, etc.) ne peut pas être représenté par une définition mathématique formelle, il faut disposer d'exemples d'images de cet objet. Puisque dans l'univers des objets on peut trouver une infinité de variations, il est impossible d'effectuer un recensement exhaustif de ces variations et donc de décrire toutes les possibilités de formes d'objets. Il devient nécessaire de déduire un modèle de chaque catégorie d'objets à l'aide d'un nombre limité d'exemples d'images (ci-après appelées images d'apprentissage) [6]. Ce modèle est censé capter l'information de base (par exemple, la forme) caractérisant la catégorie de l'objet.

1.4 Problématique

Si la segmentation d'images aide à la reconnaissance d'objets, il est pertinent de chercher à savoir si le contraire est également vrai, c'est-à-dire si la segmentation d'une image donnée, quand on connaît l'objet qu'on y recherche, peut être réalisée en la jumelant à un mécanisme de reconnaissance d'objets.

Zemel et al. [37] ont montré qu'une segmentation d'objets sur des fonds d'image (ci-après appelée segmentation figure-fond) basée sur la reconnaissance de formes connues se fait spontanément chez des adultes à qui on montre des images à segmenter. Ce processus est facilité par la capacité de l'être humain à éliminer d'emblée de sa segmentation les zones qui ne présentent pas un profil qui ressemble à ce qu'il connaît.

Peterson [31] a pu montrer que certains processus associés à la reconnaissance d'objets chez l'être humain sont effectués avant toute segmentation figure-fond. Ses expériences démontrent que le cerveau humain utilise la reconnaissance de formes pour guider la segmentation figure-fond d'une image ; elles montrent donc que la reconnaissance précède, ou se fait avec, la segmentation.

Dans ce contexte, la connaissance d'objets qu'on introduit dans le processus (on parlera d'information a priori) joue un rôle très important. La façon dont l'information a priori est structurée peut aussi contribuer de différentes manières à réaliser la segmentation. On peut organiser la forme que prend l'information a priori en deux approches principales :

- L'approche dite par *patches* [3][5][6][23][24][33], où l'information a priori consiste en un ensemble de fragments (les *patches*) extraits des images d'apprentissage. Ces fragments sont choisis en fonction de la probabilité selon laquelle on les retrouve dans les images d'apprentissage. On en infère alors qu'il est tout aussi vraisemblable de les retrouver dans une autre image de la même catégorie.

- L'approche dite par *forme complète* [4][28], où l'information a priori consiste en la forme moyenne que décrit l'objet en entier, par exemple son contour. Cette approche, bien qu'efficace si les objets recherchés varient peu en forme, perd son efficacité si, au contraire, les objets ont une forme très variable. La variabilité de la forme peut découler, par exemple dans le cas d'animaux, des différentes positions relatives que leurs membres ont les uns par rapport aux autres, des différentes résolutions possibles et de la possibilité que l'objet dans l'image soit occlus.

À la lumière de ces faits, il convient de chercher une façon efficace d'effectuer ou de corriger la segmentation d'images afin d'extraire des objets d'intérêt. La solution envisagée pour résoudre le problème de la segmentation complète comprendra l'utilisation de la reconnaissance d'objets aidée d'information a priori, car cette dernière technique semble un outil prometteur. Les applications anticipées pour cette solution se trouvent principalement en gestion d'images ainsi que dans plusieurs domaines de vision artificielle, dont la sécurité routière et la robotique.

Notre recherche s'effectuera selon les deux axes suivants :

- L'acquisition d'information a priori, où des images d'apprentissage seront employées pour définir le modèle qui représentera une catégorie d'objets. L'information a priori sera obtenue grâce à l'approche dite par *patches*, pour faciliter la reconnaissance des objets dont la forme peut varier.
- La segmentation de l'image, qui consistera en une combinaison de l'*approche ascendante* (des segments sont formés à partir de régions homogènes) et de l'*approche descendante* (les segments sont assemblés de façon à correspondre à l'information a priori). Cette approche hybride vise à utiliser les avantages respectifs des méthodes *ascendante* et *descendante*.

1.5 Structure du mémoire

Ce mémoire est organisé comme suit : le chapitre 2 explore l'état actuel des réalisations en matière de segmentation d'images ; le chapitre 3 porte sur les étapes qu'il est nécessaire d'aborder en vue de résoudre le problème de la correction de la segmentation d'images à l'aide d'information a priori ; le chapitre 4 décrit la solution que nous proposons pour résoudre la problématique avancée ; enfin, nous terminons avec une conclusion qui résume l'ensemble du document et avec la bibliographie des ouvrages et des articles dont nous nous sommes servi dans le cadre de ce projet de recherche et de la solution que nous avons développée.

Le travail de ce mémoire est publié dans l'acte de la conférence *IEEE Canadian Conference on Computer and Robot Vision 2012* [22].

Chapitre 2

État des connaissances

2.1 Segmentation d'images : une définition classique

Comme il a été mentionné au chapitre précédent, la segmentation d'une image est sa partition en régions homogènes. Le nombre de régions résultant de cette partition sera fonction du niveau de détail souhaité. Posons R_I comme étant le domaine décrit par une image I , n comme étant le nombre de régions résultant de la partition de R_I et $Q(R_k)$ comme étant un prédicat logique défini pour toute région de R_I . La segmentation de l'image I se décrit formellement comme suit [16] :

- 1) $\bigcup_{i=1}^n R_i = R_I$
 - 2) R_i forme un ensemble connexe $\forall i = 1, \dots, n$
 - 3) $R_i \cap R_j = \emptyset \forall i, j \mid i \neq j$
 - 4) $Q(R_i) = VRAI \forall i = 1, \dots, n$
 - 5) $Q(R_i \cup R_j) = FAUX$ pour toutes les régions adjacentes R_i et R_j
- (2.1)

Plusieurs conditions doivent ainsi être respectées pour qu'une segmentation soit valide. La condition 1) indique que tout pixel de l'image doit se retrouver dans une région. La

condition 2) indique la connexité des régions, c'est-à-dire : pour toute paire de pixels p et q d'une région R_i , il est possible de tracer un chemin de p vers q en ne passant que par des pixels de la région R_i [16]. La condition 3) indique la disjonction des régions, c'est-à-dire qu'aucun pixel ne fait partie de deux régions différentes à la fois. Les conditions 4) et 5) indiquent respectivement que toute région doit satisfaire au prédicat Q , mais que l'union de deux régions adjacentes ne lui satisfait pas. Par exemple, posons $Q(R_i) = \text{VRAI}$ si les pixels de R_i ont une variance maximale de σ , où σ est une constante positive proche de zéro. Posons maintenant les régions adjacentes R_1 et R_2 , où la région R_1 est composée exclusivement de pixels noirs (donc d'intensité minimale) et la région R_2 est composée exclusivement de pixels blancs (donc d'intensité maximale). Il apparaît que la condition 5) sera respectée, car la variance de l'union des régions R_1 et R_2 dépassera vraisemblablement la constante σ .

Bien que les cinq conditions demeurent vraies pour toute segmentation d'images, le prédicat utilisé variera d'une méthode de segmentation d'images à une autre. Le prédicat utilisé dans une méthode de segmentation d'images peut concerner divers aspects d'un segment : sa taille, le caractère lisse de son contour, sa couleur, sa texture, etc. Il déterminera également si la méthode employée fait partie de l'*approche ascendante* (segments formés presque exclusivement à l'aide de l'information contenue dans l'image) ou de l'*approche descendante* (segments formés conformément à une information a priori).

Il importe de préciser que l'information a priori peut être de différentes natures. Elle peut être de *bas niveau*, comme dans l'*approche ascendante* ; dans ce cas, l'information a priori peut consister, par exemple, en un certain seuil que les pixels d'un même segment ne peuvent pas dépasser. Par contre, dans le cas de l'*approche descendante*, il sera nécessaire de détenir de l'information a priori dite de *niveau élevé* ; cette information peut renseigner, par exemple, sur la forme attendue d'un objet recherché dans une segmentation.

Les méthodes de segmentation d'images se répartissent, pour la plupart, en approches

ascendante ou *descendante*; certaines méthodes empruntent cependant aux deux approches [9][17][20][39][40]. Ce présent chapitre montrera les particularités des principales méthodes de segmentation d'images et abordera les méthodes qui empruntent autant à l'*approche ascendante* qu'à l'*approche descendante*.

2.2 Approche ascendante

Les méthodes de segmentation d'images dites *ascendantes* sont caractérisées par la façon dont leurs régions sont déterminées. La règle qui dicte la formation des régions est toujours liée, lorsqu'on utilise l'*approche ascendante*, à l'homogénéité des pixels d'un même segment. Puisque la segmentation d'images qui suit l'*approche ascendante* ne se base essentiellement que sur les données contenues dans l'image, il n'est pas nécessaire de détenir de l'information a priori importante pour la guider [2][34].

Les méthodes de segmentation d'images qui suivent l'*approche ascendante* comptent parmi les premières qui ont été mises au point [41]; conséquemment, les règles sur lesquelles elles s'articulent sont souvent moins nombreuses et plus primitives. Les prochaines sections présentent quelques-unes des principales méthodes de segmentation d'images qui suivent l'*approche ascendante*.

2.2.1 Méthode du seuillage

La méthode du *seuillage* compte parmi les méthodes de segmentation d'images les plus simples, autant dans sa compréhension que dans son implémentation [16]. Elle consiste à regrouper individuellement les pixels d'une image en un certain nombre de classes déterminées au préalable. Suivant la définition classique de la segmentation d'images, les régions connexes de pixels assignés à une même classe formeront des segments. Le

nombre minimal de classes nécessaires au *seuillage* d'une image est de deux, sans quoi la segmentation ne résulterait qu'en une seule région.

Les différentes classes, dans cette méthode, sont déterminées par des seuils. Dans le cas de deux classes, on requerra un seuil ; dans le cas de trois classes, on en requerra deux, et ainsi de suite. Étant donné une image à niveaux de gris I (voir Figure 2.1.a), la classe d'un pixel (x, y) , selon le seuil T , est définie par :

$$g(x, y) = \begin{cases} 1 & \text{si } I(x, y) > T \\ 0 & \text{si } I(x, y) \leq T \end{cases} \quad (2.2)$$

Puisqu'il n'y a que deux classes définies dans cette segmentation, on leur associe soit la valeur 1, soit la valeur 0. Dans le cas où le *seuillage* doit compter trois classes, on leur associera les valeurs a , b et c ; deux seuils T_1 et T_2 sépareront ces trois classes. La définition donnée pour ce *seuillage* sera analogue à celle d'un *seuillage* selon deux classes :

$$g(x, y) = \begin{cases} a & \text{si } I(x, y) > T_2 \\ b & \text{si } T_1 < I(x, y) \leq T_2 \\ c & \text{si } I(x, y) \leq T_1 \end{cases} \quad (2.3)$$

Le *seuillage* en tant que méthode de segmentation d'images peut être utilisé pour une image complète ou pour des parties distinctes de celle-ci ; on parlera alors respectivement de *seuillage* global (voir Figure 2.1.b) ou de *seuillage* local (voir Figure 2.1.c). Un *seuillage* local peut être préférable à un *seuillage* global si seules certaines parties d'une image doivent être segmentées ou si différentes parties de l'image requièrent différents seuils.

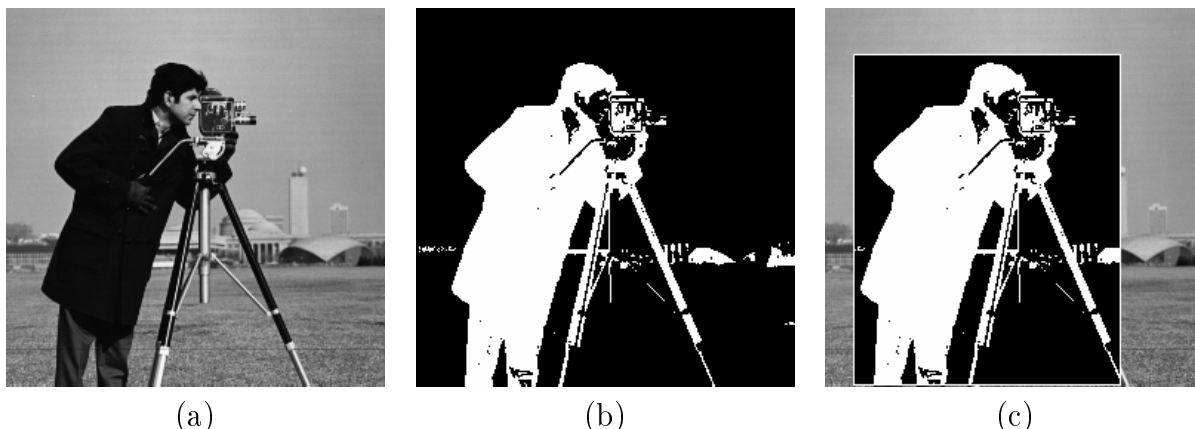


Figure 2.1 – Segmentation par seuillage

- a)* : *image à niveaux de gris*
- b)* : *seuillage global*
- c)* : *seuillage local*

2.2.2 Méthode du watershed

La méthode du *watershed* est une méthode de segmentation d'images qui, tout comme le *seuillage*, est applicable en particulier aux images à niveaux de gris. Son fonctionnement s'explique aisément si l'on change la façon selon laquelle on représente une image. En représentant l'image de façon à opposer son intensité à ses deux coordonnées spatiales (voir Figure 2.2), l'image sera décrite par un relief comportant certaines formes saillantes de même que certaines formes en creux (ou bassins). En utilisant cette conception tridimensionnelle de l'image, le *watershed* consiste à « remplir » les bassins jusqu'à leur rebord. Chaque bassin ainsi rempli définira une région dans l'image.

Les pixels d'une image à segmenter selon le *watershed* peuvent être classés en trois catégories [16] (voir Figure 2.3) : les pixels qui sont des minima locaux de l'image (chacun des points les plus bas de chacun des bassins) ; les pixels qui occupent des points d'un bassin où, si l'on y déposait une goutte d'eau, elle tomberait avec certitude vers un certain minimum local ; enfin, les pixels qui occupent des points où une goutte d'eau pourrait de

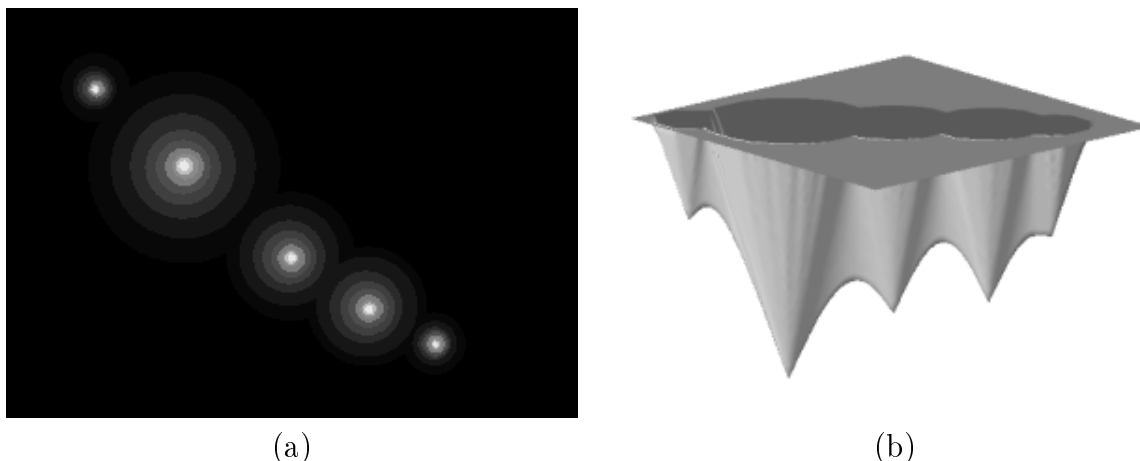


Figure 2.2 – Représentations 2D et 3D d'une image à niveaux de gris
a) : image à niveaux de gris
b) : bassins décrits dans l'image

façon équiprobable tomber vers plus d'un minimum local, c'est-à-dire les pixels qui sont sur la frontière qui sépare des bassins.

Pour accomplir la segmentation d'images selon la méthode du *watershed*, il est nécessaire de procéder en deux étapes [16]. La première étape consiste à trouver les minima locaux dans l'image (les bassins de l'exemple expliqué au paragraphe précédent). Ces minima servent d'amorces pour la formation des régions. La deuxième étape consiste à former chacune des régions en leur agrégeant les pixels qui leur sont associés ; ces pixels doivent à la fois se trouver dans le même bassin que le segment à former et ne pas se trouver sur une frontière entre des bassins.

La façon de procéder pour former les régions s'effectue, comme il a été mentionné précédemment, à la manière du remplissage de bassins. Le remplissage commence au niveau le plus bas dans chaque bassin, c'est-à-dire aux minima locaux. On remplit ensuite les bassins simultanément en faisant monter uniformément le niveau d'eau d'un pixel à la fois. À terme, les eaux de bassins différents tendront à se fusionner ; il convient alors de

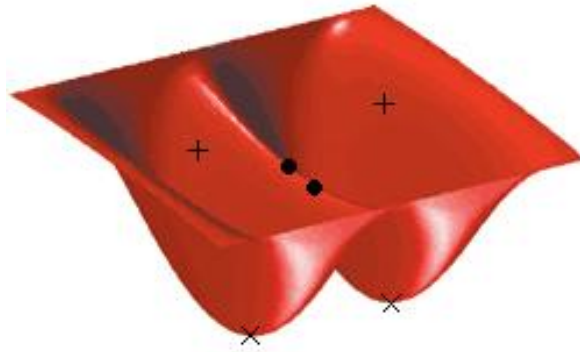


Figure 2.3 – Emplacements des points dans une image

- × : *minima locaux*
- + : *points occupant un bassin*
- : *points sur la frontière entre des bassins*

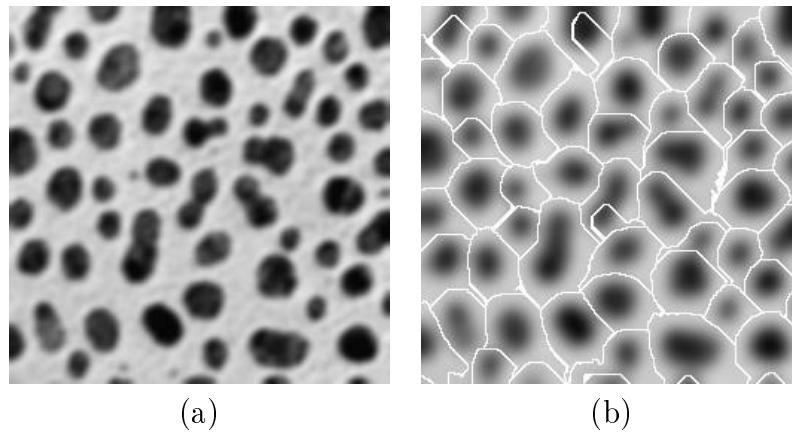


Figure 2.4 – Segmentation selon la méthode du watershed

- a) : *image à niveaux de gris*
- b) : *image segmentée selon la méthode du watershed*

poser une « digue » de pixels entre les bassins pour lesquels cette situation se produit, car le nombre de bassins ne doit jamais changer au cours du remplissage.

Le remplissage des bassins et la pose de digues se poursuit jusqu'à ce que tous les pixels de l'image fassent partie d'un bassin ou d'une frontière entre des bassins (les pixels posés comme digues). Les bassins définis au terme du remplissage constituent les régions qui résultent d'une segmentation d'images selon le *watershed* (voir Figure 2.4).

De manière formelle, posons M_1, M_2, \dots, M_R comme étant des ensembles décrivant les coordonnées des points des minima locaux d'une image $I(x, y)$, $C(M_i)$ comme étant l'ensemble des points du bassin associé au minimum local M_i , n comme étant un niveau de pixels et $T[n]$ l'ensemble des coordonnées (s, t) pour lesquelles $I(s, t) < n$. L'ensemble des coordonnées des points submergés d'un bassin dont le minimum est M_i est défini comme étant [16] :

$$C_n(M_i) = C(M_i) \cap T[n] \quad (2.4)$$

pour tout $\min(I) < n < \max(I)$. L'ensemble des pixels submergés au niveau n est quant à lui :

$$C[n] = \bigcup_{i=1}^R C_n(M_i) \quad (2.5)$$

Il s'ensuit que $C[\max(I) + 1]$ est l'ensemble de tous les bassins complètement remplis, tel que :

$$C[\max(I) + 1] = \bigcup_{i=1}^R C(M_i) \quad (2.6)$$

La procédure de remplissage des bassins peut être optimisée en évitant d'élever le niveau

de l'eau pour chaque valeur possible de n , mais en ne lui assignant que les valeurs possibles prises par les pixels de l'image, en ordre croissant [16]. Les différentes valeurs prises par les pixels d'une image, de même que leurs valeurs minimales et maximales (telles qu'utilisées un peu plus haut), peuvent être trouvées en traçant l'histogramme de l'image [36].

2.2.3 Méthode de croissance de régions

La segmentation d'images par *croissance de régions* consiste à regrouper des pixels d'une image en régions, selon un critère prédéterminé [16]. Le critère utilisé, qui peut être, par exemple, la variance ou la moyenne des régions, est une forme d'information a priori de *bas niveau*.

La *croissance de régions* commence par le choix de pixels (ou de régions connexes) de l'image, qui constituent les valeurs de départ de l'algorithme; c'est depuis ces valeurs que les régions de l'image vont croître, comme l'indique le nom de la méthode. Les valeurs de départ peuvent être choisies manuellement ou automatiquement à partir d'une heuristique. Dans le cas où une heuristique est utilisée, il est pertinent d'adapter la recherche de valeurs de départ au type d'image qu'on souhaite segmenter [16]. Si parmi les valeurs de départ se trouvent des régions composées de plus d'un pixel, on effectue l'érosion [36] de ces régions jusqu'à ce qu'elles ne comptent qu'un seul pixel. Les valeurs de départ ont ainsi toutes la même dimension, c'est-à-dire un pixel, avant qu'elles ne croissent en régions.

L'étape suivante consiste à composer une image binaire auxiliaire F_Q , de la même taille que l'image à segmenter I , où un pixel (x, y) dans F_Q prend la valeur 1 si le pixel de coordonnées (x, y) dans I satisfait au prédicat Q et la valeur 0 sinon, tel que :

$$F_Q(x, y) = \begin{cases} 1 & \text{si } Q(I(x, y)) = \text{VRAI} \\ 0 & \text{si } Q(I(x, y)) = \text{FAUX} \end{cases} \quad (2.7)$$

Le prédicat Q utilisé peut être, comme dans le cas du *seuillage*, un certain critère auquel doit satisfaire l'intensité des pixels. Il peut également être plus complexe et concerner la couleur, la texture ou le gradient des pixels.

En posant S comme étant l'ensemble des valeurs de départ, on forme une image G , de la même taille que l'image à segmenter I , en agrégeant à chaque point de S tous les pixels de F_Q qui ont la valeur 1 et qui lui sont connexes. Cette image G constitue la segmentation résultant de la *croissance de régions* (voir Figure 2.5). Dans l'hypothèse où les valeurs de départ sont adéquatement choisies et en considérant les pixels de S et les régions connexes de G et de F_Q comme les éléments d'ensembles, il découle que :

$$S \subseteq G \subseteq F_Q \quad (2.8)$$

Ainsi, la segmentation d'images selon la méthode de la *croissance de régions* dépend de deux paramètres donnés en entrée : le prédicat Q , qui dicte la formation de l'image auxiliaire F_Q , et l'ensemble des valeurs de départ fournies.

2.2.4 Méthode de division et fusion

La segmentation d'images selon la méthode de *division et fusion* consiste à diviser récursivement une image en quadrants jusqu'à ce que chaque quadrant décrive une région homogène. Une région R_i est homogène si l'ensemble de ses pixels répond à un prédicat Q . Comme le nom de la méthode l'indique, une étape de fusion suit la division de l'image. Cette étape vise à fusionner les régions adjacentes homogènes entre elles. Formellement, la méthode de *division et fusion* s'exprime selon les postulats suivants [19] :

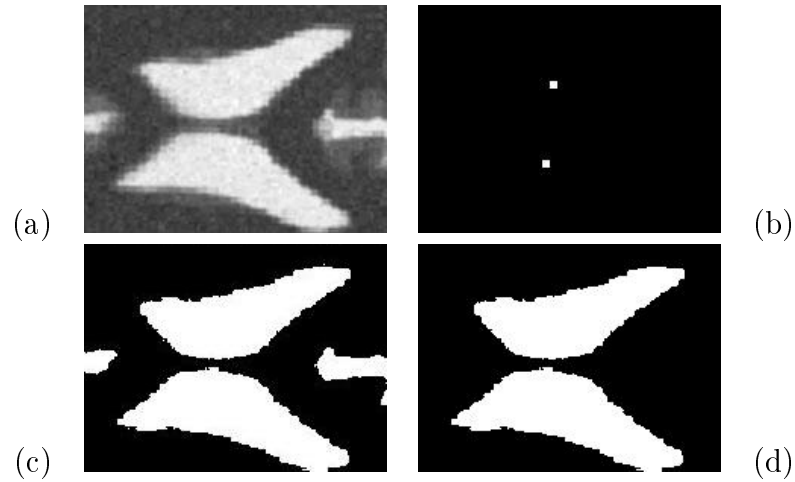


Figure 2.5 – Segmentation de l’image d’un ventricule cérébral grâce à la méthode de croissance de régions

- a)* : *image originale*
- b)* : *régions de départ (S)*
- c)* : *image auxiliaire (F_Q)*
- d)* : *image de la segmentation (G)*

- Diviser récursivement en quadrants disjoints toute région R_i tel que $Q(R_i) = \text{FAUX}$.
- Lorsque la division est terminée, fusionner récursivement chaque deux régions adjacentes R_i et R_j si $Q(R_i \cap R_j) = \text{VRAI}$.
- Lorsqu’aucune fusion n’est plus requise, la segmentation est terminée.

Comme dans le cas des méthodes du *seuillage* et de la *croissance de régions*, le prédicat Q peut concerner différentes caractéristiques des pixels qui composent les régions à diviser et fusionner (intensité, gradient, moyenne, écart-type, etc.). La segmentation qui résulte de la méthode de *division et fusion* consiste en un arbre de quadrants, où chaque feuille de l’arbre est une région homogène (voir Figure 2.6).

Selon le contexte où cette méthode est appliquée, on peut arrêter la division successive des quadrants lorsque ceux-ci atteignent une taille minimale [16] (voir Figure 2.7). Il est pertinent de préciser que, encore une fois selon l’application de la segmentation, les

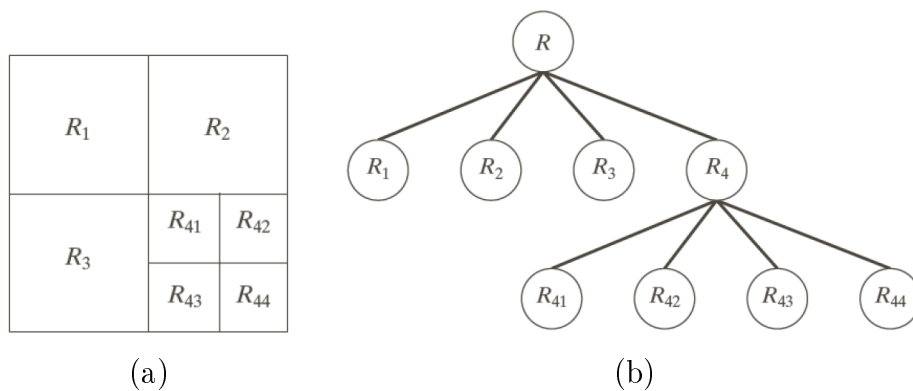


Figure 2.6 – Quadrants (régions homogènes) et leur arbre associé

a) : quadrants dans une images

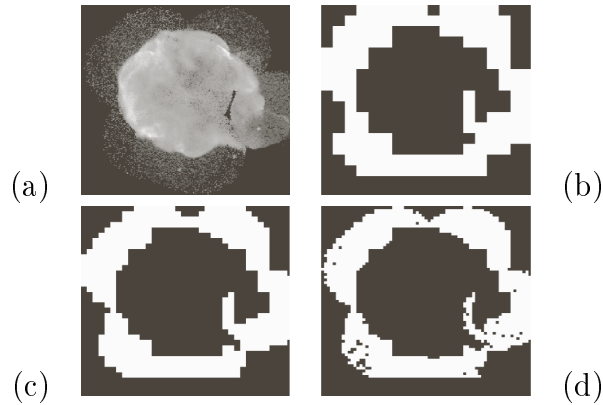
b) : arbre des quadrants

quadrants provenant de la division d'une région R_i n'ont pas forcément la même taille. Il est néanmoins possible de forcer cette condition en augmentant la taille de l'image à segmenter, de manière à ce que cette dernière soit de dimensions $n \times n$, où n est une puissance de 2.

2.2.5 Méthode du mean-shift

La méthode du *mean-shift* est une méthode statistique qui, contrairement aux quatre méthodes mentionnées précédemment, ne requiert pas d'information a priori concernant une image à segmenter. En effet, les méthodes du *seuillage*, de *croissance de régions* et de *division et fusion* requièrent toutes un prédicat qui permet de guider la formation de régions ; le prédicat utilisé constitue dans tous les cas un critère discriminant pour établir à quelle région appartient un pixel donné. La méthode du *watershed* requiert quant à elle que l'on sache selon combien de minima locaux le remplissage des bassins s'effectuera.

La méthode du *mean-shift* consiste à trouver des agrégats (*clusters*) de données dans



(cf. [16])

Figure 2.7 – *Division et fusion* : différentes tailles minimales de quadrants

- a) : *image originale*
- b) : *quadrants de taille 32×32*
- c) : *quadrants de taille 16×16*
- d) : *quadrants de taille 8×8*

l'espace de caractéristiques d'une image [11]. Les dimensions de l'espace de caractéristiques correspondent aux différents attributs des pixels de l'image qui sont retenus. Les points de l'espace de caractéristiques correspondent quant à eux aux pixels de l'image. Un agrégat présent dans l'espace de caractéristiques témoigne d'un ensemble de pixels semblables qui appartiendront potentiellement, au terme de la segmentation, à la même région.

Trouver les agrégats revient alors à trouver les endroits dans l'espace de caractéristiques où il y a une concentration plus élevée de points. La méthode des noyaux est un outil statistique non paramétrique à la base du calcul de la concentration des données. Il s'agit d'une méthode qui calcule la densité de probabilité. Précisons qu'il existe différents noyaux ; deux des principaux sont le noyau normal (K_N) et le noyau d'Epanechnikov (K_E). Posons c comme étant une constante positive qui fait en sorte que l'intégration d'un noyau quelconque $K(x)$ soit égale à 1 ($\int_{-\infty}^{\infty} K(x)dx = 1$). Les fonctions K_N et K_E

sont définies comme suit :

$$K_N(x) = c e^{-\frac{1}{2}|x|^2} \quad (2.9)$$

$$K_E(x) = \begin{cases} c(1 - |x|^2) & \text{si } |x| \leq 1, \\ 0 & \text{sinon} \end{cases} \quad (2.10)$$

La méthode du *mean-shift* concentre les points de l'espace de caractéristiques en agrégats. Ensuite, il ne reste qu'à trouver à quel agrégat chaque point doit être associé. L'agrégat auquel est associé un point dépendra des points qui sont situés dans son voisinage. Là où se trouve la plus grande densité dans le voisinage d'un point tendra à indiquer la direction de l'agrégat le plus proche. Le gradient de la distribution (autrement dit, la première dérivée de son noyau) est l'élément qui renseigne sur la direction de l'agrégat. Il en découle que :

- Un point dont le gradient est nul (ou proche de zéro) indique qu'il appartient à une région de l'espace de caractéristiques où les données ont une certaine uniformité, c'est-à-dire où elles varient peu.
- Un point dont le gradient est élevé indique que la densité de son voisinage est plus grande ; ce point est donc probablement situé sur la frontière d'une région.

Les agrégats sont trouvés de façon itérative, en partant de n'importe quel point de l'espace de caractéristiques et en se rapprochant graduellement du centre d'un agrégat (voir Figure 2.8). À partir d'un point de départ x_t et des points situés dans son voisinage, le vecteur *mean-shift* $\vec{m}(x_t)$ est calculé. Le point de départ x_{t+1} de l'itération suivante est obtenu en additionnant le vecteur *mean-shift* $\vec{m}(x_t)$ à x_t . Formellement, posons x_i comme étant un point du voisinage de x_t , n comme étant le nombre de pixels situés dans le voisinage de x_t , g comme étant le gradient d'un point et h comme étant un paramètre de lissage. Le vecteur *mean-shift* est calculé ainsi [11] :

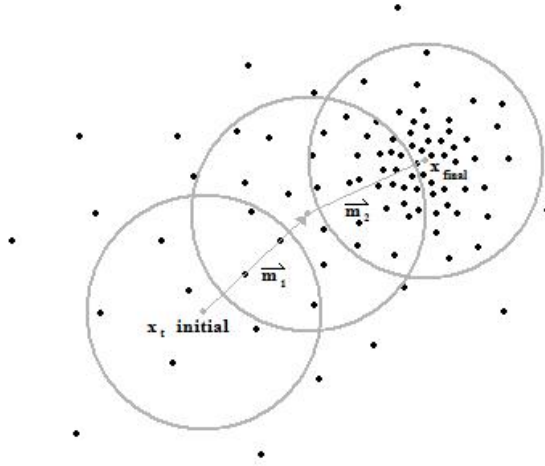


Figure 2.8 – Distribution de points, les points x_t *initial* et x_t *final* et les vecteurs *mean-shift* \vec{m}_1 et \vec{m}_2

$$\vec{m}(x_t) = \frac{\sum_{i=1}^n x_i g\left(\left|\frac{x_t - x_i}{h}\right|^2\right)}{\sum_{i=1}^n g\left(\left|\frac{x_t - x_i}{h}\right|^2\right)} - x_t \quad (2.11)$$

Après un certain nombre d'itérations, le vecteur *mean-shift* s'annule et la valeur x_t converge vers le centre de l'agrégat le plus proche [11]. En effectuant cette opération pour chaque pixel d'une image, on obtient des regroupements de pixels qui sont tous, au sein d'un même regroupement, associés au même agrégat. Une image segmentée selon la méthode du *mean-shift* résulte donc en un ensemble de régions où chacune correspond à un agrégat (voir Figure 2.9). Il importe également de noter que plus d'une région peut être associée à un même agrégat.



Figure 2.9 – Segmentation *mean-shift* d'une image
 a) : *image naturelle*
 b) : *image segmentée selon la méthode du mean-shift*

2.3 Approche descendante

Comme il a été énoncé au chapitre précédent, les méthodes de segmentation d'images dites *descendantes* définissent des segments dans une image conformément à un modèle connu au préalable. Le modèle dont on dispose constitue une information a priori, car il s'agit de connaissances préalables sur les objets qui seront recherchés au cours de la segmentation. Au centre de ces méthodes se trouve donc la forme que prend cette information a priori et la façon selon laquelle elle guide la segmentation.

L'information a priori utilisée peut être introduite sous différents aspects. Dans le cadre des quelques méthodes de segmentation d'images décrites dans cette section, l'information a priori peut être :

- de *bas niveau*, comme dans le cas du paramètre de lissage d'une distribution (comme mentionné à la sous-section 2.2.5) ;
- de *niveau moyen*, comme dans le cas de critères statistiques (moyenne, écart-type, etc.) auxquels doivent satisfaire les régions résultant de la segmentation (comme mentionnés

à la sous-section 2.2.3) ;

- de *niveau élevé*, comme dans le cas de formes particulières que doivent posséder les régions résultant de la segmentation.

2.3.1 Segmentation par contours actifs

Un *contour actif* est une courbe paramétrique définie sur le plan de l'image. La segmentation par *contours actifs* consiste à déformer un ou plusieurs contours initiaux pour capter une ou plusieurs régions dans l'image. La déformation des contours se fait sous des contraintes physiques où l'évolution de ces derniers est gouvernée par une fonctionnelle d'énergie.

La position finale du contour est déterminée par la minimisation de deux énergies : l'*énergie interne*, qui quantifie sa régularité intrinsèque (autrement dit, l'absence de zig-zags) et l'*énergie externe*, qui quantifie le fait que le contour soit dans une position qui correspond à de fortes discontinuités de l'image (autrement dit, les frontières de régions). Lorsque le contour atteint sa cible avec succès, l'énergie doit être dans son état minimal.

Cette sous-section aborde deux méthodes de *contours actifs* : la *méthode des snakes* [38] et la *méthode des ensembles de niveaux* (*level sets*).

Méthode des *snakes*

Un *snake* est une courbe paramétrique définie sur le plan de l'image. Posons $s \in [0, 1]$ comme étant le paramètre de parcours de la courbe. L'équation paramétrique d'un *snake* \vec{C} est définie comme suit [37] :

$$\begin{aligned} \vec{C} : [0, 1] &\rightarrow \mathbb{R}^2 \\ s &\mapsto (x(s), y(s)) \end{aligned} \tag{2.12}$$

Il existe des *snakes* ouverts et des *snakes* fermés. Dans ce travail, nous ne nous intéresserons qu'aux *snakes* fermés, c'est-à-dire ceux pour lesquels $\vec{C}(0) = \vec{C}(1)$.

La segmentation d'images utilisant les *snakes* consiste à déformer un ou plusieurs *snakes* à partir d'un état initial, déterminé en général manuellement, vers un état final qui correspond aux frontières de la région à segmenter. Le mouvement du *snake* est régi par un ensemble de forces dérivées de l'énergie interne du *snake* (son caractère lisse) et de l'information externe du *snake* (sa correspondance aux données de l'image). Le mouvement est effectué de manière itérative (voir Figure 2.10) en minimisant la fonctionnelle d'énergie, qui regroupe les énergies interne (E_{int}) et externe (E_{ext}) du *snake*. À chaque itération, le *snake* se déforme et se rapproche de plus en plus de la frontière de la région cible. Pour un *snake* continu sur l'intervalle de s $[0, 1]$, la fonctionnelle s'exprime formellement comme suit :

$$\int_0^1 \left(E_{int}(\vec{C}(s)) + E_{ext}(\vec{C}(s)) \right) ds \quad (2.13)$$

À terme, le résultat de cette méthode sera le *snake* qui minimise à la fois son énergie interne et son énergie externe ; c'est-à-dire, respectivement, le *snake* qui possède à la fois un certain caractère lisse et qui correspond le mieux à la frontière de la région cible.

On peut aussi ajouter de l'information sur la forme à laquelle le *snake* final doit se conformer. Dans ce cas, la fonctionnelle d'énergie doit également comporter le terme $E_{forme}(\vec{C}(s))$, qui décrit cette contrainte sur le *snake* déformé. La contrainte ainsi introduite constitue de l'information a priori.

Méthode des ensembles de niveaux

La faiblesse principale de la méthode des *snakes* consiste en son incapacité à subir des changements topologiques. En effet, puisque l'équation d'un *snake* (voir l'équation (2.12))

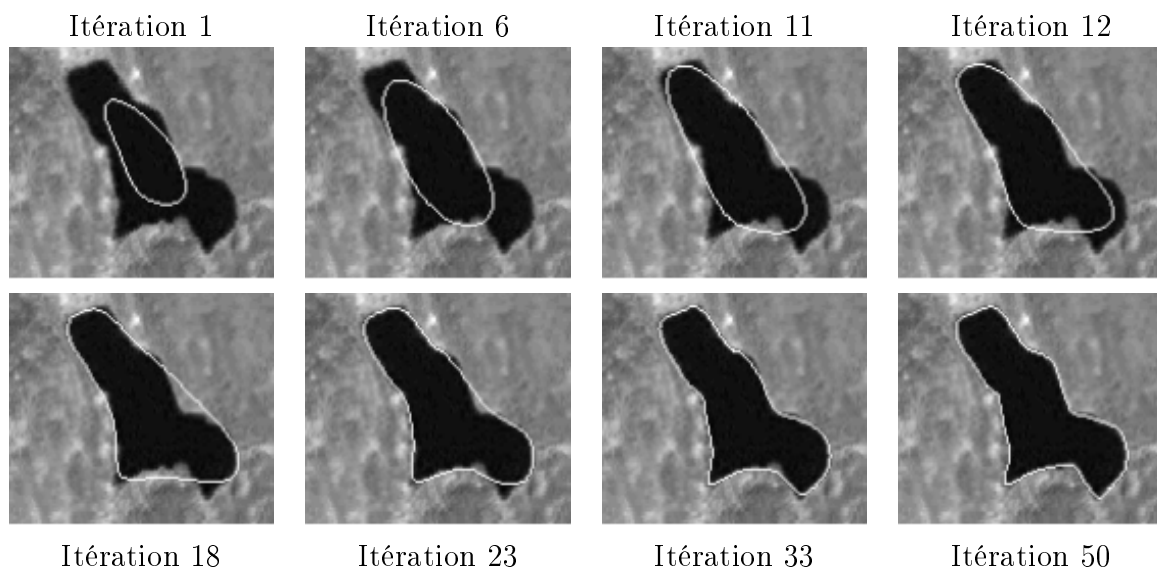


Figure 2.10 – *Snake* décrit à différentes itérations de la segmentation d'une image

ne délimite qu'un seul contour, un *snake* ne peut se décomposer en plusieurs parties. Il n'est donc pas possible de délimiter plus d'un contour (autrement dit, segmenter plus d'un objet) dans une image grâce à la méthode des *snakes*. Cette limite a été comblée par la *méthode des ensembles de niveaux* (*level sets*) [30][1], introduite après les *snakes*.

Dans cette méthode, le contour déformé correspond au niveau zéro (N_0) d'une fonction tridimensionnelle implicite $\phi(x, y)$ qui est définie comme une fonction de distance :

$$\phi(x, y) = \begin{cases} -d((x, y), N_0) & \text{si } (x, y) \text{ est à l'intérieur de } N_0 \\ d((x, y), N_0) & \text{si } (x, y) \text{ est à l'extérieur de } N_0 \\ 0 & \text{si } (x, y) \text{ est sur } N_0 \end{cases} \quad (2.14)$$

La courbe d'intérêt \vec{C} , qui est supposée capter la frontière d'une région (donc d'un objet dans une image), est définie comme l'ensemble des points de coordonnées (x, y) situés au niveau zéro de la fonction ϕ :

$$\vec{C} = \{(x, y) \mid \phi(x, y) = 0\} \quad (2.15)$$

La segmentation par la *méthode des ensembles de niveaux* se fait en déformant \vec{C} dans le temps pour minimiser la fonctionnelle d'énergie (voir l'équation (2.13)). Selon les déformations appliquées, \vec{C} peut changer de topologie (voir Figure 2.11). La segmentation converge quand la courbe d'intérêt \vec{C} correspond à la frontière de l'objet ou des objets ciblé(s) (voir Figure 2.12).

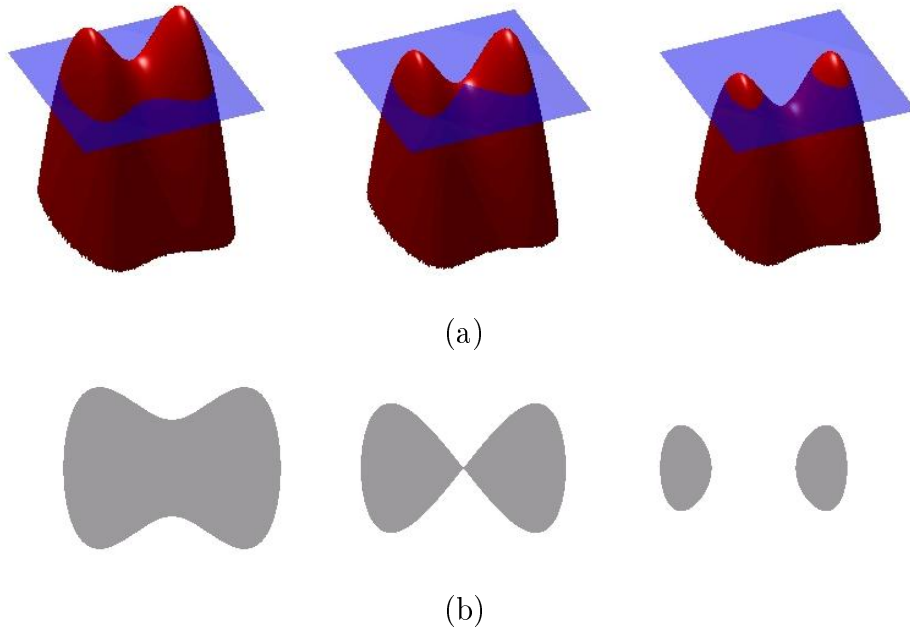


Figure 2.11 – *Méthode des ensembles de niveaux* à trois moments dans le temps
a) : fonction tridimensionnelle implicite ϕ (en rouge)
b) : régions délimitées par N_0

La fonctionnelle d'énergie utilisée dans cette méthode est similaire à celle utilisée dans la méthode des *snakes* ; il s'agit de minimiser les énergies interne et externe des contours. Il est également possible, comme dans le cas de la méthode des *snakes*, d'imposer des contraintes additionnelles, par exemple, quant à la forme des contours [26].

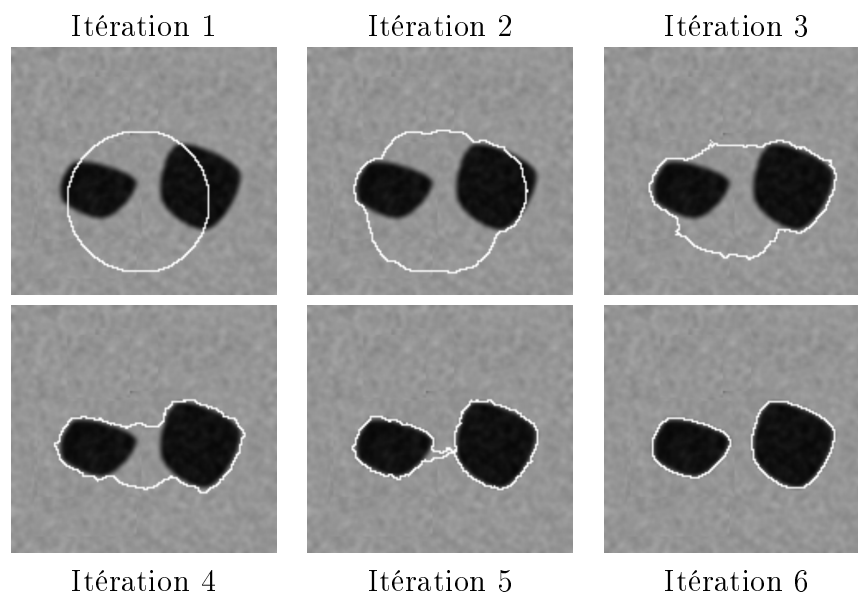


Figure 2.12 – Ensembles de niveaux décrits à différentes itérations de la segmentation d’une image

2.3.2 Méthode des *K-moyennes*

La méthode des *K-moyennes* est une méthode statistique semblable à la méthode du *mean-shift* par le fait qu’elle vise à segmenter une image en recherchant des agrégats dans la distribution de cette dernière. À la différence de la méthode du *mean-shift*, le nombre d’agrégats à rechercher dans l’image doit être précisé au préalable.

On suppose dans cette méthode que chaque région de l’image est homogène par rapport à un certain critère. Ce critère peut être le niveau de gris (pour une image à niveaux de gris), la couleur (pour une image couleur), la texture, etc. On peut alors représenter chacune des régions par la moyenne du critère utilisé. Le but de la méthode des *K-moyennes* est de trouver ces moyennes, qui constitueront les centres des agrégats.

La méthode des *K-moyennes* fonctionne de manière itérative. À chaque itération, on associe d’abord chaque pixel à l’agrégat duquel il est le plus rapproché, puis on recalcule

le centre de chaque agrégat en effectuant la moyenne des pixels qu'on leur a associés. Le processus s'arrête si les centres des agrégats se stabilisent, c'est-à-dire qu'ils ne changent pas après avoir été recalculés. Posons K comme étant le nombre d'agrégats qui sont recherchés dans l'image, les centres de chaque agrégat comme étant $\mu_1, \mu_2, \dots, \mu_K$ et x_1, x_2, \dots, x_N comme étant les données de l'image. Le processus associé à la méthode des K -moyennes se déroule ainsi :

1. Initialiser les centres $\mu_1, \mu_2, \dots, \mu_K$
2. Associer chaque donnée x_i au centre μ_k le plus près, tel que :

$$r_{ik} = \begin{cases} 1 & \text{si } k = \operatorname{argmin}_j |x_i - \mu_j| \\ 0 & \text{sinon.} \end{cases} \quad (2.16)$$

3. Recalculer chaque centre μ_k , tel que :

$$\mu_k = \frac{\sum_{i=1}^N r_{ik} x_i}{\sum_{i=1}^N r_{ik}} \quad (2.17)$$

4. Si aucun centre n'a changé de valeur à l'étape précédente, le processus est terminé ; autrement, retourner à l'étape 2.

Le résultat final de la segmentation selon cette méthode (voir Figure 2.13) dépend exclusivement de deux informations fournies a priori :

- le K choisi, c'est-à-dire le nombre d'agrégats à rechercher ;
- les valeurs auxquelles les centres μ_k sont initialisés.

2.3.3 Méthode par coupures de graphes

La méthode de segmentation d'images par coupures de graphes (*graph cuts*) [8][35] représente une image comme un graphe : les nœuds du graphe sont les pixels de l'image et ses arêtes correspondent aux liens de voisinage entre pixels, pour un maximum de quatre



Image originale



K=2



K=8



K=11



K=14



K=15

Figure 2.13 – Image segmentée selon la méthode des *K-moyennes* à l'aide de différents K

voisins par pixel. Chaque arête est pondérée par une valeur qui équivaut à la ressemblance que démontrent les deux pixels associés aux nœuds qu'elle relie. Les arêtes qui ont une valeur très basse — ou même nulle — indiquent donc les paires de pixels qui sont les plus différents. Une séquence d'arêtes consécutives qui ont toutes une valeur très basse dénote une frontière dans l'image.

La méthode en tant que telle vise à effectuer une « coupure » dans le graphe, par ses arêtes, qui soit à la fois la moins coûteuse et la plus régulière. Le coût de la coupure d'une arête est le poids qu'on lui a attribué, tandis que sa régularité (son caractère lisse) peut être calculée d'une façon similaire à ce qui est fait dans les méthodes décrites à la section 2.3.1. L'opération dans son ensemble revient à optimiser une fonction de coût qui suit le modèle de Gibbs [15]. Posons f comme étant une segmentation d'image (une coupure) donnée. La fonction de coût C de la coupure peut s'exprimer ainsi :

$$C(f) = C_{données}(f) + C_{régularité}(f) \quad (2.18)$$

Pour situer l'emplacement de la coupure, deux pixels d'amorce (ou *seeds*) sont fournis : un pour une région jugée faisant partie de l'objet (*source*) et un pour une région jugée faisant partie du fond (*sink*). Ces pixels d'amorce sont fournis comme une connaissance a priori. Au terme de la segmentation, un objet sera donc « découpé » du fond de manière à ce que son contour soit aussi régulier que possible tout en correspondant aux frontières d'objet(s) observées dans l'image (voir Figure 2.14 et Figure 2.15).

2.3.4 Méthodes utilisant des *patches*

Les méthodes qui utilisent des *patches* [3][5][6][23][24][33] diffèrent grandement des méthodes abordées jusqu'ici dans cette section par le fait qu'elles requièrent une certaine

connaissance a priori plus complexe. Cette dernière consiste en des *patches*, qui sont des fragments génériques d'image susceptibles d'être retrouvés dans un objet en particulier.

Le reste de cette sous-section explique le fonctionnement typique d'une méthode utilisant des *patches*. Pour chaque classe d'objets à segmenter dans des images, il faut au préalable bâtir une collection de fragments qui y sont associés. Cette collection de fragments constitue en quelque sorte un modèle pour la classe donnée. On bâtit une telle collection en extrayant des fragments d'images d'apprentissage fournies au préalable. Certains critères peuvent guider le choix initial des fragments. Le caractère saillant de certaines zones des images peut être estimé grâce au détecteur de Harris [18] ou encore grâce à la méthode SIFT [25]. Les fragments qui sont fréquemment observés dans différentes images d'apprentissage, ou même plus d'une fois dans une même image d'apprentissage, sont retenus.

On recherche ensuite les fragments retenus dans les images à segmenter. Les endroits dans l'image à segmenter où des fragments sont observés renseignent sur l'emplacement d'un objet de la classe recherchée. On peut également adjoindre aux fragments certaines informations additionnelles, telles que :

- leur(s) position(s) probable(s) dans une image à segmenter ;
- leur position par rapport à d'autres fragments dans une image à segmenter.

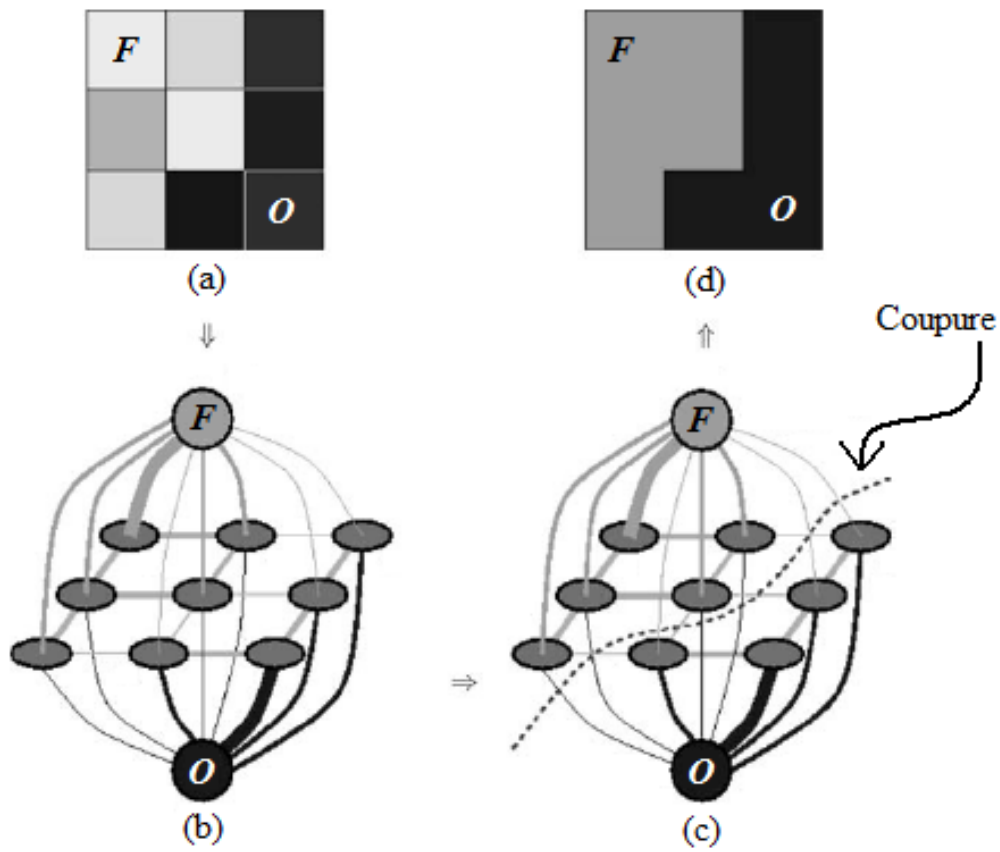
Ces informations accélèrent et facilitent la recherche, en plus de rendre plus complet le modèle d'une classe d'objets que l'on détient.

Les méthodes utilisant des *patches* se distinguent de la plupart des méthodes de segmentation d'images par le fait qu'elles effectuent implicitement la reconnaissance d'objets. La reconnaissance d'objets dans ces méthodes découle de la recherche de fragments dans des images. Comme il a été mentionné à la section 1.4, la reconnaissance d'objets peut être utilisée pour corriger, voire aider à effectuer, la segmentation d'objets.

2.4 Comparaison des méthodes présentées

Ce chapitre a abordé quelques-unes des principales méthodes de segmentation d'images. Bien qu'il n'existe pas une méthode qui soit meilleure que toutes les autres, chacune d'entre elles possède ses avantages ainsi que ses inconvénients. En effet, la performance d'une méthode peut être mesurée par plusieurs critères, tels que la rapidité de l'exécution et la qualité de la segmentation obtenue ainsi que la facilité de sa généralisation à plusieurs régions. D'autres critères, comme le degré de lissage des contours ou le besoin d'interactivité avec l'utilisateur, peuvent aussi être considérés.

Le tableau 2.1 vise à mettre en perspective les différentes méthodes qui ont été décrites en résumant leurs avantages et inconvénients respectifs.



(cf. [7])

Figure 2.14 – Illustration de la méthode des coupures de graphes

- a)* : image originale et pixels d'amorce
- b)* : image représentée par un graphe
- c)* : coupure du graphe
- d)* : résultat de la segmentation

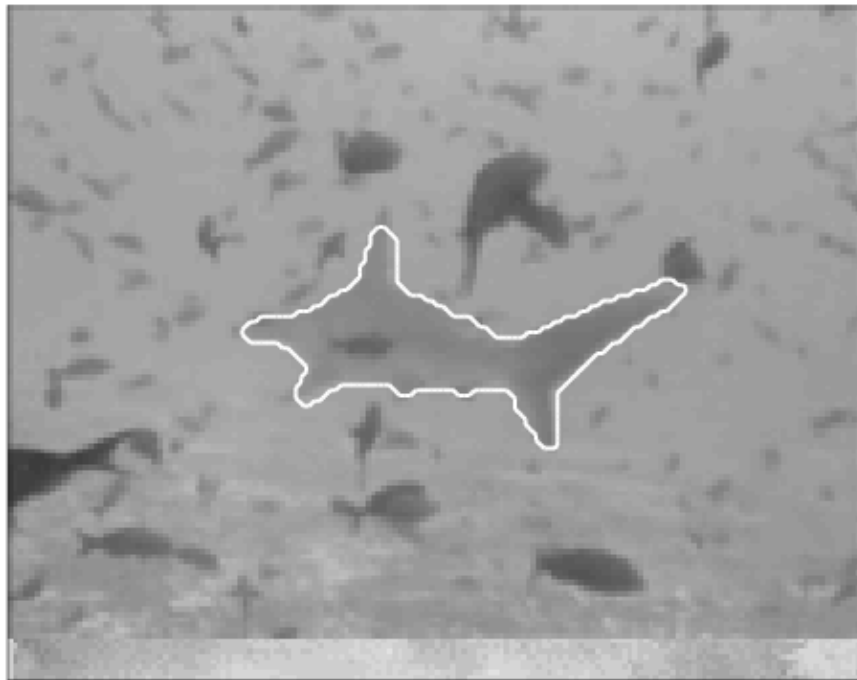


Figure 2.15 – Méthode des coupures de graphes appliquée à une image naturelle

Tableau 2.1 – Méthodes de segmentation d'images : avantages et inconvénients

Méthode	Avantages	Inconvénients
Seuillage	-Implémentation simple -Exécution rapide	-Difficilement applicable aux images en couleurs -Ne tient pas compte de l'information spatiale -Difficilement généralisable pour N classes (où $N > 2$)
Watershed	-Bien adaptée aux images topographiques -Exécution rapide	-Difficilement applicable aux images naturelles
Croissance de régions	-Plus flexible que le seuillage -Exécution rapide	-Prédicat difficile à définir -Difficilement généralisable pour N classes (où $N > 2$)
Division et fusion	-Implémentation simple -Tient compte de l'information spatiale	-Prédicat difficile à définir -Frontières en zigzags
Mean-shift	-Tient compte de l'information spatiale -Facilement extensible à N classes (où $N > 2$)	-Procédé lent
Contours actifs (<i>snakes</i>)	-Possibilité d'introduire de l'information a priori	-Incapacité d'adaptation aux changements topologiques -Implémentation complexe -Procédé lent -Sensible à l'initialisation
Contours actifs (ensembles de niveaux)	-Flexible quant à la topologie des régions	-Implémentation complexe -Procédé lent
K-moyennes	-Implémentation simple -Exécution rapide	-Valeur K doit être bien choisie -Ne tient pas compte de l'information spatiale
Coupures de graphes	-Frontières lisses	-Implémentation complexe -Difficile d'introduire de l'information a priori
Méthodes utilisant des <i>patches</i>	-Combine segmentation d'images et reconnaissance d'objets	-Dépend des images d'apprentissage

Chapitre 3

Problématique

3.1 Aperçu de la problématique

Les deux chapitres précédents ont expliqué en quoi consiste la segmentation d'images et quelles sont les principales méthodes utilisées pour la réaliser. Les méthodes de segmentation d'images peuvent être réparties entre méthodes par *forme complète*, où l'entière silhouette d'un objet est considérée, et méthodes par *patches*, abordées en 1.4 et en 2.3.4, où un objet est décrit par un certain nombre de fragments. Bien que l'état des connaissances en ce qui concerne les méthodes par *forme complète* soit plus étendu que celui des méthodes par *patches*, les méthodes par *forme complète* ont des faiblesses inhérentes qui les rendent incommodes pour l'usage que nous voulons en faire dans le cadre de ce travail. Notamment, elles ne peuvent être appliquées de façon fructueuse lorsque, par exemple, les objets à segmenter ont une forme variable, ou encore lorsque les objets à segmenter ne sont que partiellement visible. Au vu de ces lacunes, et en accord avec ce que nous avons avancé à la section 1.2 à propos de la segmentation d'images, il convient d'adopter une approche qui est davantage similaire à la vision humaine. Par conséquent, l'approche

par *patches* est retenue et c'est ce que vise à explorer plus profondément ce chapitre.

La motivation derrière l'approfondissement de cette approche, plutôt que celle par *forme complète*, repose principalement sur les points suivants :

- il s'agit d'une approche prometteuse, car elle est inspirée de la vision humaine ;
- il y a possibilité de segmenter partiellement des objets ;
- la variabilité des objets est prise en compte ;
- l'information a priori utilisée est extraite directement d'exemples d'objets.

Comme il a été mentionné à la section 1.4, le travail effectué dans le cadre de ce mémoire s'articule autour de deux axes : l'acquisition de la connaissance a priori dans le but de former un modèle pour une classe d'objets donnée et la segmentation d'objets de cette classe dans de nouvelles images à l'aide du modèle.

3.2 Modèle : formation et utilisation

Le premier problème auquel nous nous intéressons est la forme que prend le modèle associé à une classe d'objets donnée. Ce modèle est composé d'un ensemble de *patches* (ou fragments). Les fragments constituent la connaissance a priori introduite dans le procédé de segmentation.

Le modèle d'une classe d'objets représente ces derniers par la forme que prend leur contour. Nous motivons le choix de décrire les objets par leur contour par le fait que le contour est une des principales caractéristiques utilisées par l'être humain pour identifier un objet. Dans une série d'expériences, Needham [29] a observé le comportement de nourrissons à qui l'on a montré des objets qui variaient en couleur, en orientation ou en forme. L'une des conclusions qu'elle tire est qu'un nourrisson semble reconnaître plus rapidement un objet dont seule la couleur ou l'orientation a changé qu'un objet dont

la forme paraît différente. La méthode utilisée pour faire paraître différente la forme d'un objet consiste à cacher partiellement cet objet derrière un autre, masquant ainsi une partie de son contour. Ceci démontre que la forme d'un objet est d'une grande importance quant à son identité.

Chaque fragment du modèle couvre une partie du contour probable d'un objet de la classe en question. Le fait de définir suffisamment de fragments qui se chevauchent fait en sorte que la totalité du contour qui décrit la silhouette d'un objet peut être couverte.

Ainsi, la seule connaissance que le modèle contient concerne les différentes configurations des contours d'un objet donné. Aucune autre information, par exemple en ce qui a trait à la texture ou à la couleur d'une classe d'objets, n'est contenue dans le modèle. Ce type d'information supplémentaire peut cependant être rajoutée plus tard au modèle.

3.2.1 Extraction de la connaissance a priori

Les fragments sont extraits des images d'apprentissage. L'extraction d'un fragment se fait de manière à maximiser l'information que ce dernier possède au regard de la classe d'objets qu'il vise à caractériser. L'ensemble de fragments qui définit une classe doit ne pas démontrer une trop grande redondance. Posons un fragment potentiel I_c et un ensemble de fragments (un modèle d'objet) existant M_c . Pour qu'il soit ajouté au modèle, un fragment I_c doit d'abord satisfaire au critère suivant : $I_c \notin M_c$. Autrement dit, chacun des fragments que contient un modèle doit être différent de tous les autres.

Une fois qu'il a été établi que le fragment I_c n'appartient pas déjà au modèle M_c , il faut s'assurer qu'il y apporte une quantité appréciable d'information. Pour cela, nous pouvons comparer le fragment I_c à tous les fragments contenus dans le modèle pour déterminer s'il ressemble trop à l'un d'eux. Cette comparaison peut être réalisée en mettant en parallèle le fragment I_c avec chacun des autres fragments et en calculant leur correspondance. Nous

posons qu'un fragment potentiel qui correspond fortement à un fragment se trouvant déjà dans le modèle (autrement dit, qui contient essentiellement la même information que lui) ne peut être ajouté au modèle.

Définition des fragments

Comme il a été mentionné à la section 3.2, un fragment contient une partie du contour d'un objet d'une certaine classe. De façon générale, un fragment est une image à niveaux de gris. Dans notre cas, et ce, pour simplifier le modèle que nous formons, un fragment est une image binaire (donc un ensemble de pixels) où les pixels blancs sont les pixels de l'objet et où les pixels noirs sont les pixels du fond. Un fragment typique (voir Figure 3.1) définit deux régions distinctes : une région d'objet (les pixels blancs) et une région de fond (les pixels noirs). La frontière entre ces deux régions constitue la partie du contour d'un objet que décrit le fragment.

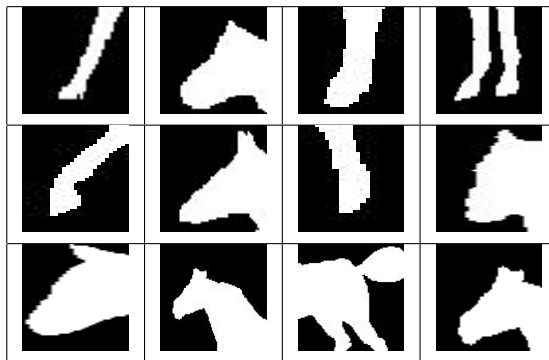


Figure 3.1 – Exemples de fragments de la classe *cheval*

Extraction des bons fragments

Les fragments sont extraits de l'une ou l'autre des images d'apprentissage qui contiennent un objet de la classe de laquelle on souhaite former un modèle (ces images sont ci-

après appelées *exemples positifs*). Un fragment potentiel est d'abord obtenu en cadrant une partie d'un de ces exemples. Cela dit, pour être ajouté au modèle, un fragment potentiel doit satisfaire à certains critères. Ces critères concernent la *quantité minimale d'information* que doit contenir ce fragment, sa *validité*, sa *pertinence* ainsi que sa *taille*.

Le premier critère concerne la *quantité minimale d'information* qu'un fragment doit détenir à propos de la classe. Un fragment ne peut, par exemple, contenir que des pixels de l'objet ou que des pixels du fond. En effet, dans l'un ou l'autre de ces cas, aucune information sur la frontière entre l'objet et le fond (donc sur le contour de l'objet) n'est fournie. Le même problème se présente si le nombre de pixels du fragment qui décrivent une frontière entre l'objet et le fond est trop petit. Il appert dès lors convenable d'introduire une contrainte sur les proportions minimale et maximale du nombre de pixels d'objet par rapport au nombre total de pixels. Posons I_c comme étant un fragment, F_{objet} comme étant l'ensemble des pixels d'objet de I_c , et δ_p comme étant un seuil de nombre de pixels (où $\delta_p \in]0, 0.5[$). L'intervalle dans lequel la proportion de pixels d'objet dans un fragment doit se situer s'énonce formellement comme suit :

$$\delta_p \leq \frac{|F_{objet}|}{|I_c|} \leq (1 - \delta_p) \quad (3.1)$$

où $|F_{objet}|$ et $|I_c|$ représentent respectivement les cardinalités de l'ensemble des pixels d'objet et de l'ensemble de tous les pixels de I_c .

Le deuxième critère concerne la *validité* d'un fragment. Pour être valide, un fragment doit caractériser les images qui font partie de la classe en question. Un fragment caractérise une image si un contour semblable à ce qu'il contient peut être retrouvé à une certaine position dans l'image. La similitude entre le contour contenu dans le fragment et celui observé dans l'image s'obtient en calculant leur correspondance. La *validité* d'un fragment est vérifiée en calculant sa meilleure correspondance dans chacun des *exemples positifs*.

Posons I_c comme étant un fragment de la classe c , I comme étant un *exemple positif*, i comme étant la position d'un pixel et x comme étant la position de I_c dans I où on calcule la correspondance. La fonction de correspondance C qui calcule la proportion des pixels identiques entre I_c et I à la position x est définie comme suit :

$$C(I_c, I(x)) = 1 - \frac{\sum_i (I_c(i) - I(x+i))^2}{|I_c|}, \quad I_c(i) \in \{0, 1\} \text{ et } I(x+i) \in \{0, 1\} \quad (3.2)$$

où $|I_c|$ représente le nombre de pixels du fragment I_c . Après avoir calculé la fonction (3.2) d'un couple (I_c, I) pour toutes les positions x possibles de I_c dans I , on calcule la position m qui maximise C . Ce calcul s'énonce formellement comme suit :

$$m = \operatorname{argmax}_x (C(I_c, I(x))) \quad (3.3)$$

La valeur optimale de C doit satisfaire à un certain seuil de correspondance δ_c (où $\delta_c \in [0, 1]$) tel que :

$$C(I_c, I(m)) \geq \delta_c \quad (3.4)$$

Ce seuil δ_c est le critère discriminant qui détermine la *validité* d'un fragment par rapport à un *exemple positif*. Le seuil doit être choisi adéquatement. Un seuil δ_c trop élevé exclura une trop grande proportion de fragments potentiels et la représentation que l'on aura de la classe sera insuffisante. Par contre, un seuil δ_c trop bas admettra des fragments qui ne caractérisent pas assez la classe d'objets en question. La valeur optimale de δ_c est celle qui maximise le terme R_C , décrit à l'équation (3.7).

L'étape suivante consiste à étendre le test de *validité* à tous les *exemples positifs*. On détermine ainsi si un fragment est valide au regard de suffisamment d'*exemples positifs*.

Posons un ensemble d'*exemples positifs* E_P qui contiennent un objet d'une classe donnée et un seuil de validité Δ_P (où $\Delta_P \in [0, 1]$). La *validité* d'un fragment peut s'exprimer formellement à l'aide de l'équation (3.5), qui se sert de la fonction (équation (3.6)).

$$\frac{\sum_i S(I_c, E_P(i))}{|E_P|} \geq \Delta_P \quad (3.5)$$

$$\text{où } S(I_c, E_P(i)) = \begin{cases} 1 & \text{si } C(I_c, E_P(i)) \geq \delta_c \\ 0 & \text{sinon.} \end{cases} \quad (3.6)$$

et où i est l'indice d'un *exemple positif* dans E_P et $|E_P|$ représente la cardinalité de l'ensemble des *exemples positifs*.

Le troisième critère auquel doit satisfaire un fragment potentiel concerne sa *pertinence*. Un fragment pertinent doit être caractéristique d'une classe d'objets sans cependant être si commun qu'on puisse également le retrouver dans des images qui ne font pas partie de cette même classe d'objets. Un fragment est jugé non pertinent s'il caractérise autant des images de la classe d'objets que des images qui ne sont pas de la classe d'objets. Il en découle que des contre-exemples d'images de la classe en question (ces images sont ci-après appelées *exemples négatifs*) sont nécessaires à l'extraction de fragments pertinents. Nous pouvons calculer la *pertinence* d'un fragment de façon analogue à ce qui est fait à l'équation (3.5). Posons E_N comme étant l'ensemble des *exemples négatifs* définis pour une classe donnée c . La *pertinence* du fragment I_c se calcule à l'aide de la fonction décrite en (3.6), avec comme paramètres $(I_c, E_N(k))$, où $E_N(k)$ est le k ième *exemple négatif* de l'ensemble E_N .

Pour un fragment donné, on peut dès lors définir les quatre cas selon lesquels il peut se présenter (ou non) dans les images d'apprentissage. Un fragment I_c peut soit être détecté ($D(I_c) = 1$) ou non ($D(I_c) = 0$) dans un *exemple positif* ($P_E = 1$) ou dans un *exemple négatif* ($P_E = 0$). Nous pouvons ainsi bâtir le tableau de contingence 3.1 :

Tableau 3.1 – Tableau de contingence d’un fragment

	$P_E = 1$	$P_E = 0$
$D(I_c) = 1$	$\#P_E^+$	$\#N_E^+$
$D(I_c) = 0$	$\#P_E^-$	$\#N_E^-$

En vérifiant la *pertinence* d’un fragment quant aux *exemples positifs* et *negatifs*, on peut remplir pour ce fragment une matrice de contingence M_C qui a la même structure que le tableau 3.1. Les deux éléments contenus dans la diagonale principale de cette matrice ($\#P_E^+$ et $\#N_E^-$) représentent, respectivement, le nombre de fois où le fragment a été trouvé dans les *exemples positifs* ($D(I_c) = 1 \wedge P_E = 1$) et celui où il n’a pas été trouvé dans les *exemples négatifs* ($D(I_c) = 0 \wedge P_E = 0$). Les deux éléments contenus dans la diagonale secondaire ($\#P_E^-$ et $\#N_E^+$) représentent quant à eux respectivement le nombre de fois où le fragment n’a pas été trouvé dans les *exemples positifs* ($D(I_c) = 0 \wedge P_E = 1$) et celui où il a été trouvé dans les *exemples négatifs* ($D(I_c) = 1 \wedge P_E = 0$). La trace de cette matrice (la somme des éléments de sa diagonale principale) répertorie tous les cas souhaitables pour un fragment. Le rapport de la trace et de la somme des éléments de la matrice de contingence se nomme rapport de contingence (R_C). Ce dernier s’énonce formellement comme suit :

$$R_C(I_c) = \frac{\text{trace}(M_C(I_c))}{\text{somme}(M_C(I_c))}, R_C(I_c) \in [0, 1] \quad (3.7)$$

On peut combiner les vérifications de *validité* et de *pertinence* d’un fragment en introduisant un seuil de contingence (Δ_C). Un fragment est admissible (valide et pertinent) s’il satisfait au seuil de contingence, c’est-à-dire si $R_C(I_c) \geq \Delta_C$.

Pour améliorer le rapport de contingence d’un fragment, il peut être nécessaire de modifier le seuil de correspondance (voir l’équation (3.4)). En effet, la matrice de contingence

dépend directement de ce dernier. Le meilleur seuil de correspondance est donc celui qui maximise le rapport de contingence d'un fragment.

Le dernier critère concerne la *taille* d'un fragment. Ce critère est nécessaire pour éviter d'obtenir des fragments si petits qu'ils ne contiennent pas d'information suffisante ou si grands qu'ils sont difficiles à retrouver dans une image de dimensions plus petites. Il est cependant possible de satisfaire implicitement à ce critère lorsque le fragment est initialement cadré depuis une image d'apprentissage. Le fait de remplir cette condition en amont, en particulier avant le calcul de *validité* et de *pertinence* du fragment, permet d'accélérer le procédé de sélection des fragments.

Un fragment qui satisfait à ces quatre critères est ainsi ajouté au modèle de la classe d'objets.

3.2.2 Segmentation utilisant les fragments

Comme il a été mentionné aux sous-sections 1.4 et 2.3, il est possible d'effectuer la segmentation d'images en tenant compte de la connaissance a priori. Cette connaissance consistant ici en un ensemble de fragments d'images, il importe de savoir comment tirer profit de la forme sous laquelle elle est utilisée.

L'objectif visé est de segmenter correctement un objet dans une image avec l'aide du modèle de la classe de cet objet. La première étape consiste à faire une segmentation initiale de l'image avec une méthode de segmentation connue. La méthode choisie doit satisfaire aux critères suivants :

1. Elle doit être applicable aux images couleurs.
2. Elle doit tenir compte de l'information spatiale.
3. Elle doit être extensible à un nombre quelconque de classes.

Il importe de préciser que la vitesse d'exécution du procédé n'est pas un critère pris en compte pour les besoins de ce mémoire. À la lumière de ces faits, nous retenons la méthode du *mean-shift* [11] pour accomplir la segmentation initiale. Le résultat de cette segmentation initiale consiste en un ensemble de segments. Cet ensemble est conforme aux conditions énoncées à la section 2.1. Nous posons que certains des segments de cet ensemble appartiennent à l'objet dans l'image, tandis que les autres appartiennent au fond. Les segments forment ainsi deux sous-ensembles disjoints : 1) *le sous-ensemble des segments d'objet* et 2) *le sous-ensemble des segments de fond*. Dans ce qui suit, l'appartenance des segments à un sous-ensemble ou à l'autre est déterminée grâce à l'information contenue dans le modèle.

Un sous-ensemble de segments donné équivaut à une certaine segmentation d'objets dans l'image. Le meilleur sous-ensemble de segments d'objet est celui dont l'union des segments résulte en la meilleure correspondance avec le modèle. Une façon naïve de procéder pour trouver de quels segments est composé ce sous-ensemble consiste à calculer la correspondance de chacune des combinaisons de segments avec le modèle dont on dispose. Il importe de mentionner que pour n segments définis dans la segmentation initiale de l'image, il existe 2^n combinaisons de ces segments (ou, pratiquement, $2^n - 2$, car nous pouvons d'emblée exclure les cas du sous-ensemble \emptyset qui ne contient aucun segment et celui du sous-ensemble qui contient tous les segments).

Posons $\mathcal{S} = \{s_1, \dots, s_m\}$ comme étant l'ensemble des segments obtenus lors de la segmentation initiale, I_c comme étant un fragment contenu dans le modèle de la classe c , IS comme étant une image binaire composée d'un sous-ensemble de \mathcal{S} (voir Figure 3.2) et δ_c comme étant un seuil de correspondance. Nous pouvons, de façon analogue à ce qui est fait à l'équation (3.4), définir une fonction C qui vérifie si la meilleure correspondance entre un fragment I_c et l'image IS satisfait au seuil δ_c :

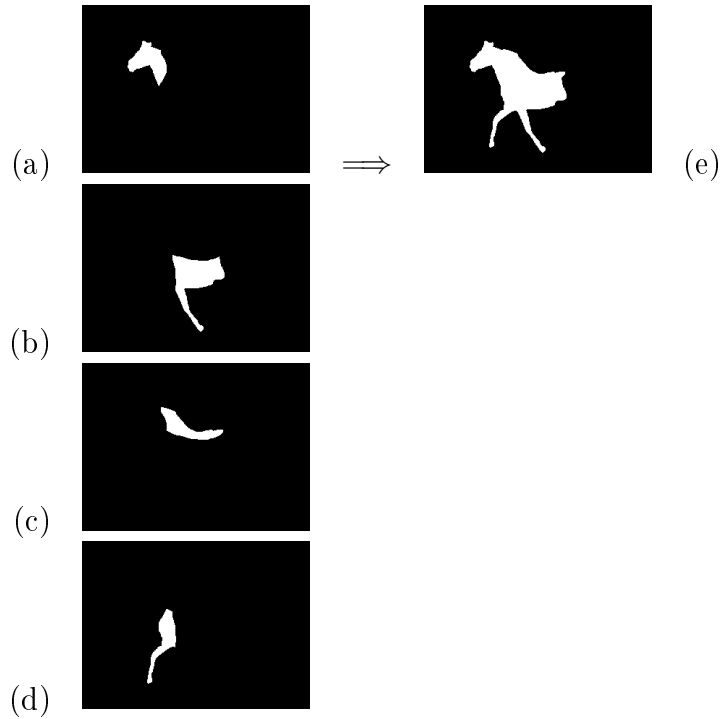


Figure 3.2 – Combinaison de segments

$\{a,b,c,d\}$: sous-ensemble de segments
 $e)$: image IS résultant de la combinaison des segments $\{a,b,c,d\}$

$$C(I_c(i), IS) \geq \delta_c \quad (3.8)$$

où i est l'indice d'un fragment I_c . Le principe décrit aux équations (3.5) et (3.6), utilisé pour déterminer la *validité* d'un fragment au regard d'un ensemble d'images, est utilisé dans les équations (3.9) et (3.10) pour déterminer, à l'inverse, si une image est valide au regard d'un ensemble de fragments. Posons un seuil de validité Δ_P . En se servant de la fonction S (voir l'équation (3.6)), la *validité* de l'image IS est déterminée comme suit :

$$\frac{\sum_i S(I_c(i), IS)}{|I_c|} \geq \Delta_P \quad (3.9)$$

$$\text{où } S(I_c(i), IS) = \begin{cases} 1 & \text{si } C(I_c(i), IS) \geq \delta_c \\ 0 & \text{sinon.} \end{cases} \quad (3.10)$$

et où i est l'indice d'un fragment I_c et $|I_c|$ représente la cardinalité de l'ensemble I_c . Il est possible que plus d'une combinaison de segments satisfasse au seuil Δ_P ; dans ce cas, la combinaison IS qui affiche la correspondance la plus élevée (telle que calculée à l'équation (3.9)) est retenue. Les segments que compte cette combinaison constituent le sous-ensemble des segments d'objet, tandis que les segments qu'elle ne compte pas constituent le sous-ensemble du fond.

La figure 3.3 illustre à l'aide d'organigrammes les processus d'extraction des fragments et de segmentation d'objets utilisant ces fragments décrits plus haut dans cette section.

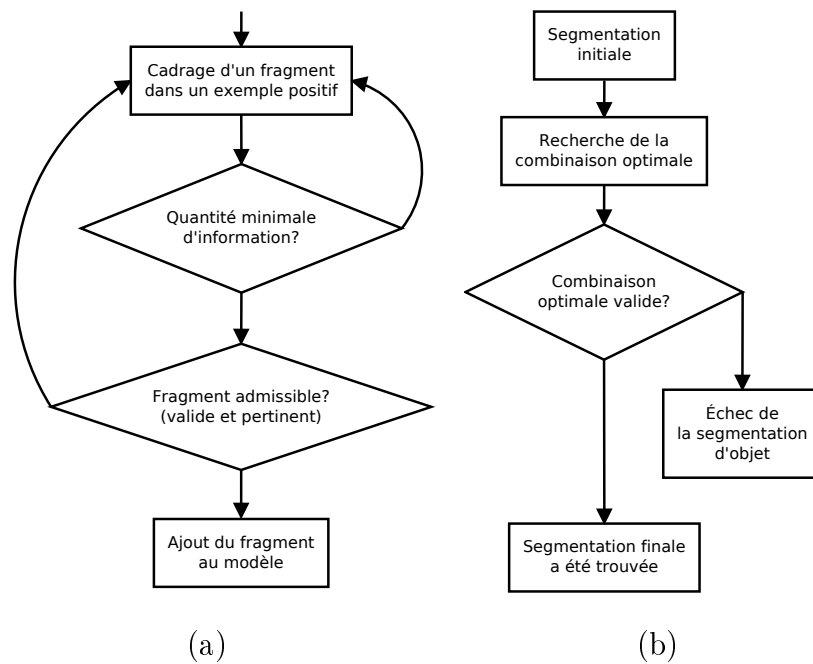


Figure 3.3 – Organigrammes
 a) : *extraction d'un fragment*
 b) : *segmentation utilisant les fragments*

3.3 Expérimentations

Notre expérimentation utilise des *exemples positifs* représentant la classe *cheval* (voir Figure 3.4) et des *exemples négatifs* représentant des objets qui ne font pas partie de la classe *cheval* (voir Figure 3.5). Nous extrayons les fragments selon les paramètres suivants : taille des fragments (en pixels) = 75×75 , seuil de nombre de pixels $\delta_p = 0,35$, seuil de correspondance $\delta_c = 0,85$ et seuil de contingence $\Delta_C = 0,85$. La valeur de chaque paramètre a été choisie parce qu'elle semblait produire les meilleurs résultats. Ce processus est effectué selon des seuils de correspondance et de contingence relativement bas (comparativement à un maximum de 1) dans le but d'accélérer l'extraction des fragments en tant que tels, et ce, sans compromettre la *validité* et la *pertinence* de ces derniers. La figure 3.6 montre des fragments extraits selon ces paramètres.

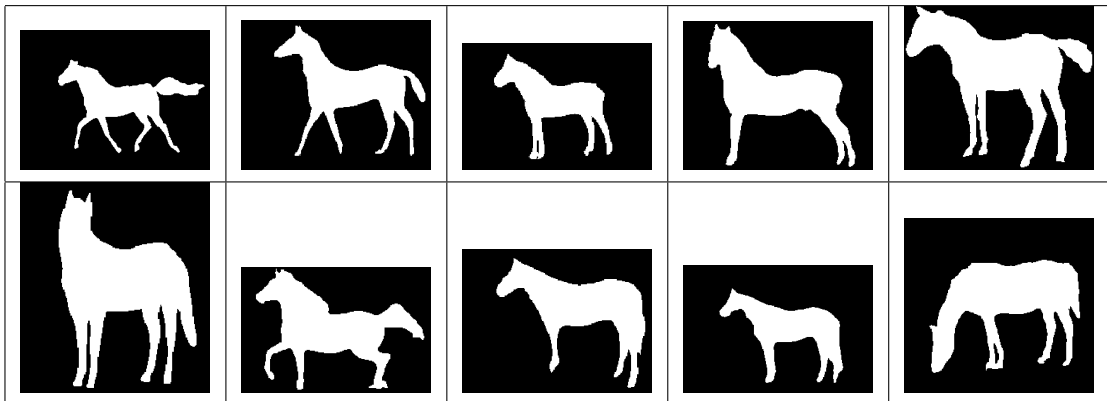


Figure 3.4 – *Exemples positifs* de la classe *cheval*

La figure 3.7 montre les résultats de notre expérimentation sur quelques images naturelles de la classe *cheval*. La première colonne montre l'image naturelle telle qu'utilisée. La deuxième colonne montre le résultat de la segmentation initiale ; dans notre expérimentation, celle-ci est réalisée grâce à la méthode du *mean-shift* [11]. La troisième colonne montre l'union de segments qui présente la meilleure correspondance avec le modèle de

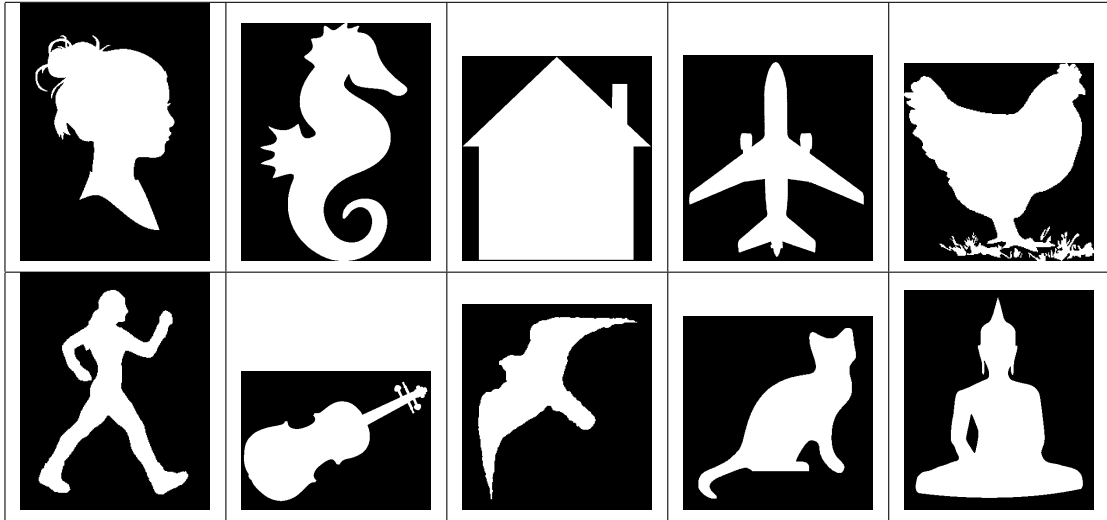


Figure 3.5 – *Exemples négatifs* de la classe *cheval*

la classe d'objets recherchée dans l'image. La quatrième colonne montre l'image naturelle d'origine à laquelle on a ajouté la segmentation d'objet obtenue selon notre méthode. Enfin, la cinquième et dernière colonne montre la vérité terrain (*ground truth*) de la segmentation de l'objet dans l'image, c'est-à-dire l'ensemble des pixels de l'image naturelle qui appartiennent véritablement à l'objet.

Il est possible de quantifier la qualité de la segmentation d'une image. Dans le cadre de notre expérimentation, nous utilisons la *précision* et le *rappel*. Posons VP (*vrais positifs*) comme étant le nombre de pixels d'une image correctement attribués à l'objet à segmenter, FP (*faux positifs*) comme étant le nombre de pixels de l'objet incorrectement attribués à l'objet à segmenter et FN (*faux négatifs*) comme étant le nombre de pixels incorrectement attribués au fond de l'image. La *précision* (équation (3.11)) et le *rappel* (équation (3.12)) s'énoncent formellement comme suit :

$$Précision = \frac{VP}{VP + FP} \quad (3.11)$$

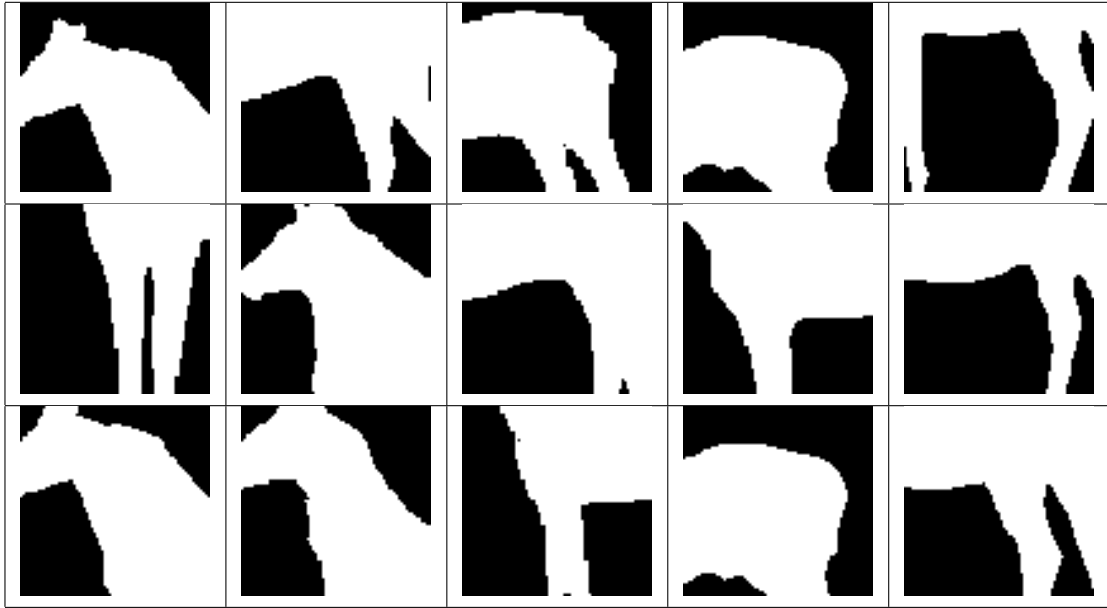


Figure 3.6 – Fragments extraits pour la classe *cheval*

$$Rappel = \frac{VP}{VP + FN} \quad (3.12)$$

Une *précision* élevée signifie qu'une grande proportion des pixels attribués à l'objet le sont correctement. Un *rappel* élevé signifie, quant à lui, qu'une grande proportion des pixels qui doivent être attribués à l'objet le sont effectivement.

Les calculs de la *précision* et du *rappel* ont été effectués pour chacune des segmentations corrigées que nous avons obtenues dans notre expérimentation. Le tableau 3.2 montre les moyennes et les écarts-types respectifs des *précisions* et *rappels* calculés pour ces segmentations. Les deux moyennes obtenues étant plutôt élevées, notre méthode corrige de manière satisfaisante la segmentation des images ; les deux écarts-types obtenus étant quant à eux bas, notre méthode donne des résultats assez uniformes. Dans notre expérimentation, la segmentation corrigée ne correspond pas précisément à la vérité terrain. Selon notre interprétation de ces résultats, ceci est principalement dû à l'une des

trois causes suivantes : la qualité de la segmentation initiale, la qualité du modèle et l'heuristique utilisée pour corriger la segmentation.

Tableau 3.2 – Moyennes et écarts-types des *précisions* et *rappels* calculés

	Précision	Rappel
Moyenne	0.8134	0.9047
Écart-type	0.0344	0.0703

La première cause concerne la segmentation initiale. Si cette dernière n'est pas adéquate (par exemple, si une partie de l'objet est fusionnée avec une partie du fond), sa correction ne le sera pas non plus. La qualité de la segmentation corrigée fournie par notre méthode est donc tributaire de la qualité de la segmentation initiale.

La deuxième cause concerne la qualité du modèle. Si ce dernier ne qualifie pas adéquatement la classe d'objets (par exemple, si les fragments qu'il contient ne couvrent pas la totalité du contour de l'objet à segmenter), il est possible que la correction de la segmentation en soit affectée négativement.

La troisième cause concerne l'heuristique utilisée pour la correction de la segmentation. Comme il a été mentionné à la sous-section 3.2.2, le nombre de segmentations possibles qui peuvent être obtenues à partir d'un nombre n de segments augmente de manière exponentielle par rapport à n . Par conséquent, et ce, pour éviter de devoir tester toutes ces possibilités et pour être en mesure d'obtenir des résultats dans un délai raisonnable, l'utilisation d'une heuristique est nécessaire. Dans le cas qui nous intéresse, l'heuristique utilisée consiste d'abord à trouver le segment individuel qui correspond le mieux à notre modèle, puis à successivement lui adjoindre des segments adjacents qui améliorent la correspondance au modèle. Cette heuristique peut faire en sorte d'exclure des segments qui font en fait partie de l'objet si ces derniers sont dans une configuration telle que leur adjonction n'améliore pas immédiatement la correspondance avec le modèle.

3.4 Discussion

Jusqu'à maintenant, nous avons supposé dans la définition de la problématique que la classe de l'objet est connue à l'avance par l'utilisateur. Ceci nous permet d'utiliser directement le modèle spécifique de l'objet à segmenter. Cependant, si la classe de l'objet est inconnue, le choix du modèle, dans notre méthode telle que décrite jusqu'ici, doit se faire par tâtonnement. La connaissance de la classe de l'objet à segmenter est d'autant plus importante, car comme on l'a supposé, la segmentation d'images et la reconnaissance d'objet vont de pair. Le prochain chapitre présente une façon de traiter les cas où la classe de l'objet à segmenter est inconnue.


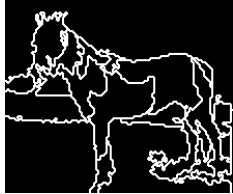


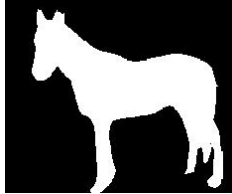


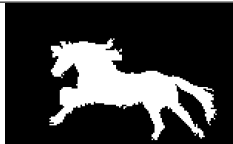

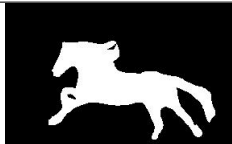

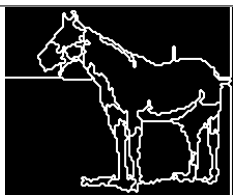

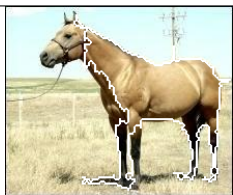
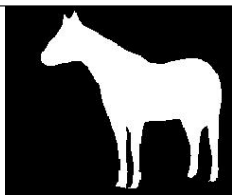

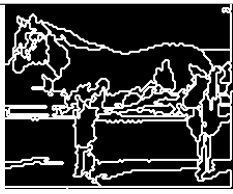
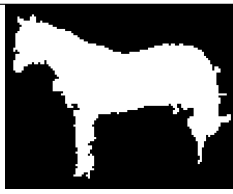

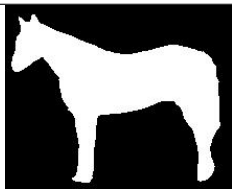




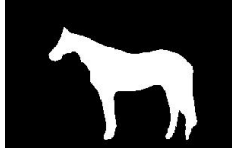
Image naturelle	Segmentation initiale	Segments optimaux	Segmentation corrigée	Vérité terrain
				
				
				
				
				

Figure 3.7 – Expérimentation sur cinq images naturelles

Chapitre 4

Contribution

4.1 Extension aux cas de multiples classes d'objets

La motivation quant à l'élaboration de notre solution s'explique principalement par la nécessité d'étendre la correction de la segmentation aux cas de multiples classes d'objets. Dans un contexte où plus d'une classe d'objets peut être reconnue, notre solution requiert la formation d'un modèle pour chacune des classes d'objets.

Suivant ce contexte, nous devons concevoir une façon de récupérer à la fois la classe de l'objet et d'effectuer sa segmentation de manière automatique. Un critère à prendre en considération est la nature variable des objets dans les images et de leur environnement. Cette variabilité fait en sorte que la correspondance calculée entre un objet et un modèle de classe ne peut jamais être parfaite ni totalement imparfaite ; elle se situe entre les deux. Puisque la reconnaissance d'un objet ne donne pas un résultat binaire pour une classe donnée mais plutôt un degré de confiance, il convient d'adopter une approche qui tient compte de cette réalité. Pour ce faire, nous proposons d'adopter une approche probabiliste.

Il est tout d'abord nécessaire, comme exigé dans la description de notre solution, de former un modèle de chacune des classes d'objets que nous envisageons pouvoir segmenter dans des images. Posons C comme étant le nombre de classes dont on possède les modèles, c comme étant la classe d'un objet quelconque (où $c \in \{1, \dots, C\}$), O_c comme étant un objet de la classe c , x comme étant le centre de gravité (la position) d'un objet dans une image, \mathcal{S} comme étant l'ensemble $\{s_1, \dots, s_m\}$ de tous les segments d'une segmentation d'image donnée et I_c comme étant un fragment du modèle qui décrit la classe c . La probabilité de retrouver un objet O appartenant à la classe c à une position x peut se calculer comme suit :

$$\begin{aligned}
P(O_c, x|\mathcal{S}) &= \sum_{I_c} P(O_c, x, I_c|\mathcal{S}) \\
&= \sum_{I_c} P(O_c|I_c, x)P(I_c, x|\mathcal{S}) \\
&= \sum_{I_c} P(O_c|I_c, x)P(I_c|x, \mathcal{S})P(x|\mathcal{S})
\end{aligned} \tag{4.1}$$

Les trois termes en lesquels se développe l'équation (4.1) s'expliquent mieux s'ils sont lus de droite à gauche. Le terme $P(x|\mathcal{S})$ (ci-après appelé P_1) représente la probabilité de retrouver un objet à la position x , quand on connaît sa segmentation. Le terme $P(I_c|x, \mathcal{S})$ (ci-après appelé P_2) représente la probabilité de retrouver un fragment du modèle de la classe c dans une image en connaissant la position de l'objet et sa segmentation. Le terme $P(O_c|I_c, x)$ (ci-après appelé P_3), quant à lui, représente la probabilité de retrouver un objet de la classe c dans une image, sachant qu'un fragment du modèle de la classe c s'y trouve et connaissant la position de l'objet dans l'image.

La première ligne du développement indique la marginalisation de la probabilité recherchée $P(O_c, x|\mathcal{S})$ sur l'ensemble des fragments I_c . Les développements de la première à la deuxième ligne et de la deuxième à la troisième ligne emploient une extension du

théorème de Bayes :

$$P(A \cap B|C) = P(A|B, C)P(B|C) \quad (4.2)$$

La probabilité $P(O_c, x|\mathcal{S})$ résulte donc de la somme des produits des probabilités P_1 , P_2 et P_3 obtenues pour chacun des fragments ; il s'agit de la probabilité de retrouver un objet d'une classe donnée à une position donnée dans l'image. Nous obtenons la classe de l'objet qui se trouve dans l'image ainsi que sa position grâce à la fonction suivante :

$$\operatorname{argmax}_{O_c, x}(P(O_c, x|\mathcal{S})) \quad (4.3)$$

La classe et la position de l'objet à segmenter dans l'image sont ainsi trouvées par la maximisation de la probabilité décrite à l'équation (4.1).

Nous justifions l'emploi de cette équation comme modèle par deux raisons : premièrement, elle prend en charge toute l'information a priori contenue dans les modèles de classe (cette information est décrite à la sous-section 4.4.1) ; deuxièmement, elle prend également en charge les différentes configurations spatiales selon lesquelles un objet peut être représenté dans une image. La configuration spatiale d'un objet peut, par exemple, concerner la position de celui-ci dans l'image ou encore l'agencement de ses différentes parties.

Ce chapitre traite des détails de l'implémentation d'une solution pour la présente problématique. La solution proposée est composée d'une série d'opérations qui respectent le cadre de l'approche probabiliste que nous avançons. Nous posons quelques hypothèses quant aux images auxquelles notre solution peut être appliquée, notamment que la perspective, l'échelle, l'orientation et la direction selon lesquelles elles sont représentées doivent être les mêmes que celles selon lesquelles le modèle de la classe correspondante a été formé.

4.2 Facteurs de correspondance fragment / image

La correspondance entre un fragment et un cadrage d'une image (ci-après appelé cadre) peut être calculée de plusieurs façons différentes. Cette section présente quelques facteurs de correspondance explorés dans le cadre de l'élaboration de notre solution. Il importe de mentionner que chaque facteur est normalisé pour se situer dans l'intervalle $[0, 1]$, où une valeur de 0 indique une correspondance nulle et une valeur de 1 indique une correspondance totale.

Étant donné une image et un fragment, il importe d'extraire de cette image un cadre qui coïncide fortement avec le fragment. Selon nos expérimentations, il appert que le cadre qui possède le plus grand *chevauchement* par rapport au fragment courant est celui qui est retenu (le calcul du *chevauchement* est décrit plus bas, à la sous-section 4.2.4). La figure 4.1 montre un exemple de fragment et le cadre correspondant qui a été extrait d'une image.

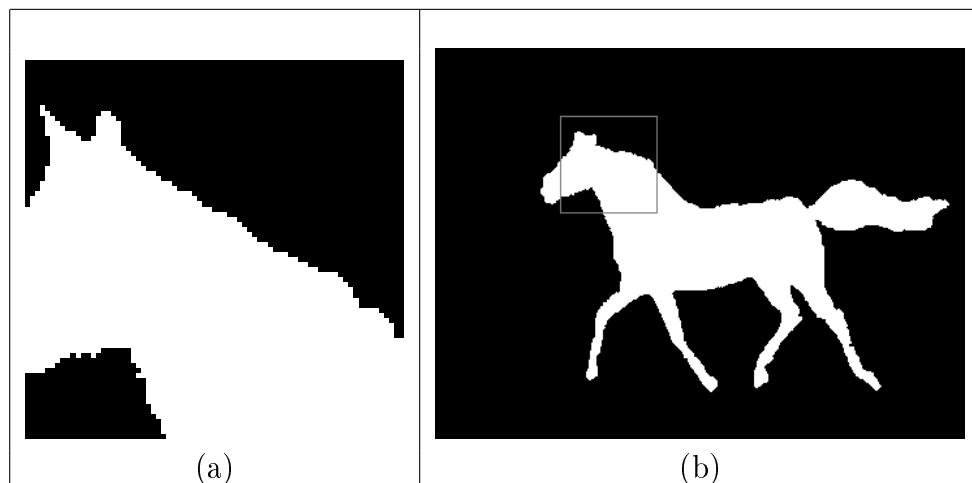


Figure 4.1 – Exemple d'extraction du meilleur cadre

- a) : *fragment*
- b) : *cadre (encadré gris)*

À noter qu'un cadre doit toujours posséder les mêmes dimensions que le fragment à partir duquel il a été extrait, puisque le calcul de leur correspondance se fait pixel à pixel.

Dans ce qui suit, nous entendons par *vrais positifs* (*VP*) les pixels d'objet d'un fragment correspondant aux pixels d'objet d'un cadre, par *faux positifs* (*FP*) les pixels d'objet d'un fragment correspondant aux pixels de fond d'un cadre et par *faux négatifs* (*FN*) les pixels de fond d'un fragment correspondant aux pixels d'objet d'un cadre.

4.2.1 Score algébrique

Nous définissons le *score algébrique* comme étant la proportion de pixels correspondants d'un fragment et d'un cadre qui ont la même valeur. Posons I_c comme étant un fragment de la classe c , C comme étant un cadre, i comme étant la position d'un pixel, et 0 et 1 comme étant les valeurs que prennent respectivement les pixels de fond et les pixels d'objet. Ce score s'énonce formellement ainsi :

$$score_{alg}(I_c, C) = 1 - \frac{\sum_i (I_c(i) - C_i)^2}{|I_c|}, \quad I_c(i) \in \{0, 1\} \text{ et } C_i \in \{0, 1\} \quad (4.4)$$

où $|I_c|$ représente le nombre de pixels dans un fragment I_c . Le *score algébrique* d'un fragment par rapport à un cadre est ainsi nul lorsqu'ils n'ont aucun pixel en commun et maximal lorsque tous leurs pixels correspondants ont chacun la même valeur.

4.2.2 Score géométrique

Le *score géométrique* est défini comme étant le produit des nombres de pixels correspondants ayant la même valeur dans un fragment et un cadre, divisé par le produit des nombres de pixels d'objet et de fond se trouvant dans le même cadre. Posons I_c comme

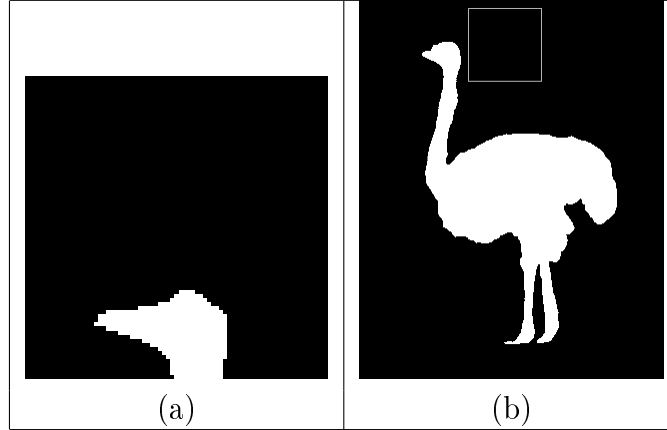
étant un fragment de la classe c , C comme étant un cadre, i comme étant l'indice d'un pixel du fragment ou du cadre, C_{fond} comme étant l'ensemble des pixels de fond du cadre C , C_{objet} comme étant l'ensemble des pixels d'objet du cadre C , et 0 et 1 comme étant les valeurs que prennent respectivement les pixels de fond et les pixels d'objet. Le *score géométrique* s'énonce formellement ainsi :

$$score_{geo}(I_c, C) = \frac{\sum_i (I_c(i)C_i) \cdot \sum_i (1 - I_c(i))(1 - C_i)}{|C_{fond}| \cdot |C_{objet}|}, \quad I_c(i) \in \{0, 1\} \text{ et } C_i \in \{0, 1\} \quad (4.5)$$

où $|C_0|$ et $|C_1|$ représentent respectivement le nombre de pixels de fond et le nombre de pixels d'objet du cadre. Le *score géométrique* d'un fragment par rapport à un cadre est nul lorsqu'aucun des pixels de fond (ou d'objet) correspondants n'a la même valeur et maximal lorsque le produit du nombre de pixels de fond correspondants et du nombre de pixels d'objet correspondants est maximal.

L'avantage principal que possède le *score géométrique* par rapport au *score algébrique* est que son calcul pénalise davantage les correspondances où l'on observe un petit nombre de pixels d'objet ou de pixels de fond. Pour illustrer cette propriété, la figure 4.2 montre un fragment dont un petit nombre de pixels sont des pixels d'objet, un cadre dont tous les pixels sont des pixels de fond et le résultat du calcul des scores *algébrique* et *géométrique* entre ce fragment et ce cadre.

Le *score géométrique* est donc utile pour détecter les cas où la correspondance entre les pixels d'objet (ou les pixels de fond) d'un fragment et d'un cadre est nulle ou très basse. Ces cas peuvent survenir pour deux raisons : soit le fragment contient proportionnellement peu de pixels d'objet ou de fond, soit le cadre est constitué totalement de pixels d'objet ou de fond. Dans notre solution, nous évitons ces cas en contrôlant tout d'abord la proportion des pixels d'objet et de fond (comme nous l'avons décrit à l'équation 3.1 du chapitre précédent), puis en excluant d'emblée un cadre constitué totalement de pixels



- a) : fragment de dimensions 75×75 où 412 pixels ($\approx 7\%$) sont des pixels d'objet
 b) : cadre (encadré gris) choisi dans une image

VP	VN	$ I_c $	C_{objet}	C_{fond}	$score_{alg}$	$score_{geo}$
0	5213	5625	0	5625	0.93	0

Figure 4.2 – Fragment, cadre et calcul de leurs scores *algébrique* et *géométrique*

d'objet ou de fond. Ces critères exclusifs que nous avons introduits en amont du calcul des différents facteurs de correspondance rendent superflu le calcul du *score géométrique*.

4.2.3 Distance de Hausdorff

La *distance de Hausdorff* est une mesure topologique de l'éloignement (la distance) entre deux polygones (voir Figure 4.3). Dans le cadre de notre solution, les deux polygones desquels on calcule la *distance de Hausdorff* sont constitués des points du contour d'un fragment et des points du contour d'un cadre.

Posons p_1 et p_2 comme étant deux polygones quelconques composés respectivement des points (a_1, a_2, \dots, a_k) et (b_1, b_2, \dots, b_l) . La *distance de Hausdorff* H entre le polygone ∂I_c , formé par les points du contour du fragment, et le polygone ∂C , formé par les points du contour du cadre, s'énonce formellement ainsi :

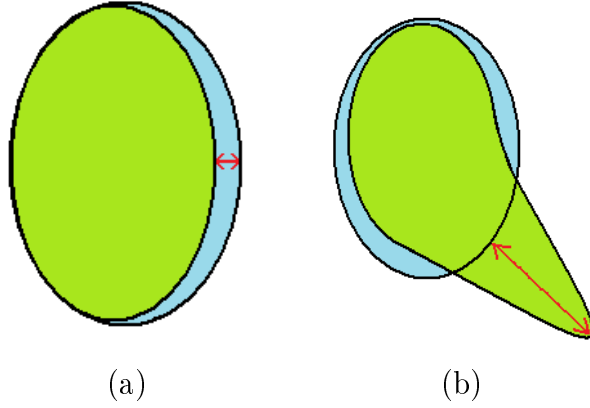


Figure 4.3 – *Distance de Hausdorff* entre les polygones p_1 (bleu) et p_2 (vert)
 a) : *courte distance de Hausdorff*
 b) : *longue distance de Hausdorff*

$$h(p_1, p_2) = \max_{a_i \in p_1} \{ \min_{b_j \in p_2} \{ \|a_i - b_j\| \} \} \quad (4.6)$$

$$H(\partial I_c, \partial C) = \max(h(\partial I_c, \partial C), h(\partial C, \partial I_c)) \quad (4.7)$$

Pour que le facteur de correspondance calculé à partir de la *distance de Hausdorff* respecte les contraintes énoncées au début de cette section, il est nécessaire de normaliser la valeur obtenue. Posons d comme étant la longueur de la diagonale qui traverse un fragment. La normalisation de la *distance de Hausdorff* pour obtenir un facteur de correspondance utilisable dans notre solution s'énonce formellement ainsi :

$$facteur_{Hausdorff}(\partial I_c, \partial C) = 1 - \frac{H(\partial I_c, \partial C)}{d} \quad (4.8)$$

Le facteur ainsi calculé est minimal lorsque la *distance de Hausdorff* est maximale, donc lorsqu'on observe une importante différence entre les contours du fragment et du cadre, et maximal lorsque la *distance de Hausdorff* est minimale, donc lorsque les contours du

fragment et du cadre sont fort semblables.

4.2.4 Chevauchement

Nous définissons le *chevauchement* en tant que facteur de correspondance comme étant une mesure de la proportion de pixels correspondants entre un fragment et un cadre qui n'ont pas la même valeur (voir Figure 4.2.4). Le *chevauchement* augmente proportionnellement au rapport entre le nombre de *vrais positifs* et la somme des nombres de *faux positifs* et de *faux négatifs*.

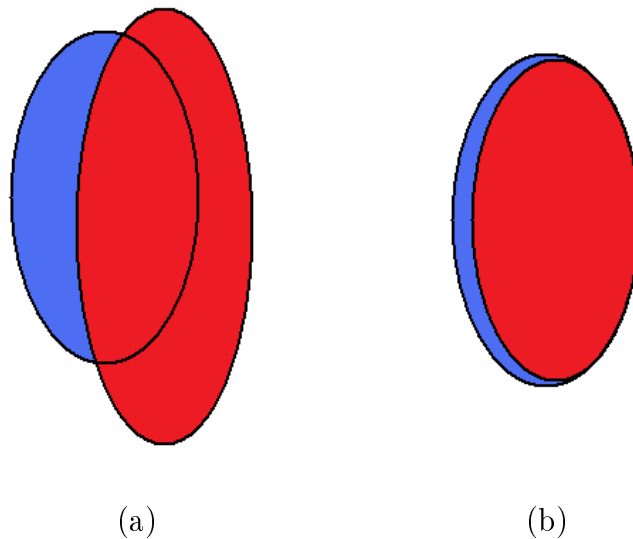


Figure 4.4 – Exemples de *chevauchements*

- a) : *chevauchement bas*
- b) : *chevauchement élevé*

Posons I_c comme étant un fragment, C comme étant un cadre, VP comme étant le nombre de *vrais positifs*, FP comme étant le nombre de *faux positifs* et FN comme étant le nombre de *faux négatifs*. Une fois normalisé aux fins de notre solution, nous énonçons formellement le *chevauchement* ainsi :

$$facteur_{chevauchement}(I_c, C) = 1 - \frac{FP + FN}{FP + FN + VP} \quad (4.9)$$

Ce facteur, tel que nous le calculons, est nul lorsqu'il n'y a aucun *vrai positif* ($VP = 0$) et maximal lorsqu'il n'y a ni *faux positifs* ni *faux négatifs* ($FP + FN = 0$).

4.2.5 Position d'un fragment

Pour accélérer la recherche de fragments dans une image, nous enregistrons dans le modèle d'une classe la position attendue de chacun de ses fragments par rapport au centre de l'objet ; cette position est la position de référence. Il est dès lors possible de calculer la distance qui sépare la position de référence d'un fragment de sa position observée dans une image.

La position d'un fragment peut constituer un facteur de correspondance entre un fragment et un cadre, en indiquant la proximité (ou l'éloignement) du second par rapport à sa position escomptée. Posons I_c comme étant un fragment, C comme étant un cadre, $p\vec{o}s_o$ comme étant la position observée du cadre dans l'image, $p\vec{o}s_r$ comme étant sa position de référence et N_{max} comme étant la norme de l'éloignement maximal d'un cadre par rapport à sa position attendue. Le facteur de correspondance, calculé à partir de la position du fragment et normalisé pour satisfaire aux exigences de notre solution, s'énonce formellement ainsi :

$$facteur_{pos}(I_c, C) = 1 - \frac{\|p\vec{o}s_o - p\vec{o}s_r\|}{N_{max}} \quad (4.10)$$

Ce facteur est minimal lorsque le cadre est éloigné de sa position attendue et maximal lorsqu'il en est très proche.

4.3 Facteurs de correspondance fragment / modèle

Une fois que la correspondance entre un certain fragment et une image est quantifiée, il reste à quantifier la correspondance entre ce même fragment et le modèle dont il est issu. Ceci est nécessaire pour respecter l'approche probabiliste proposée au début de ce chapitre. Tout comme les facteurs de correspondance décrits à la section précédente, ces facteurs de correspondance se situent dans l'intervalle $[0, 1]$, où une valeur de 0 indique une correspondance nulle et une valeur de 1 indique une correspondance totale. Les deux facteurs dont il est question dans cette section, la *validité* et la *pertinence*, sont calculés une seule fois par fragment par modèle lors de la formation de chacun de ceux-ci et sont ultérieurement consultés au besoin.

4.3.1 Validité

La *validité* est, telle que nous l'avons décrite à la sous-section 3.2.1, une mesure du caractère d'un fragment par rapport à la classe qu'il décrit. Une *validité* élevée signifie qu'un fragment d'une classe donnée se trouve avec une forte probabilité dans une image d'un objet de cette même classe. À l'opposé, une *validité* basse dénote qu'un fragment d'une classe donnée se trouve avec une faible probabilité dans une image d'un objet de la classe en question.

4.3.2 Pertinence

La *pertinence*, comme la *validité*, a été décrite à la sous-section 3.2.1. Il s'agit d'une mesure de la non-caractérisation d'un fragment d'une certaine classe par rapport aux objets qui n'appartiennent pas à cette classe. Ainsi, une *pertinence* élevée signifie qu'un fragment d'une classe donnée se trouve avec une faible probabilité dans une image d'un

objet d'une autre classe et une *pertinence* basse signifie qu'un fragment se trouve avec une plus forte probabilité dans une image d'un objet d'une autre classe.

4.4 Méthodologie

Nous avons montré à la section 3.3 qu'une méthode heuristique simple permet, pour une image naturelle donnée et lorsque la classe de l'objet qu'elle contient est connue, de distinguer les segments d'objet des segments de fond. Notre solution permet quant à elle d'effectuer essentiellement la même opération, mais pour un nombre quelconque de classes. En d'autres mots, notre solution peut distinguer les segments qui constituent un objet dans une image de ceux qui ne le constituent pas, en plus d'identifier à quelle classe appartient l'objet décrit par ces mêmes segments. Ces deux opérations sont, en quelque sorte, réalisées en parallèle dans notre solution. Dans un premier temps, et ce, pour chaque classe dont on possède le modèle, nous calculons la probabilité qu'un objet de cette classe se trouve dans l'image. Un sous-produit du calcul de cette probabilité consiste en l'ensemble des segments qui correspondent le mieux au modèle de classe utilisé. Enfin, la classe associée à la probabilité obtenue qui est la plus élevée est retenue et de même pour l'ensemble des segments en résultant.

Notre solution consiste en quatre étapes distinctes :

1. Formation des modèles des classes susceptibles d'être utilisés.
2. Segmentation initiale de l'image.
3. Calcul de la probabilité que chacune des classes d'objets se retrouve dans l'image.
4. Correction de la segmentation.

La présente section détaille chacune de ces étapes.

4.4.1 Formation des modèles de classes

Chaque classe d'objets doit être décrite par un modèle. La formation du modèle d'une classe, tel qu'utilisé dans notre solution, se rapproche de ce qui est décrit à la section 3.2 du chapitre précédent. Le modèle d'une classe constitue la totalité de l'information relative à cette classe que nous utilisons dans notre solution. Un modèle consiste en un ensemble de fragments (extraits comme nous l'avons énoncé à la sous-section 3.2.1) auxquels nous avons adjoint quatre paramètres. Ces paramètres sont la position, l'écart-type de la position, la *validité* et la *pertinence* de chaque fragment.

Position

Puisque les objets acceptés dans notre solution ne sont pas rigides, nous enregistrons la position moyenne où se trouvent les fragments dans les *exemples positifs* utilisés (la position est calculée relativement au centre de l'objet se trouvant dans un *exemple positif*). Cette information accélère grandement le processus de correction de la segmentation, car la recherche de fragments ne s'effectue qu'aux endroits où nous savons qu'ils se trouvent. Posons I_c comme étant un fragment, E_P comme étant un ensemble d'*exemples positifs*, i comme étant l'indice d'un *exemple positif* dans E_P et $p\vec{o}s(I_c, E_P(i))$ comme étant la position d'un fragment I_c dans un *exemple positif* $E_P(i)$. La position moyenne $p\vec{o}s_{I_c}$ enregistrée pour un fragment I_c est définie ainsi :

$$p\vec{o}s_{I_c} = \frac{\sum_i p\vec{o}s(I_c, E_P(i))}{|E_P|} \quad (4.11)$$

où $|E_P|$ représente la cardinalité de l'ensemble des *exemples positifs*.

Écart-type de la position

Nous enregistrons l'écart-type de la position pour guider plus précisément le processus de correction de la segmentation. L'écart-type est ici une mesure de l'éparpillement de la position d'un fragment où il est trouvé dans les *exemples positifs*. Posons I_c comme étant un fragment, E_P comme étant un ensemble d'*exemples positifs*, i comme étant l'indice d'un *exemple positif* dans E_P , $p\vec{o}s(I_c, E_P(i))$ comme étant la position d'un fragment I_c dans un *exemple positif* $E_P(i)$ et $p\vec{o}s_{I_c}$ comme étant la position moyenne enregistrée pour un fragment I_c . L'écart-type σ_{I_c} de la position est défini ainsi :

$$\sigma_{I_c} = \sqrt{\frac{1}{|E_P| - 1} \cdot \sum_i \|p\vec{o}s(I_c, E_P(i)) - p\vec{o}s_{I_c}\|^2} \quad (4.12)$$

où $|E_P|$ représente la cardinalité de l'ensemble des *exemples positifs*.

Validité

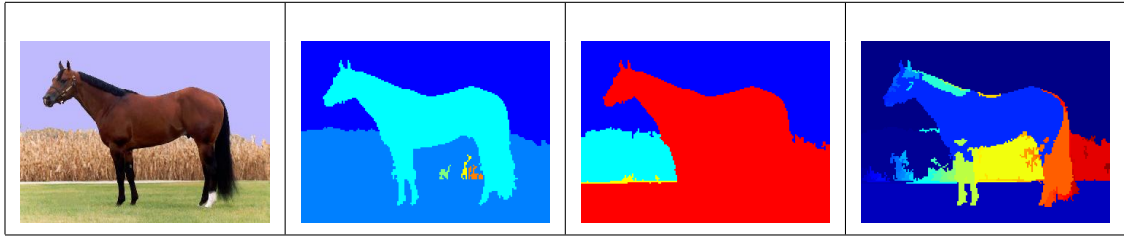
Voir la sous-section 4.3.1 pour la description de ce paramètre.

Pertinence

Voir la sous-section 4.3.2 pour la description de ce paramètre.

4.4.2 Détection d'objets

Dans notre solution, nous effectuons la segmentation initiale d'une image grâce à la méthode du *mean-shift*. Le résultat de cette segmentation initiale consiste en un ensemble de segments. Nous supposons que ces segments appartiennent à l'une de deux catégories : des segments d'objet ou des segments de fond.



- a) : *image naturelle à segmenter*
- b) : *segmentation de bonne qualité*
- c) : *segmentation de mauvaise qualité (sous-segmentation)*
- d) : *segmentation de mauvaise qualité (sur-segmentation)*

Figure 4.5 – Exemples de segments significatifs et non significatifs d’une image naturelle

Il est important que la segmentation initiale de l’image fournisse des segments significatifs : nous entendons par segments significatifs des segments qui ne résultent ni d’une *sous-segmentation* ni d’une *sur-segmentation*. On dit qu’il y a occurrence de *sous-segmentation* lorsqu’un ou plusieurs segments chevauchent à la fois des zones d’objet et de fond, ou plus simplement, lorsque la forme de leur contour n’est pas caractéristique de la forme que décrit l’objet (voir Figure 4.5.c). Il y a occurrence de *sur-segmentation* lorsqu’un trop grand nombre de segments est obtenu, tel que chaque segment compte pour une très petite partie de l’objet, ou du fond, de l’image (voir Figure 4.5.d). Dans le cadre de notre solution, la *sur-segmentation* pose deux problèmes distincts : d’abord, chaque segment contient, proportionnellement à son propre contour, très peu (ou pas) d’information sur le contour de l’objet qu’il doit caractériser ; enfin, le temps d’exécution de notre solution croît proportionnellement au nombre de segments qui sont utilisés. De façon générale, une segmentation où le contour de l’objet est suffisamment reconnaissable par un être humain est une segmentation de bonne qualité.

4.4.3 Calcul de la probabilité de chaque classe

Le calcul de la probabilité d'observer un objet de la classe c est effectué conformément à l'approche probabiliste décrite à la section 4.1. Cette probabilité est obtenue en additionnant le produit des termes P_1 , P_2 et P_3 obtenus pour chacun des fragments de la classe c . Nous amenons pour la première fois ces termes à l'équation (4.1). Il importe donc, pour expliquer en détails le fonctionnement de notre solution, de préciser comment ces termes sont calculés.

Le terme P_1 , défini comme étant la probabilité de retrouver un objet à une certaine position dans l'image quand on connaît sa segmentation, est considéré comme constant dans notre solution, et ce, dans le but d'accélérer et de simplifier le processus de correction de la segmentation. Nous caractérisons cette invariance en supposant que l'objet dont on souhaite corriger la segmentation se situe au centre de l'attention (centre de l'image); autrement dit, nous ne recherchons l'objet qu'en un endroit (appelé point de recherche) dans l'image. Il est cependant possible d'adapter notre solution pour admettre davantage de points de recherche.

De façon plus générale, et ce, sans égard pour notre implémentation du modèle probabiliste, le terme P_1 peut être calculé pour une position autre que celle supposée a priori. Posons $p\vec{s}_{hyp}$ comme étant la position avancée par hypothèse et $p\vec{s}_{priori}$ comme étant la position supposée a priori. Le calcul du terme P_1 s'énonce formellement ainsi :

$$P_1 = \frac{1}{1 + (\|p\vec{s}_{hyp} - p\vec{s}_{priori}\|)^2} \quad (4.13)$$

Le terme P_2 est quant à lui défini comme étant la probabilité de retrouver un fragment du modèle de la classe c dans une image en connaissant la position de l'objet et sa segmentation. Ce terme dépend exclusivement de la correspondance d'un fragment avec

une image ; par conséquent, nous le construisons à partir de facteurs de correspondance fragment / image. Selon nos expérimentations, les facteurs qu'il convient d'utiliser pour construire le terme P_2 sont le *score algébrique*, la *distance de Hausdorff* ainsi que le *chevauchement*. Chacun de ces facteurs contribue différemment à quantifier la correspondance fragment / image. Le *score algébrique* renseigne sur la proportion de pixels correspondants qui ont la même valeur ; il tend à pénaliser les couples fragment / image où une grande proportion de pixels d'objet ou de fond ne se correspondent pas (voir Figure 4.6.a). La *distance de Hausdorff* renseigne sur la correspondance des contours respectifs du fragment et de l'image ; ce facteur pénalise les couples fragment / image où, par exemple, les contours diffèrent de façon notable (voir Figure 4.6.b). Le *chevauchement*, de façon complémentaire au *score algébrique*, renseigne, quant à lui, sur le degré d'ajustement entre les pixels d'objet du fragment et de l'image ; le facteur qu'on lui associe pénalise les couples fragment / image dont les pixels d'objet respectifs ne correspondent pas (voir Figure 4.6.c). Posons I_c comme étant un fragment de la classe c et C comme étant le cadre correspondant à ce fragment. Nous définissons formellement le terme P_2 ainsi :

$$P_2 = score_{alg}(I_c, C) \cdot facteur_{Hausdorff}(I_c, C) \cdot facteur_{chevauchement}(I_c, C) \quad (4.14)$$

Enfin, le terme P_3 est défini comme étant la probabilité de retrouver un objet de la classe c dans une image, sachant qu'un fragment du modèle de la classe c s'y trouve et connaissant la position de l'objet dans l'image. Il s'agit donc, en quelque sorte, de la qualité relative du fragment courant par rapport au modèle de classe utilisé. Dans notre solution, ce caractère est mesuré grâce à la *validité* et à la *pertinence* ; un fragment à fois fortement valide et fortement pertinent caractérise par conséquent fortement le modèle auquel il appartient. Posons I_c comme étant un fragment de la classe c , le terme P_3 est défini formellement ainsi :

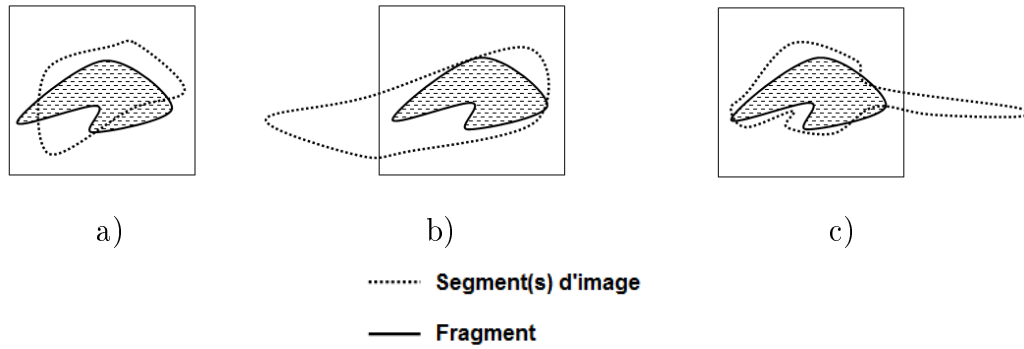


Figure 4.6 – Cas de correspondance fragment / image pénalisés

- a) : *pixels correspondants*
- b) : *pixels d'objet correspondants*
- c) : *contours*

$$P_3 = \text{validite}(I_c) \cdot \text{pertinence}(I_c) \quad (4.15)$$

À noter que dans notre solution, le calcul de la *validité* et la *pertinence* d'un certain fragment n'a à être effectué qu'une seule fois, c'est-à-dire au moment où le modèle contenant ce fragment est formé. Les valeurs de *validité* et de *pertinence* sont ainsi consultées à même le modèle plutôt que d'être recalculées chaque fois qu'elles sont requises.

Une fois les termes P_1 , P_2 et P_3 calculés pour chacun des fragments d'une classe donnée, il s'ensuit, de par l'équation (4.1), que nous avons également calculé la probabilité de retrouver dans une image un objet de cette même classe. En effectuant le même processus pour chaque classe dont on détient un modèle et en comparant les différentes probabilités ainsi calculées, nous obtenons la classe ainsi que la position de l'objet qui se trouve dans l'image.

4.4.4 Correction de la segmentation

La dernière étape consiste à déterminer quels segments, parmi ceux produits par la segmentation initiale, font partie de l'objet. Dans notre solution, nous réalisons cette opération en compilant, pour chaque fragment observé dans une image, les segments qui ont contribué à former l'objet. Nous obtenons ainsi, pour chaque segment, la fréquence avec laquelle il est utilisé pour former un objet. Un segment qui possède une grande fréquence est donc plus susceptible de faire partie de l'objet qu'un segment qui possède une fréquence moindre. Posons $freq_{s_j}$ comme étant la fréquence d'un segment s_j produit par la segmentation initiale ; pour qu'un segment soit considéré comme segment d'objet, sa fréquence doit être non nulle, tel que $freq_{s_j} > 0$. L'union des segments qui satisfont à ce critère forme la segmentation corrigée de l'image.

4.4.5 Algorithmes

Cette sous-section présente des algorithmes, sous forme de pseudocodes, qui décrivent le fonctionnement de notre méthode. L'algorithme 4.1 décrit l'extraction des fragments qui forment les modèles de classe, tandis que l'algorithme 4.2 décrit la segmentation de l'objet, permettant l'identification de la classe de l'objet et la détermination de sa position dans l'image.

4.5 Expérimentations

Dans nos expérimentations, nous extrayons d'abord un modèle pour chacune des classes *autruche*, *cheval* et *cygne*. Ces modèles comptent chacun vingt fragments et sont extraits avec comme paramètres $\delta_p = 0,15$, $\delta_c = 0,75$ et $\Delta_C = 0,9$; les valeurs retenues pour

```

pour (chaque classe d'objet  $c \in \mathcal{C}$ ) :
   $i \leftarrow 0$ ;
  tant que (nombre de fragments de classe  $i < N_c$ ) :
    1. Cadrer un fragment quelconque à partir d'un exemple positif ;
    2. Tester le fragment par rapport aux équations (3.1), (3.5) et (3.7);
    si (le fragment satisfait à tous les critères) alors
      Ajouter le fragment au modèle de la classe  $c$ ;
      Enregistrer la position relative du fragment ;
      Enregistrer la validité et la pertinence du fragment ;
       $i \leftarrow i + 1$ ;
    fin si
  fin tant que
fin pour

```

Algorithme 4.1: Algorithme pour l'extraction d'un modèle de classe

```

Entrée: Ensemble de segments  $\mathcal{S} = \{s_1, \dots, s_M\}$ 
pour (chaque classe  $c \in \mathcal{C}$ ) :
  pour (chaque position  $\mathbf{x}$  dans l'image) :
     $\tilde{s}_{c,\mathbf{x}} \leftarrow \emptyset$ ;
    pour (chaque fragment  $I_c$ ) :
      1. Trouver le meilleur cadre ;
      2. Ajouter les segments du cadre à  $\tilde{s}_{c,\mathbf{x}}$  ;
      3. Calculer le score de position  $P(\mathbf{x}|\mathcal{S})$  ;
      4. Calculer la correspondance fragment / image  $P(I_c|\mathbf{x}, \mathcal{S})$  ;
      5. Calculer la correspondance fragment / modèle  $P(c|I_c, \mathbf{x})$  ;
    fin pour
    Calculer le score final grâce à l'équation (4.1);
  fin pour
fin pour
retourner valeurs  $(\tilde{c}, \tilde{\mathbf{x}})$  qui maximisent le score final
retourner segments  $\tilde{s}_{\tilde{c}, \tilde{\mathbf{x}}}$ 

```

Algorithme 4.2: Algorithme pour la segmentation de l'image

ces paramètres ont été choisies expérimentalement et donnent de très bons résultats. Les ensembles d'*exemples positifs* et d'*exemples négatifs* utilisés comptent tous deux vingt images.

Nous avons d'abord validé notre méthode en la testant sur les images à niveaux de gris des *exemples positifs* (voir Figure 4.7), dans le but de simuler des segmentations figure-fond idéales. Cette validation a montré que la classe de 100% des *exemples positifs* a été correctement identifiée. Nous avons ensuite utilisé notre méthode sur diverses images naturelles contenant une autruche, un cheval ou un cygne. En ce qui a trait au point de recherche pour chaque segmentation (l'endroit où le centre de l'objet doit en principe se trouver), nous avons supposé qu'il se situait au centre de l'image.

Les figures 4.8, 4.9 et 4.10 décrivent quelques-uns de nos résultats; elles montrent les images naturelles utilisées, les segmentations *mean-shift* résultantes, les corrections de segmentation obtenues par notre méthode et les vérités terrain pour chacune de ces trois classes d'objets. Enfin, dans le but de quantifier la qualité de nos résultats, les figures montrent également le *rappel* et la *précision* obtenus en comparant les corrections de segmentation et vérités terrain correspondantes de chacune des images naturelles à laquelle nous avons appliqué notre méthode.

4.6 Interprétation des résultats

À la lumière de nos expérimentations, nous pouvons interpréter les résultats obtenus. Une première interprétation concerne la *précision* et le *rappel* obtenus pour chaque segmentation corrigée. Dans presque tous les cas, la *précision* est plus élevée que le *rappel*; ceci signifie que notre méthode exclut mieux les mauvais segments d'une segmentation qu'elle n'en identifie les bons. Une des raisons qui peuvent expliquer cette particularité

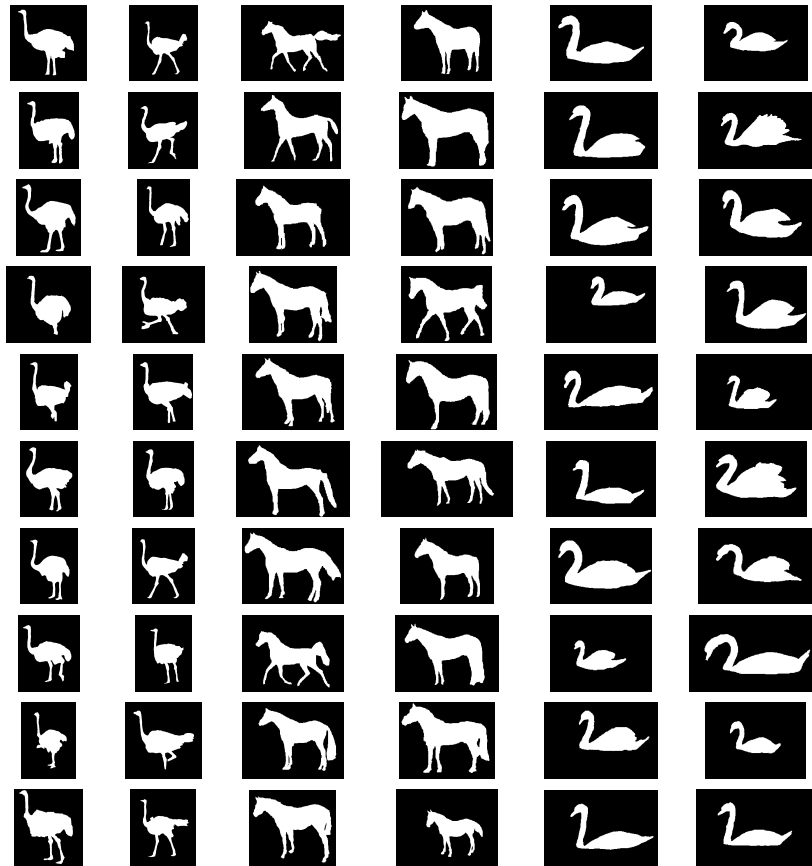


Figure 4.7 – *Exemples positifs* utilisés dans nos expérimentations


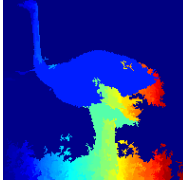



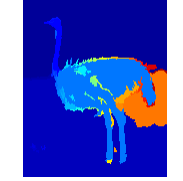
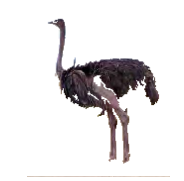


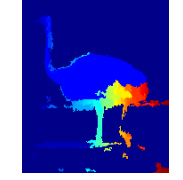


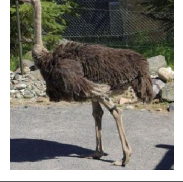
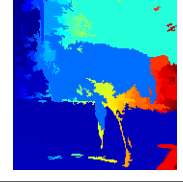

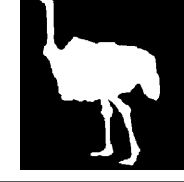
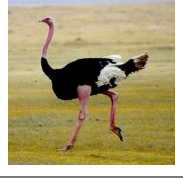
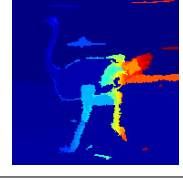

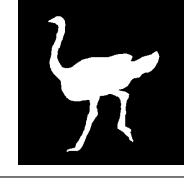
Image naturelle	Segmentation <i>mean-shift</i> de l'image	Correction de la segmentation	Vérité terrain	<i>Précision</i>	<i>Rappel</i>
				84.08%	60.11%
				97.82%	78.81%
				98.75%	63.81%
				87.90%	77.38%
				91.20%	43.67%

Figure 4.8 – Résultats de nos expérimentations pour la classe *autruche*


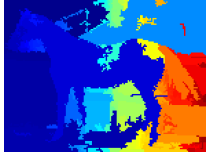

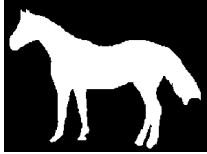

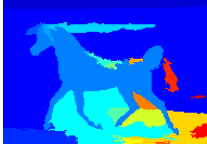








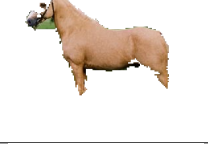


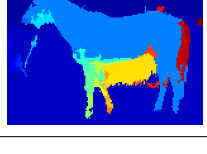
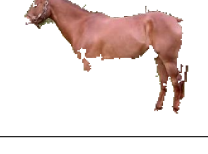

Image naturelle	Segmentation <i>mean-shift</i> de l'image	Correction de la segmentation	Vérité terrain	<i>Précision</i>	<i>Rappel</i>
				92.32%	72.27%
				96.05%	85.63%
				97.45%	97.03%
				95.51%	68.75%
				96.23%	77.19%

Figure 4.9 – Résultats de nos expérimentations pour la classe *cheval*


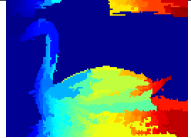



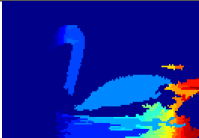



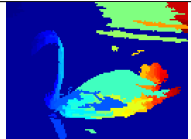


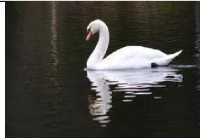
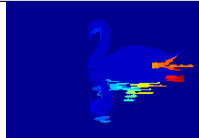
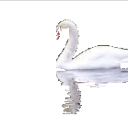


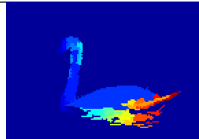


Image naturelle	Segmentation <i>mean-shift</i> de l'image	Correction de la segmentation	Vérité terrain	<i>Précision</i>	<i>Rappel</i>
				96.49%	62.64%
				96.45%	55.38%
				98.12%	37.05%
				55.87%	83.48%
				97.73%	55.90%

Figure 4.10 – Résultats de nos expérimentations pour la classe *cygne*

de notre méthode réside dans le processus de détermination des bons segments décrit à la sous-section 4.4.4. Atténuer le critère d'inclusion des bons segments serait une façon d'augmenter le *rappel* des segmentations corrigées.

Une deuxième interprétation de nos résultats concerne l'absence de caractère lisse des segmentations corrigées. Ceci est dû au résultat de la segmentation initiale des images naturelles, qui est, dans notre méthode, réalisée à l'aide du *mean-shift*. Pour obtenir une segmentation corrigée au caractère plus lisse, il conviendrait, par exemple, d'ajouter un filtre morphologique au processus qui fournit la segmentation initiale.

Enfin, nous interprétons l'absence fréquente de membres inférieurs, de cous et de têtes dans les segmentations corrigées des images naturelles d'animaux par le fait qu'il s'agit de parties aux configurations très variables. Cette variabilité fait en sorte que ces parties sont moins susceptibles d'être identifiées correctement dans leur modèle correspondant. Une solution à ce problème consiste à admettre dans le modèle un plus grand nombre de fragments qui décrivent les parties d'objets qui ont des configurations variables. Il conviendrait aussi de permettre des degrés de liberté pour certains fragments (par exemple, avec l'ajout d'une rotation).

4.7 Complexité algorithmique

La complexité algorithmique de notre méthode est calculée séparément pour les opérations 1) d'*extraction de la connaissance a priori* et de 2) *correction de la segmentation*. Il est d'autant plus important de considérer séparément ces calculs, car l'*extraction de la connaissance a priori* est une opération qui n'a à être effectuée qu'une seule fois avant de pouvoir effectuer la *correction de la segmentation* sur un nombre illimité d'images naturelles. Pour cette raison, il ne conviendrait pas de combiner les complexités algorithmiques

de ces deux opérations en un seul calcul.

4.7.1 Extraction de la connaissance a priori

La complexité algorithmique de l'extraction de la connaissance a priori dépend des facteurs suivants : le nombre de classes pour lesquelles on souhaite extraire un modèle (n_c), le nombre de fragments désiré par modèle de classe (n_f) ainsi que le nombre d'*exemples positifs* (n_{EP}) que l'on détient pour chaque classe. On peut exprimer la complexité algorithmique de l'extraction de la connaissance a priori comme suit : $\theta(n_c n_f n_{EP})$.

4.7.2 Correction de la segmentation

La complexité algorithmique de la correction de la segmentation dépend quant à elle des facteurs suivants : le nombre de modèles de classe (n_c), le nombre de fragments par modèle de classe (n_f), le nombre de points de recherche (n_r) et le nombre de segments (n_s). La borne supérieure de la correction de la segmentation est décrite par l'expression $\mathcal{O}(n_c n_f n_r 2^{n_s})$.

4.8 Discussion

Le succès de notre méthode appliquée à une image naturelle donnée dépend de quelques facteurs cruciaux. Tout d'abord, la segmentation *mean-shift* de l'image doit être de bonne qualité (voir Figure 4.5.b), c'est-à-dire ne dénoter ni *sous-segmentation* ni *sur-segmentation* (se référer à la sous-section 4.4.2 pour des précisions sur ces deux phénomènes). La qualité d'une segmentation est cependant difficile à quantifier.

Il importe de préciser que la segmentation *mean-shift* d'une image de mauvaise qualité est

davantage susceptible d'être elle-même de mauvaise qualité, comme l'ont indiqué Crowley et al. [12]. Par image de mauvaise qualité, nous entendons : soit une image où les contours sont diffus ou mal définis (comme dans le cas d'une image à laquelle on aurait fait subir une compression avec pertes) ; soit une image où la variance des pixels d'une même région homogène est élevée ; soit une image où l'éclairage est inégal au point de faire se confondre des parties de l'objet avec le fond. Le critère le plus déterminant demeure toutefois la netteté des contours présents dans l'image ; ces derniers doivent avoir un caractère suffisamment lisse et régulier pour que la forme de l'objet soit conservée au terme de la segmentation. Si, au contraire, les contours sont insuffisamment nets, la forme même des segments qui composent l'objet dans l'image est compromise. Puisque notre méthode se base exclusivement sur la silhouette d'un objet dans une image pour en identifier la classe, il va sans dire qu'une silhouette d'objet mal définie nuit à l'identification de cet objet.

Un second facteur à considérer concerne la variabilité de la forme des objets. Un objet (par exemple, un cheval) typiquement composé de différentes parties pouvant être positionnées et orientées de multiples façons (la tête, la queue, les jambes) peut se présenter selon une multitude de configurations (voir Figure 4.11). Étant donné que le modèle d'une classe ne peut contenir qu'un nombre fini de fragments et que, par conséquent, seules certaines configurations sont contenues dans ces fragments, il est possible qu'une image contienne un objet qui, dû à la configuration de ses différentes parties, puisse difficilement être distingué à l'aide du modèle de sa classe. Ainsi, notre méthode est plus habile à segmenter des objets dont la forme est commune, mais l'est moins lorsque les objets à segmenter ont une forme fortement inhabituelle.

Enfin, notre méthode a de la difficulté à différencier des classes d'objets qui présentent des silhouettes similaires. En effet, puisque les modèles de classes dans notre méthode se basent exclusivement sur la forme du contour des objets, il est conséquemment im-



Figure 4.11 – Exemples de formes variées chez des objets de la classe *cheval*

portant que ces modèles présentent des différences dans l'information que contiennent leurs fragments. En l'absence de telles différences dans leurs contours, davantage d'information à propos de ces classes (couleur, texture, etc.) est nécessaire pour les distinguer. Voici des exemples de classes qui, à cause de leurs formes similaires, seraient difficilement différenciées par notre méthode :

- autruche, émeu, casoar ;
- cheval, zèbre, âne ;
- cygne, canard, oie ;
- femme, homme, enfant ;
- saumon, truite, corégone ;
- souris, rat, dègue.

4.9 Extensions de ce travail

Notre solution, telle que décrite dans ce travail, peut bénéficier de différentes extensions. Quelques-unes de ces extensions visent à rendre notre solution applicable aux cas où l'objet subit une transformation géométrique (par rapport au modèle de classe correspondant) de sorte que la correction de la segmentation est a priori impossible. En effet, notre solution, bien qu'applicable à toute image dont on possède un modèle convenable, ne peut réussir que si l'objet à segmenter respecte certains aspects du modèle qui lui

correspond. Puisque le modèle est formé à partir d'images d'apprentissage, les images auxquelles on souhaite appliquer notre solution doivent s'y conformer. Les aspects du modèle que doivent respecter les nouvelles images sont :

- l'échelle utilisée ;
- la perspective utilisée ;
- l'orientation utilisée ;
- la direction utilisée.

Notre solution peut donc être étendue en permettant la segmentation d'objets indépendamment de leur échelle ou encore de la perspective, de l'orientation ou de la direction selon laquelle ils sont représentés. Dans ce qui suit, nous proposons quelques façons de remédier aux cas où l'objet à segmenter dans l'image correspond au modèle modulo une de ces transformations géométriques.

4.9.1 Échelle

En ce qui concerne l'échelle de l'objet à segmenter, notre solution peut être adaptée pour couvrir une gamme d'échelles. En définissant au préalable un certain nombre d'échelles selon lesquelles peut se présenter un objet, il suffit d'appliquer notre solution à l'image à segmenter transformée selon ces échelles jusqu'à ce que l'objet ait été correctement segmenté. La figure 4.12.a montre une image à segmenter transformée selon différentes échelles.

4.9.2 Perspective

Le changement de perspective (de point de vue) d'un objet dans une image complexifie grandement la segmentation lorsque celle-ci est effectuée selon les paramètres de notre

solution. En effet, la formation de notre modèle se base sur le fait qu'un objet est toujours représenté selon la même perspective. Par exemple, notre solution telle qu'elle est décrite jusqu'ici n'est pas en mesure de corriger adéquatement la segmentation de l'image d'un cheval vu de face si le modèle est formé comme celui décrit à la figure 3.6. En effet, puisque ce dernier modèle décrit le contour d'un cheval vu de profil et que le contour d'un cheval vu de face est entièrement différent, la correction de la segmentation ne sera pas possible.

Une façon de résoudre ce problème consiste à d'abord former un modèle qui tient compte des différentes perspectives selon lesquelles peut se présenter un objet, puis à chercher cet objet selon ces différentes perspectives. Ces dernières sont définies en formant des sous-modèles pour une même classe d'objets, où chaque sous-modèle contient un ensemble de fragments qui décrivent le contour de l'objet lorsque ce dernier est vu selon une perspective donnée. Par exemple, en ce qui concerne la classe *cheval*, nous pourrions définir un premier sous-modèle pour l'objet vu de profil, un deuxième pour l'objet vu de face et un troisième pour l'objet vu d'un angle de 45° (autrement dit, à moitié chemin entre une vue de face et une vue de profil).

Une autre solution à ce problème implique la formation d'un modèle tridimensionnel pour une classe d'objets [32]. Comme Marr le propose [27], un modèle tridimensionnel est plus proche de la représentation d'un objet que possède l'être humain. Un tel modèle n'est donc pas limité à une seule perspective, comme le serait le modèle d'une classe tel que décrit dans notre solution. Comme ce modèle est tridimensionnel, on peut extraire le contour de l'objet selon n'importe quelle perspective ; il n'est donc pas nécessaire, suivant cette solution, de bâtir un sous-modèle pour chacune des perspectives selon lesquelles un objet peut être représenté.

4.9.3 Orientation

Le changement d'orientation de l'objet dans une image présente essentiellement les mêmes difficultés que le changement de perspective, dans la mesure où les contours qui sont observés dans l'image ne correspondent pas à ceux décrits dans le modèle. La façon de résoudre ce problème se rapproche de celle proposée pour le changement d'échelle ; elle consiste à choisir un certain nombre d'orientations possibles pour l'objet et à appliquer successivement notre solution à une image transformée selon ces différentes orientations (voir Figure 4.12.b).

4.9.4 Direction

La dernière transformation géométrique que nous abordons est le changement de direction. Le changement de direction d'une image équivaut géométriquement à une symétrie. La direction selon laquelle un objet est représenté dans une image à segmenter doit correspondre à la direction empruntée par le modèle de sa classe. De plus, pour former ce modèle, tous les *exemples positifs* utilisés doivent représenter l'objet selon la même direction. Par exemple, les objets contenus dans les images de la figure 3.4 ont tous la même direction. La segmentation d'une image contenant un objet représenté dans une direction autre que celle utilisée pour former le modèle de sa classe ne peut pas a priori être corrigée selon notre solution.

Pour résoudre ce problème, il faut procéder de manière analogue à ce qui est fait pour le changement d'échelle ou d'orientation, c'est-à-dire appliquer notre solution à une même image en changeant la direction de celle-ci et trouver selon quelle direction la correction de sa segmentation correspond le mieux au modèle. La figure 4.12.c montre trois directions selon lesquelles on peut représenter une image.



Image originale
(échelle = 1)



Image originale
(orientation = 0°)



Image originale



Échelle = 0,75



Image orientée à 180°



Image dont la direction
horizontale a changé



Échelle = 1,5

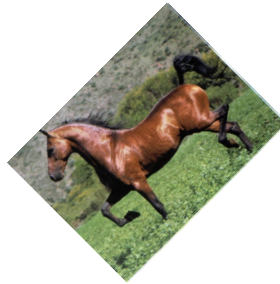


Image orientée à 45°



Image dont la direction
verticale a changé

(a)

(b)

(c)

Figure 4.12 – Transformations géométriques appliquées à une image naturelle

- a) : *changement d'échelle*
- b) : *changement d'orientation*
- c) : *changement de direction*

Conclusion

Dans ce travail, nous avons présenté une méthode qui effectue la segmentation d'objets dans des images. Non seulement permet-elle de corriger des segmentations partielles en segmentations complètes, mais elle permet aussi d'identifier la classe et la position d'un objet contenu dans l'image. La classe est décrite par un modèle de forme ; ce dernier est dans un premier temps formé par notre méthode grâce à des images d'apprentissage et ensuite utilisé dans l'opération de correction de la segmentation de l'image. La méthode que nous présentons, utilisée dans nos expérimentations sur trois classes d'objet distinctes, montre des résultats prometteurs.

Nous montrons, par ailleurs, que la qualité des résultats que fournit notre méthode est fortement tributaire de la segmentation des images. Il s'avère aussi que le temps d'exécution de notre méthode augmente de façon linéaire par rapport au nombre de modèles de classes qui ont été définis. Les perspectives envisagées pour notre méthode concernent la possibilité de corriger la segmentation d'images naturelles qui auraient subi une transformation géométrique ; nous avançons quelques façons de pallier ces situations.

Bibliographie

- [1] M.S. Allili et Djemel Ziou. Globally Adaptive Region Information for Automatic Color-Texture Image Segmentation. *Pattern Recognition Letters*, 28(15), 1946-1956, 2007.
- [2] M.S. Allili, Djemel Ziou, N. Bouguila et S. Boutemedjet. Image and Video Segmentation by Combining Unsupervised Generalized Gaussian Mixture Modeling and Feature Selection. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(10), 1373-1377, 2010.
- [3] S. Bagon, O. Boiman et M. Irani. What is a Good Image Segment? A Unified Approach to Segment Extraction. *European Conference on Computer Vision*, 5305(1), 30-44, 2008.
- [4] S. Belongie, J. Malik et J. Puzicha. Shape Matching and Object Recognition Using Shape Contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(24), 509-522, 2002.
- [5] E. Borenstein et J. Malik. Shape Guided Object Segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, 1, 969-976, 2006.
- [6] E. Borenstein et S. Ullman. Combined Top-Down/Bottom-Up Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(12), 2109-2125, 2008.

- [7] Y. Boykov et M. Jolly. Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images. *IEEE International Conference on Computer Vision*, 1, 105-112, 2001.
- [8] Y. Boykov, O. Veksler et R. Zabih. Fast Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11), 1222-1239, 2001.
- [9] T. Brox, L.D. Bourdev, S. Maji et J. Malik. Object Segmentation by Alignment of Poselet Activations to Image Contour. *IEEE Conference on Computer Vision and Pattern Recognition*, 1, 2225-2232, 2011.
- [10] C. Carson, S. Belongie, H. Greenspan et J. Malik. Blobworld : Image Segmentation Using Expectation-Maximization and Its Application to Image Querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8), 1026-1038, 2002.
- [11] D. Comaniciu et P. Meer. Mean Shift : A Robust Approach Toward Feature Space Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), 603-619, 2002.
- [12] J.L. Crowley, H.I. Christensen et A. Chehikian. *Vision as Process*. Springer, 1995.
- [13] G. Dougherty. *Digital Image Processing for Medical Applications*. Cambridge, 47-118, 2009.
- [14] D.A. Forsyth et J. Ponce. *Computer Vision : A Modern Approach*. Prentice Hall, 2003.
- [15] S. Geman et D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6), 721-741, 1984.

- [16] R.C. González et R.E. Woods. *Digital Image Processing*. Prentice Hall, 2008.
- [17] F. Han et S.C. Zhu. Bottom-Up/Top-Down Image Parsing with Attribute Grammar. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(1), 59-73, 2009.
- [18] C. Harris et M. Stephens. A Combined Corner and Edge Detector. *Proceedings of The Fourth Alvey Vision Conference*, 147-151, 1988.
- [19] S.L. Horowitz et T. Pavlidis. Picture Segmentation by a Directed Split and Merge Procedure. *International Conference on Pattern Recognition*, 424-433, 1974.
- [20] G. Hua, Z. Liu, Z. Zhang et Y. Wu. Iterative Local-Global Energy Minimization for Automatic Extraction of Objects of Interest. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10), 1701-1706, 2006.
- [21] S. Jayaraman, S. Esakkirajan et T. Veerakumar. *Digital Image Processing*. Tata McGraw Hill, 2009.
- [22] G. Larivière et M.S. Allili. A Learning Probabilistic Approach for Object Segmentation. *IEEE Canadian Conference on Computer and Robot Vision*, 2012.
- [23] B. Leibe et B. Schiele. Interleaved Object Categorization and Segmentation. *British Machine Vision Conference*, 759-768, 2003.
- [24] B. Leibe et B. Schiele. Interleaving Object Categorization and Segmentation. *Cognitive Vision Systems*, LNCS 3948, 145-161, 2006.
- [25] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2), 91-110, 2004.
- [26] M. Lynch, O. Ghita et P.F. Whelan. Left-Ventricle Myocardium Segmentation Using a Coupled Level-Set With A Prior Knowledge. *Computerized Medical Imaging and Graphics*, 30(4), 255-262, 2006.

- [27] D. Marr, S. Lal et H.B. Barlow. Visual Information Processing : The Structure and Creation of Visual Representations. *Philosophical Transactions of the Royal Society of London*, 290(1038), 199-218, 1980.
- [28] K. Mikolajczyk, A. Zisserman et C. Schmid. Shape Recognition with Edge-Based Features. *British Machine Vision Conference*, 2, 779-788, 2003.
- [29] A. Needham. Object Recognition and Object Segregation in 4.5-Month-Old Infants. *Journal of Experimental Child Psychology*, 78(1), 3-22, 2001.
- [30] S. Osher et J.A. Sethian. Fronts Propagating with Curvature Dependent Speed : Algorithms Based on Hamilton-Jacobi Formulations. *Journal of Computational Physics*, 79(1), 12-49, 1988.
- [31] M.A. Peterson. Object Recognition Processes Can and Do Operate Before Figure-Ground Organization. *Current Directions in Psychological Science*, 3(4), 105-111, 1994.
- [32] M. Riesenhuber et T. Poggio. Models of Object Recognition. *Nature neuroscience*, 3, 1199-1204, 2000.
- [33] O. Rotem, H. Greenspan et J. Goldberger. Combining Region and Edge Cues for Image Segmentation in a Probabilistic Gaussian Mixture Framework. *IEEE Conference on Computer Vision and Pattern Recognition*, 1, 1-8, 2007.
- [34] P. Salembier et L. Garrido. Binary Partition Tree as an Efficient Representation for Image Processing, Segmentation, and Information Retrieval. *IEEE Transactions on Image Processing*, 9(4), 561-576, 2000.
- [35] J. Shi et J. Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 888-906, 2000.

- [36] M. Sonka, V. Hlavac et R. Boyle. *Image Processing, Analysis, and Machine Vision*. Thomson, 2008.
- [37] R.S. Zemel, M. Behrmann, M.C. Mozer et D. Bavelier. Experience-Dependant Perceptual Grouping and Object-Based Attention. *Experimental Psychology*, 28(1), 202-217, 2002.
- [38] S.C. Zhu et A. Yuille. Region Competition : Unifying Snakes, Region Growing, and Bayes/MDL for Multiband Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9), 884-900, 1996.
- [39] L. Zhu, Y Chen et A. Yuille. Learning a Hierarchical Deformable Template for Rapid Deformable Object Parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6), 1029-1043, 2010.
- [40] L. Zhu, Y. Chen, Y. Lin, C. Lin et A. Yuille. Recursive Segmentation and Recognition Templates for Image Parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2), 359-371, 2012.
- [41] S. Zucker. Region Growing : Childhood and Adolescence. *Computer Graphics and Image Processing*, 5(3), 382-399, 1976.