République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Hadj Lakhdar BATNA
Faculté de Technologie
Département d'Électrotechnique

THÈSE

Présentée pour l'obtention du diplôme de

DOCTORAT en SCIENCES en Électrotechnique

OPTION

Électrotechnique

Par

HAMADA MAHFOUD

Thème

Contribution à la Segmentation d'images en Vision par ordinateur utilisant une approche mixte coopérative basée sur les Réseaux de Neurones et la Transformée de Hough.

Soutenue le Jeudi **26 / 11 / 2015** devant le jury composé de :

Dr. ABDESSEMED Rachid…	**Prof.**	**U. Hadj-Lakhdar de BATNA..**	**Président**
Dr. BOUTARFA Abdelhalim..	**Prof.**	**U. Hadj-Lakhdar de BATNA**	**Rapporteur**
Dr. ADANE Abdelhamid	**Prof.**	**U. Houari Boumediene d'Alger..**	**Examinateur**
Dr. ZIANI Rezki……	**Prof.**	**U. Mouloud Maameri de Tizi-ouzou**	**Examinateur**
Dr. TEBBIKH Hicham…	**Prof.**	**U. 8mai 45 de Guelma……**	**Examinateur**
Dr. BOUGUECHAL N-eddine	**Prof.**	**U. Hadj-Lakhdar de BATNA…**	**Examinateur**

بسم الله الرحمان الرحيم

يا رب علمني أن أحب الناس كلهم كما أحب نفسي وعلمني أن أحاسب نفسي كما أحاسب الناس وعلمني أن التسامح هو أكبر مراتب القوة وأن الانتقام هو أول مظاهر الضعف اللهم إن أعطيتني مالا فلا تأخذ سعادتي وإن أعطيتني قوة فلا تأخذ عقلي وإن أعطيتني نجاحا فلا تأخذ تواضعي وإن أعطيتني تواضعا فلا تأخذ اعتزازي بكرامتي

اللهم إن أسأت الى الناس فاعطني شجاعة الاعتذار وإذا أساء الي الناس فاعطني شجاعة العفو وإذا نسيتك فلا تنساني واجعلني أعمل على مرضاتك في كل حين

اللهم آميـــــــــــن

DÉDICACE

A mes parents,

A ma mère,

A ma femme,

A mes cinq enfants.

M. HAMADA

REMERCIEMENTS

Je tiens avant tout à remercier mon directeur de thèse Monsieur BOUTARFA Abdelhalim Professeur des Universités. Ses conseils lors de nos entrevues ont toujours été fructueux et au demeurant son optimisme m'a été d'une aide précieuse tout au long de cette thèse.

Je le remercie aussi pour la liberté qu'il m'a laissée dans mes recherches et pour tout le temps qu'il m'a consacré lors de l'écriture des articles et les paragraphes de ce manuscrit.

J'exprime ma sincère reconnaissance à Monsieur Abdessemed Rachid, Professeur à l'UHLB pour l'honneur qu'il m'a fait en acceptant de juger ce travail et de présider le jury de ma soutenance.

Je souhaite vivement remercier les membres du jury de m'honorer de leur présence lors de cette soutenance de thèse. Merci respectivement aux Professeurs Adane Abdelhamid (USTHB), Ziani Rezki (UMMTO), Tebbikh Hicham (Université 08 mai 45 de Guelma) et Bouguechal Nour-Eddine (UHLB) qui ont accepté d'être examinateurs pour cette thèse.

Le temps très limité qui leur était laissé pour relire le manuscrit ne les a pas empêché d'écrire des rapports très détaillés et constructifs, et ce malgré leurs nombreuses obligations. Je les remercie pour leurs remarques et commentaires qui m'ont permis de grandement améliorer ce manuscrit de thèse.

Merci également aux Professeurs Boulemden Mohamed et Naceri Farid de notre université de nous avoir soutenu dans cette dure épreuve malgré leurs nombreuses contraintes professionnelles.

Cette thèse n'aurait pas pu se dérouler dans d'aussi bonnes conditions sans le concours de l'ensemble du personnel du Centre de Développement des Technologies Avancées (Cdta) et plus particulièrement ceux de la division Robotique et de la Micro-électronique.

Merci également aux efforts conjugués des ingénieurs, techniciens et enseignants-chercheurs sans qui le laboratoire d'électronique avancée (LEA) de notre université ne pourrait fonctionner.

Je remercie particulièrement le Professeur Frédéric Morain-Nicolier du Centre de Recherche en Sciences et Techniques de l'Information et de Communication (Crestic) à l'IUT de Troyes en Champagne-Ardenne pour nous avoir accueillis et permis d'effectuer un fructueux stage dans son laboratoire.

Je tiens également à ne pas omettre tous les amis (es) extérieurs de m'avoir régulièrement permis de m'ouvrir l'esprit et de réaliser qu'il y a une vie en dehors de la thèse. La liste serait trop longue, et le risque qu'elle soit incomplète trop grand, mais qu'ils soient tous remerciés pour leur bonne humeur et leur soutien permanent.

Enfin, pour finir et pour être sûr de n'oublier personne, je remercie toute ma famille grande et petite, particulièrement ma femme qui m'a soutenu tout au long de ce travail.

RÉSUMÉ

En robotique mobile en général et plus particulièrement en vision artificielle, le processus de calcul d'orientation des primitives représente un des aspects les plus importants quand il s'agit de modéliser le milieu dans lequel évolue le système mobile. Nonobstant l'impérieuse nécessité de percevoir et d'appréhender un environnement pas ou peu connu, il y a lieu de prendre en charge les incertitudes liées au caractère aléatoire de cet environnement sur un double plan spatial et évènementiel.

La problématique globale, objet de notre travail de recherche, consiste à concevoir et à élaborer une approche systémique efficace en vue de l'extraction de caractéristiques d'ensembles cohérents de primitives d'une part et, d'autre part, de représenter adéquatement ces ensembles.

La méthode hybride que nous proposons implique un algorithme de description et d'apprentissage qui combine l'intelligence artificielle et la Transformée de Hough (TH). Cette technique nous permet de mettre à contribution les avantages inhérents aux deux approches notamment en conciliant réalisme biologique et simplicité calculatoire d'une part avec robustesse et efficacité d'autre part.

En vue de rendre encore plus simple la prise de décision par le système mobile, nous élargissons notre champ d'investigation à l'extraction et la représentation des attributs ainsi qu'à implémentation de la TH sur un circuit FPGA. De plus, et dans un souci d'assurer un déplacement sans collisions au système mobile, nous intégrons un protocole de navigation hybride nommé DVFF qui incorpore une stratégie de planification globale associée à une approche qui s'inspire du principe des champs de forces virtuelles VFF.

Nous indiquons finalement comment ce système peut être adapté à d'autres applications utilisant la vision et nous proposons des pistes prospectives pour l'adaptation d'un comportement en temps réel sur un robot.

Mots Clés : Vision, Traitement d'images, Segmentation, Contours actifs, Extraction de primitives, Reconnaissance de formes, Transformée de Hough, Réseaux de neurones, Robotique mobile.

ABSTRACT

In mobile robotics in general and most particularly when it comes to artificial vision, the process to calculate the primitives orientation represents one of the most challenging aspect regarding the modeling of the environment in which the mobile system evolves. Notwithstanding the necessity to perceive and comprehend a little or even not known environment, some uncertainties related to the random character of such an environment have to be taken into account on both spatial and event levels.

The global problematic we are tackling in the present research work consists in conceiving and elaborating an efficient and yet systematic approach in order to extract the characteristics of coherent sets of primitives on the one hand and to represent these sets in a adequate manner on the other hand.

The hybrid method we propose involves a descriptive and learning algorithm that combines both artificial intelligence and Hough Transform. Such a technique enables us to take profit of the advantages inherent to the two approaches therefore conciliating biological realism and computing simplicity on the one hand and robustness and efficiency on the other hand.

In view to make it even simpler for the mobile system to take a decision, we enlarge the investigation field to the extraction and representation of the attributes and the implementation of the Hough Transform on a FPGA circuit. Furthermore, and in order to ensure a motion free of collisions to the mobile system, we devise a hybrid navigational protocol called DVFF that incorporates a global planning strategy associated with an approach that derives from the principle of virtual forces fields VFF.

Finally, we indicate how this system can be adapted to other applications using vision and propose prospective leads for the adaptation of a real-time behavior on a robot.

Keywords: Vision, Image processing, Segmentation, Active edges, Primitives extraction, Pattern recognition, Hough Transform, Neural networks, Mobile robotics.

ملخص

إن أهمية هذا البحث تتمحور في مجال الروبوتات المتنقلة آليا بشكل عام وحساب التوجه نحو العملية البدائية يمثل واحدا من الجوانب الأكثر أهمية عندما يتعلق الأمر بالحاجة الماسة الى معرفة وفهم البيئة الغير معروفة وغير المناسبة.

المشكلة العامة و الشاملة التي هي مجال بحثنا هو تصميم و تطوير أسلوب منهجي فعال لاستخراج مقاربة متماسكة ومضبوطة من أجل الحصول على مميزات ومعرفة مبدئيا من جهة ومن جهة أخرى إيجاد تمثيلا مناسبا لها.

الطريقة الهجينة التي نقترحها تتضمن بروتوكولا تدريبيا وتعليميا يمزج بين الذكاء الاصطناعي ومحول هوق (Transformée de Hough) وهذه التقنية تسمح لنا بالحصول والاستفادة من المزايا الكامنة في كلا النهجين (الطريقتين) بما في ذلك التوفيق بين البيولوجي بصفة خاصة من الواقعية والبساطة الحسابية أولا والمتانة والكفاءة من جهة ثانية.

وبهدف تقديم المزيد من المساعدة الإضافية في اتخاذ القرارات من قبل النظام الآلي نوسع مجال عملنا في التحقيق لاستخلاص وتمثيل المعطيات بإضافة (TH) على شريحة **FPGA** زد على ذلك ومن أجل السير الحسن و الآمن وبدون حوادث للروبوت نزوده ببروتوكول سير هجين(مزيج) يسمى **D*VFF*** الذي يحوي تقنية تخطيط شاملة ومترابطة ناتجة من مجال القوى الظاهرية ***VFF*** .

كلمات مفتاحية : الرؤية، معالجة الصور، تجزئة، الملامح النشطة، استخراج البدائيون، التعرف على الأنماط، تحويل Hough، الشبكات العصبية، شريحة FPGA، والروبوتات المتنقلة.

LISTE DES FIGURES

Figure Int.1. : Vue d'ensemble de la spécialité..……..... **7**

Figure 1.1. : Représentation du problème d'asservissement visuel……... **15**

Figure 1.2. : Suivi d'un objet pendant une expérience d'asservissement visuel…….. **16**

Figure 1.3. : Illustration simplifiée du processus de cartographie et localisation simultanées moyennant la vision…….. **19**

Figure 1.4. : Fonction de coût du RANSAC en présence (a) 20% et (b) 40% de mesures aberrantes…….. **25**

Figure 1.5. : Robot CESA…….. **30**

Figure 1.6 : Structure du robot avec coordonnées de son centre de référence…….. **30**

Figure 1.7. : Image prise par la caméra lorsque le robot CESA est au milieu du couloir…….. **31**

Figure 1.8. : Robot CESA est au milieu et à proximité de la paroi droite du couloir…… **31**

Figure 1.9. : Robot CESA est au milieu et à proximité de la paroi gauche du couloir…… **32**

Figure 2.1. : **a :** Vue d'une scène artificielle, **b :** Coupe et agrandissement de la fenêtre la "hauteur" des surface correspond à leur niveau de gris. Les filtres sont généralement écrits pour détecter de tels contours…… **37**

Figure 2.2. : **a :** Vue d'une scène artificielle, **b :** Coupe et agrandissement de la fenêtre F. Les filtres sont souvent inopérants au voisinage du point triple T et **c :** Exemples de résultats: mauvaise détection de la jonction des lignes [34]……..…….. **38**

Figure 2.3. : Division récursive de régions suivant un critère d'homogénéité [34]……... **51**

Figure 2.4. : Image originale. 65536 points de 256 niveaux de gris [34]……... **52**

Figure 2.5. : Partition initiale obtenue par balayage séquentiel de l'image. Le sens de traitement des données influe sur la forme des régions [34]…… **53**

Figure 2.6. : Segmentation finale. Seuil = 10 (3578 régions) [34]…….. **54**

Figure 2.7. : Segmentation finale, seuil = 15 (2786 régions) [34]……... **55**

Figure 2.8. : Segmentation finale, seuil = 20 (2162 régions) [34]……... **55**

Figure 2.9. : Partition obtenue par l'utilisation optimale de trois prédicats d'homogénéité successifs (162 régions) [34]…….... **58**

Figure 2.10. : Superposition de la carte de points de contraste et de la partition en régions en cours de création. La région A est suffisamment entourée de points de contraste, sa croissance est arrêtée [34]…….. **59**

Figure 2.11. : Extraction des points de contraste par la méthode Deriche [34] et [37] …… **61**

Figure 2.12. : Résultat de la segmentation où la croissance de régions est contrôlée par une carte de points de contraste pré calculée [34]…… **62**

Figure 2.13. : Schéma récapitulatif des traitements pour la segmentation en régions des images stéréoscopiques [34]…… **65**
Figure 2.14. : Coupe stéréoscopique. Images initiales [34]…… **66**
Figure 2.15 : Couple stéréoscopique. Segmentation finales [34] …….. **66**
Figure 2.16. : Coupe stéréoscopique. Images initiales [34]…… **67**
Figure 2.17. : Couple stéréoscopique. Segmentations finales [34]…….. **67**

Figure 3.1. : Représentation simplifiée d'un neurone biologique [54]…….. **73**
Figure 3.2. : Neurone formel [54] ……. **73**
Figure 3.3. : Architectures de réseaux neuronaux [54] ……. **75**
Figure 3.4. : Réseau à représentation locale [54] …….. **75**
Figure 3.5. : Exemple de codage semi distribué par micro-traits [54] ……. **76**
Figure 3.6. : Structure pyramidale de la transformée de Hough hiérarchique…….. **86**
Figure 3.7. : Transformée de Hough, (a): Plan cartésien (xy) et (b): Plan des paramètres (ab) **87**
Figure 3.8. : Quantification du plan des paramètres (ab)…….. **87**
Figure 3.9. : Paramétrage polaire d'une droite…… **88**
Figure 3.10. : Quantification du plan des paramètres (ab)……. **89**
Figure 3.11. : Transformée de Hough.(a): Plan cartésien (xy).(b): Plan des paramètres(ab) **89**
Figure 3.12. : Paramètres polaires de deux droites opposées……. **90**
Figure 3.13. : Champ de la dimension de ρ…….. **91**
Figure 3.14. : Champ de la dimension de ρ…….. **91**
Figure 3.15. : Le plan de Hough en relief……... **93**
Figure 3.16. : Organigramme de la TH……..... **95**
Figure 3.17. : Droites réelles et insignifiantes…….. **96**
Figure 3.18. : Image en niveaux de gris…… **96**
Figure 3.19. : Image contour…… **97**
Figure 3.20. : Images Résultats t_{Tr} est le temps de traitement de la TH…….. **98**

Figure 4.1. : Deux obstacles renvoyant la même mesure……..... **111**
Figure 4.2. : Répartition des énergies…….. **111**
Figure 4.3. : Phénomène de spécularité……. **112**
Figure 4.4. : Exemple de réflexions multiples……... **113**
Figure 4.5. : Exemple de diaphonie……. **113**
Figure 4.6. : (a): Le système mobile ATRV2, (b): Perception panoramique…….. **114**
Figure 4.7. : Système de vision (ATRV2)…… **115**
Figure 4.8. : Système de vision (ATRV2)…… **117**
Figure 4.9. : Modèle sténopé d'une caméra…… **117**
Figure 4.10. : Mise en œuvre du calibrage du banc stéréoscopique du système ATRV2
(a): en utilisant la première mire pour le calibrage fort.
(b): en utilisant la deuxième mire pour le calibrage faible……. **121**

Figure 4.11. : Reconstruction 3D de la mire.
(a): Deux mires utilisées.
(b): La reconstruction 3D des deux mires dans le repère caméra gauche…… **122**
Figure 4.12. : Mise en œuvre du calibrage du banc stéréoscopique de l'ATRV2
(a): En utilisant le calibrage fort
(b): En utilisant le calibrage faible…….. **123**
Figure 4.13. : Algorithme de planification de chemin optimal D*…….. **127**
Figure 4.14. : Diagramme de l'approche de navigation DVFF……... **128**
Figure 4.15. : Chemins de navigation du robot mobile ATRV2…….. **130**
Figure 4.16. : Segmentation de notre processus de reconnaissance de formes…… **132**
Figure 4.17. : Architecture du réseau…….. **134**
Figure 4.18. : Image originale d'objets de formes cylindrique et polyédrique…….. **135**
Figure 4.19. : Résultat comparatif.
(a): Pour un réseau (25-15-5-1), T = 0.125.
(b): Pour un réseau (9-6-3-1), T = 0.125…….. **135**
Figure 4.20. : Image originale d'une scène d'intérieur de Laboratoire……. **138**
Figure 4.21. : Détection de contours par Deriche (α=1.5)……. **138**
Figure 4.22. : Tableau accumulateur……. **139**
Figure 4.23. : Construction de la fenêtre de recherche F(S_i)……. **140**
Figure 4.24. : Contrainte de compatibilité entre les appariements gauche-droite et droite-gauche…….. **142**
Figure 4.25. : Cas de recouvrement entre deux segments de droites…….. **143**
Figure 4.26. : Structure du Réseau de Hopfield……... **144**
Figure 4.27. : Paire stéréo 3D Labo……..…… **146**
Figure 4.28. : Paire stéréo Ball……..…. **147**
Figure 4.29. : Couple stéréoscopique, images originales 3D Labo…….. **148**
Figure 4.30. : Résultats de l'Algorithme appliqué sur l'image 3D gauche du Labo.
(a): Image initiale.
(b): Image contour binaire.
(c): Résultat de la reconnaissance…… **148**
Figure 4.31. : Résultats de l'Algorithme appliqué sur l'image 3D droite du Labo.
(a): Image initiale.
(b): Image contour binaire.
(c): Résultat de la reconnaissance…… **148**
Figure 4.32. : Application de l'Algorithme sur des balises (Polyédrique, Cylindrique et Rectangulaire)
(a): Image originale
(b): Image résultat…… **149**

Figure 4.33. : Résultats de l'Algorithme appliqué sur des objets multiformes.
(a): Image initiale
(b): Extraction des segments
(c): Espace de Hough
(d): Résultat de la reconnaissance…….. **150**
Figure 4.34. : Taux de reconnaissance en % du classifieur hybride en fonction des seuils de confusion et d'ambiguïté…….. **151**
Figure 4.35. : Circuit FPGA réalisé……. **151**

LISTE DES TABLEAUX

Tableau 2.1. : Nombre moyen de régions voisines par région……..…….....…….... **50**

Tableau 4.1. : Résultats d'appariements…….. **145**

Tableau 4.2. : Résultats obtenus par le logiciel…….. **145**

Tableau 4.3. : L'occupation de l'espace FPGA……..……..... **151**

ABREVIATIONS

I : Image initiale a segmenté.

S : Partition dont on calcule la qualité.

P : Prédicat d'homogénéité.

ε : Résolution de θ.

Res : Résidu partiel.

Q : Fonction de qualité locale.

C : Fonction de la qualité de la partition a optimisé.

N : Vecteur d'attributs d'une région (nœud du graphe d'adjacence).

A : Vecteur d'attributs d'une relation entre deux régions.

F : Nombre de facettes (une facette est définie par les arcs qui la délimitent).

X_S : L'ensemble des couples de points connexes de I appartenant à une même région.

M : Nombre de valeurs de ρ générées en même temps.

K : Nombre de division de θ.

ρ_i : Valeur de ρ obtenue pour un angle θ_i de l'axe de θ.

m_{ij} : Coefficients de la matrice de projection M.

$I_{(k, l)}$: Valeur du point image à la position (k, l).

Sgn : Fonction signe.

$H[J]$: Résidu complet.

R_i, R_j : Deux Régions de la partition.

$Seuil$: Seuil accepté pour qualifier l'homogénéité des régions.

R_i , R_j : Deux régions de la partition.

F_p , F_q : Fonctions d'évaluation des prédicats de la qualité locale.

F_n , F_a : Fonctions de mise à jour des attributs des nœuds et des arcs du graphe d'adjacence.

Max_i , Min_i : Niveau de gris maximum, respectivement minimum des points de la régions R_i .

Moy_i, Moy_j : Moyennes respectives des niveaux de gris des points des régions R_i et R_j.

Table des Matières

INTRODUCTION GENERALE **1**

1. Contexte Général **1**
2. Problématique **2**
3. Motivation et Contributions **4**
4. Structure de la Thèse **5**

CHAPITRE 1
OUTILS FONDAMENTAUX DE LA VISION POUR LA ROBOTIQUE **8**

1.1. Contexte et motivations **8**
1.1.1. Motivations pour l'utilisation de la vision monoculaire **8**
1.1.2. Apprentissage et adaptation **10**
1.2. Différents aspects de la vision pour la robotique **11**
1.2.1. Représentation de l'environnement **12**
1.2.2. Structure de l'environnement **12**
1.2.3. Extraction ou utilisation des informations visuelles **13**
1.3. Approches analytiques **13**
1.3.1. Asservissement visuel **13**
1.3.1.1 Asservissement visuel basé sur la position **16**
1.3.1.2 Asservissement visuel basé sur l'image **16**
1.3.2. Localisation et cartographie basée sur la vision (Vision-Based SLAM) **18**
1.4. Présentation de la chaîne de traitement **21**
1.4.1. Filtrage des images **22**
1.4.2. Extraction d'informations **22**
1.4.3. Utilisation des informations **23**
1.5 Méthodes robustes de vote **23**
1.5.1. Transformée de Hough **23**
1.5.2. Ransac (Random Sample Consensus) **24**

1.6 Exemples d'applications des algorithmes à la vision **26**
1.6.1. Détection de points d'intérêt **26**
1.6.2. Reconstruction 3D **27**
1.6.3. Reconnaissance d'amers visuels **28**
1.7. Conclusion **29**

CHAPITRE 2

SEGMENTATION D'IMAGES 33

2.1. Introduction : Que devrait être une bonne segmentation ? **34**
2.2. Choisir entre "approche contour" et "une approche région" **35**
2.2.1. Approche "contour " **36**
2.2.2. Approche "région" **39**
2.2.3. Notre approche **39**
2.3. Définition formelle de la segmentation **39**
2.4. Algorithme **43**
2.4.1. Structure de l'algorithme **44**
2.4.2. Structure de données et implantation de l'algorithme **44**
2.4.2.1 Graphe image **44**
2.4.2.2 Accès rapide au meilleur arc **45**
2.4.2.3 Mise à jour facile des attributs du graphe image **46**
2.4.3. Complexité de l'algorithme **47**
2.4.4. Partition initiale **50**
2.4.5. Résumé **52**
2.5. Prédicats d'homogénéité **53**
2.5.1. Utilisation d'un prédicat d'homogénéité **54**
2.5.2. Utilisation de plusieurs prédicats **56**
2.6. Coopération régions-contours **59**
2.7. Conclusion **62**
2.7.1 Discussion de notre approche **62**
2.7.2 Applications **64**
2.7.3 Utilisation de l'information « contour » **64**
2.7.4 Application à la stéréovision **64**

CHAPITRE 3

APPORT DES RESEAUX DE NEURONES ET LA TRANSFORMEE DE HOUGH 68

3.1. Approche neuronale **68**

3.1.1. Introduction **68**

3.1.2. Caractéristiques architecturales d'un réseau neuronal **72**

3.1.2.1. Réseaux à une seule couche **74**

3.1.2.2. Réseaux à couche unidirectionnels **74**

3.1.2.3. Réseaux récurrents **74**

3.1.2.4. Réseaux d'ordre supérieur **75**

3.1.3. Apprentissage connexionniste **76**

3.1.3.1. Niveaux de difficulté de l'apprentissage **76**

3.1.3.2. Types d'apprentissage **77**

3.1.3.3. Apprentissage par rétro-propagation du gradient **78**

3.1.3.4. Problèmes d'apprentissage **79**

3.1.4 Avantages, inconvénients des réseaux neuronaux **80**

3.1.4.1. Avantages de l'approche neuronale **80**

3.1.4.2. Inconvénients de l'approche neuronale **81**

3.1.5. Applications de l'approche neuronale **82**

3.1.5.1. Réseaux pour l'approximation de fonctions **83**

3.1.5.2. Réseaux pour la classification **83**

3.2. Méthode robuste de vote, la Transformée de Hough **83**

3.2.1. Principe de la Transformée de Hough **86**

3.2.2. Dimension des paramètres θ et ρ **90**

3.2.2.1. Champ de la dimension de θ **90**

3.2.2.2 Champ de dimension deρ **91**

3.2.2.3 Propriétés de la Transformée de Hough **92**

3.2.3. Utilisation de la Transformée de Hough **92**

3.2.4. Implémentation de la Transformée de Hough **93**

3.2.5. Etat de l'art : Evaluation en ligne d'Algorithmes de la TH **99**

3.2.5.1. Algorithme de H. Koshimizu et M. Numada [82] **99**

3.2.5.2. Algorithme de S. Tagzout et al [83] **101**

3.2.5.3. Algorithme proposé [84] **102**

3.3. Intérêt de l'hybridation **104**

3.4. Conclusion **106**

CHAPITRE 4

APPLICATION A LA SEGMENTATION D'IMAGESEN VISION ROBOTIQUE 109

4.1. Modèle de l'environnement du système mobile **110**
4.1.1. Système de perception utilisé **110**
4.1.1.1 Capteurs à ultrasons [89] **110**
4.1.1.2 Système de vision **115**
4.1.2. Représentation des mesures issues des capteurs embarqués **115**
4.2. Expérimentations de vision **123**
4.2.1. Problématique et solution **123**
4.2.2. Planification du robot ATRV2 ..... **126**
4.2.3. Résultats expérimentaux **129**
4.2.4. Notre processus Rdf **131**
4.2.5. Implémentation neuronale **133**
4.2.6. Extraction des chaines de points de contour **137**
4.2.6.1. Explication **137**
4.3. Application de la Transformée de Hough dans l'appariement des images **139**
4.3.1. Algorithme d'appariement **140**
4.3.2. Mise en correspondance par la méthode de Hopfield **143**
4.4. Résultats expérimentaux **144**
4.4.1 Implémentation sur FPGA **151**
4.5. Discussion et interprétation **152**
4.6. Conclusion **154**

CONCLUSION GENERALE **157**

ANNEXES **163**
Annexe A : **Calculs liés à la géométrie du flux optique** **164**
Annexe B : **Robot Mobile ATRV2** **171**
Annexe C : **Transformée de Hough** **174**

REFERENCES BIBLIOGRAPHIQUES **185**

VALORISATION DE MES TRAVAUX DE RECHERCHE **192**

Introduction Générale

Les travaux présentés dans ce mémoire ont été réalisés au sein du Laboratoire d'électronique avancée (LEA, UHLB), avec un financement de type Allocation de Recherche PNR. Ils se focalisent sur la segmentation en vision par des robots mobiles non holonomes (CESA et l'ATRV2).

Les résultats de cette recherche ont été en partie validés d'une part sur la plate-forme de robots mobiles au Crestic de l'UIT de Troyes (France) dans le cadre d'une coopération mixte avec notre établissement et d'autre part au Centre des Technologies Avancées (CDTA) au sein de la division robotique et productique de Bab-Hassen à Alger.

1. Contexte Général

Que cela soit sur internet, au cinéma, à la télévision, sur les téléphones, dans le domaine médical, l'image est partout.

Aujourd'hui il ne s'agit plus uniquement de traiter les images pour les améliorer mais aussi de les comprendre et de les interpréter. C'est dans ce contexte que la reconnaissance d'objets dans les images devient un sujet de recherche important. Et pour reconnaître des objets afin d'interpréter les images, il faut souvent au préalable les segmenter, c'est-à-dire séparer les objets d'intérêt du fond de l'image.

Le but de toute méthode de segmentation est l'extraction d'attributs caractérisant ces entités. Les attributs étudiés correspondent à des point d'intérêt où à des zones caractéristique de l'image (contours et régions). La détection des contours implique la recherche des discontinuités locales de la fonction des niveaux de gris de l'image.

Par exemple, dans le cas d'images réelles, les contours correspondent aux frontières des objets et les régions à leurs surfaces. Ces deux approches contour et région sont duales en ce sens qu'une région définit une ligne par son contour et qu'une ligne fermée définit une région. Elles mènent cependant à des algorithmes

complètement différents et ne fournissant pas les mêmes résultats. Cette dualité est cependant peu exploitée dans la plupart des méthodes existantes.

Un autre aspect de la segmentation est celui qui consiste à retrouver la géométrie des objets à partir des images. On obtient ainsi des représentations intrinsèques, aisément manipulables et utilisables, à partir de la réalité physique induite par l'image. De manière à obtenir ces caractéristiques géométriques, on est souvent conduit à définir une suite hiérarchique de représentations de l'information image permettant finalement d'obtenir des indices visuels servant à résoudre une tâche donnée. La reconnaissance consiste essentiellement à comparer des indices visuels bi-où tridimensionnels avec les indices des objets à reconnaitre.

Un des créneaux de la recherche en robotique mobile est de permettre à un système mobile (robot mobile) de se déplacer de manière autonome dans son environnement pour accomplir un certain nombre de tâches. Ces tâches sont, par exemple, se déplacer vers une cible fixe ou mobile, éviter les obstacles, percevoir et intervenir dans un milieu hostile. Pour cela, les problèmes à résoudre sont parfois complexes. Parmi eux, la détermination d'une représentation interne de l'environnement (carte) du système mobile au moyen de la perception pour mieux agir dans la planification des actions et le contrôle dans l'exécution. C'est dans ce contexte que s'inscrivent tous les travaux de notre thèse.

2. Problématique

En robotique mobile, les techniques d'apprentissage qui utilisent la vision artificielle représentent le plus souvent l'image par un ensemble de descripteurs visuels. Ces descripteurs sont extraits en utilisant une méthode fixée à l'avance, ce qui compromet les capacités d'adaptation du système à un environnement visuel changeant. Nonobstant les nombreux organes sensoriels développés par la nature, la vision est celui qui apporte l'information la plus riche et la plus complète sur le milieu environnant. La vue permet en effet de percevoir simultanément l'environnement proche et lointain avec une quantité de détails inaccessible à tout autre sens. Les animaux évolués peuvent grâce à la vue s'orienter et se déplacer, repérer et chasser une proie ou au contraire fuir un prédateur. Son utilisation représente un tel avantage que l'évolution de la vision fait partie des explications plausibles à l'explosion cambrienne.

Cette période qui s'étend d'environ 520 millions d'années avant notre ère a vu l'extinction soudaine d'un grand nombre d'espèces, l'apparition tout aussi rapide de traits anatomiques totalement nouveaux et la formation de la plupart des grands groupes d'animaux actuels. L'évolution de la vision sur quelques millions d'années pourrait avoir déclenché une course au développement entre proies et prédateurs, ce qui expliquerait ces changements soudains dans la morphologie des animaux.

La vision s'avère de plus remarquablement adaptative. Si la recherche de proie ou la fuite de prédateur ne représente généralement plus le quotidien de l'homme aujourd'hui, la vision est toujours aussi indispensable pour l'orientation et la navigation, pour la reconnaissance d'objets et de personnes ainsi que pour la communication gestuelle et écrite. Paradoxalement, son utilisation est tellement naturelle et simple pour l'homme qu'il est difficile d'appréhender la complexité des traitements réellement effectués pour acquérir et interpréter cette information.

Pourtant la perception visuelle est le résultat d'une multitude d'opérations réparties sur plusieurs chaînes de traitement parallèles qui occupent une grande partie du cortex humain [1].

De ce point de vue, il est intéressant de constater que les chercheurs en robotique ont rapidement essayé de faire bénéficier les robots de capacités de vision en supposant qu'un robot doté de vision serait plus à même d'effectuer des tâches essentielles telles que la navigation dans un environnement peu connu par exemple. Il s'est rapidement avéré que le problème était beaucoup plus complexe qu'il n'y paraissait et paradoxalement, les recherches actuelles en vision pour la robotique sont souvent moins ambitieuses que celles qui ont été menées dans les années 80.

Cela ne signifie pas qu'aucun progrès n'a été réalisé dans le domaine. Des tâches comme le positionnement d'un bras robot par rapport à des repères visuels ou la détection d'anomalies sur une pièce peuvent maintenant être effectuées de manière fiable. Cependant il s'agit plutôt dans ce cas de problèmes liés à l'utilisation de la vision dans l'industrie, pour lesquels l'environnement et les conditions d'acquisition peuvent être parfaitement maîtrisés. Des systèmes ont également été développés pour des robots mobiles autonomes, permettant par exemple de construire une carte métrique de l'environnement ou de suivre une route et détecter les véhicules en utilisant uniquement la vision.

Néanmoins ces systèmes restent très sensibles à toute variation de l'information visuelle, qu'il s'agisse d'un changement de luminosité, de texture ou de point de vue.

En termes de capacité d'apprentissage ou d'adaptation à un environnement changeant, la vision humaine reste encore très largement inégalée.

Une des difficultés majeures qui se pose aux systèmes artificiels pour l'apprentissage de la vision est que la quantité de données délivrée par les capteurs d'images est très importante et qu'il est difficile d'extraire l'information pertinente de ce flot de données. Les techniques d'apprentissage automatique existantes nécessitent généralement de disposer d'une information réduite et significative en entrée. De ce fait, les systèmes effectuant de l'apprentissage sur des données visuelles vont le plus souvent commencer par extraire des caractéristiques ou descripteurs représentant l'information visuelle de manière condensée.

L'apprentissage en lui-même est ensuite réalisé à partir de ces descripteurs. Ce parti pris, très compréhensible d'un point de vue computationnel, restreint toutefois grandement les possibilités d'utilisation de la vision et explique en grande partie le manque d'adaptativité de la plupart des systèmes artificiels.

En effet, les mécanismes d'attention visuelle mis en œuvre dans le cerveau humain permettent d'adapter les traitements visuels de bas niveau en fonction du contexte et de la tâche en cours [2].Cette influence réciproque entre traitements bottom-up et sélection top-down semble être centrale pour l'adaptativité de la vision humaine.

En final, pour lever un des aspects essentiels de la vision robotique, nous proposons une méthode permettant de décrire et d'apprendre des algorithmes de vision de manière globale, depuis l'image perçue jusqu'à la décision finale, moyennant des outils fondamentaux liés à la robotique.

3. Motivation et Contributions

Les travaux présentés dans cette thèse portent d'abord sur le développement d'une méthode de localisation et d'une méthode de navigation pour le système mobile (*ATRV2 ou CESA)* dans un environnement naturel peu connu. Nous nous intéressons plus particulièrement au développement des méthodes de localisation et navigation beaucoup plus qu'à l'environnement où le système mobile se déplace.

L'environnement naturel nécessite un certain type de capteur (inclinomètre, GPS, etc.) que notre système mobile ne possède pas.

Nous nous sommes restreints au déplacement de ce système mobile sur un sol plat et n'ayant aucune connaissance a priori sur son environnement. Notre

motivation primordiale est le développement de techniques de perception, de localisation et de navigation appropriées rendant ce système mobile capable de percevoir et d'utiliser ses données perceptuelles pour augmenter son autonomie de déplacement, pouvoir agir et réagir face à son environnement et accomplir ses tâches tout en évitant les obstacles naturels imprévus (amers).

Pour cela, le système mobile (*ATRV2 ou CESA)* doit d'abord modéliser et construire sa carte de l'environnement à partir de données issues de ses capteurs embarqués, se localiser dans cette carte et en suite générer un chemin sans collisions lui permettant d'atteindre son but. En suite, une fois atteint, commence le processus de vision dans le système de reconnaissance de formes (Rdf) proposé.

On peut résumer nos principales contributions de cette thèse de la manière suivante :

- Etude et mise en œuvre d'une méthode de navigation hybride nommée D^*V_{FF} combinant les méthodes basées sur les algorithmes D^* et V_{FF}.
- La définition formelle d'une structure d'algorithmes de vision facilement extensible et réutilisable avec le développement des méthodes associées permettant de faire évoluer ces algorithmes efficacement sur cette structure.
- La prise en compte par notre robot mobile du processus de vision dans sa globalité.
- Implémentation de l'outil TH (Transformée de Hough) dans un circuit FPGA.

4. Structure de la Thèse

Une des hypothèses principales de cette thèse est ainsi que l'apprentissage du processus de vision dans sa globalité est bien plus à même de résoudre des tâches complexes et de s'adapter à des conditions changeantes qu'un apprentissage utilisant uniquement des descripteurs visuels extraits de l'image par une fonction fixée à l'avance. Afin de démontrer la validité de cette hypothèse, cette thèse s'articule selon le plan suivant :

Le chapitre 1 débute par une présentation du contexte et des motivations de cette recherche et justifie le choix de restreindre cette étude d'abord à la vision monoculaire puis l'étaler ensuite à la vision stéréoscopique (3d).

Nous présentons ensuite différentes approches qui ont été proposées pour aborder les problématiques d'apprentissage et d'adaptation de la vision en robotique mobile et nous situons cette thèse par rapport à ces approches.

Le second chapitre est consacré à l'état de l'art de la segmentation d'images.

Devant la diversité des méthodes auxquelles la littérature scientifique fait référence, le choix d'une stratégie de segmentation demeure un problème ardu.

Nous présentons une étude assez exhaustive et comparative des différentes méthodes de segmentation dans le but de justifier le choix de celles qu'il faut adopter dans notre processus de reconnaissance de formes.

Dans le troisième chapitre, nous nous sommes initiés aux réseaux de neurones (*Rna's*). Les efforts menés avec les systèmes à base de connaissance ont permis de mettre en évidence leurs inadéquations avec le monde réel, du fait de la question toujours ouverte relative au passage du numérique (capteur) au symbolique (symbols grounding problem). Ceci nous a amené à mieux comprendre la tendance vers les systèmes hybrides neuro-mimétiques. Par suite, on s'est intéressé aux méthodes d'estimation robustes utilisées en vision par ordinateur, avec une attention particulière aux applications robotiques. Tout cela, nous a aidés à opter pour la Transformée de Hough et bien cerner l'approche dans le domaine de la reconnaissance de formes (Rdf) et son impact applicatif en robotique mobile. Les concepts de base liés au champ de dimension de θ et deρde la TH sont abordés, ainsi que son implémentation hardware.

Enfin, dans le chapitre 4, plusieurs expériences permettant de démontrer les capacités d'adaptation au contexte visuel de notre système sont présentées.

En effet, dans les premiers paragraphes, nous exposons l'analyse des outils de détection de contours tels que le filtre de Sobel, le filtre Laplacian, l'impact visé étant de justifier notre choix qui s'est porté sur le filtre optimal de Canny Deriche.

Nous montrons par ce choix judicieux comment l'utilisation de méthodes d'évolution guidée peut améliorer la performance des algorithmes évolués et accélérer leur développement.

Nous testons les capacités de généralisation de ces algorithmes.

Nous finalisons notre manuscrit avec une conclusion générale en évoquant les points forts de notre méthode, ses limitations ainsi que quelques perspectives futures. Un bilan sur nos différentes contributions tout au long des chapitres est présenté avec l'analyse qualitative de l'ensemble des résultats obtenus.

Les dernières parties en relation respectivement avec la bibliographie et les annexes sont établies en fin de manuscrit de la thèse.

Par ailleurs et à titre illustratif, la Figure (Int.1) ci-contre montre une vue d'ensemble des liens du domaine de la vision.

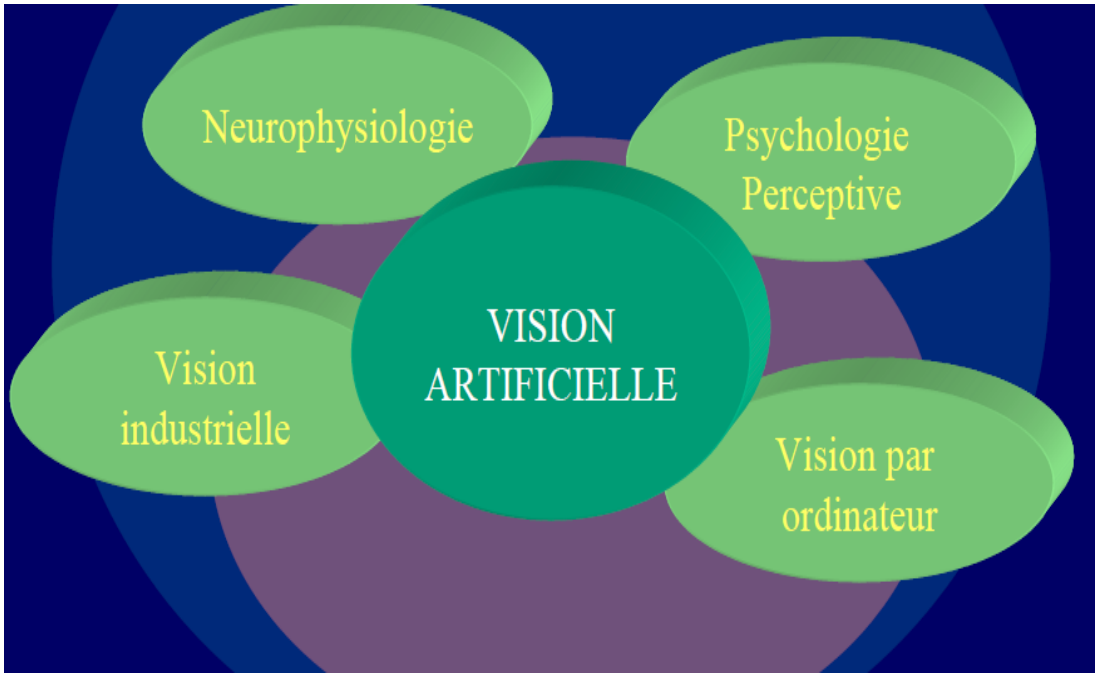


Figure (Int.1). : Vue d'ensemble de la spécialité.

CHAPITRE 1

Outils fondamentaux de la vision pour la robotique

L'objectif de ce chapitre est de présenter les outils fondamentaux de la vision par ordinateur - détection et segmentation, extraction d'indices visuels, géométrie et calibration des capteurs, stéréoscopie, localisation et reconnaissance d'objets, reconstruction, traitement d'images volumiques dans un langage mathématique simple, le souci de base étant la clarté. Ce chapitre contient par ailleurs de nombreux exemples d'utilisation de la vision par ordinateur dans un domaine de technologie de pointe : la robotique.

1.1. Contexte et motivations

Le contexte global de cette thèse est la conception d'un système de vision adaptatif aidant à la navigation des robots. Nous allons présenter dans ce premier chapitre les différentes approches de la vision pour la robotique, en nous concentrant sur celles qui permettent au robot d'apprendre ou de s'adapter à son environnement.

Cet état de l'art est loin d'être exhaustif, les travaux dans ce domaine étant nombreux depuis vingt-cinq ans. Le but est surtout de donner un aperçu des différentes méthodes qui ont été employées dans ce domaine, de leurs avantages et de leurs limitations. Nous essaierons ensuite de situer cette thèse par rapport à ces approches.

1.1.1. Motivations pour l'utilisation de la vision monoculaire

À l'heure actuelle, les fonctions de navigation des robots utilisent le plus souvent des capteurs de distance (sonars, radars, télémètres lasers, capteurs infrarouges) et/ ou des capteurs faisant appel à des éléments extérieurs (balises, GPS). Le principal avantage de ces systèmes est que l'information de position ou de distance est très facile à calculer. Ils présentent néanmoins plusieurs inconvénients.

Les capteurs de distance sont des capteurs actifs ce qui les rend généralement plus gourmands en énergie et plus facilement détectables. Cela peut être rédhibitoire pour des applications militaires par exemple. De plus, les sonars et les capteurs infrarouges souffrent d'un manque de précision. Ils sont généralement utilisés uniquement pour détecter des obstacles proches. Les télémètres lasers sont quant à eux beaucoup plus précis mais ils sont potentiellement dangereux et restent relativement chers.

Les capteurs utilisant un élément extérieur sont par définition dépendants du système de positionnement. L'utilisation de balises limite les applications du système aux environnements préparés, et l'installation des balises nécessite en soi une source extérieure d'énergie et un accès à différentes zones de l'environnement. Cela exclut donc l'exploration de zones inconnues. De plus, le robot devient incapable de se positionner si une balise est détruite ou ne fonctionne plus.

Le GPS est plus fiable mais il est inutilisable en environnement intérieur ou dans un contexte d'exploration planétaire.

À l'inverse, les caméras sont aujourd'hui des capteurs bon marché et peu gourmands en énergie. Elles fournissent de plus une information très riche sur l'environnement du robot. Le fait que la vision soit également très utilisée par les animaux et les humains pour se localiser et naviguer dans leur environnement tend à montrer qu'il s'agit d'une capacité sous-exploitée en robotique à l'heure actuelle. Bien sûr, les données visuelles sont complexes à traiter pour en extraire une information utilisable par un système de navigation. C'est cette difficulté que nous cherchons à surmonter dans cette thèse.

Notons également qu'un grand nombre de systèmes de vision, aussi bien en robotique que dans la nature, utilisent en fait la stéréovision. Cela consiste à mettre en correspondance deux images ou plus, issues de capteurs distants généralement de quelques centimètres, afin d'extraire une information de distance. Nous avons choisi au contraire de nous limiter à la vision monoculaire pour différentes raisons :

- Les systèmes de stéréovision nécessitent plusieurs caméras, précisément positionnées et calibrées. Ils sont donc plus chers et plus complexes à mettre en place.

- La nature nous montre qu'il est tout à fait possible de se déplacer dans un environnement sans utiliser la stéréovision. Une grande partie des animaux ont les yeux placés sur les côtés de la tête. Cela leur procure un champ de vision très large pour détecter au mieux les prédateurs mais le champ de recouvrement permettant la stéréovision est alors quasi-nul. Cela ne les empêche pas pour autant de se déplacer efficacement dans leur environnement. De plus, le rapprochement des yeux chez la plupart des animaux capables de stéréovision ne permet pas d'utiliser celle-ci sur des distances supérieures à un ou deux mètres. Enfin, un animal ou un humain borgne est tout à fait capable de s'orienter et de se déplacer, même dans un environnement inconnu et sans période d'adaptation.

- La mise en correspondance d'une paire d'images stéréoscopique ne nécessite pas une puissance de calcul moindre en général. Si l'on considère à nouveau l'œil des animaux, il a été montrée que des traitements monoculaires rétiniens très simples sont suffisants pour réaliser certaines tâches, comme la détection de contraste ou de mouvement (voir [3] par exemple). La stéréovision est réalisée par des traitements plus complexes dans le contexte visuel, et nécessite de toute manière un prétraitement monoculaire important.

- Une des plates-formes cibles pour notre système est une rétine artificielle numérique programmable que nous présenterons rapidement à la fin de ce chapitre. Il s'agit d'un capteur de vision "intelligent", c'est-à-dire que le traitement de l'image est effectué directement sur le capteur. Ce genre de système perd de son intérêt s'il faut utiliser plusieurs images issues de différents capteurs.

1.1.2. Apprentissage et adaptation

Quels que soient les capteurs utilisés, la plupart des systèmes robotiques utilisent encore aujourd'hui un programme codé "en dur" au préalable pour accomplir une tâche donnée. Ce type de programme peut s'avérer très efficace et fiable tant que l'environnement du robot ne change pas. La plupart des robots utilisés dans l'industrie sont ainsi préprogrammés, ce qui n'empêche pas la réalisation de tâches complexes comme par exemple le positionnement précis d'un bras robot par rapport à des amers visuels (problème de l'asservissement visuel) [4].

Cependant, ces systèmes sont souvent complexes à programmer et fonctionnent très mal lorsque l'environnement change ou en cas d'événement imprévu. Dans le cas d'un robot mobile, les conditions et l'environnement sont presque toujours variables. Les recherches dans ce domaine s'intéressent donc aux mécanismes d'apprentissage et d'adaptation que l'on peut observer dans la nature pour concevoir des robots plus robustes aux changements et imprévus.

L'apprentissage peut toutefois avoir lieu à différents moments et de différentes manières. Les systèmes de type SLAM (Simultaneous Localization And Mapping) permettent au robot d'apprendre de manière autonome la carte de son environnement et de se localiser dans celui-ci [5]. Le comportement du robot est cependant toujours pré-codé et ne pourra donc pas s'adapter en fonction de l'environnement.

Les systèmes d'apprentissage offline sont utilisés pour programmer le robot de manière plus souple et plus facile. Il s'agit typiquement de systèmes d'apprentissage supervisé où un opérateur va "montrer" au robot la tâche à effectuer. Le programme utilisé par le robot est alors développé ou paramétré automatiquement pour accomplir cette tâche mais il ne pourra plus être modifié par la suite pendant le fonctionnement du robot.

Enfin, les systèmes d'apprentissage ou d'adaptation online sont conçus pour que le robot puisse changer son comportement pendant le fonctionnement afin de s'adapter à de nouvelles conditions. Il s'agit le plus souvent de systèmes d'apprentissage non supervisé où une fonction de coût va indiquer au robot si son comportement actuel est adapté ou non.

1.2. Différents aspects de la vision pour la robotique

La réalisation d'un état de l'art sur les travaux en vision pour la robotique est une tâche quelque peu ardue, tout d'abord car les recherches sur ce sujet ont été très nombreuses, mais surtout car il ne s'agit pas d'un domaine de recherche clairement défini. Les problématiques de vision en robotique se situent à l'intersection entre plusieurs domaines très différents comme le traitement d'images, la planification, la construction de cartes ou la compréhension de la vision d'un point de vue neurologique. Un état de l'art sur ces recherches risque donc fort de ressembler à un inventaire à la Prévert rassemblant des travaux sans lien apparent entre eux.

Un bon point de départ pour tenter de classifier ces travaux est la revue réalisée par DeSouza sur ce sujet, qui n'est pas très récente mais présente bien les différents aspects de la vision pour la robotique [6]. À partir de celle-ci, nous allons décrire rapidement les aspects qui nous semblent importants pour caractériser les différents travaux dans ce domaine.

1.2.1. Représentation de l'environnement

Cet aspect n'est pas directement lié à la vision et se retrouve dans tous les systèmes robotiques. Les questions qui se posent ici sont de savoir comment le robot se représente l'environnement et comment cette représentation est acquise. L'environnement peut être représenté sous la forme d'une carte métrique ou d'une carte topologique (c'est-à-dire sous la forme d'un ensemble de lieux reliés entre eux sans notion explicite de distance). Le robot peut également ne disposer que d'une représentation implicite de l'environnement (par exemple un ensemble de prises de vue associées à des règles ou actions, sans lien explicite entre elles), voire n'utiliser aucune représentation auquel cas la planification d'un déplacement n'est pas possible. La représentation peut être fournie au robot au préalable, généralement sous forme de carte, ou elle peut être apprise par le robot au cours de son déplacement dans l'environnement. Les systèmes de SLAM, par exemple, se situent dans ce dernier cas.

1.2.2. Structure de l'environnement

Un environnement structuré dispose de points de repères qui peuvent être utilisés par le robot pour se déplacer, comme par exemple le fait qu'un mur est généralement vertical ou qu'une route est une bande continue entourée de deux lignes blanches. À l'inverse, un environnement non structuré ne dispose pas de tels repères. C'est le cas par exemple pour les systèmes d'exploration planétaire comme le Mars Pathfinder Rover. Les systèmes de vision sont généralement conçus pour une utilisation intérieure ou extérieure, rares sont ceux qui fonctionnent dans les deux cas. Les caractéristiques principales d'un environnement intérieur sont une illumination relativement constante, un sol uniforme et plat ainsi qu'une structure géométrique constituée de couloirs, de portes et de pièces ou bureaux. Les environnements extérieurs subissent quant à eux de grandes variations d'illumination et sont généralement moins structurés. Il existe donc souvent un lien entre la

structure de l'environnement et le fait qu'il soit intérieur ou extérieur mais ce n'est pas forcément le cas. Une route est un environnement extérieur structuré et à l'inverse un bâtiment effondré peut être vu comme un environnement intérieur non structuré.

1.2.3. Extraction ou utilisation des informations visuelles

La plupart des applications de vision pour la robotique nécessitent tout d'abord d'extraire certaines informations pertinentes de l'image, puis utilisent ces informations pour une tâche donnée (contrôle, cartographie, etc.). La première phase d'extraction de l'information est plutôt un problème de traitement d'image, tandis que la seconde n'est pas forcément spécifique aux applications de vision. Les travaux en vision pour la robotique ne s'intéressent souvent qu'à une seule de ces phases, auquel cas il est utile de préciser quel est le problème traité. Notons que dans certains cas cette distinction est effectuée seulement de manière implicite, voire pas du tout. Nous allons présenter dans les sections suivantes différentes approches de la vision pour la robotique et quelques travaux significatifs pour chacune d'elles. Nous essaierons de situer ces travaux, ainsi que cette thèse, par rapport aux aspects présentés ici.

1.3. Approches analytiques

Les approches analytiques aux problèmes de vision utilisent de manière extensive les propriétés géométriques liées à la projection de l'environnement en trois dimensions sur le plan image à deux dimensions. Cela suppose en général que les données précises de calibration de la caméra sont disponibles, même si certains travaux s'intéressent à l'utilisation d'une caméra partiellement calibrée ou à la calibration automatique.

1.3.1. Asservissement visuel

Les recherches en asservissement visuel s'intéressent à l'utilisation d'informations visuelles pour définir une loi de contrôle permettant à un robot d'atteindre une position fixée à l'avance. Cette position cible est en réalité définie par la position que prennent plusieurs amers visuels dans l'image provenant de la caméra lorsque le robot se trouve à la position cible.

Le robot peut être initialement placé de manière aléatoire dans l'environnement tant que les amers visuels en question se trouvent dans son champ de vision.

Les premiers travaux dans ce domaine s'intéressaient principalement au positionnement d'un bras robot disposant de six degrés de liberté [7] et [8].

Cependant un certain nombre de recherches depuis une dizaine d'années s'intéressent au contrôle de robots mobiles non-holonomes avec les mêmes méthodes [7] et [8]. Notons que les amers visuels définissant la position cible peuvent également se situer sur un objet mobile. Dans ce cas les techniques d'asservissement visuel sont utilisées pour effectuer du suivi de cible ou du maintien de formation entre plusieurs robots mobiles [11].

Nous allons maintenant présenter les principes mis en œuvre par les techniques d'asservissement visuels. Il s'agit uniquement d'une introduction rapide sur le sujet. Le lecteur peut se reporter notamment à l'article de François Chaumette pour un tutoriel plus détaillé [4].

La Figure 1.1 présente les données de base du problème d'asservissement visuel, c'est-à-dire la position initiale, la position cible et les images correspondantes. Les croix rouges représentent les amers visuels qui vont être utilisés. Le but est de déterminer la loi de commande qui va permettre d'amener ces amers de leur position initiale dans l'image vers la position cible.

On distingue deux approches différentes d'asservissement visuel:

L'une basée sur la position et l'autre sur l'image. Dans le premier cas, les amers sont utilisés pour déterminer la position de la caméra, et la loi de contrôle sera établie à partir de cette information de position.

Dans le deuxième cas, on utilise directement la position des amers visuels dans l'image pour déduire la loi de contrôle.

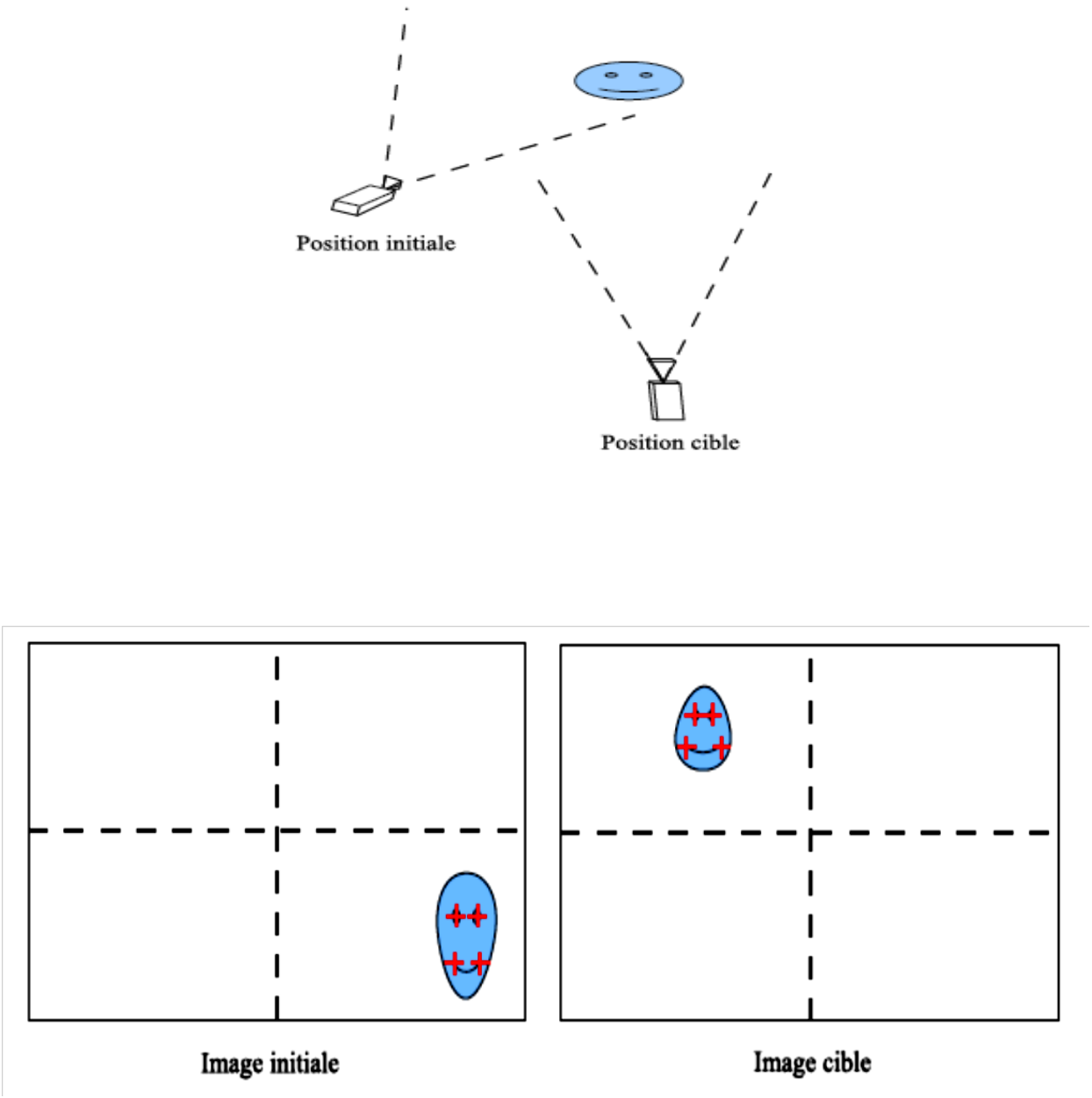


Figure 1.1 : Représentation du problème d'asservissement visuel.

La Figure 1.2 nous montre un exemple de suivi d'un objet pendant une expérience d'asservissement visuel 2D. L'objet suivi est en vert et sa position désirée dans l'image est en bleu. Les images de la première ligne correspondent à l'étape initiale de positionnement. Dans la suivante, à la fois l'objet et le robot sont en mouvement et subi de multiples occultations correctement traitées par les estimateurs robustes.

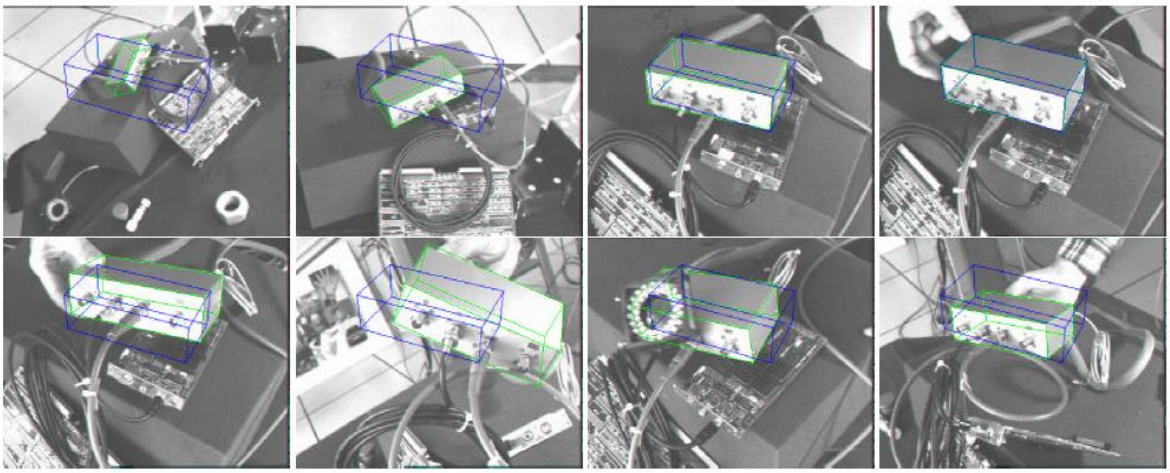


Figure 1.2. : Suivi d'un objet pendant une expérience d'asservissement visuel.

1.3.1.1. Asservissement visuel basé sur la position

Ce cas se ramène principalement à un problème de reconstruction 3D et à l'utilisation d'une loi de contrôle relativement simple. L'idée est ici d'utiliser les amers pour déterminer la position actuelle de la caméra dans l'espace en trois dimensions et la position cible. La loi de contrôle va chercher à réduire la distance entre ces deux positions. Il peut s'agir simplement d'effectuer une translation directe de la position initiale vers la position cible et une rotation pour placer la caméra selon l'orientation souhaitée, mais d'autres lois de contrôle sont également possibles. Le principal inconvénient de cette méthode est qu'elle ne traite pas directement le problème de la reconstruction 3D, la caméra étant en fait considérée à la base comme un capteur 3D. Des méthodes ont été proposées en traitement d'image pour résoudre ce problème mais elles nécessitent généralement une connaissance préalable de l'objet 3D sur lequel sont situés les amers visuels. Cette méthode étant particulièrement sensible aux erreurs sur l'estimation de la position de la caméra, elle est difficile à employer lorsque l'environnement n'est pas parfaitement connu.

1.3.1.2. Asservissement visuel basé sur l'image

Cette technique d'asservissement visuel utilise directement les coordonnées des amers visuels dans le plan image afin de déterminer la loi de contrôle.

Plus précisément, on va chercher à exprimer le déplacement de ces amers dans l'image en fonction du déplacement de la caméra. Il faut pour cela disposer des paramètres de calibration de la caméra et également d'une approximation de la profondeur des points dans l'image. Cette estimation de la profondeur nécessite

également l'utilisation de techniques de reconstruction 3D, mais cette méthode s'avère beaucoup plus robuste que la précédente aux imprécisions dans cette estimation.

Les erreurs à ce niveau ralentiront la convergence vers la position cible mais n'auront que peu d'impact sur la précision de la position finale. À partir de cette analyse, la loi de contrôle va déterminer le déplacement à effectuer pour réduire l'erreur sur la position des amers dans l'image. Un inconvénient majeur de cette méthode est qu'il existe des minima locaux dans cette fonction d'erreur qui peuvent empêcher la loi de contrôle d'amener la caméra jusqu'à la position cible.

Notons que ces deux méthodes peuvent être adaptées pour une utilisation en stéréovision, auquel cas la reconstruction 3D est grandement facilitée. Essayons à présent de situer les approches d'asservissement visuel en fonction des aspects de la vision pour la robotique présentés dans la section précédente (1.2). Tout d'abord, ces approches n'utilisent pas de représentation cartographique de l'environnement. Elles nécessitent toutefois certaines informations préalables comme l'image acquise par la caméra à la position cible et éventuellement une connaissance de l'objet 3D utilisé comme repère visuel. Cela limite l'utilisation de ce genre de technique aux cas où l'environnement est accessible et connu au préalable. Par contre, elles n'utilisent pas d'autres informations sur la structure de l'environnement, ce qui les rend utilisables en environnement intérieur comme extérieur, structuré ou non. Enfin, ces approches ne répondent pas au problème d'extraction de l'information visuelle, elles considèrent la caméra comme un capteur fournissant directement la position des amers visuels.

En résumé, ces méthodes servent au positionnement précis d'un robot pendant ce qu'on peut appeler une phase d'approche, durant laquelle un repère visuel donné est toujours visible. Une application industrielle typique est le positionnement d'un bras robot par rapport à une pièce à usiner (productique). Ces méthodes peuvent aussi être utilisées en robotique spatiale pour l'arrimage d'un module à une station spatiale par exemple. En robotique mobile, des applications possibles sont le positionnement du robot sur sa station de rechargement, le suivi d'un autre robot ou le maintien en formation de plusieurs robots. Ce type de méthode peut aussi être utilisé pour contrôler le robot à travers une suite de waypoints représentés par des images prises par la caméra en différents lieux. Cependant il est nécessaire dans ce cas d'acquérir au préalable un certain nombre d'images assez rapprochées les unes des autres et

ces méthodes sont mal adaptées pour faire face à des environnements variables. De ce fait, l'asservissement visuel est peu utilisé pour des tâches de navigation.

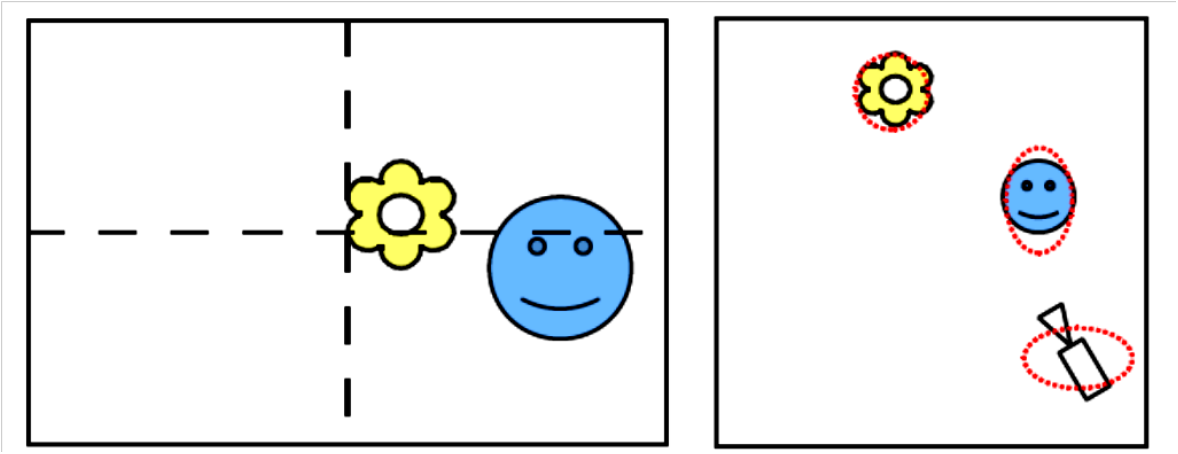
1.3.2. Localisation et cartographie basée sur la vision (Vision-Based SLAM)

Cette approche a pour but d'effectuer simultanément la construction d'une carte de l'environnement dans lequel se déplacent la caméra et la localisation de la caméra dans celle-ci. Elle s'inspire d'un côté des travaux de localisation et cartographie simultanée (Simultaneous Localization And Mapping ou SLAM) qui se basent généralement sur des capteurs de distance (voir [12] par exemple).

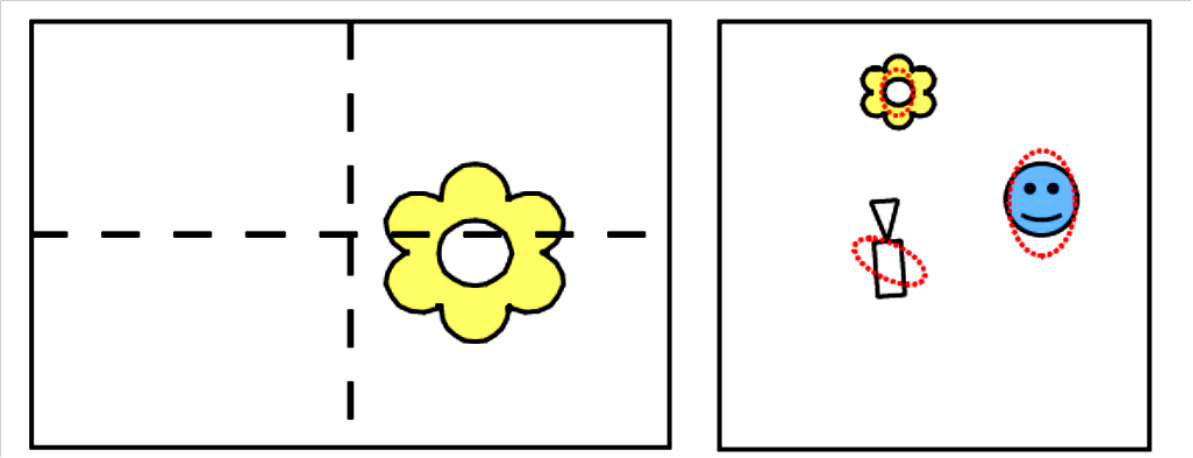
D'un autre côté, cette approche reprend un certain nombre de concepts développés dans les travaux de reconstruction 3D à partir du mouvement (Structure From Motion ou SFM, voir [13] par exemple) en leur ajoutant la notion de traitement en temps réel. Cette approche de SLAM basée sur la vision est apparue il y a une dizaine d'années avec les travaux d'Andrew Davison principalement [5] et [14].

Elle est très utilisée aujourd'hui car elle représente un moyen pratique et efficace d'effectuer de la cartographie et de la localisation en utilisant uniquement la vision. Il s'agit d'une approche bayésienne, les positions de la caméra et des différents repères visuels de l'environnement étant représentées avec une certaine incertitude.

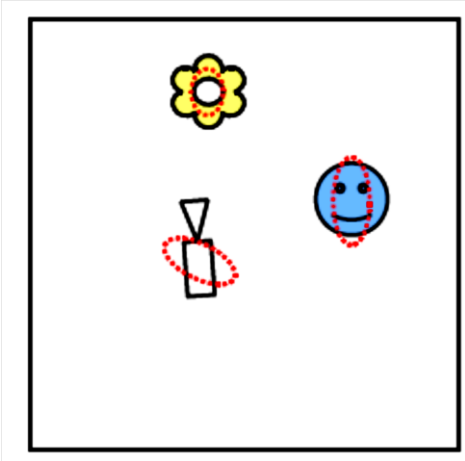
Ces incertitudes sont de plus liées entre elles : si l'on a observé plusieurs repères proches dans l'environnement, une diminution dans l'incertitude sur la position de l'un d'eux nous permettra également de diminuer l'incertitude sur la position des autres. Cette idée générale est représentée sur la Figure 1.3 où l'on voit bien que les ellipses en pointillés rouges représentent les incertitudes sur les positions.



(a) : Observation de deux repères visuels proches et estimation de leurs positions et celle de la caméra.



(b) : Nouvelle observation d'un des repères visuels et réduction de l'incertitude sur sa position et sur celle de la caméra.



(c) : Réduction de l'incertitude sur la position du deuxième repère visuel.

Figure 1.3. : Illustration simplifiée du processus de cartographie et localisation simultanées moyennant la vision.

En pratique, toutes les positions, les incertitudes et les corrélations entre elles sont représentées sous la forme d'un seul vecteur d'état et d'une matrice de covariance complète. L'idée est de mettre à jour ce vecteur et cette matrice à chaque nouvelle image pour prendre en compte les nouvelles informations. Cette mise à jour est effectuée en utilisant un filtre de Kalman étendu [15]. Cette méthode présente plusieurs avantages :

- L'estimation des nouvelles positions et la mise à jour du vecteur d'état et de la matrice de covariance ne nécessitent que les données à l'étape précédente et les nouvelles observations. Cela permet donc un traitement en temps réel des données, contrairement aux techniques de SFM qui nécessitent de traiter l'ensemble de la séquence d'images pour extraire les informations.

- Les incertitudes sur les mesures et sur le processus en cours étant prises en compte de manière explicite, le système sera plus adapté à l'utilisation de capteurs bruités qu'un système supposant des mesures exactes.

- Cette méthode facilite grandement la prise en compte de plusieurs types de capteurs. Pour un robot mobile roulant, on peut par exemple fusionner les données de la caméra et celles de l'odométrie pour obtenir des estimations de position plus précises.

Les méthodes de SLAM présentent également certains inconvénients qui font toujours l'objet de recherches :

- La taille de la matrice et la complexité de la mise à jour à chaque étape varient en $O\ (N^2)$, N étant le nombre de repères visuels présents dans la carte. En pratique, cela signifie qu'il devient difficile d'effectuer du SLAM en temps réel au-delà de quelques centaines de repères dans la carte. Plusieurs méthodes ont été proposées pour tenter de réduire cette complexité [16] par exemple).

- Un autre problème, commun à toutes les méthodes de localisation et de cartographie, est celui de la détection de fermeture de boucle et de la réinitialisation du système. Ce problème provient du fait qu'après un long trajet dans l'environnement sans retour en arrière, l'incertitude sur la position du robot va croître. Lorsque le trajet effectué est une boucle, il devient difficile de

détecter quand le robot revient à un endroit qu'il avait déjà visité auparavant. Cette détection est pourtant cruciale pour ne pas créer de doublons dans la carte et causer d'importantes erreurs de positionnement. Là aussi, des méthodes ont été proposées pour résoudre ce problème [17] (par exemple), mais il s'agit encore d'un sujet de recherche ouvert car la détection de fermeture de boucle doit être accompagnée d'une mise à jour majeure de la carte qui n'est pas triviale.

Si l'on considère les différents aspects sur la vision pour la robotique présentés à la section 1.2, il est clair que la question de la représentation de l'environnement est centrale pour les méthodes de SLAM basées sur la vision.

Cette représentation est ici construite par le système lui-même, généralement sous la forme d'une carte métrique. Notons toutefois que dans certains cas la carte est décomposée de manière hiérarchique avec une carte topologique au plus haut niveau et des sous-cartes métriques à chaque nœud topologique. La structure de l'environnement n'est pas utilisée en général, ce qui rend ces systèmes adaptés à tout type d'environnement. Ces systèmes ne traitent généralement pas le problème de l'extraction des informations visuelles et se concentrent plutôt sur l'utilisation qui en est faite, c'est à dire la cartographie et la localisation. Le choix des primitives visuelles utilisées est issu d'un compromis entre la robustesse de la détection et la complexité algorithmique.

Dans ses travaux, Davison utilise le détecteur de Shi et Tomasi [18] avec des fenêtres assez larges (11 x 11 pixels typiquement). D'autres travaux utilisent plutôt les propriétés SIFT (Scale Invariant Feature Transform [19]), plus robustes mais plus longues à calculer. Notons finalement que les travaux de SLAM n'abordent pas du tout le problème du contrôle du robot, qui est généralement effectué manuellement.

1.4. Présentation de la chaîne de traitement

La section précédente avait pour but de présenter différentes techniques utilisées en robotique basés sur la vision monoculaire. Il en découle, et plus généralement des travaux en traitement d'images, que les algorithmes de vision se présentent toujours plus ou moins selon la même structure.

Nous allons présenter cette structure dans cette section afin de justifier les choix réalisés pour notre système en termes de primitives de traitement et d'organisation de ces primitives.

1.4.1. Filtrage des images

La première étape d'un algorithme de traitement d'image consiste à appliquer un certain nombre de filtres sur celle-ci. Ces filtres peuvent faciliter les traitements suivants (utilisation de filtres gaussiens passe-bas avant un calcul de flux optique par exemple) ou directement mettre en évidence les parties de l'image qui nous intéressent pour une application donnée (filtre de texture permettant de segmenter le sol dans l'image par exemple). Il est possible également d'utiliser plusieurs filtres séquentiellement.

Par exemple, l'utilisation du flux optique pour déterminer les distances des obstacles (amers) peut se voir comme l'utilisation de trois filtres successifs :

1. Filtre gaussien permettant d'atténuer le bruit dans l'image.
2. Calcul de flux optique qui va transformer l'image en un champ de vecteurs.
3. Projection de ce champ de vecteur pour obtenir une image dépendant de la profondeur.

Cette projection peut s'effectuer sur un axe (horizontal ou vertical) ou par un calcul de norme. Ce filtrage a pour but de préparer l'image à l'étape d'extraction d'information qui vient en suite. Cependant cette étape de filtrage n'est pas toujours présente, certains algorithmes extraient l'information directement à partir de l'image originale.

1.4.2. Extraction d'informations

Cette deuxième étape a pour but d'extraire de l'image les informations nécessaires pour l'application choisie. Le but est de réduire la grande quantité de données que représente une image en un petit ensemble d'informations plus facilement exploitables. Ces informations sont généralement représentées par quelques valeurs scalaires. Les primitives utilisées pour cette étape d'extraction d'information peuvent se concentrer sur des portions réduites de l'image (extracteurs de coins ou de bords) ou sur des zones plus importantes (calcul de moyenne sur une partie de l'image). Dans notre système, nous nous concentrons sur les primitives permettant d'extraire de l'information sur les zones plus importantes de l'image. Ce choix vient du fait que l'évitement d'obstacles se fait généralement en identifiant des zones comme les murs ou le sol plutôt que des objets particuliers.

Par exemple, l'utilisation du flux optique se fait généralement en découpant l'image en plusieurs zones d'une certaine taille, voire uniquement deux zones lors

d'un simple équilibrage droite-gauche. De même, la détection de sol se fait en utilisant des fenêtres d'une certaine taille dans la partie inférieure de l'image.

Notons toutefois qu'en environnement intérieur, les détecteurs de bords peuvent extraire des informations géométriques utiles pour déterminer la structure de l'environnement (séparation entre les murs et le sol, emplacement des portes). Il pourrait donc être intéressant par la suite d'adapter ce système à l'utilisation de telles primitives.

1.4.3. Utilisation des informations

Une fois les informations extraites sous forme d'un ensemble de valeurs scalaires, la dernière étape consiste à prendre une décision en fonction de ces informations. Selon l'application visée, cette décision peut être une simple valeur booléenne (l'image fait partie de la catégorie visée ou non), une commande moteur (ce sera le cas pour l'évitement d'obstacles) ou un ensemble plus complexe de traitements (mise à jour d'une carte par exemple).

Dans notre cas la commande motrice sera représentée sous la forme de deux valeurs scalaires : La vitesse linéaire et la vitesse angulaire requises. Cette dernière étape consiste donc à transformer un ensemble de valeurs issues de l'image en un couple de valeurs utilisé pour générer la commande moteur. Cette transformation s'effectue à l'aide d'opérateurs scalaires classiques (addition, soustraction, multiplication et division).

1.5. Méthodes robustes de vote

1.5.1. Transformée de Hough

L'estimation des paramètres avec les méthodes de vote repose sur l'utilisation du minimum de données nécessaires à l'estimation. Chaque estimation, avec un jeu de données particulier, correspond à un "vote" pour les paramètres obtenus. Le jeu de paramètres élu, c'est-à-dire, le plus "voté", est retenu comme résultat de l'estimation.

La Transformée de Hough [20] est une méthode de vote très robuste. La version originale de la méthode proposée par Hough a été modifiée par [21].

Depuis plusieurs variantes ont été proposées [22]. Cette approche repose sur une discrétisation de l'espace des paramètres. On obtient alors des hyper-cubes dans l'espace d'état auquel sont associés des accumulateurs. Pour un jeu de données de

taille minimale, les paramètres recherchés sont estimés et l'accumulateur correspondant de l'hyper-cube est incrémenté.

Ce processus est itéré jusqu'à considérer toute les combinaisons possibles des données à disposition. L'accumulateur ayant la valeur la plus importante correspond alors à la meilleure estimation des paramètres.

La Transformée de Hough est bien adaptée aux problèmes ayant un nombre important de données par rapport aux nombre des paramètres à estimer. En effet, si les données et les inconnues sont de taille équivalente il est difficile de trouver un accumulateur prépondérant par rapport aux autres. En plus, dû à la discrétisation et au bruit il est possible que l'optimum soit délocalisé. La Transformée de Hough est très robuste car elle effectue une recherche globale et exhaustive. Finalement, cette technique est capable de segmenter les données en plusieurs populations qui vérifient le modèle de référence. Toutefois, la transformée de Hough est rarement utilisée seule en vision robotique car pour des problèmes qui nécessitent l'estimation de plus de trois ou quatre paramètres, les temps de calculs deviennent prohibitifs.

Pour parer à cet inconvénient, on associe l'IA dans notre processus Rdf qui sera désormais hybride et coopératif afin de parachever et par la même, justifier l'objectif de notre thèse.

1.5.2. Ransac (Random Sample Consensus)

La méthode RANSAC [23] est une méthode de vote probabiliste qui a été proposée afin de réduire le temps de calcul des méthodes de votes classiques comme par exemple, la TH. À partir d'un sous ensemble minimal de ''s'' signaux mesurés, il est possible de calculer les paramètres (équations 1 et 2) dans une situation non dégénérée. Ensuite, on calcule la fonction coût suivante:

$$C(x) = \sum_k^n \rho(r_k(x)) \tag{1.1}$$

$$\text{ou : } \rho\left(r_k(x)\right) = \begin{cases} 0 & si \quad r_k^2 \ (x) \leq C \\ 1 & si \quad r_k^2 \ (x) > C \end{cases} \tag{1.2}$$

$$c = \mathbf{2.5}\hat{\sigma}$$

Soit ‘’p’’ la probabilité de trouver la bonne solution, ‘’s’’ le nombre minimum de signaux nécessaire pour l’estimation des paramètres et ‘’r’’ le pourcentage d’inliers.

Le nombre ‘’m’’ de tirages aléatoires nécessaire pour avoir une probabilité ‘’p’’ de retrouver les bons paramètres est :

$$m = \frac{log(1-p)}{log(1-(1-r)^{s})} \tag{1.3}$$

Dans la Figure 1.4 est représentée la fonction coût du RANSAC. Le barres rouges représentent les résultats d’estimation des paramètres à partir des tirages aléatoires (pour certains tirages on obtient le même jeu de paramètres).

On peut voir que pour 20% d’outliers, 5 tirages aléatoires seulement sont suffisants pour avoir une probabilité de 95% de trouver la bonne solution. Quand le pourcentage d’outliers est de 40 %, 13 tirages sont nécessaires.

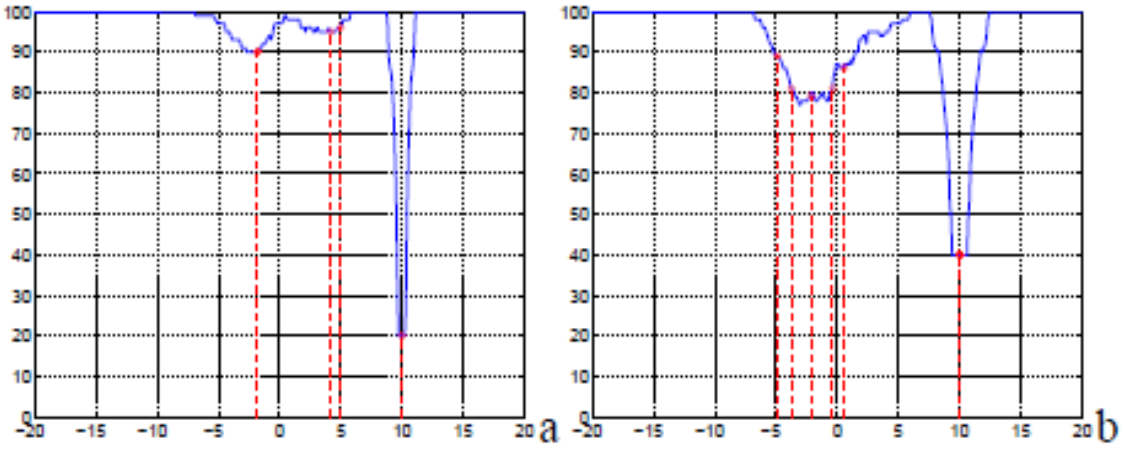


Figure 1.4. : Fonction de coût du RANSAC en présence (a) 20% et (b) 40% de mesures aberrantes.

1.6. Exemples d'applications des algorithmes à la vision

Les algorithmes évolutionnaires et la programmation génétique en particulier ont été utilisés pour nombre d'applications, de la conception de circuits électriques analogiques aux systèmes de stabilisation pour des drones.

Nous présentons ici quelques exemples d'applications plus directement liées au domaine qui nous intéresse dans cette thèse, à savoir la vision par ordinateur.

1.6.1. Détection de points d'intérêt

Marc Ebner a développé un système utilisant la programmation génétique pour détecter automatiquement des points d'intérêt dans une image et les mettre en correspondance dans une suite d'images [24]. Cette approche présente plusieurs points communs avec le système développé dans le cadre de cette thèse. Tout d'abord, il considère qu'un algorithme de vision est lié de manière inhérente à la tâche pour laquelle il est utilisé. Dans son cas, la tâche visée est le calcul d'un flux optique épars sur une séquence d'images. Il définit donc pour sa fonction de fitness un certain nombre de critères qui vont refléter la qualité des points détectés pour le calcul d'un flux optique épars. Ces critères sont le nombre de points détectés, la qualité de la mise en correspondance, le ratio entre le nombre de points extraits et le nombre de points pour lesquels une correspondance a été trouvée, la non-ambiguïté des correspondances, la régularité et la densité du flux optique ainsi calculé. Il combine ensuite ces critères en une seule fonction de fitness et prouve que son système fournit de meilleures correspondances que les systèmes existants selon cette fonction. L'idée majeure est que ces opérateurs ne sont pas forcément meilleurs dans l'absolu, mais qu'ils sont plus adaptés pour effectuer la tâche fixée. Un autre point commun avec nos travaux est la liste des primitives visuelles utilisées qui est assez proche de celle que nous avons définie. Il utilise notamment des filtres de Gabor, des filtres gaussiens ainsi que des différences de gaussiennes.

Leonardo Trujillo a utilisé une approche similaire pour détecter des points d'intérêt, la principale différence résidant dans la fonction de fitness utilisée [25]. Ici, les détecteurs sont évalués par leur taux de répétabilité (nombre de points qui sont détectés systématiquement lorsqu'on change l'angle de vue de la scène) et la répartition des points dans l'image. Il arrive ainsi à recréer des détecteurs de points

d'intérêts très simples mais efficaces comme la différence de gaussiennes ou le déterminant de la matrice hessienne.

Dans des développements plus récents de ces travaux, il utilise des techniques d'évolution multicritères basées sur le principe d'optimalité au sens de Pareto. Cela permet de ne pas comparer les différents critères directement et ainsi de sélectionner des opérateurs qui représentent différents compromis entre ces critères [26].Ces systèmes se rapprochent fortement de notre propre approche, tout d'abord car il s'agit de générer des algorithmes de vision par programmation génétique, mais surtout parce qu'ils traitent directement le problème de l'apprentissage de la vision bas niveau. Toutefois ils prennent en compte uniquement cette première étape du processus de vision tandis que nous essayons de traiter la vision dans sa globalité, depuis la perception et l'extraction d'informations jusqu'à l'application finale.

1.6.2. Reconstruction 3D

Une autre application des algorithmes évolutionnaires à la vision est la reconstruction de scènes en trois dimensions à partir de couples d'images (stéréovision). Cette méthode a été développée initialement par Jean Louchet sous le nom d'algorithme des mouches [27]. Elle s'inscrit dans l'approche dite parisienne, c'est à dire que la solution n'est pas représentée par un individu unique mais par un groupe d'individus. Les individus sont ici des points appelés mouches dans l'espace en trois dimensions.

L'évaluation va consister à calculer la corrélation entre la région entourant la mouche dans chaque image. Si la mouche se situe sur un objet, cette corrélation sera élevée, et inversement si la mouche est loin de tout objet cette corrélation sera très probablement faible. Les opérateurs de transformation sont géométriques : pour le croisement, la nouvelle mouche se situe à mi-distance des deux parents et pour la mutation, elle est déplacée aléatoirement d'une petite distance. Après quelques générations, les mouches convergent vers la surface des objets et permettent ainsi d'effectuer une reconstruction 3D de l'environnement. Cette méthode a été utilisée également pour éviter les obstacles [28]. Une variante de cette approche a été proposée par Gustavo Olague. Celle-ci consiste à spécialiser les individus, qui sont ici appelés abeilles en référence aux capacités d'organisation et de communication des abeilles au sein d'une ruche. Une première catégorie d'abeilles va explorer l'image et repérer les différents objets.

Une seconde catégorie va ensuite se répartir sur ces objets pour obtenir une représentation plus dense de ceux-ci [29]. Même si cette approche est très différente de la nôtre, il est intéressant de constater que des algorithmes évolutionnaires ont déjà permis d'obtenir de bons résultats en évitement d'obstacles basé sur la vision. Néanmoins comme indiqué précédemment, les systèmes basés sur la stéréovision nécessitent deux caméras précisément positionnées et calibrées ce qui les rend plus coûteux et complexes à mettre en œuvre.

1.6.3. Reconnaissance d'amers visuels

Un problème souvent associé au précédent est celui de l'encodage de descripteurs robustes à partir des points d'intérêts détectés dans l'image, et la reconnaissance de ces points dans d'autres images à partir de ces descripteurs. Les descripteurs SIFT par exemple sont très employés actuellement pour cela [19].

Ils utilisent des histogrammes sur l'orientation des gradients dans plusieurs patchs d'images autour du point d'intérêt pour décrire celui-ci. Ils présentent l'avantage d'être très robustes mais sont relativement longs à calculer et leur représentation reste assez lourde (chaque point d'intérêt est décrit par un vecteur de 128 éléments).

Une représentation beaucoup plus simple consiste simplement à encoder les valeurs des pixels autour du point d'intérêt, éventuellement en filtrant auparavant l'image. De multiples combinaisons de ces principes sont par ailleurs possibles. Il serait tout à fait envisageable de reprendre ces idées dans notre système pour faire évoluer des descripteurs adaptés à un contexte donné.

L'évolution multi-objective permettrait en outre de produire des descripteurs représentant différents compromis intéressants avec des en mise des fonctions d'évaluation qui pourraient être utilisées ainsi comme par exemple :

- La robustesse des mises en correspondance (taux de reconnaissance des points).

- La complexité de l'algorithme d'encodage et la longueur de la représentation utilisée.

1.7. Conclusion

L'objectif principal de cette thèse est la conception d'un système permettant d'adapter les capacités du robot en fonction du contexte visuel.

Dans notre approche, ce terme de contexte visuel regroupe les informations liées à la structure de l'environnement dans lequel évolue le robot et celles de plus bas niveau liées à l'aspect des objets dans cet environnement (texture, illumination, etc.) jusqu'au haut niveau (reconnaissance d'objets).

La grande diversité de ces informations visuelles et des manières de les exploiter nous a poussés à nous intéresser aux techniques d'évolution artificielle, bien adaptées pour gérer des représentations à structure variable et explorer de grands espaces d'état. Cette thèse s'inscrit donc directement dans le cadre de la vision robotique. La prise en compte du contexte visuel implique la nécessité d'inclure toute la chaîne de vision dans le processus d'apprentissage, et plus particulièrement la vision bas niveau. La plupart des travaux présentés précédemment font en réalité une utilisation très restreinte de la vision.

Toute la partie concernant l'extraction d'informations depuis l'image est fixée, et l'apprentissage s'effectue uniquement sur l'utilisation qui est faite de cette information extraite. Ce parti pris est très réducteur par rapport à l'utilisation qui est faite de la vision par les animaux, où tout le processus est intimement lié au contexte visuel et à la tâche effectuée. Nous pensons qu'une approche plus globale de la vision est indispensable pour développer de vraies capacités d'apprentissage et d'adaptation sur des robots mobiles. Nous chercherons donc ici à apprendre conjointement quelle information doit être extraite de l'image et comment l'utiliser.

Comme nous l'avons indiqué précédemment, les travaux utilisant la vision en robotique traitent généralement des images de quelques pixels seulement. Cela s'explique aisément d'un point de vue computationnel, mais l'utilisation d'images en haute résolution est indispensable dans notre approche. Nous voulons en effet prendre en compte un maximum d'indices visuels, et notamment des informations de texture qui ne sont disponibles qu'en haute résolution.

Heureusement, nous bénéficions de ce point de vue du développement continu de la puissance de calcul disponible, ce qui rend envisageable des expériences qui restaient impensables il y a dix ans. Notons toutefois que ce que nous entendons par "haute résolution" reste limité, puisqu'il s'agit ici d'images de 320 x 160 pixels.

Cependant le coût computationnel reste élevé et les expériences présentées dans cette thèse durent généralement des dizaines de jours. Un des inconvénients majeurs est que cela rend impossible toute analyse statistique digne de ce nom sur la variabilité des résultats obtenus.

A cet effet, ci-contre sur les Figures 1.5 et 1.6, nous présentons le robot CESA élaboré au CDTA au niveau de la division robotique et productique qui a été le support de la partie test et de l'expérimentation.

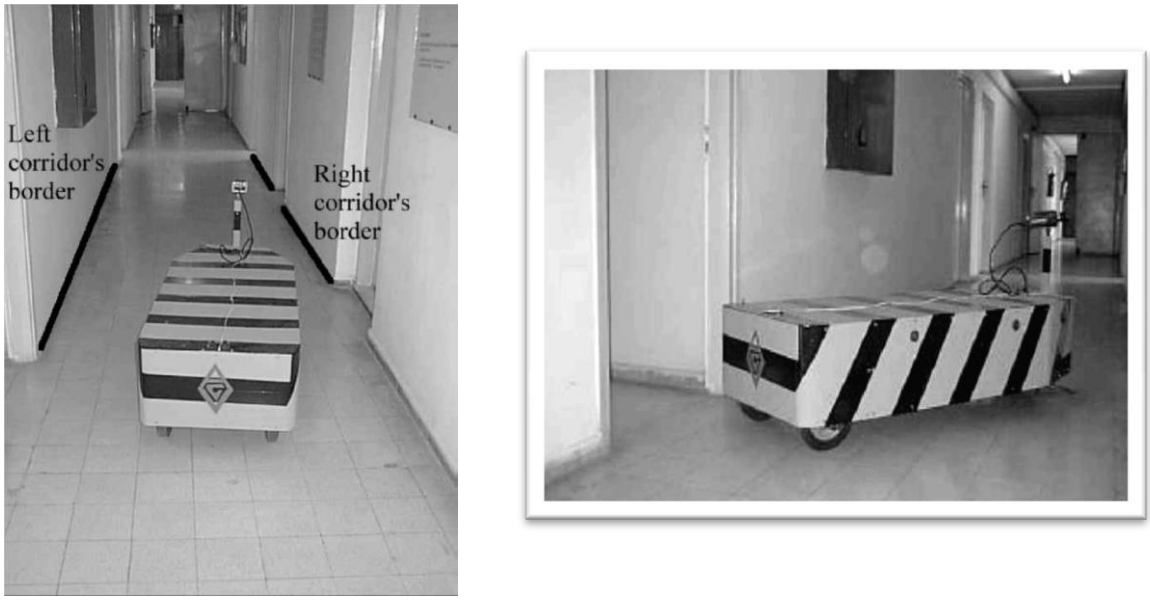


Figure 1.5. : Robot CESA.

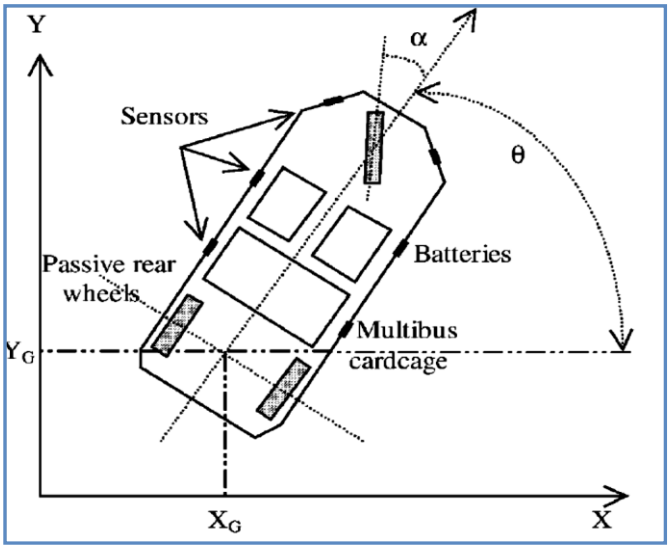


Figure 1.6. : Structure du robot avec coordonnées de son centre de référence.

Sur les Figures 1.7, 1.8 et 1.9, nous représentions les frontières du couloir (corridor) détectés par les segments pleins, la station cible dans le couloir est représentée par un petit carré et l'orientation que le CESA doit avoir une fois à cette position cible par un petit triangle. Au cours de cette expérience, le robot mobile est complètement autonome, mais cette autonomie est sensible à la fois à l'éclairage de l'environnement et à l'algorithme de détection des frontières dans le couloir.

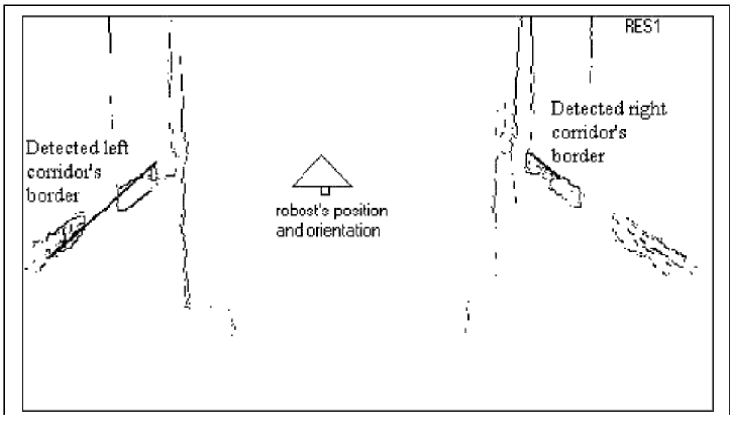


Figure 1.7. : Image prise par la caméra lorsque le robot CESA est au milieu du couloir.

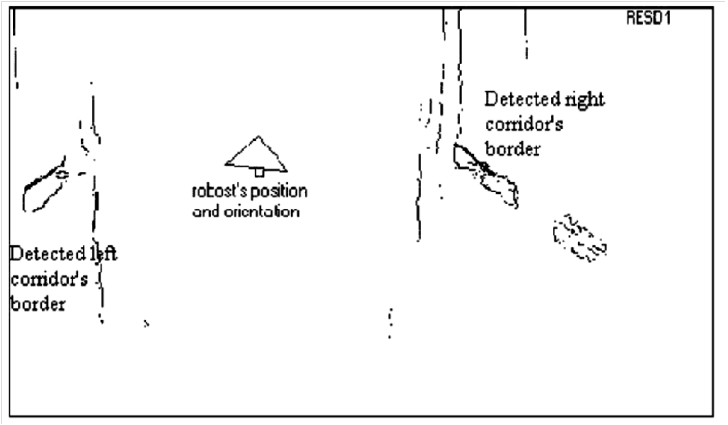


Figure 1.8. : Robot CESA est au milieu et à proximité de la paroi droite du couloir.

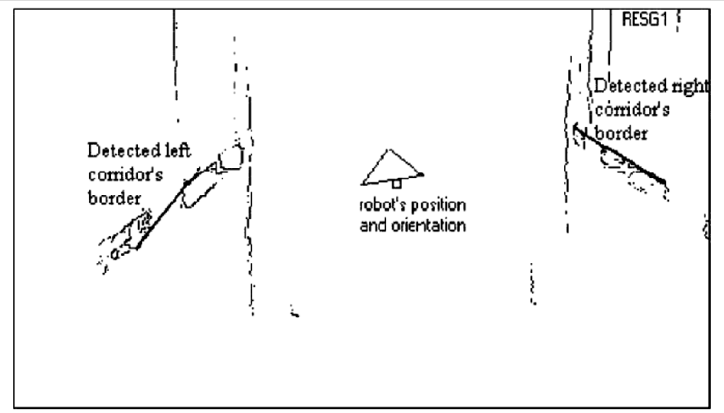


Figure 1.9. : Robot CESA est au milieu et à proximité de la paroi gauche du couloir.

Plusieurs algorithmes ont été développés pour résoudre cet aspect de la problématique de la vision en robotique mobile à savoir:

- L'algorithme développé par Mian [30] et Boutarfa [31], pour restaurer les niveaux de gris des images afin d'obtenir des meilleures images de pointe et de meilleure segmentation.

- L'algorithme mis au point par Aloimonos [32] et Achour et al [33] qui utilisent la technique d'adaptation stéréo pour calculer le modèle 3D de l'environnement afin de permettre au robot mobile de contrôler sa trajectoire et, éventuellement avoir une orientation automatique pertinente. Ces algorithmes sont maintenant au stade de l'expérimentation dans notre laboratoire.

Une des contributions majeures de cette thèse a été de concevoir une structure extensible pour représenter des algorithmes de vision, et de développer les outils permettant de les faire évoluer automatiquement. Ces concepts sont réutilisables pour beaucoup d'autres applications, potentiellement toutes celles qui utilisent la vision. Les modifications nécessaires pour cela consistent à déterminer les types de données, la base de primitives et les données d'entrée et de sortie adaptés à l'application visée.

Enfin, nous allons chercher à développer des algorithmes de vision implémentables et efficaces sur différentes architectures matérielles. Nous visons en particulier un système Rdf embarqué de type capteurs de vision "intelligents", basé sur une coopération hybride RN-THG programmable sur un circuit FPGA. Celle-ci présente l'intérêt de pouvoir à la fois acquérir et traiter l'image en consommant extrêmement peu d'énergie. Ce qui est un des buts recherchés.

CHAPITRE 2

Segmentation d'images

L'objectif de ce chapitre est de présenter la méthode de segmentation d'images développée pour obtenir les partitions utilisées lors de la mise en correspondance dans notre processus de reconnaissance de formes (Rdf).

Dans les premiers paragraphes, nous exposons le problème de la segmentation et les deux grandes approches (extraction de contours ou de régions) permettant de le résoudre, puis nous précisons la méthode que nous avons choisie.

Nous donnons ensuite une définition formelle de la segmentation, et présentons, d'une part, l'outil algorithmique général mis en œuvre, et d'autre part, les prédicats d'homogénéité de régions utilisés. Ces prédicats dépendent du type des images traitées.

En dernière partie, nous montrons une façon simple de contrôler notre processus de croissance de régions par une carte de points de contraste pré-calculée [34]. Ceci représente un essai, très encourageant, d'utilisation conjointe de deux informations duales: les discontinuités locales dans l'image (points de contraste) et les zones homogènes (régions).

2.1. Introduction: Que devrait être une bonne segmentation?

Il est actuellement difficile de qualifier la qualité d'une partition en régions car le résultat d'une segmentation est généralement jugé par l'homme en fonction de critères sémantiques difficilement implantables. De plus, cette qualité dépend souvent du traitement ultérieur choisi, elle diffère selon que l'objectif est, par exemple, la réduction du volume des données stockées ou l'interprétation de scènes.

Nous pouvons toutefois essayer de donner une définition générale d'une "bonne segmentation".

- Les régions devraient être uniformes et homogènes en fonction de caractéristiques comme le niveau de gris ou l'histogramme de répartition des niveaux de gris;

- Les régions adjacentes devraient prendre des valeurs relativement différentes du point de vue des critères en fonction desquels elles sont uniformes;

- Les frontières des régions devraient vérifier une certaine régularité; on souhaite que les régions ne soient pas déchiquetées, qu'elles ne comportent pas de petits trous et que les frontières soient lisses, non bruitées et correctement positionnées.

Effectuer une segmentation qui atteint de tels objectifs est très difficile pour les deux raisons suivantes:

- Des régions uniformes et homogènes comportent justement des contours bruités:
  - La zone de frontière entre deux surfaces dans une image réelle s'étend généralement sur une largeur de plusieurs pixels, alors qu'elle devrait "idéalement" être filiforme. Cette frontière ne peut donc être déterminée qu'approximativement.
  - Les valeurs des pixels dans la zone de la frontière sont relativement différentes de celles composant les deux régions à séparer:
    * Les intégrer aux régions rend celles-ci beaucoup moins homogènes et uniformes.

*Ne pas les intégrer crée des contours bruités ou même une région "fictive" car n'existant pas dans la réalité et représentant toute la "zone" de la frontière.

- Insister sur le fait que les régions adjacentes doivent comporter des différences significatives entre les valeurs de leurs caractéristiques a pour effet la fusion intempestive de ces régions :
  - Il est naturel de trouver, dans une image réelle, des zones adjacentes de caractéristiques similaires et représentant pourtant des surfaces différentes.

D'une manière générale, nous pouvons dire qu'il n'est pas simple d'extraire des indices visuels sans ambiguïté et sans qu'ils soient entachés d'erreurs, une partie de ces erreurs provenant notamment des incertitudes dues au capteur. Cette difficulté explique le nombre très important des travaux menés dans le domaine de la segmentation d'images. Elle explique également que les systèmes de plus haut niveau, comme la stéréovision ou l'interprétation de scènes, préfèrent travailler avec des primitives dont ces contours sont plus faciles à éxtraires.

2.2. Choisir entre "approche contour" et une "approche région"

Dans notre cas, nous souhaitons que la segmentation nous fournisse une "bonne" partition dans laquelle les régions correspondent aux projections d'objets ou parties d'objets présents dans la scène d'où est issue l'image. Le problème est de trouver la fonction "qualité" à optimiser afin d'obtenir cette partition.

Par exemple, nous pouvons définir la qualité d'une partition comme étant la différence de chaque point avec la moyenne de la région à laquelle il appartient. Ceci est décrit par la formule suivant:

$$C_2(S) = \sum_{k,l \in I} (I(k,l) - M_{kl})^2 \qquad (2.1)$$

avec:

C : Fonction de la qualité de la partition à optimiser.

S : Partition dont on calcule la qualité.

I : Image à segmenter.

$I(k,l)$: Valeur du point image à la position (k,l).

M_{kl} : Moyenne arithmétique des valeurs des points de la région à laquelle appartient le point (k,l).

Optimiser la qualité d'une partition revient dans ce cas à minimiser cette différence, c'est-à-dire à tenter de créer des régions où les points ont sensiblement les mêmes valeurs. Nous détaillons page 82 la fonction de qualité utilisée.

- Les techniques de segmentation existantes sont nombreuses, mais elles sont généralement fondées sur l'un des deux principes de base: **la discontinuité** ou **la similarité**.
- Les approches s'appuyant sur la recherche des discontinuités locales de la fonction de niveau de gris correspondent aux techniques de recherche de contours. Par définition, un point de contour est un lieu de forte variation de l'intensité dans l'image, ou une "crête" dans l'image gradient. Ces points de fort gradient sont souvent attachés à la présence d'un objet ou d'un reflet. Ils correspondent généralement à des transitions entre deux zones de caractéristiques différentes dans l'image. L'objectif est alors de décrire l'image avec un ensemble de points constituant les frontières des différentes entités physiques présentes dans la scène.
- Les approches fondées sur le principe de similarité correspondent aux techniques de recherche de **régions** homogènes dans l'image; elles font intervenir la recherche de la continuité de certaines caractéristiques sur des groupes de points adjacents.

Ces deux approches sont **duales** en ce sens qu'une région définit une ligne par son contour et qu'une ligne fermée définit une région. Elles mènent cependant à des algorithmes différents et ne fournissent pas les mêmes résultats.

Pour parvenir à segmenter une image en régions nous avons donc le choix entre ces deux approches.

2.2.1. Approche "contour"

Les avantages d'une méthode par recherche des lignes de contours sont multiples [35] :

- Les points de contour sont physiquement bien définis puisqu'ils correspondent aux endroits de l'image gradient où le rapport signal/bruit est maximal.
- Les crêtes de l'image gradient sont indépendantes du niveau d'éclairage, et également des variations faibles ou systématiques de l'éclairage. De plus, de par leur définition, les crêtes sont précisément les endroits où l'image gradient

possède le meilleur rapport signal/bruit si l'on suppose que le bruit est uniforme et indépendant du niveau du signal.

L'opération d'extraction de points de contours se fait généralement par filtrage de l'image. Cette extraction fait apparaître des discontinuités, les séquences de points voisins regroupés en lignes sont interrompues. La cause de ces discontinuités est double:

- Le gradient est trop faible à certains endroits de ligne de contours pour donner naissance à un point de contour.

- Les jonctions de plusieurs lignes de crêtes de gradient sont mal détectées: les opérateurs privilégient généralement une ligne. Ce problème provient des filtres utilisés pour détecter les contours "simples" entre seulement deux zones différentes. Ce principe est illustré (figure 2.1). Ces mêmes filtres ne sont pas adaptés au traitement des jonctions des lignes de contours, c'est-à-dire où un contour "multiple" passe par un point (figure 2.2).

Ce problème peut être partiellement résolu en utilisant des opérateurs directionnels qui ne détectent les contours que dans une seule direction sur l'image[36].

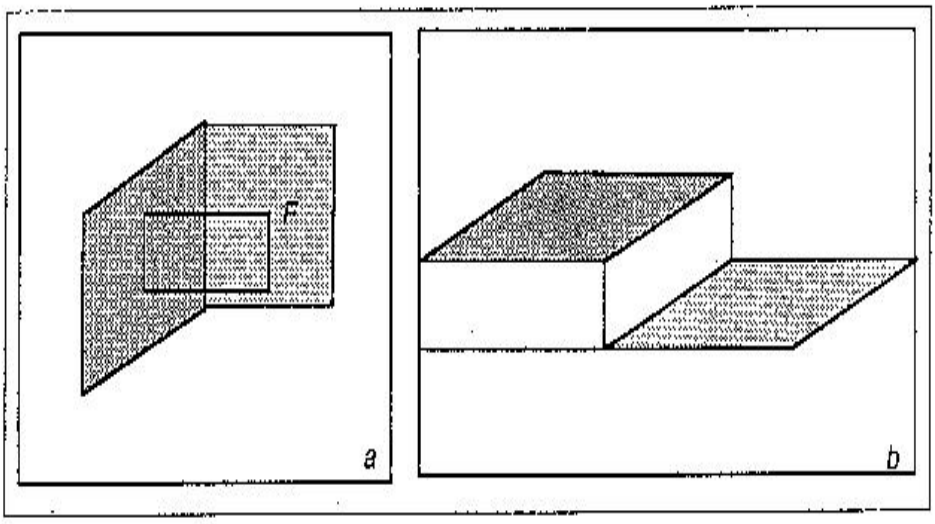


Figure 2.1.a: Vue d'une scène artificielle, **b:** Coupe et agrandissement de la fenêtre : La "hauteur" des surfaces correspond à leur niveau de gris. Les filtres sont généralement écrits pour détecter de tels contours [34].

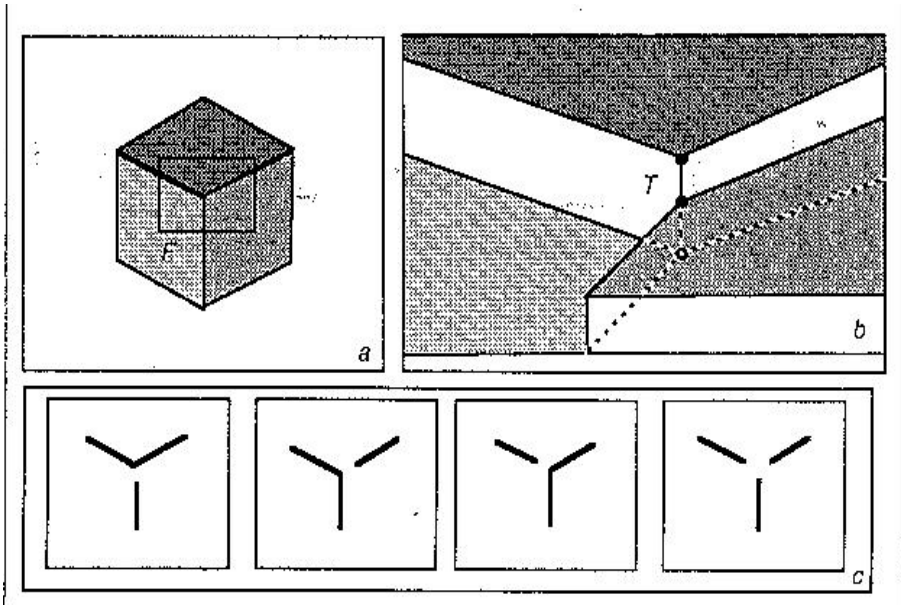


Figure 2.2. a: Vue d'une scène artificielle, **b:** Coupe et agrandissement de la fenêtre F. Les filtres sont souvent inopérants au voisinage du point triple T, **c:** Exemples de résultats: mauvaise détection de la jonction des lignes [34].

Puisque les lignes de contours sont discontinues, il devient nécessaire d'utiliser un algorithme de fermeture de contours. Certaines méthodes réalisent l'extraction et la fermeture en même temps. La difficulté est qu'une ligne de contraste peut traverser l'image en sautant d'un objet à un autre. Passer par des ombres et des reflets. Ceci signifie que son interprétation en terme bord d'objet, ou d'élément de la scène, peut varier le long de la ligne.

Un processus de fermeture de contours doit donc résoudre les problèmes suivants:

- déterminer le seuil de qualité afin d'interrompre le suivi d'une "mauvaise ligne".
- trouver les critères de sélection du point par lequel on poursuit le suivi: module du gradient, plus forte dérivée dans une direction…

Il s'avère que ces algorithmes fournissent de bons résultats sur des images bien contrastées et sont utilisés avec succès en robotique. Mais il faut souligner que cette approche s'accommode mal de la présence de texture dans les images. Les images d'intérieur que nous traitons se situent entre ces deux types d'images en ce sens où les images ne pas très bien contrastées car l'éclairage n'est pas contrôlé, et les surfaces des objets ont une texture relativement "fine" donc discrète. Cependant, la méthode d'extraction et de fermeture de contours développée par R. Deriche et J.P. Cocquerez et testée sur des images de scène d'intérieur fournit de très bons résultats [37].

2.2.2. Approche "région"

La notion de région n'a pas les propriétés que nous venons d'énoncer. Au contrainte, la segmentation d'une image en régions dépend souvent des conditions d'éclairage. Le niveau et la direction de l'éclairage influent directement sur la valeur de l'intensité lumineuse de chaque point. Les attributs utilisés pour caractériser les régions sont calculés directement à partir de ces valeurs; l'influence du niveau du signal et du bruit n'est pas annulée par l'utilisation d'un opérateur tel que le gradient.

Plus précisément, dans les zones à forte variation d'intensité, la nature même des frontières entre zones homogènes ne nous permet pas de les détecter de façon sûre; de tels contours ne sont pas parfaits et leur localisation par recherche de zones homogènes n'est pas garantie.

2.2.3. Notre approche

Plutôt que de choisir entre l'une ou l'autre de ces deux approches, nous avons décidé de les faire "coopérer", c'est-à-dire d'utiliser conjointement des propriétés d'homogénéité et des discontinuités locales dans l'image. L'idée est de contrôler un algorithme de recherche de régions par une carte de points de contours pré-calculée.

La suite de ce chapitre est consacrée à la définition formelle de la segmentation d'image en régions, puis à la présentation d'un algorithme de recherche d'une partition pour un processus Rdf par optimisations successives de sa qualité globale.

Nous montrons ensuite comment caractériser les propriétés d'homogénéité des régions et la manière d'améliorer les résultats par l'utilisation d'une carte de points de contours.

2.3. Définition formelle de la segmentation

Une segmentation d'une image I utilisant le prédicat d'homogénéité P est généralement défini comme un ensemble $S = \{R_1, ..., R_n\}$ de I telle que [38]:

(a)- $I = U_{i=1,n} R_i$

(b)- $\forall i \in [1, n], R_i$ *est connexe;*

(c)- $\forall i \in [1, n], P(R_i)$ = vrai

(d)- $\forall i \neq j \in [1, n], R_i$ *et* R_j *étant deux ensemble voisins,* $P(R_i \cup R_j)$ = *faux.*

- La première condition indique simplement que la segmentation est complète: tout point de l'image doit appartenir à une région et une seule. Un algorithme ne doit pas s'arrêter avant d'avoir traité tous les points de l'image.
- La seconde condition indique que les régions sont des ensembles de points connexes. C'est la raison pour laquelle les algorithmes prennent en compte le voisinage des points.
- Le critère de segmentation est défini à la troisième condition. Une région doit vérifier un prédicat d'homogénéité, lequel dépend généralement de la nature de l'image traitée. A titre d'exemple, citons un prédicat P imposant à des régions de posséder des points de niveaux de gris quasi-identiques:

$$P(R_i) = \left[Max_i - Min_i \prec seuil \right] \qquad (2.2)$$

Max_i et Min_i : niveau de gris maximum et minimum des points de la région R_i ;

S : est le seuil accepté pour qualifier l'homogénéité des régions. Par exemple, seuil = 10 pour des niveaux de gris variant de 0 à 255 pour l'image.

- La quatrième condition est une façon d'exprimer la maximalité de chaque région.

La vérification de ces quatre conditions est une **condition nécessaire et suffisante** pour qu'une partition d'une image soit une segmentation; rien, toutefois, n'implique son unicité. En particulier, les résultats de la segmentation dépendent de l'ordre dans lequel les régions sont traitées et non uniquement du contenu de l'image comme cela devrait être le cas. L'exemple donné ci-après illustre cet inconvénient:

Soit une image 2 × 2 où chacun des quatre points est représenté par la valeur de son intensité lumineuse:

1	2
4	3

Si nous prenons le critère de ressemblance suivant.

"Deux points voisins appartiennent à la même région si l'écart total entre leurs valeurs est inférieur ou égal à 1".

Nous obtenons plusieurs partitions qui dépendent de l'algorithme utilisé pour segmenter:

1 2	
4 3	

ou

1	2
4	3

Pour les mêmes raisons, une segmentation effectuée sur la transposée d'une image ne donne généralement pas la transposée de la segmentation de l'image originale.

Il est cependant possible de réduire cette indétermination en ajoutant l'optimisation d'une fonction C caractérisant la qualité globale d'une segmentation. Nous ajoutons alors aux quatre conditions précédentes, la condition **(e)** suivante :

(e)- De toute les segmentations S possibles qui vérifient les quatre conditions **(a)**, **(b)**, **(c)** et **(d)**, nous cherchons la (ou une) segmentation S^* qui optimise la fonction de qualité globale C, c'est-à-dire:

$$\forall S \in S_p(I),\ C(S^*) \geq C(S) \qquad (2.3)$$

où $S_p(I)$ est l'ensemble de toutes les partitions possibles de I, et C est une fonction positive, évaluant la qualité globale de la partition *S.*

Cette condition n'est pas suffisante pour obtenir une segmentation unique puisque plusieurs partitions de même valeur minimale pour la fonction C peuvent exister. Cependant, cette condition nous fournit une définition plus précise de la segmentation.

L'objectif est alors de trouver une *partition optimale* vérifiant des *contraintes de voisinage*. Si un tel problème peut être formulé de la façon suivante:

"*Soit G un graphe valué, trouver une partition en sous graphes connexes optimisant un critère donné*",

Alors le problème est *NP-difficile* [39].

En effet, soit $I = \{ i_1, i_2, ...i_n \}$ un ensemble de n valeurs;

Soit *G* le graphe complet de n sommets valués par les n valeurs $i_1, i_2, ...i_n$;

Soit *C* le critère sur la partition $S = \{ R_1, R_2, ...R_k \}$, défini par:

$$C(S) = \begin{cases} \infty & si\ k \succ 2 \\ Max_i(\sum_{j \in R_i}(i_j)) - Min_i(\sum_{j \in R_i}(i_j)) & si\ non \end{cases} \qquad (2.4)$$

S'il existe une partition S telle-que $C(S) = 0$, alors il existe une partition de *I* en deux sous-ensembles tels que la somme des entiers de ces deux sous-ensembles soit égale.

Or le problème de l'existence d'une partition d'un ensemble d'entiers en deux sous-ensembles de sommes égales est $NP-complet$. Optimiser le critère *C(S)* permettrait de répondre à ce problème d'existence. Cependant, notre propos n'est pas de décider de l'existence d'une telle partition, mais d'en déterminer une; ce problème est donc $NP-difficile$.

Pour notre problème, notons ici que ni le critère, ni le graphe, ne correspondent exactement à notre représentation. Notons que notre algorithme travaille avec un graphe planaire représentant une partition image. Or, à aucun moment, la planéité du graphe n'est prise en compte. L'utilisation de cette caractéristique permet peut être au problème de ne pas être $NP-difficile$, mais nous n'avons pas cherché à le vérifier. Nous pouvons donc seulement conclure que la recherche d'une partition optimale est probablement un problème $NP-difficile$.

Nous nous proposons donc de résoudre ce problème de façon sous optimale. Notre idée est d'adopter une stratégie d'optimisation locale au cours des étapes de l'algorithme, dont on espère que la solution ne soit pas trop distante de l'optimum réel.

Nous pouvons alors décomposer le problème de la segmentation d'images en régions en deux parties:

- Etablir une stratégie d'utilisation du prédicat d'homogénéité pour optimiser la qualité globale de la segmentation; cette optimisation s'effectue au fur et à mesure du déroulement de l'algorithme;

- Définir le prédicat d'homogénéité à utiliser; ce prédicat caractérise les régions à extraire et dépend donc de la nature de l'image traitée.

Cette première partie correspond à la définition de l'outil algorithmique à utiliser pour optimiser la qualité de la segmentation; la seconde partie est la définition de l'outil mathématique. Nous présentons ces deux outils ci-après, leur définition et leur formalisation sont le résultat de la synthèse de nos travaux et de ceux d'Olivier Monga de l'INRIA, et de plus amples détails peuvent être trouvés dans [40].

2.4. Algorithme

Il existe de nombreuses techniques de segmentation d'images, mais notre propos n'est pas d'en discuter dans ce document, et l'on pourra se reporter à la comparaison de ces méthodes par R. Haralick [41] et à leur critique par T. Pavlidis [42]. Il apparait que les trois principales classes des techniques de segmentation sont les suivantes:

- Techniques de fusion,
- Techniques de division,
- Techniques de fusion/division.

Les méthodes fondées sur des fusions de régions procèdent généralement par fusions successives jusqu'à trouver une partition qui satisfasse la définition donnée précédemment. La difficulté de ces méthodes est de trouver le critère de regroupement à utiliser.

Les méthodes procèdent par division de régions possèdent deux difficultés: trouver d'une part le critère de division et d'autre part la façon dont on divise une région non homogène.

Les méthodes **mixtes** combinent à la fois la nature ascendante des techniques de fusion et la nature descendante des techniques de division. Le principe est de partitionner arbitrairement l'image initiale, puis, à chaque étape, soit de diviser les régions si elles ne sont pas homogènes, soit de les fusionner si deux régions voisines sont similaires. Ces techniques cumulent les difficultés des deux techniques précédentes et posent également le problème de la stratégie à adopter.

Une solution simple consiste à diviser tout d'abord l'image initiale en régions très homogènes en appliquant un découpage simple (chacune des zones est partagée en quatre sous-régions par exemple), puis à fusionner des régions adjacentes vérifiant un prédicat d'homogénéité [43].

Nous avons choisi de mettre en œuvre une technique de croissance de régions, mais pour des raisons de coût en temps de calcul, l'algorithme de fusion débute, non pas sur l'image originale, mais sur une partition initiale obtenue par divisions successives jusqu'à obtenir de petites régions très homogènes. Notre algorithme s'apparente alors à la troisième classe de technique: division/fusion.

2.4.1. Structure de l'algorithme

L'idée de base de l'algorithme est d'optimiser la qualité globale C de la segmentation par fusions successives de régions vérifiant le prédicat *P* [44].

Parmi toutes les fusions de régions adjacentes permises à une étape, seule la meilleure va être réalisée. Ceci revient à chercher, dans toute l'image, les deux régions adjacentes dont la réunion optimise le critère de fusion *Q*.

Nous avons donc la procédure Croissance suivante:

Procédure Croissance $Prédicat\ P, Qualité\ Q$:

- Tant qu'il existe un couple de régions adjacentes dont la réunion vérifie le prédicat d'homogénéité P **faire**
  - Choisir parmi tous les couples (R_i, R_j) vérifiant $P(R_i \cup R_j)$, celui pour *lequel la qualité locale fusion* $Q(R_i \cup R_j)$ *est optimale.*
  - Mettre à jour la partition S :

$$S = S - \{R_i\} - \{R_j\} \cup \{R_i \cup R_j\} \qquad (2.5)$$

avec S : partition en cours de création.

L'algorithme de segmentation en régions a alors la structure suivante:

- Données:

  I : Image initiale;

  P : Prédicat d'homogénéité;

  Q : Fonction de qualité locale.
- $S = \{\{(k,l)\} / (k,l) \in I\}$
- *Croissance* (P,Q)

2.4.2. Structure de données et implantation de l'algorithme

La détermination, à chaque étape, de tous les couples de régions candidats à une fusion, sous-tend une complexité algorithmique importante. Des structures de données adaptées permettent cependant d'aboutir à une implantation algorithmique de coût réduit.

2.4.2.1 Graphe image

La partition en cours de création se présente sous la forme d'un ensemble de régions, chacune de ces régions est caractérisée par un ensemble de valeurs (attributs) et possède un lien avec ses voisins; ce lien est également caractérisé.

Ceci nous conduit naturellement à représenter une partition en régions sous forme d'un graphe d'adjacence value :

- A chaque nœud du graphe, nous associons une région et ses attributs (nombre de points, somme des niveaux de gris des points, …);
- A chaque arc du graphe, nous associons deux régions et les attributs qui caractérisent la relation existant entre les deux nœuds concernés. Nous ne conservons bien évidemment que les arcs entre les paires de nœuds qui vérifient certaines propriétés comme la connexité (images 2d) ou le recouvrement (images 3d).

L'implantation de cette structure de données doit essentiellement favoriser les deux traitements suivants :

- Accès rapide au meilleur arc du graphe; c'est-à-dire le meilleur couple de régions adjacentes qui vérifient le prédicat d'homogénéité *P*;
- Mise à jour facile des propriétés locales (attributs) des nœuds et des arcs modifiés par la fusion.

2.4.2.2 Accès rapide au meilleur arc

L'accès facile au meilleur arc du graphe impose une gestion triée de ces arcs. Nous utilisons un arbre binaire de recherche [45], pour stocker les couples de régions proposés à la fusion : nous plaçons chaque pointeur d'arc liant deux régions qui vérifient le prédicat d'homogénéité P dans un arbre. Cet arbre stocke les arcs triés en fonction des valeurs de la qualité locale de fusion *Q* calculée sur les couples de nœuds concernés. A la feuille d'une branche située d'un côté de l'arbre se trouve le pointeur sur le couple de régions à fusionner. L'extraction de cette valeur se fait en *0(log(n))*. Remarquons que nous ne nous somme pas préoccupés de l'équilibrage de l'arbre, les nœuds qui servent à la construction de l'arbre sont insérés selon un ordre quelconque. Nous espérons qu'en moyenne l'arbre soit proche de l'équilibre.

Lors de la fusion de deux nœuds *N1* et *N2*, nous créons un nouveaux nœud *N3* représentant l'union des deux anciens nœuds et autant de nouveaux arcs que de régions adjacentes aux deux régions *N1* et *N2*. Chaque nouvel arc, liant deux nœuds et vérifiant le prédicat d'homogénéité *P*, est ajouté dans l'arbre binaire de recherche en fonction de la qualité de fusion *Q* des deux régions concernées.

Dans ce cas, la recherche d'un arc de coût minimal dans l'arbre se poursuit jusqu'à trouver un arc entre deux nœuds qui existent encore.

2.4.2.3 Mise à jour facile des attributs du graphe image

La mise à jour des attributs des nœuds et des arcs, pour être rapide ne doit évaluer les nouveaux attributs qu'en fonction des anciens et éviter ainsi tout retour coûteux sur les données de l'image initiale. Il faut également que le prédicat d'homogénéité P et la fonction de qualité locale de fusion Q d'un couple de régions puissent être calculée directement en fonction des attributs des deux régions concernées. Ceci se traduit par les conditions (1) et (2) suivantes :

Posons :

S : Partition de l'image

R_i *et* R_j : Deux régions de la partition,

P : Prédicat d'homogénéité,

Q : Qualité de la fonction locale de fusion,

N : Vecteur d'attributs d'une région (nœud du graphe d'adjacence),

A : Vecteur d'attributs d'une relation entre deux régions (arc du graphe d'adjacence).

(1) $\forall R_i,\ R_j \in S$,

Il existe deux fonctions F_p et F_q d'évaluation du prédicat d'homogénéité P et de la qualité locale Q des deux régions, telles que :

$$P(R_i \cup R_j) = F_p(N(R_i).N(R_j), A(R_i, R_j)) \qquad (2.6)$$

$$Q(R_i \cup R_j) = F_q(N(R_i).N(R_j), A(R_i, R_j)) \qquad (2.7)$$

(2) $\forall\ R_i, R_j \in S.$

Il existe deux fonctions F_n et F_a de mise à jour des attributs des nœuds et des arcs, telle que :

$$N(R_i \cup R_j) = F_n(N(R_i), N(R_j)) \qquad (2.8)$$

$$A(R_i, R_j \cup R_k) = F_a(A(R_i, R_j), A(R_i, R_k)) \qquad (2.9)$$

Des détails supplémentaires peuvent être trouvés dans [46].

L'algorithme prend alors la structure suivante :

données :

F_p et F_q : Fonctions d'évaluation des prédicats et de la qualité locale,

F_n et F_a : Fonctions de mise à jour des attributs des nœuds et des arcs du graphe d'adjacence,

P : Prédicat d'homogénéité;

Q : Fonction de qualité locale.

- Construire le graphe image d'adjacence.

- Mettre dans l'arbre binaire de recherche tous les couples régions adjacentes (R_i, R_j) vérifiant $P(R_i, Rj)$ en fonction de la qualité de fusion $Q(R_i, R_j)$.

- Tant que arbre non vide faire
  - Rechercher un arc de coût minimal dans l'arbre jusqu'à trouver un arc entre deux nœuds qui existent encore.
  - Effectuer la fusion par une mise-à-jour des nœuds et des arcs à l'aide des deux fonctions F_n et F_a.

Le résultat de la segmentation se trouve dans le graphe image d'adjacence.

2.4.3. Complexité de l'algorithme

Etudions la complexité de l'algorithme proposé :

Soit *N* le nombre de nœuds du graphe initial. Soit A le nombre d'arcs du graphe initial. Chaque nœud du graphe est connecté en moyenne à $2A/N$ nœuds. Posons $V = 2A/N$. Cette valeur reste sensiblement identique pendant toute la durée du processus de segmentation. En effet, d'après la relation d'Euler, nous avons la relation suivante pour un graphe connexe :

$$N + F - A = 2 \qquad (2.10)$$

avec:

N : Nombre de sommets (nombre de nœud dans notre cas),

A : Nombre d'arcs,

F : Nombre de facettes (une facette est définie par les arcs qui la délimitent).

Cette formule est également vraie dans le cas particulier d'un graphe planaire, comme celui avec lequel nous travaillons. Or,

- une facette est délimitée par un ensemble d'arcs, une facette est donc limitée par au moins 3 arcs.

- chaque arc délimite au plus 2 facettes;

Nous pouvons donc majorer le nombre de facettes par :

$$F \leq \frac{A \times \boldsymbol{2}}{\boldsymbol{3}} \tag{2.11}$$

De (2.10) et (2.11), nous déduisons un majorant du nombre d'ars :

$$A \leq \boldsymbol{3} \times (N - \boldsymbol{2}) \tag{2.12}$$

Ceci signifie que le nombre d'arcs dans un graphe est toujours inférieur à 3 fois le nombre de facettes :

$$\frac{A}{N} \leq \boldsymbol{3} \tag{2.13}$$

de plus :

$$\frac{N}{2} \leq A \tag{2.14}$$

de (2.13) et (2.14), nous concluons que :

$$\boldsymbol{0{,}5} \leq \frac{A}{N} \leq \boldsymbol{3} \tag{2.15}$$

d'où :

$$\boldsymbol{1} \leq \frac{\boldsymbol{2}A}{N} \leq \boldsymbol{6} \tag{2.16}$$

$$\boldsymbol{1} \leq V \leq \boldsymbol{6}$$

La valeur *V* peut varier dans un intervalle très restreint. Le nombre moyen de nœuds connectés à un autre nœud reste donc sensiblement constant pendant la segmentation comme le montre le tableau de valeurs présenté à la fin de ce paragraphe.

Pour cet exemple *V* varie entre 5.153 et 5.430.

Soit F le nombre de fusions réalisées pendant tout le traitement.

Soit T_n le nombre d'opérations effectuées par la fonction F_n de mises à jour des attributs des nœuds après une fusion.

Soit T_a le nombre d'opérations effectuées par la fonction T_a de mise à jour des attributs des arcs après une fusion.

Soit T_x le nombre d'opérations effectuées pour l'évaluation d'un prédicat d'homogénéité et de la fonction de qualité locale entre deux régions.

Soit M le nombre de fusions potentielles de chaque nœud avec l'un de ses voisins qui vérifient le prédicat d'homogénéité.

- Le nombre d'opérations nécessaire pour mettre à jour les attributs des nœuds et des arcs dans le graphe image est de :

$$F \times \left(T_n + T_a \times V\right) \qquad (2.17)$$

- Après la fusion de deux nœuds, il faut calculer V nouvelles valeurs du prédicat d'homogénéité puisque ce nouveau nœud est connecté à une moyenne de V autres nœuds, le nombre d'opérations demandé est de :

$$F \times \left(V \times T_x\right) \qquad (2.18)$$

- Seuls M arcs parmi V nouveaux arcs vérifient le prédicat P et doivent donc être insérés dans l'arbre binaire de recherche, le nombre d'opérations est de :

$$F \times \left(M \times log\left(M \times N\right)\right) \qquad (2.19)$$

Puisque le nombre maximal d'éléments dans cette structure est de: $M \times N$.

- La complexité de tout l'algorithme s'écrit alors :

$$O(F \times (T_n + V \times T_a + V \times T_x + M \times log(M \times N))) \qquad (2.20)$$

Le nombre F de fusions est directement lié au nombre de régions à traiter, en revanche, les autres coefficients intervenant dans le calcul de la complexité dépendent seulement de la "densité d'arc" de ce graphe. Il est donc très coûteux d'appliquer cet algorithme sur l'image originale, le graphe est alors composé de tous les points de l'image comme nœuds et de tous les points en relation comme arcs.

L'idée est alors d'effectuer une partition initiale de l'image avant d'utiliser notre algorithme de regroupement.

Nous présentons ci-après le tableau de valeurs illustrant le fait que chaque région de la partition a un nombre quasi constant de régions voisines, compris entre 5 et 6 pour notre exemple. Les étapes de l'algorithme sont arbitraires, elles correspondent à différents arrêts lors du déroulement du processus et sont déterminées aléatoirement.

Tableau 2.1. : Nombre moyen de régions voisines par région.

Image de la lampe présentée figure 2.4 De taille 256 × 256 points			
Etapes du processus de segmentation	Nombre de régions (nœuds) N	Nombre de relation de voisinage (arcs) A	Nombre moyen de régions voisines par région V= 2A/N
1	6259	16128	5.153
2	5622	14584	5.188
3	4781	12483	5.220
4	3965	10368	5.229
5	3191	8579	5.210
6	2772	7181	5.181
7	2643	6819	5.160
8	393	1074	5.460
9	295	803	5.440
10	288	782	5.430

2.4.4. Partition initiale

Cette première partition est réalisée par une technique classique de division de régions: la procédure SPLIT de Pavlidis [43]. La méthode consiste à diviser récursivement l'image en quatre parties égales jusqu'à ce que chaque zone obtenue vérifie un prédicat d'homogénéité (figure 2.3). Pour implanter ceci, nous utilisons une description de l'image sous forme de "quad-tree", aussi appelé arbre quaternaire.

Le prédicat utilisé peut être identique à celui utilisé dans l'algorithme de fusion, mais en diminuant sa tolérance. Ceci permet d'obtenir des régions petites mais très homogènes, et de réduire le volume des données d'environ 75%, ce qui est nettement appréciable.

Par exemple, le prédicat d'homogénéité P suivant peut être utilisé pour effectuer une partition initiale :

$$P(R_i) = [Max_i - Min_i < seuil] \quad (2.21)$$

avec :

R_i : Région traitée;

Max_i, respectivement Min_i : niveau de gris maximum, respectivement minimum, des points de la région R_i;

Seuil: Seuil accepté pour qualifier l'homogénéité des régions.

Pour une valeur de seuil de 30 sur une image 256 x 256 de niveaux de gris variant de 0 à 255 (figure 2.4), nous obtenons les résultats présentés dans la figure 2.6.

Une première version de l'algorithme de création de la partition initiale a été de construire les régions par agglomération de points au fur et à mesure du balayage séquentiel de l'image [47]. Mais il s'est avéré que le sens de parcours des données avait une influence non négligeable sur la forme des régions créées, qui se retrouvait ensuite dans la partition finale. L'image de la figure 2.5 illustre la déformation des régions de la partition initiale, le balayage de l'image étant effectué de la gauche vers la droite et haut en bas. Cette technique de création de la partition initiale a donc été abandonnée au profit de la méthode de division présentée ci avant.

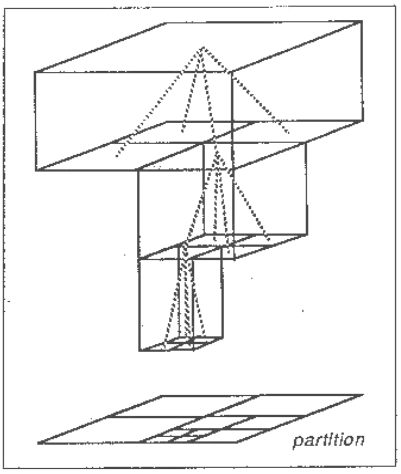


Figure 2.3.: Division récursive de régions suivant un critère d'homogénéité [34].

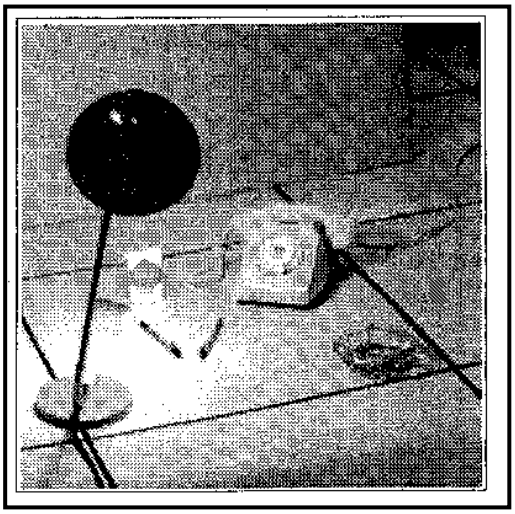


Figure 2.4.: Image originale. 65536 points de 256 niveaux de gris [34].

2.4.5. Résumé

Nous venons de définir une stratégie de segmentation par:

- Un algorithme itératif de fusions de régions qui optimise, localement au fur et à mesure de son déroulement, la qualité globale de la segmentation.

- Un graphe image d'adjacence valué et deux fonctions de mise à jour des attributs des nœuds et des arcs de ce graphe :

- Des structures de données adaptées aux traitements effectués.

L'outil algorithmique étant défini, nous pouvons passer à l'étude de la segmentation d'images de scènes naturelles et définir le (ou les) prédicats(s) d'homogénéité à utiliser qui, eux, dépendent des caractéristiques de l'image traitée.

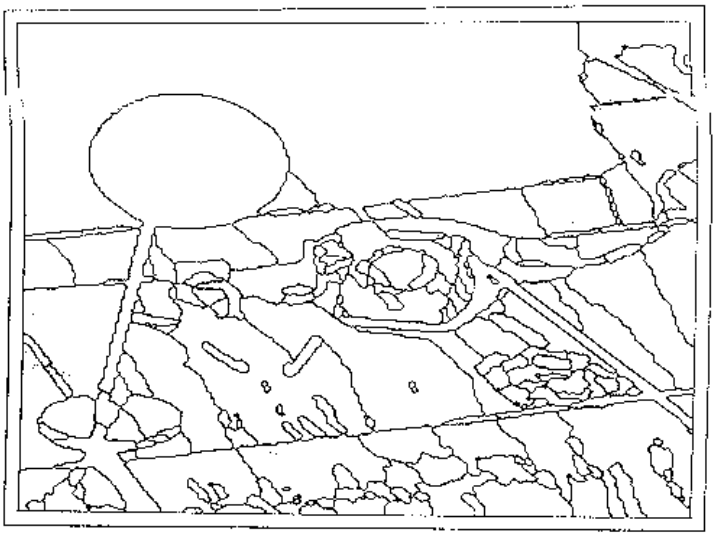


Figure 2.5. : Partition initiale obtenue par balayage séquentiel de l'image. Le sens de traitement des données influe sur la forme des régions [34].

2.5. Prédicats d'homogénéité

L'homogénéité des régions dépend de la nature de l'image traitée. Les régions extraites d'une image satellite ne peuvent pas posséder les mêmes propriétés d'homogénéité que les régions extraites d'image de scènes d'intérieur.

L'objectif de la segmentation est de déterminer la projection des différents objets présents dans la scène observée. Or une image naturelle dépend des diverses caractéristiques de la scène, telle que l'orientation et les matériaux des objets, l'éclairage, l'angle de prise de vue de la caméra, etc. La conjonction de tous ces paramètres fournit une image ne contenant qu'un seul type d'information : le **pixel**. C'est-à-dire l'association d'une intensité lumineuse à une position dans l'image.

Il est évidemment plutôt difficile de séparer l'influence de ces différents facteurs afin de trouver les projections des objets.

La qualité de la segmentation est maximale si chaque région correspond à un objet et réciproquement. Pour arriver à une telle segmentation, il faut connaître précisément les propriétés d'homogénéité des régions ce qui n'est généralement pas le cas ; il est en effet difficile de trouver une caractérisation de l'homogénéité globale et adaptée à toutes les régions.

Nous montrons ci-après que l'utilisation d'un prédicat d'homogénéité simple est insuffisante pour obtenir une bonne segmentation. Il s'avère nécessaire :

- Soit d'utiliser un prédicat plus complexe, combinaison de prédicats ¨élémentaire:
- Soit, plus simplement, d'utiliser plusieurs prédicats ¨élémentaires¨ de manière séquentielle.

2.5.1. Utilisation d'un prédicat d'homogénéité

Il semble assez naturel de travailler avec les moyennes de niveaux de gris des régions, par exemple, et de fusionner des régions adjacentes de moyennes quasi identiques. Le prédicat d'homogénéité s'écrit alors comme suit :

$$P(R_i \cup R_j) = \left[\left| Moy_i - Moy_j \right| \prec Seuil \right] \qquad (2.22)$$

avec : R_i et R_j : les deux régions qui sont proposées au fusionnement si le prédicat est vrai,

Moy_i et Moy_j : Moyennes respectives des niveaux de gris des points des régions R_i et R_j,

*S*euil : Seuil de fusionnement.

Seuil est à déterminer, expérimentalement, aucune valeur n'est satisfaisante. Prenons par exemple Seuil = 10 pour une image où les niveaux de gris des points varient entre 0 et 255 ; la partition résultat est sur segmentée (figure 2.6).

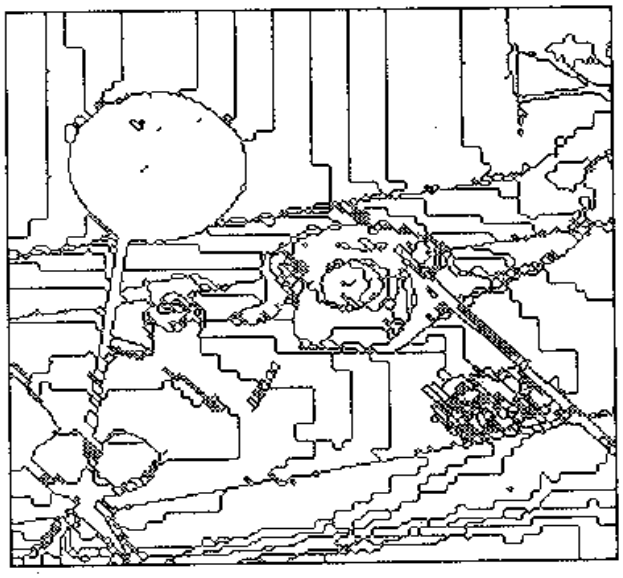


Figure 2.6.: Segmentation finale. Seuil = 10 (3578 régions) [34].

L'augmentation de la valeur de seuil. Pour s'affranchir des problèmes de sur-segmentation essentiellement dus aux dégradés de lumière, provoque des fusions intempestives de régions adjacentes appartenant à des objets différents. Ceci est illustré (figures 2.7 et 2.8) par la fusion des régions correspondantes au téléphone et au support sur lequel il est posé.

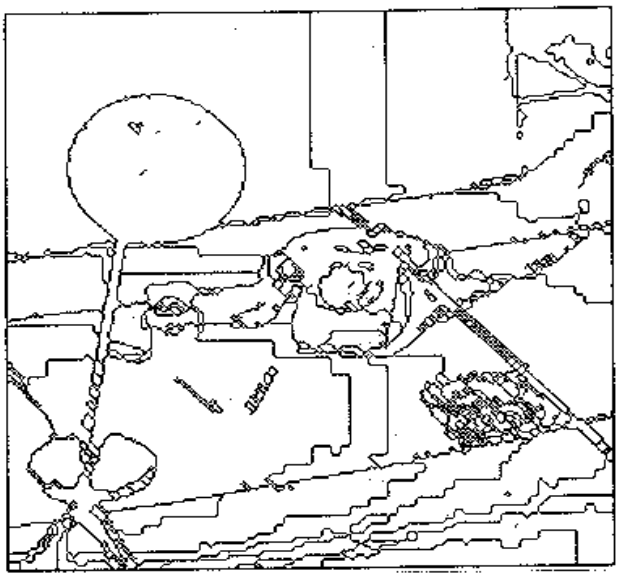


Figure 2.7. : Segmentation finale, seuil = 15 (2786 régions) [34].

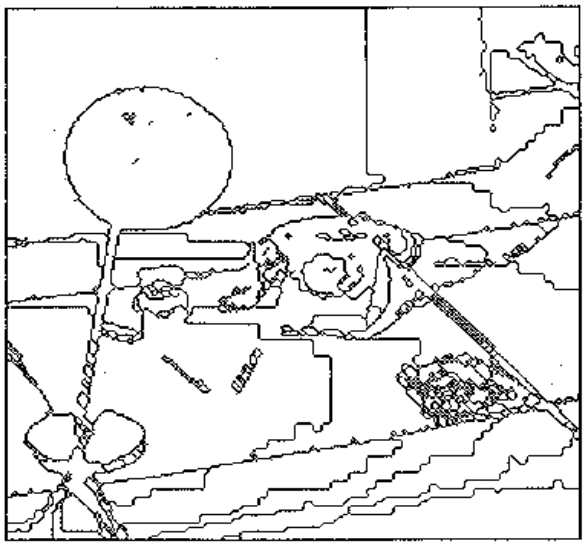


Figure 2.8. : Segmentation finale, seuil = 20 (2162 régions) [34].

La conclusion de ces essais est qu'il n'est pas suffisant de caractériser l'homogénéité de régions par la moyenne des niveaux de gris de leurs points.

L'utilisation isolée d'autres prédicats aboutit aux mêmes constatations. Lorsque la scène étudiée est aussi complexe que celle présentée.

5.2. Utilisation de plusieurs prédicats

La solution que nous proposons pour obtenir de meilleurs résultats est d'utiliser de façon optimale ***plusieurs propriétés d'homogénéité***, ces propriétés devant être décomposées de manière hiérarchique. A chacune de ces propriétés est associé un prédicat, chaque prédicat est utilisé séquentiellement par l'algorithme sous-optimal décrit précédemment. Cet algorithme prend alors la structure suivante :

données :

$(P_1,Q_1),.....,(P_n,Q_n)$, séquence de prédicats et fonction de qualité locale associée.

- Construire la partition initiale (généralement à l'aide du prédicat P_1).
- **Pour** chacune de ces n paires (P_k,Q_k) **faire**

- Croissance (P_k,Q_k)

Dans le cas de la segmentation d'images naturelles à niveaux de gris, les propriétés souhaitées pour chaque région sont les suivantes :

(a) - La moyenne des gradients calculée sur les points d'une région est faible, c'est-à-dire qu'aucun contour ne traverse la région.

(b) - Une région peut être décomposée en petites zones de moyennes quasi-identiques.

(c) - Chacune de ces zones peut être décomposée en zones de points de niveaux de gris quasi-identiques.

Ces trois propriétés servent à définir les trois prédicats d'homogénéité; il est évident que le premier prédicat P_1 de la séquence travaille au niveau de l'image initiale, le second prédicat P_2 travaille sur les régions obtenues à l'étape précédente par le prédicat P_1, etc.

Soit I une image *I, (k ,l)* un point de cette image, et $S=\{R_1,.....R_n\}$, *une partition de* I *, les trois prédicats issus des trois propriétés sont les suivants :*

(a) - Prédicat P_1 :

$$P_1(R_i \cup R_j) = \lfloor Max_{ij} - Min_{ij} \prec seuil_1 \rfloor \qquad (2.23)$$

avec : Max_{ij}, respectivement Min_{ij}: niveau de gris maximum, respectivement minimum, des points de la "nouvelle" région $R_i \cup R_j$:

Seuil$_1$: Seuil accepté pour l'homogénéité des régions.

- La fonction de qualité globale à optimiser s'écrit sous la forme :

$$C_1(S) = \sum_{i=1}^{n} \sum_{k,l \in R_i} \left((I(k,l) - Max_i)^2 - (I(k,l) - Min_i)^2 \right) \quad (2.24)$$

- La fonction de qualité locale s'écrit:

$$Q_1(R_i \cup R_j) = Max_{ij} - Min_{ij} \quad (2.25)$$

(b) - Prédicat P_2 :

$$P_2(R_i \cup R_j) = \left[\left| Moy_i - Moy_j \right| \prec seuil_2 \right] \quad (2.26)$$

avec : R_i et R_j : les régions candidates à une fusion,

Moy_i et Moy_j : moyennes respectives des niveau de gris des points des régions R_i et R_j ;

Seuil$_2$: Seuil de fusion.

- La fonction de qualité globale à optimiser s'écrit sous la forme :

$$C_2(S) = \sum_{k,l \in I} (I(k,l) - M_{kl})^2 \quad (2.27)$$

Où M_{kl} est la moyenne des niveaux de gris des points de la région à laquelle appartient le point (k,l).

- La fonction de qualité locale s'écrit :

$$Q_2(R_i \cup R_j) = \left| Moy_i - Moy_j \right| \quad (2.28)$$

(c) - Prédicat P_3 :

$$P_3(R_i \cup R_j) = \left[\frac{\sum_{(i,j),(k,l) \in (R_i,R_j)} |I(i,j) - I(k,l)|}{Card(F(R_i,R_j))} \prec Seuil_3 \right] \quad (2.29)$$

Avec : $F(R_i, R_j)$: l'ensemble des couples de points situés le long de la frontière entre les régions R_i et R_j (figure 2.10),

$Card(F(R_i,R_j))$: Longueur de la frontière entre R_i et R_j,

Seuil$_3$: Seuil de fusion.

- La fonction de qualité globale s'écrit sous la forme :

$$C_3(S) = \sum_{(i,j),(k,l)\in X_s(I)} I(i,j) - I(k,l) \qquad (2.30)$$

avec X_S : l'ensemble des couples de points connexes de I appartenant à une même région, et ce, pour toutes les régions de la partition S.

- La fonction de qualité locale s'écrit :

$$Q_3(R_i \cup R_j) = \frac{\sum_{(i,j),(k,l)\in(R_i,R_j)} |I(i,j) - I(k,l)|}{Card(F(R_i,R_j))} \qquad (2.31)$$

Nous donnons figure 2.9, un exemple de résultat obtenu avec l'utilisation séquentielle de ces trois prédicats d'homogénéité. Les améliorations apportées à la segmentation sont évidentes. Une adaptation plus fine des différents seuils utilisés permet d'obtenir une meilleure partition, mais puisqu'une détection automatique de tels paramètres en fonction de l'image traitée n'est pas réalisée, nous préférons travailler avec un jeu de valeurs identiques pour tous les types d'images traités.

L'utilisation optimale d'une suite de critères de regroupement permet de mieux utiliser les propriétés d'homogénéité des régions que l'on recherche. Ces critères dépendent des caractéristiques des images traitées et doivent être redéfinis pour chaque type d'application.

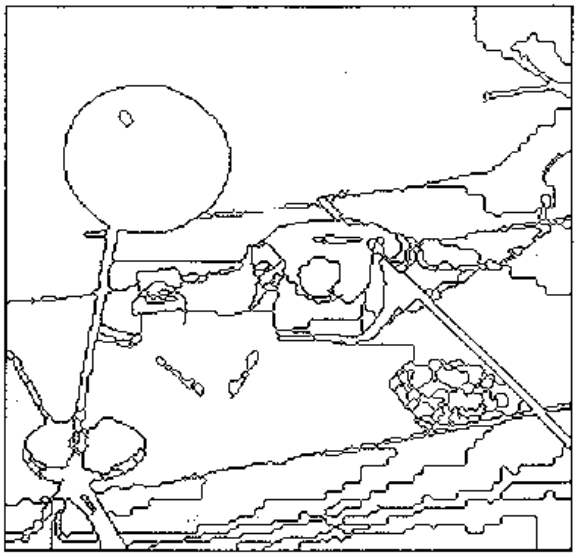


Figure 2.9. : Partition obtenue par l'utilisation optimale de trois prédicats d'homogénéité successifs (162 régions) [34].

2.6. Coopération régions-contours

L'idée implicite est d'utiliser conjointement les caractéristiques d'homogénéité des régions et les discontinuités locales (points de contours).

Avant de poursuivre, notons que nous préférons utiliser le terme "**point de contraste**" plutôt que "**point de contour**" qui est pourtant plus fréquemment utilisé dans la littérature. Cependant, il faut remarquer que le terme point de contraste est plus adapté car il se réfère seulement aux propriétés de l'image et non aux formes qui peuvent être présentes dans la scène [48]. Par exemple, les points de contraste dû à la texture d'une surface ne correspondent à aucun contour d'objet.

Les points de contraste, préalablement calculés, sont utilisés pour guider le processus de segmentation; ils contrôlent la croissance des régions comme l'illustre la figure 2.10.

Ce contrôle de la croissance des régions est effectué en interdisant toute fusion entre régions adjacentes pour lesquelles la proportion de points de contraste situés sur leur frontière commune dépasse un certain seuil. Ce seuil est fixé à 0 pour tous les prédicats sauf celui évaluant les "forces" des frontières (prédicat P_3). Les trois prédicats précédemment cités sont donc modifiées de la façon suivante:

$$P_1(R_i) = [Max_i - Min_i < seuil_1] \wedge [\, Ct(R_i, R_j) = 0 \,] \qquad (2.32)$$

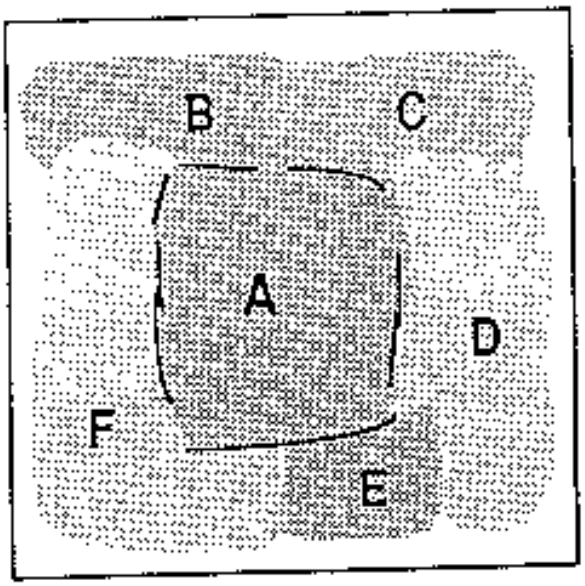


Figure 2.10.: Superposition de la carte de points de contraste et de la partition en régions en cours de création. La région A est suffisamment entourée de points de contraste, sa croissance est arrêtée [34].

$$P_2(R_i, R_j) = \left[\left|Moy_i - Moy_i\right| \prec Seuil_2\right] \wedge \left[Ct(R_i, R_j) = 0\right] \quad (2.33)$$

$$P_3(R_i \cup R_j) = \begin{cases} \left[\dfrac{\sum_{(i,j),(k,l)\in F(R_i,R_j)} |I(i,j) - I(k,l)|}{Card(F(R_i,R_j))} \prec Seuil_3\right] & si \left(\dfrac{Ct(R_i,R_j)}{Card(F(R_i,R_j))} \prec Seuil_4\right) \\ faux & si\ non \end{cases} \quad (2.34)$$

Pour refuser toute fusion comportant une proportion trop importante de points de contraste sur la frontière commune des deux régions concernées.

avec $Ct(R_i, R_j)$: nombre de points de contraste sur la frontière entre les deux régions R_i et R_j .

Cette "coopération" régions-contours permet d'utiliser des seuils plus tolérants lors de l'évaluation des prédicats d'homogénéité, donc d'autoriser plus de fusions fondées sur une comparaison des attributs des régions, sans dégrader la partition obtenue. Notre algorithme apparaît comme un moyen très naturel de concilier l'information contours et l'information régions [49]. Ceci est utile en particulier dans les zones de l'image où aucune de ces deux informations prise séparément ne permet de déterminer correctement les contours des objets.

Dans ce cas, assez fréquent pour les images de type scène d'intérieur, la projection d'un objet est entourée de lignes de contraste brisées et voisine de régions homogènes de mêmes caractéristiques.

La hiérarchie du processus de regroupement permet la formation de la région correspondant à la projection de l'objet. Cette région n'est plus fusionnée, car elle est entourée de points de contraste.

Notre méthode peut être résumée de la façon suivante: nous effectuons une première segmentation (extraction de points de contraste) avec des critères stables; puis une seconde segmentation où les critères sont plus tolérants car portant sur des valeurs peu stables.

Les régions grandissant, soit jusqu'à buter sur les lignes de contraste, soit jusqu'à atteindre le seuil maximal définissent leur homogénéité. De cette manière, la localisation des frontières des régions est correcte; ces frontières sont généralement situées sur les lignes de contraste pré calculées.

Cette approche nous permet de tirer profit de la richesse des informations contenues dans les deux types de primitives, et également de leur dualité.

Nous approche nous permet de tirer profit de la richesse des informations contenues dans les deux types de primitives, et également de leur dualité.

Nous présentons ci-après une carte de contraste (figure 2.11) et la segmentation obtenue (figure 2.12) avec les prédicats indiqués précédemment. L'extraction des points de contraste est effectuée selon la méthode développée par R. Deriche à l'Inria- Rocquencourt [50]. Son implantation dans notre laboratoire a été réalisée par D. Ziou. Nous en utilisons donc ici seulement les résultats.

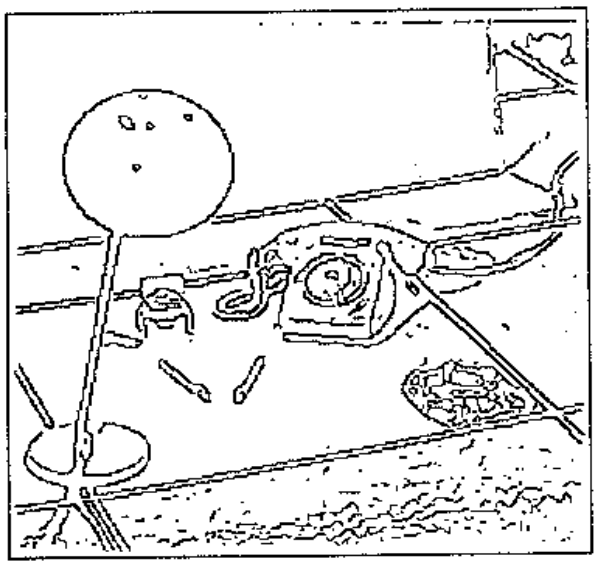


Figure 2.11. : Extraction des points de contraste par la méthode de Deriche [34, 47].

Remarquons que n'importe quelle carte de points de contraste définissant des points d'arrêt peut, en fait, être utilisée.

2.7. Conclusion

Le choix des primitives à apparier entre les deux images stéréoscopiques s'étant porté sur les régions, nous avons été confrontés au problème de leur extraction. Les nombreuses techniques de segmentation d'images existantes ne nous ont pas semblé satisfaisantes dans la mesure où elles dépendent trop souvent du domaine d'application considéré, mais nous nous en sommes cependant fortement inspirés pour construire notre approche.

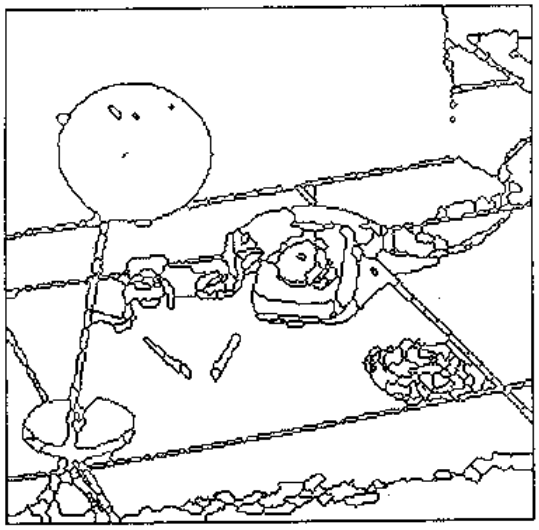


Figure 2.12. : Résultat de la segmentation où la croissance de régions est contrôlée par une carte de points de contraste pré calculée [34].

2.7.1.Discussion de notre approche

La méthode développée possède quatre points en sa faveur:

- Elle sépare l'algorithme de la définition des prédicats d'homogénéité qui caractérisent les régions recherchées. Cette séparation confère à notre algorithme sa généralité.

- Aucun sens d'examen des données n'est défini a priori, l'algorithme est essentiellement guidé par les données.

- L'algorithme optimise au fur et à mesure de son déroulement la qualité globale de la partition. Ceci nous permet d'obtenir une segmentation que nous espérons proche de l'optimum réel.

- L'utilisation de plusieurs prédicats d'homogénéité autorise une meilleure définition des caractéristiques des régions. Ils sont utilisés de manière séquentielle des caractéristiques des régions. Ils sont utilisés de manière séquentielle par l'algorithme sous optimal.

Cependant, avec l'approche que nous avons déterminée, nous ne proposons aucun moyen pour définir les seuils apparaissant dans les prédicats. Les seuils utilisés Prennent des valeurs trouvées empiriquement et qui sont employées pour toutes les images de type scène d'intérieur. La détermination automatique des paramètres est un objectif que nous avons tenté d'atteindre, sans toutefois aboutir à une solution satisfaisante. Cette phase de test d'adaptation des seuils correspond au premier inconvénient de notre méthode.

Le deuxième inconvénient est de déterminer les prédicats à utiliser selon le type d'images traitées : nous ne proposons pas de manière pour les construire, ce travail est laissé au soin de l'utilisateur. Il faut donc décrire l'image sous forme d'une suite de propriétés que les régions doivent vérifier. De là sont déduits les prédicats d'homogénéité. Leur utilisation séquentielle dépend de la façon dont les propriétés sont "emboîtées" .les propriétés peuvent être de toutes catégories, comme par exemple de type sémantique : quand une région a une signification sémantique, elle est reconnue et sa croissance peut être arrêtée.

Le travail qui reste à faire pour l'utilisateur se résume de la façon suivante :

- Décrire la suite des propriétés que doivent vérifier les régions à extraire;
- Définir la suite des prédicats d'homogénéité et les fonctions locales de qualité à optimiser;
- Déterminer les différents seuils utilisés dans les prédicats.

Remarquons que les prédicats d'homogénéité peuvent se combiner pour ne former qu'un seul prédicat contrôlant la fusion des régions. Ceci pose cependant les problèmes de déterminer la fonction de combinaison des prédicats et de trouver leurs coefficients de pondération, et risque également d'alourdir le traitement par des calculs inutiles.

L'idée d'utiliser plusieurs critères de regroupement apparaît aussi dans le travail de F. Zaroli [51], qui propose une approche liant à la fois les techniques de classification et de fusion. Cet outil a été créé pour un module d'interprétation de scène et utilise plusieurs images de la scène étudiée : images à niveaux de gris, images de distance, etc.

2.7.2. Applications

Pour illustrer la généralité de cette approche, citons son utilisation pour la segmentation d'images 3d sismiques [52] où dans le domaine du mouvement.

Dans ce domaine, la segmentation représente une étape intermédiaire importante lors du processus de traitement de séquences d'images et d'analyse de scène dynamique, et l'un des objectifs est d'obtenir une segmentation selon des critères de mouvement. L'outil algorithmique est identique. Les prédicats d'homogénéité au sens du mouvement représentent des modèles de vitesse.

Deux prédicats sont actuellement utilisés, le premier pour obtenir une partition en régions de mouvement constant, le second pour déterminer les régions de mouvement linéaire [53]. Des prédicats qualifiant des mouvements plus complexes peuvent ensuite être utilisés.

2.7.3 Utilisation de l'information ¨contour¨

Nous avons également montré comment réaliser la ¨coopération¨ entre un détecteur- contour et un détecteur région : l'utilisation conjointe de deux informations duales permet d'obtenir de meilleures partitions. Une carte de points de contraste pré calculée permet de contrôler la croissance des régions ; ceci joue le rôle d'une adaptation locale des prédicats d'homogénéité. En effet, les fusions de régions se font plus facilement dans les zones de l'image ou il n'y a que peu de points de contraste, donc représentant certainement une seule surface, même si les caractéristiques locales des régions sont assez différentes. Ce cas est illustré par les régions de la partition initiale situées sur une surface ou la lumière est dégradée.

L'inconvénient de la segmentation en régions ¨classique ¨est ici corrigé : les frontières des régions sont bien localisées car s'appuyant sur des lignes de contraste fiables, de plus l'homogénéité des régions, généralement assez délicate à définir, n'est plus la seule caractéristique prise en compte.

2.7.4 Application à la stéréovision

Nous donnons sur la Figure 2.13, le schéma récapitulatif des traitements à effectuer sur chaque image du couple stéréoscopique et les Figures 2.15 et 2.17, les résultats obtenus sur les images originales présentées dans les Figures 2.14 et 2.16.

Il faut cependant mentionner que, si la segmentation de chaque image d'une paire stéréoscopique est satisfaisante, il subsiste des différences, quelquefois très

importantes, entre les deux partitions à apparier comme nous pouvons le constater sur les images présentées. Ces différences sont essentiellement dues à l'écart entre les angles d'observation de la scène.

Notre idée est de tenir compte de ces différences entre les partitions lors de l'appariement. Le processus d'appariement accepte en données les deux partitions et cette redondance d'information permet de jouer le rôle, dans certaines parties de l'image et lorsque le besoin s'en fait sentir, d'un processus de segmentation. L'objectif est de remettre en cause certaines zones de la partition pour améliorer la qualité de correspondance entre les deux images. Ceci fait l'objet du chapitre suivant.

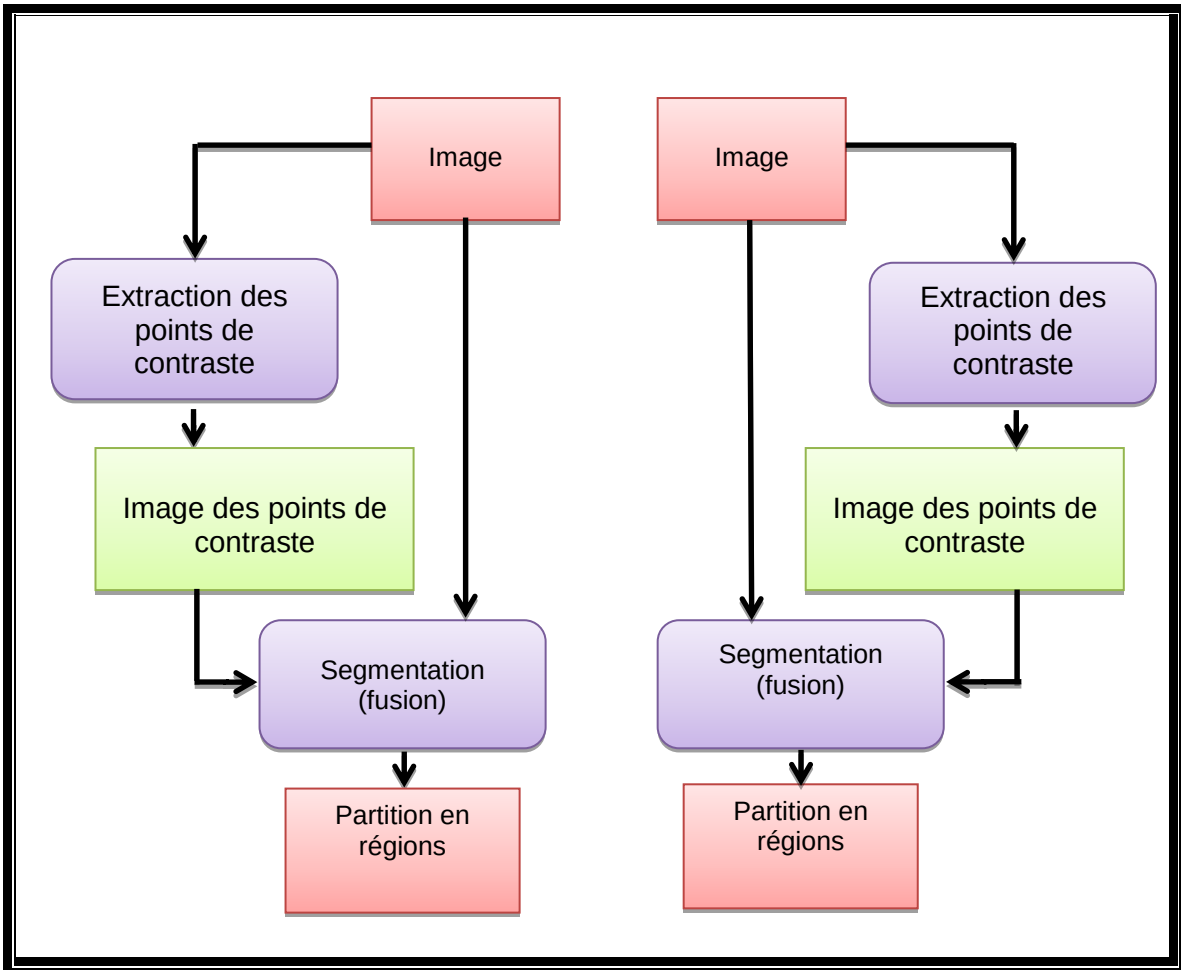


Figure 2.13. : Schéma récapitulatif des traitements pour la segmentation en régions des images stéréoscopiques [34].

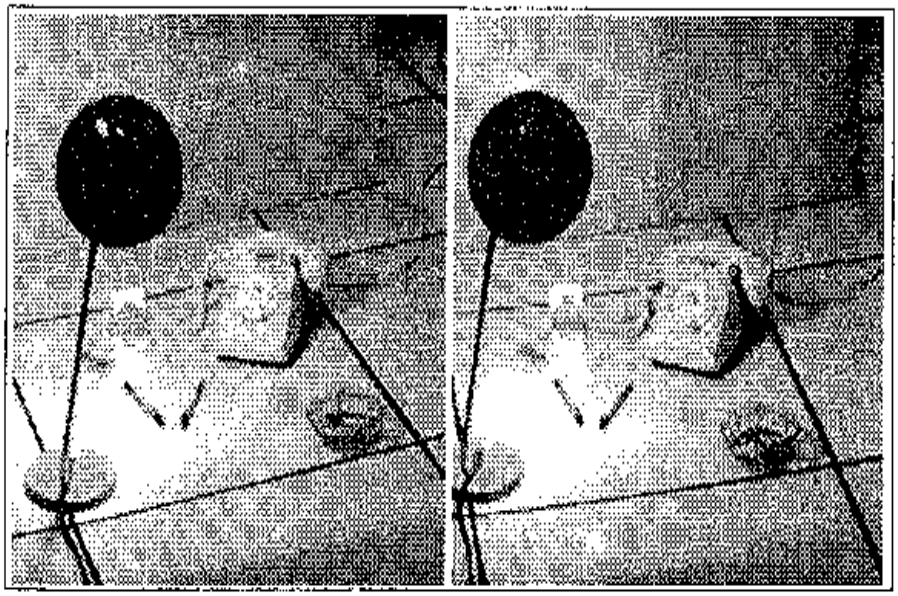


Figure 2.14. : Coupe stéréoscopique. Images initiales [34].

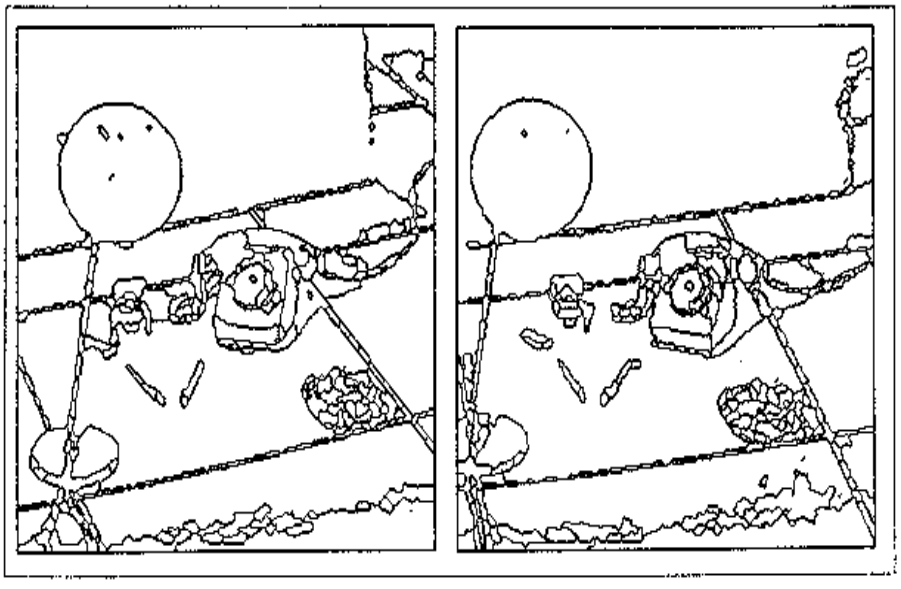


Figure 2.15. : Couple stéréoscopique. Segmentations finales [34].

Figure 2.16. : Coupe stéréoscopique. Images initiales [34].

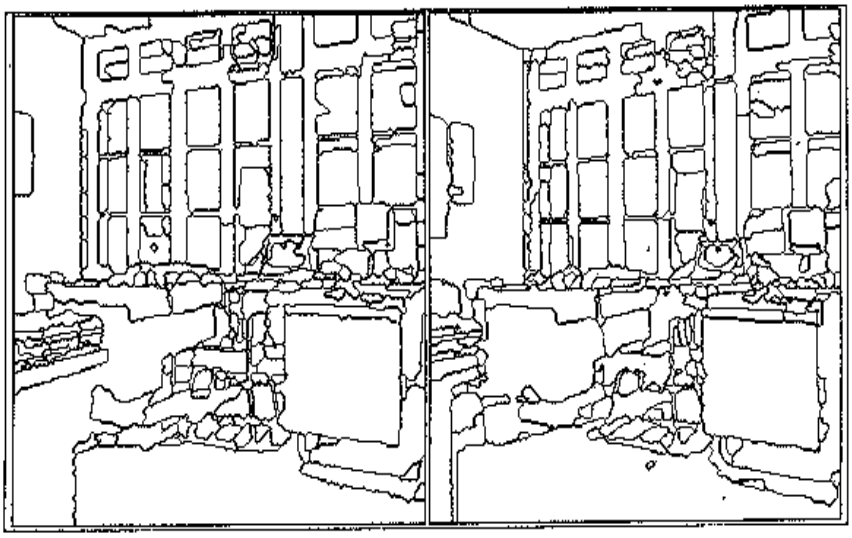


Figure 2.17. : Couple stéréoscopique. Segmentations finales [34].

CHAPITRE 3

Apport des réseaux de neurones et de la Transformée de Hough

L'objectif de ce chapitre est de présenter un état de l'art des méthodes d'estimation robustes utilisées en vision par ordinateur, avec une attention particulière aux applications robotiques.

Dans ce contexte particulier, les contraintes dues au temps de calcul doivent être prises en compte pour le choix des algorithmes d'estimation. Parmi les nombreuses techniques qui ont été proposées dans la littérature pour obtenir des estimations robustes, on peut citer, sans être exhaustif, la Transformée de Hough, RANSAC (Random Sample Consensus), les LMS (Least Median of Squares), les M-estimateurs, etc.

Dans cette partie de cette thèse, nous justifions le choix de l'hybridation de la Transformée associée aux RNA's.

3.1. Approche neuronale

3.1.1. Introduction

Les méthodes connexionnistes ne sont pas destinées exclusivement, loin de là, à être appliquées en reconnaissance des formes. Il faut signaler leur introduction en automatique (commande de processus), en traitement du signal (compression, prédiction, filtrage), en synthèse de la parole (NetTalk) et aussi dans certains domaines où les méthodes de l'intelligence artificielle sont prédominantes. Nous cantonnerons nos propos aux notions utiles de reconnaissance des formes.

Commençons par citer un extrait de la thèse de Yann Le Cun :

Ces dernières années ont vu l'apparition de nouvelles techniques regroupées sous le nom de « connexionnisme » dont le principe directeur est l'utilisation d'un réseau de “processeurs” très simples interconnectés, et dont la connaissance réside précisément dans les connexions. Cette idée est en fait très ancienne, pour ne pas dire plus ancienne que l'informatique. L'idée sous-jacente et assez présomptueuse étant de copier le cerveau. Ce dernier, qu'il soit humain ou animal est en effet le seul exemple de machine intelligente que nous connaissions. Il peut donc paraître naturel de s'en inspirer non seulement au niveau fonctionnel, mais aussi au niveau architectural. Il est bon de mettre en garde le lecteur en insistant sur le fait que les systèmes connexionnistes ne sont qu'un moyen de plus de résoudre certains problèmes dont la longue histoire des espoirs déçus de l'intelligence artificielle nous conduit à une prudence extrême quant aux résultats possibles d'une nouvelle technique, quelle ’’qu'elle soit’’. Notre but n'est pas de jouer au jeu des définitions mais il est important de bien cerner l'objet technique appelé réseau connexionniste.

Cet objet est entouré d'un rendant certes, sa diffusion plus facile vers un vaste public, mais conduisant à des excès d'enthousiasme que Yann Le Cun, de manière prémonitoire, dénonçait dès 1987. Une certaine presse se jette sur la moindre, le plus petit, comme jadis sur les produits de l'IA (intelligence artificielle).

Ainsi, aujourd’hui, ce n’est pas le paradigme neurobiologique qui contribue au développement des réseaux de neurones formels : au contraire, ce sont les réseaux de neurones formels qui contribuent, de plus en plus fréquemment, à la compréhension des systèmes neuronaux vivants, car ils constituent des outils précieux pour en construire des modèles, simples mais utiles. Peut-être cette situation changera-t-elle dans l’avenir : les progrès réalisés dans l’analyse des systèmes vivants pourraient conduire, à leur tour, à la conception de réseaux de neurones formels plus efficaces que ceux que nous décrivons aujourd’hui. Il y a là un champ de recherche fascinant, complètement ouvert. On rêve, jusque dans l'antre des grands décideurs (privés, publics, européens, nippons,…), de machines extraordinaires, nouvelles pierres philosophales de l'intellect, d'engins neuromimétiques bourrés de silicium, de circuits optoélectroniques ou même de composants moléculaires; inutile même de les programmer, ces machines apprennent quasiment seules, elles dépassent leurs maîtres si on leur fournit

quelques bons exemples de la tâche à accomplir, certaines fonctionnent même sans professeur.

Pour définir un réseau connexionniste spécifique, il y a trois éléments à fournir :

1) L'architecture : la manière dont les unités seront interconnectées et les contraintes liant les poids.

2) La dynamique : comment les activités des unités sont couplées entre elles, comment les signaux se propagent dans le réseau. Cette dynamique peut être déterministe ou bien probabiliste.

3) L'apprentissage : les règles qui tentent de déterminer les valeurs (optimales) des poids pour que le réseau accomplisse une certaine tâche. Cette tâche est caractérisée par une relation prédéfinie (en général par une liste d'exemples) entre les activités de certaines unités.

Pour rendre moins abstraites ces définitions, anticipons un peu sur l'algorithme du perceptron de Rosenblatt et analysons-le en termes d'architecture, de dynamique et d'apprentissage. L'architecture comprendra deux couches de cellules. La première couche appelée entrée comprendra autant de cellules que de composantes au vecteur X (qui code la forme). La seconde couche appelée sortie comprendra une seule cellule. Toutes les cellules d'entrée sont connectées à la cellule de sortie par un arc dont le poids sera la composante du vecteur A. La dynamique sera de type feed-forward (propagation vers l'avant), la cellule de sortie aura pour activité une combinaison linéaire des activités des cellules d'entrée:

$$x_s = \sum_{i=1}^{N+1} a_i x_i \qquad (3.1)$$

où x_i est l'activité de la cellule d'entrée n° i et aᵢ le poids associé à la connexion entre la cellule d'entrée n° i et la cellule de sortie.

La sortie du réseau ne sera pas x_S mais Sgn(x_S) où Sgn est la fonction signe. On aura donc, en notant O la sortie :

$$O = Sgn(x_s) = Sgn(\sum_{i=1}^{N+1} a_i . x_i) \qquad (3.2)$$

Notons que puisque la dernière composante vaut toujours + 1, on peut écrire :

$$O(x) = Sgn(\sum_{i=1}^{N} a_i . x_i + a_N) \qquad (3.3)$$

où : a_N peut être vu comme un seuil ou offset associé à la sortie.

Cette expression est celle proposée par MacCullogh et Pitts dans leurs travaux

des années quarante sur les rapports entre neurones formels et logique. Il s'agit d'un modèle ultra simplifié d'un neurone biologique. Les coefficients a¡ représentent les poids synaptiques, le seuil et la fonction signe modélisent l'activation du neurone. L'apprentissage consiste à faire coïncider la sortie effective du réseau $O(X)$ avec une sortie désirée, ici $C(X)$ (qui vaut + 1 pour la classe 1 et –1 pour la classe 2).

On notera que le critère du perceptron est calculé à partir de $\sum_{i=1}^{N} a_i x_i + a_N$ et non de la sortie $O(X)$.Néanmoins, la règle d'apprentissage est la suivante :

Tant qu'il y a des exemples X tels que $O(X) \neq C(X)$:

Présenter un exemple $x = (x_1, ..., x_{N+1})$

Calculer la sortie $O(X)$ du réseau

Si : $O(X) = C(X)$, pas de modification

Sinon : $a_i = a_i + r.x_i$ pour $i = 1, ..., N+1$

Nous venons de reformuler l'algorithme du perceptron sous une forme connexionniste qui correspond aux intuitions initiales de F. Rosenblatt. De manière analogue, l'algorithme de Widrow-Hoff a d'abord été présenté par les auteurs sous le nom d'Adaline (Adaptative Linear Elements) vers 1960. Après quelques succès, le perceptron et l'Adaline plafonnent. Comme une unité (neurone, automate) ne sait calculer qu'une somme pondérée de ses entrées suivie d'un seuillage (plus ou moins brutal), un dispositif où une seule couche de poids est modifiable (le fameux apprentissage) sera limité aux problèmes linéairement séparables.

Les inventeurs ont bien sûr pensé à combiner ces éléments de manière plus complexe (fonctions quadratiques et non linéaires, logiques majoritaires et non à seuil, multiple adaline,…) mais si ces idées concernaient les points 1 et 2 (architecture & dynamique), elles ne proposaient pas de solution efficace au point 3 (apprentissage). Hors ce point est crucial, sauf si l'on dispose d'une technique permettant de définir directement (de manière non itérative) les poids à partir des exemples. Le livre de Minsky & Papert (1968), en montrant les limitations théoriques du perceptron, a contribué à la mise en sommeil de ce domaine de recherche, au moins pendant 15 ans. Les idées cybernetico-connexionnistes sont cependant restées vivantes notamment dans la modélisation biologique (Caianiello, Amari,…), en automatique et traitement du signal (Widrow,…), dans la théorie des mémoires associatives (Kohonen).

Le début des années quatre-vingt voit le redémarrage des travaux connexionnistes. En Finlande, T. Kohonen introduit des cartes topologiques auto-organisatrices (connues maintenant sous le nom de cartes de Kohonen).

Le physicien J. Hopfield propose un modèle à architecture complètement connectée (voisin de celui d'Amari) dont la dynamique peut être caractérisée par une fonction énergie inspirée des états désordonnés de la matière (on retrouve une des sources d'inspiration cybernétique: la physique statistique). L'informaticien G. Hinton associé à Anderson et d'autres propose des modèles distribués. En 1983, Hinton, Sejnowski et Ackley introduisent la machine de Boltzman dont la dynamique et l'apprentissage sont probabiliste, et dont les performances extrêmement intéressantes sont handicapées par des temps de convergence énormes.

En 1985-1986, Y. Le Cun, Parker et Rumelhart proposent indépendamment une extension de l'Adaline (bien que tout le monde parle de multi-perceptron) avec une architecture multicouche, une dynamique feed-forward et surtout un algorithme d'apprentissage des poids très efficace connu sous le nom de rétro-propagation (BP).

3.1.2. Caractéristiques architecturales d'un réseau neuronal

Les réseaux connexionnistes sont des assemblages d'unités de calcul, les neurones formels. Ces derniers ont pour origine un modèle du neurone biologique, dont ils ne retiennent d'ailleurs qu'une vision fort simplifiée (voir Figure 3.1). Le neurone, comme toute cellule, est composé d'un corps (ou soma), qui contient son noyau et où se déroulent les activités propres à sa vie cellulaire. Cependant, il est aussi doté d'un axone et de dendrites, structures spécialisées dans la communication avec les autres neurones. Cette communication entre cellules nerveuses s'effectue via des impulsions nerveuses. Les impulsions sont générées à l'extrémité somatique de l'axone et vont vers les terminaisons axonales. Là, elles affecteront tous les neurones reliés au neurone générateur, par l'intermédiaire de jonctions entre les terminaisons axonales et les autres cellules. Cette jonction est appelée synapse.

Cet héritage de la neurobiologie forme une composante importante de l'étude des réseaux connexionnistes, et le souci de maintenir une certaine correspondance avec le système nerveux humain a animé et continue à animer une part importante des recherches dans ce domaine. Malgré cet héritage, l'essentiel des travaux d'aujourd'hui ont pour objet les réseaux de neurones formels et non son corrélât neurobiologique. Vus comme des systèmes de calcul, les réseaux de neurones

possèdent plusieurs propriétés qui les rendent intéressants d'un point de vue théorique, et fort utile en pratique.

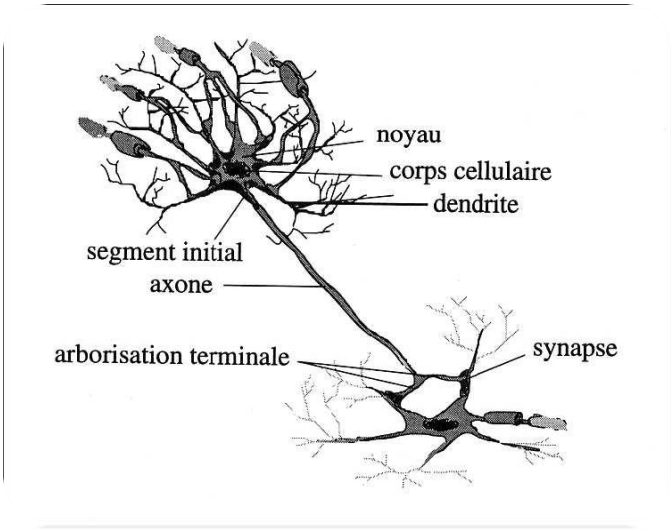


Figure 3.1. : Représentation simplifiée d'un neurone biologique [54].

Un réseau neuronal est constitué par un graphe orienté et pondéré. Les nœuds de ce graphe sont des automates simples nommés neurones formels (Figure 3.2) ou tout simplement unités du réseau, dotées d'un état interne, que l'on appelle état d'activation. Les unités peuvent propager leur état d'activation aux autres unités du graphe en passant par des arcs pondérés appelés connexions ou liens synaptiques dotés de poids synaptiques. La règle qui détermine l'activation d'un neurone en fonction de l'influence venue de ses entrées et de leurs poids respectifs s'appelle règle d'activation ou fonction d'activation.

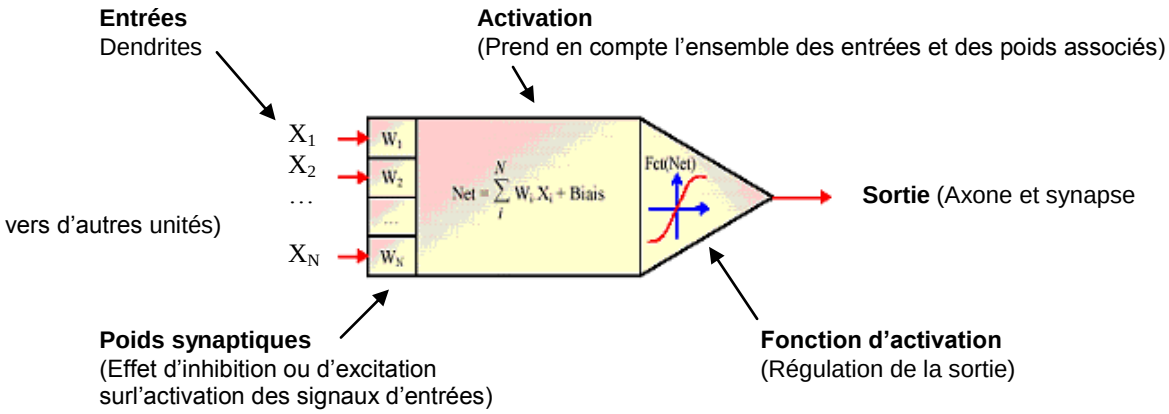


Figure 3.2. : Neurone formel [54].

L'architecture d'un réseau neuronal est donnée par le nombre et le type des neurones ainsi que la connectivité dont les contraintes permettent de distinguer les différents modèles voir (Figure 3.3). Les architectures les plus importantes sont :

3.1.2.1. Réseaux à une seule couche

Les unités sont toutes sur le même niveau. Dans ce type d'architectures, les unités sont connectées directement aux entrées et sont aussi les sorties du réseau. Les réseaux à une seule couche ont normalement des connexions latérales (entre les neurones d'une même couche). Un exemple de ce type d'architecture est le modèle de Kohonen "Kohonen Feature Map" [55], [56], [57] et [58].

3.1.2.2. Réseaux à couches unidirectionnels

Les unités sont organisées en plusieurs niveaux bien définis. En général, chaque unité d'une couche reçoit ses entrées à partir de la couche précédente et envoie ses sorties vers la couche suivante (feed-forward nets). La Figure 3.3.a montre un exemple de réseaux à trois couches. Cette architecture à trois couches (entrée, couche cachée et sortie) est très utilisée dans la pratique. Le célèbre modèle du Perceptron Multi-Couches (PMC) [55], [56], [57], [58] et [59] se compose en général d'une architecture de ce type, c'est-à-dire avec une seule couche cachée (hidden layer), mais rien n'empêche d'avoir plus d'une seule couche cachée dans ce modèle. Un autre type d'interconnexions dans les réseaux à couches sont les raccourcis (short-cuts) qui permettent de lier des unités en passant à travers des niveaux. Ainsi, on peut sauter d'une couche à l'autre avec les raccourcis, à condition de ne pas créer une boucle (voir Figure 3.3.b).

3.1.2.3. Réseaux récurrents

Les réseaux récurrents [56], [57], [58] et [59] peuvent avoir une ou plusieurs couches, mais leur particularité est la présence d'interconnexions depuis la sortie d'une unité vers une autre unité de la même couche ou d'une couche inférieure. Ce type d'interconnexions permet de modéliser des aspects temporaux et des comportements dynamiques, où la sortie d'une unité dépend de son état antérieur. Les boucles internes rendent ces réseaux instables, ce qui oblige à utiliser des algorithmes plus spécifiques (et généralement plus complexes) pour l'apprentissage. Un type particulier de réseau récurrent sont les réseaux totalement connectés, tels que le modèle de Hopfield représenté dans la Figure 3.3.d.

3.1.2.4. Réseaux d'ordre supérieur

Les unités de ce type permettent la connexion directe entre deux entrées ou plus, avant d'appliquer la fonction de calcul de l'activation de l'unité [60]. Ce type de réseau sert à modéliser les synapses de modulation, c'est-à-dire quand une entrée peut moduler (agir sur) le signal provenant d'une autre entrée. Un modèle particulier de réseaux d'ordre supérieur (high order neural net) est le réseau Sigma-Pi représenté dans la Figure 3.3.e.

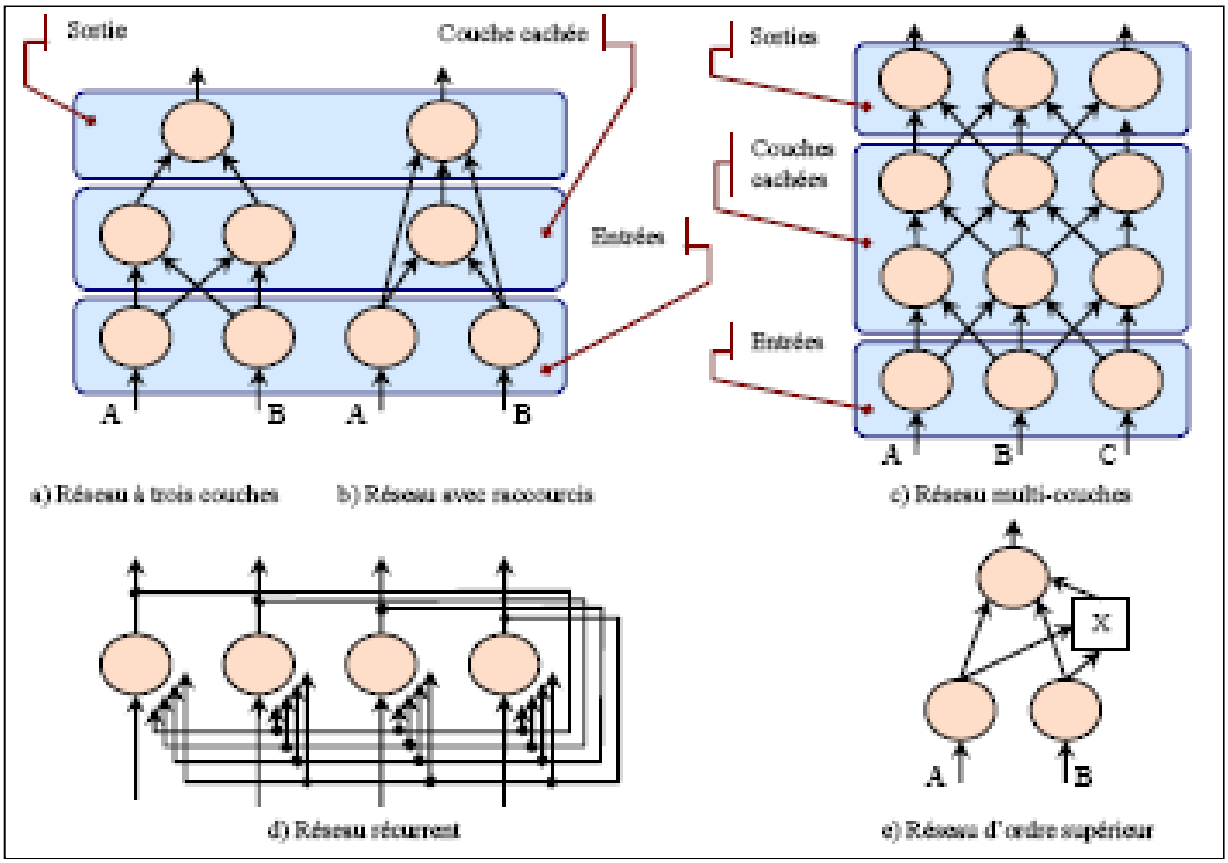


Figure 3.3. : Architectures de réseaux neuronaux [54].

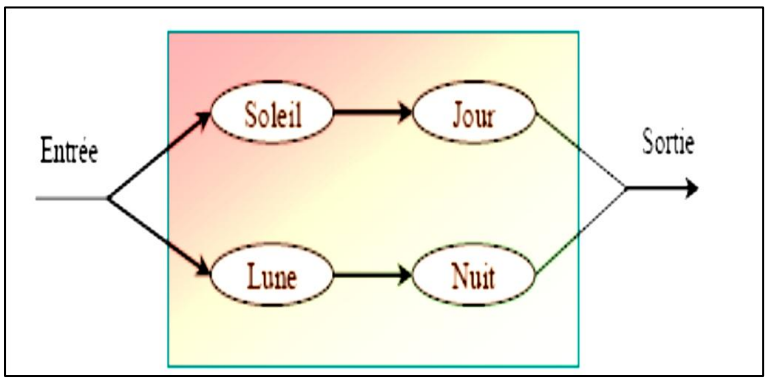


Figure 3.4. : Réseau à représentation locale [54].

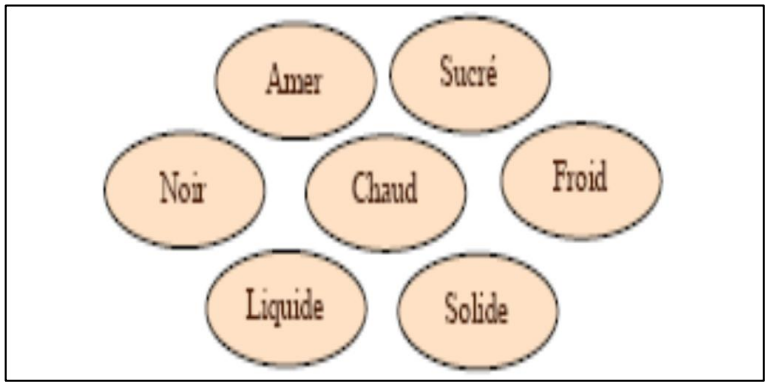


Figure 3.5. : Exemple de codage semi distribué par micro-traits [54].

3.1.3. Apprentissage connexionniste

Un des attraits principaux du formalisme des réseaux de neurones provient de sa capacité d'apprentissage et de généralisation à partir d'exemples. L'apprentissage est vu comme étant la possibilité de modifier la fonction d'un réseau en contrôlant ses coefficients synaptiques pour adapter son comportement par retouches successives.

3.1.3.1. Niveaux de difficulté de l'apprentissage

Le but de l'apprentissage est d'améliorer la réponse du réseau en modulant ses coefficients de contrôle selon un algorithme qui présente trois niveaux de difficulté :

1er niveau : Exprimer le lien entre la réponse du système à améliorer et les paramètres qui le définissent de façon à évaluer la direction suivant laquelle il est préférable d'évoluer. Ce lien étant établi, il apparaît que l'information donnée par un seul exemple n'est pas nécessairement pertinente. Il se peut, en effet, que plusieurs informations n'arrivent pas à être départagées à la seule vue de cet exemple. Si l'on essaye, en réponse à ce problème, de tenir compte simultanément de plusieurs exemples, il se peut alors que l'on obtienne des informations contradictoires, que chaque exemple indique des directions d'amélioration opposées et qu'aucune direction ne permette au système d'être simultanément amélioré sur tous les points.

2ème niveau : Tenir compte de tous les exemples dont on dispose. Un algorithme d'apprentissage doit tenir compte de ces deux niveaux de difficulté: le calcul de la direction d'évolution permettant de tenir compte d'un exemple et le choix d'un compromis pour tenir compte de tous les exemples.

3ème niveau : Ce niveau ne se situe pas sur le plan de l'algorithme, mais plutôt sur le plan de sa mise en œuvre. En effet, en fixant un graphe de connexion, on peut très bien, par inadvertance, ne pas y inclure le système que l'on souhaite obtenir, ou, au contraire, se donner un espace de recherche si grand que les quelques exemples dont on dispose ne sont plus pertinents. Dans les deux cas, notre recherche ne peut aboutir: dans un espace trop confiné, on ne peut qu'obtenir une machine réalisant un vague compromis et, avec trop de liberté, on risque d'obtenir un réseau au comportement fort éloigné de celui souhaité aux points non spécifiés comme exemples.

3.1.3.2. Types d'apprentissage

Il existe trois grandes classes d'apprentissage : non supervisé, supervisé et semi-supervisé.

a- Apprentissage non supervisé

Les réseaux à apprentissage non supervisé (à capacité acquise ou sans professeur) effectuent généralement des traitements comparables à des techniques d'analyse de données (problèmes de classification). La modification des poids synaptiques du réseau se fait en fonction d'un critère interne, indépendant de l'adéquation entre le comportement du réseau et la tâche qu'il doit effectuer. La règle d'apprentissage est en fonction du comportement local des neurones: on renforce les connexions entre le neurone ayant le mieux reconnu un exemple et les cellules d'entrée activées par cet exemple, ainsi, deux entrées proches produiront en sortie deux valeurs proches. La première règle d'apprentissage connexionniste (règle de Hebb) était non supervisée, elle était basée sur la proportionnalité entre la modification de l'efficacité synaptique des connexions entre deux neurones et l'activité simultanée de ces deux dernières.

b- Apprentissage supervisé

Le but de ce type d'apprentissage (à capacité enseignée ou avec professeur) est d'inculquer un comportement de référence au réseau en recherchant un jeu de poids synaptiques qui minimise l'erreur de sortie pour l'ensemble des patrons afin d'améliorer graduellement le comportement du réseau.

1. A chaque patron d'entrée, on associe une sortie désirée.
2. Un patron est présenté en entrée, puis l'activité est propagée à travers le réseau. Les sorties obtenues sont comparées aux sorties désirées.
3. On détermine la correction à apporter aux poids du réseau en utilisant une fonction permettant de calculer la modification à apporter aux poids en fonction de l'erreur et d'un pas d'apprentissage constant.

En général, les règles d'apprentissage supervisé sont des formes de descente du gradient où se pose le problème du choix du pas des itérations successives.

c. Apprentissage semi-supervisé

Ce type d'apprentissage, appelé aussi apprentissage par renforcement (qualitatif ou pénalité-récompense) suppose qu'un comportement de référence précis n'est pas disponible, mais, qu'en revanche, il est possible d'obtenir des indications qualitatives (correct/incorrect) sur les performances du réseau. Le jeu d'échecs est un exemple typique des problèmes d'apprentissage semi-supervisé. La seule information dont dispose le système pour corriger l'ensemble des coups qui constitue son environnement est l'évaluation finale de la partie.

3.1.3.3. Apprentissage par rétro-propagation du gradient

L'algorithme de rétro-propagation du gradient est l'aboutissement de l'évolution du modèle à couches des réseaux de neurones. C'est probablement aujourd'hui l'algorithme le plus utilisé, qui permet d'obtenir les résultats les plus satisfaisants dans beaucoup de domaines d'application. Dans cette section nous nous limitons à une présentation intuitive, le lecteur intéressé par une présentation plus formelle peut se référer à des documents tels que [55], [57], [58], [59], [60] et [61].

L'apprentissage est supervisé et fonctionne selon le principe suivant:

- On dispose d'un ensemble d'exemples qui sont les couples (entrées, sorties désirées). A chaque étape, un exemple est présenté au réseau, une sortie réelle est calculée. Ce calcul est effectué de proche en proche de la couche d'entrée à la couche de sortie. Cette phase est appelée "propagation avant" ou "relaxation du réseau".
- Le réseau apprend en essayant de diminuer son erreur à chaque itération. Il le fait en changeant l'intensité des connexions en sens inverse du signal d'erreur. Chaque neurone de la couche de sortie calcule son signal d'erreur puis

considère que les cellules de la couche cachée sont responsables de cette erreur de manière proportionnelle à leur contribution dans sa valeur. Par exemple, pour une cellule de la couche cachée qui doit répondre avec la valeur 1 mais répond avec la valeur 0, l'erreur consiste à n'avoir pas prêté assez attention aux cellules de la couche cachée qui lui suggéraient d'être actives ou d'avoir trop prêté attention à celles qui lui suggéraient d'être inactive. La solution est d'augmenter l'intensité des connexions positives et de diminuer celle des connexions négatives en provenance des cellules actives de la couche cachée.

- Le signal d'erreur des couches cachées n'est pas calculé directement, il doit être évalué en fonction de l'erreur des cellules de la couche de sortie. Chaque cellule de la couche cachée considère que son erreur peut s'estimer par la moyenne des erreurs qu'elle a fait commettre aux neurones de sortie. L'importance de l'erreur dépend de l'intensité de la connexion entre la cellule cachée et la cellule de sortie: il faut donc pondérer l'erreur par l'intensité de la connexion reliant la cellule de la couche de sortie puis renvoyer le signal d'erreur obtenu à la cellule de la couche cachée en réutilisant les mêmes connexions mais dans le sens inverse, d'où le nom de Rétro-propagation.
- Ce processus est répété, en présentant successivement chaque exemple. Si, pour tous les exemples, l'erreur est inférieure à un seuil choisi, on dit que le réseau a convergé. Bien que l'on ne dispose pas de preuve de sa convergence, cet algorithme donne de bons résultats dans de nombreuses applications pratiques. En plus, il n'existe pas de résultat théorique, ni de règle empirique satisfaisante, qui permette de dimensionner correctement un réseau en fonction du problème à résoudre.

3.1.3.4. Problèmes d'apprentissage

Plusieurs problèmes peuvent intervenir au cours de l'apprentissage [58] :

- Insuffisance de la règle d'apprentissage car rien ne garantit que la règle d'apprentissage soit capable de tirer profit du potentiel du réseau.
- Minima locaux car la technique de descente du gradient, utilisée par la majorité des apprentissages supervisés peut conduire à des solutions sous-optimales.
- Un mauvais choix de paramètres tels que le nombre de neurones cachés, par exemple, peut compromettre l'apprentissage.

- Le sur-apprentissage (over-fitting) peut se produire quand l'apprentissage d'un réseau est prolongé, ses poids reflèteront de trop près les particularités des exemples appris.
- Un mauvais échantillonnage du corpus d'apprentissage peut engendrer une mauvaise généralisation.
- Oublis et interférences quand le corpus est grand ou que le comportement à apprendre présente beaucoup de cas exceptionnels.

3.1.4. Avantages et inconvénients des réseaux neuronaux

L'intérêt porté aux réseaux de neurones tient sa justification dans les propriétés intéressantes qu'ils possèdent; ils présentent aussi un certain nombre de limitations ou d'inconvénients. A partir de ces propriétés, les applications potentielles de l'approche neuronale peuvent être déduites [55], [60] et [62].

3.1.4.1. Avantages de l'approche neuronale

L'exploitation de connaissances empiriques : L'apprentissage à partir d'exemples (méthode d'apprentissage empirique) se fait d'une façon assez simple et permet d'obtenir de bons résultats par rapport aux autres techniques d'apprentissage automatique [63].

La robustesse : Dans les réseaux de neurones, la mémoire est distribuée, elle correspond à une carte d'activation de neurones. Cette carte est en quelque sorte un codage des faits mémorisés ce qui attribue à ces réseaux l'avantage de résister aux bruits (pannes) car la perte d'un élément ne correspond pas à la perte d'un fait mémorisé. Ainsi, un réseau peut bien fonctionner même quand des unités sont en panne. De plus, de nombreux modèles de réseaux donnent de bons résultats même quand leurs entrées sont bruitées [64].

La dégradation progressive : Les réseaux, de par leur nature continue, ne fonctionnent pas en tout ou rien et leurs performances ont plutôt tendance à diminuer progressivement en cas de problème (bruit, panne, entrée inconnue...). Cette propriété est très recherchée car les systèmes cognitifs vivants montrent une telle faculté. Les réseaux permettent de bien généraliser les connaissances présentes dans la base d'apprentissage et sont moins sensibles aux perturbations que les systèmes symboliques. Le fait de travailler sur une représentation numérique des connaissances rend les réseaux plus adaptés pour manipuler des données

quantitatives (valeurs continues). Les réseaux de neurones sont moins vulnérables aux données approximatives et à la présence de données incorrectes dans la base d'apprentissage.

Le parallélisme massif : Les réseaux sont composés d'un ensemble d'unités de traitement de l'information qui peuvent opérer en parallèle. Bien que la plupart des implémentations des réseaux connexionnistes soient réalisées sur des simulateurs séquentiels, il est possible de faire des implémentations (logicielles ou matérielles) exploitant la possibilité d'activer simultanément les unités. La plupart des implémentations des réseaux de neurones peuvent être facilement converties d'une version séquentielle vers une version parallèle.

La prise en compte du non linéarité et du temps : Les réseaux de neurones artificiels présentent l'intérêt de pouvoir prendre en compte la non linéarité (les fonctions d'activation sont en général non linéaires).

Certains réseaux peuvent aussi prendre en compte les aspects temporels (cas des réseaux récurrents).

3.1.4.2. Inconvénients de l'approche neuronale

La difficulté de choix de l'architecture et des paramètres : Il n'existe pas de méthode automatique pour choisir la meilleure architecture possible pour un problème donné. Il est assez difficile de trouver la bonne topologie du réseau ainsi que les bons paramètres de réglage de l'algorithme d'apprentissage. L'évolution du processus d'apprentissage est très influencée par ces deux éléments (l'architecture du réseau et les paramètres de réglage) et dépend beaucoup du type de problème traité. Le simple fait de changer la base d'apprentissage utilisée, peut nous obliger à reconfigurer le réseau en entier.

Le problème d'initialisation et de codage : Les algorithmes d'apprentissage connexionniste sont en général très dépendants de l'état initial du réseau (initialisation aléatoire des poids) et de la configuration de la base d'apprentissage. Un mauvais choix des poids employés pour initialiser le réseau, de la méthode de codage des données, ou même de l'ordre des données, peut bloquer l'apprentissage ou poser des problèmes pour la convergence du réseau vers une bonne solution.

Le manque d'explicabilité : Les connaissances acquises par le réseau sont codées par l'ensemble des valeurs des poids synaptiques ainsi que par la façon dont les unités sont interconnectées. Il est très difficile pour un être humain de les

interpréter directement. Les réseaux connexionnistes sont en général des boîtes noires, où les connaissances restent enfermées et sont inintelligibles pour l'utilisateur ou pour l'expert. Un réseau ne peut pas expliquer le raisonnement qui l'a amené à une solution spécifique.

Le manque d'exploitation de connaissances théoriques :

Les réseaux classiques ne permettent pas de profiter des connaissances théoriques disponibles sur le domaine du problème traité. Ils sont dédiés à la manipulation de connaissances empiriques. Une façon simpliste de profiter des connaissances théoriques consiste à convertir des règles en exemples (prototypes). Cependant, cette méthode ne garantit pas que ces exemples vont être bien représentés dans les connaissances du réseau à la fin de l'apprentissage, car nous sommes obligés à passer pour une phase d'apprentissage où se mélangent sans distinction des connaissances empiriques avec des connaissances théoriques codées par des exemples.

3.1.5. Applications de l'approche neuronale

Voici quelques-unes des caractéristiques des problèmes bien adaptés à une résolution par les réseaux de neurones :

- Les règles qui permettent de résoudre le problème sont inconnues ou très difficiles à expliciter ou à formaliser. Cependant, on dispose d'un ensemble d'exemples qui correspondent à des entrées du problème et à des solutions qui leurs sont données par des experts.
- Le problème fait intervenir des données bruitées ou incomplètes.
- Le problème peut évoluer et nécessite une grande rapidité de traitement.
- Il n'y a pas de solutions technologiques courantes.

Grâce à ces caractéristiques, il est possible de retrouver les domaines privilégiés qui constituent le cœur des applications des réseaux de neurones: la reconnaissance des formes [65], la classification, la transformation de données (compression), la prédiction, le contrôle de processus et l'approximation de fonctions. Ces tâches peuvent être regroupés en deux catégories principales selon le type des sorties fournies par le réseau et le comportement qui est recherché :

3.1.5.1. Réseaux pour l'approximation de fonctions

Ce type de réseaux doit avoir une sortie continue et sera employé pour l'approximation exacte (interpolation) ou pour l'approximation approchée d'une fonction représentée par les données d'apprentissage Ces réseaux sont capables d'apprendre une fonction de transformation (ou d'association) des valeurs de sortie aux valeurs d'entrée. Cette fonction acquise par le réseau permet de prédire les sorties étant données les entrées. On appelle ce type de problème, un problème de régression. En général, les fonctions représentées sont des fonctions avec des variables d'entrée et de sortie continues.

3.1.5.2. Réseaux pour la classification

Ce type de réseau doit attribuer des classes (valeur de sortie non continue) aux exemples qui lui sont fournis. La classification est un cas particulier de l'approximation de fonctions où la valeur de sortie est discrète et appartient à un ensemble limité de classes. Cet ensemble de classes peut être connu d'avance dans le cas de l'apprentissage supervisé. Un réseau adapté à la classification doit avoir des sorties discrètes ou implémenter des méthodes de discrétisation des sorties (e.g. application d'un seuil de discrimination).

3.2. Méthode robuste de vote, la Transformée de Hough

L'estimation des paramètres avec les méthodes de vote repose sur l'utilisation du minimum de données nécessaires à l'estimation. Chaque estimation, avec un jeu de données particulier, correspond à un "vote" pour les paramètres obtenus. Le jeu de paramètres élu, i.e. le plus "voté", est retenu comme résultat de l'estimation. A cet effet, la Transformée de Hough outil de détection de courbes paramétriques dans une image a été proposée par P.V.C Hough dans un brevet déposé en 1960, [66].

Inaperçu pendant plusieurs années, cette dernière a été vulgarisée par les travaux de Rosenfield [67], Duda et Hart [21] au début des années 70 et fait l'objet par la communauté scientifique depuis cette date à ce jour d'une particulière attention. Depuis les années 80, elle a quitté les laboratoires de recherche pour trouver des champs d'applications dans de nombreux domaines industriels [22], [68] et [69], tels que la vision par ordinateur et le traitement d'images. Elle est devenue une solution plus adaptée au problème de détection des lignes droites, cercles ou toute autre

forme paramétrique dans l'image. Cependant, le calcul de la TH requiert de larges délais de traitement et beaucoup d'espace mémoire, alors plusieurs nouveaux algorithme tels que la TH probabiliste, la TH aléatoire, la TH hiérarchique, la TH incrémentale, ont été proposés pour améliorer ce calcul, le rendre efficace et praticable pour son utilisation dans le traitement d'images en temps réel.

Pour illustrer le principe de base de la TH, on considère par exemple, le problème de détection d'un objet polyédrique dans une image, on doit avant toute détection, extraire les traits de ce dernier puis reconstruire son image. Donc on est amené à extraire des lignes droites où autrement dit les ensembles de points colinéaires. La méthode robuste qui peut faire ce traitement doit tester la présence de droites formées par toutes les paires de points de l'image, ce qui est extrêmement inefficace, vu que pour tester n points de l'image deux par deux, on arriverait à un nombre exagéré d'itérations au moins supérieur à n^2.

Dans une image 512*512 cette méthode devient prohibitive [70]. La TH résout ce problème car elle transforme les lignes du plan 2D de l'image en points dans le plan des paramètres qui définit ces lignes, par conséquent elle convertit le problème de détection de ligne en un autre plus simple celui de la détection des points d'intersection. Le principe est de prendre toutes les paires de pixels appartenant aux contours de l'image et construire un histogramme à deux dimensions appelé aussi le ''tableau accumulateur'' qui servira à enregistrer les droites ainsi trouvées pour chaque paire de pixel tirée. Tout point de l'image faisant parti d'une de ces droites se voit incrémenter d'un crédit dans l'histogramme.

Tout en gardant la même idée principale du calcul de la TH citée ci-dessus; ils existent plusieurs façon de traiter les points de l'image, ce qui nous donne diverses techniques de la TH. Celles-ci sont énumérées comme suit:

a) Transformée de Hough probabiliste

Contrairement à la transformée de Hough standard qui applique la TH pour tous les pixels d'une image contours, la transformée de Hough probabiliste [71] et [72] affirme qu'il suffit de calculer la TH seulement pour une portion α de pixels de cette image $(0<\alpha<100)$. Ces pixels sont choisis aléatoirement à partir de la densité de probabilité uniforme définit sur l'image. Kiraty et al [71] préconisent d'utiliser une valeur α comprise entre 10% et 20%, cette variation dépend de l'application.

b) Transformée de Hough aléatoire

La TH aléatoire est présentée dans [73] et les explications peuvent être aussi trouvées dans [74] et [75]. Brièvement, la TH aléatoire, utilise une autre technique pour générer des valeurs dans le tableau accumulateur définit sur le plan de paramètres. Dans le cas de la détection d'une ligne droite dans la TH standard, un pixel de l'image correspond à une courbe dans le plan de paramètres, celui-ci est discrétisée et enregistré dans le tableau accumulateur, par contre dans la TH aléatoire, une paire de pixels est choisi aléatoirement et les paramètres de la ligne unique qui passe par ces pixels sont calculés.

Cette ligne est enregistrée comme la seule entrée dans le tableau accumulateur et les pixels de cette dernière sont ensuite enlevés, et laissent une image simple à analyser. De cette façon, les entrées sont accumulées dans l'espace des paramètres. Cet algorithme est ensuite répété pour détecter les lignes suivantes un nombre de fois dans le temps, où le nombre des itérations est beaucoup moins que le nombre de paires de pixels dans l'image. L'algorithme s'arrête quand aucune ligne n'est détectée pour ce nombre d'itérations.

c) Transformée de Hough hiérarchique

La TH hiérarchique combine une structure pyramidale avec la Transformée de Hough (voir la Figure3.6). L'image est organisée en grille de sous-images et la TH est appliquée sur chacune d'elle. Typiquement, chacune de ces sous-images va contenir au plus deux lignes. Ces résultats sont propagés vers le haut à travers la pyramide. Pour chaque nœud du prochain niveau, on prend le chevauchement à l'aide d'un masque 4*4 du niveau précèdent. Les lignes de ces dernières sont fusionnées en utilisant l'algorithme de la TH. Ces lignes sont propagées au prochain niveau hiérarchique. Cet algorithme est répété jusqu'au niveau le plus haut de la pyramide, où on trouve un seul nœud. Ce nœud représente alors l'image entière. Les détails sont donnés dans [76].

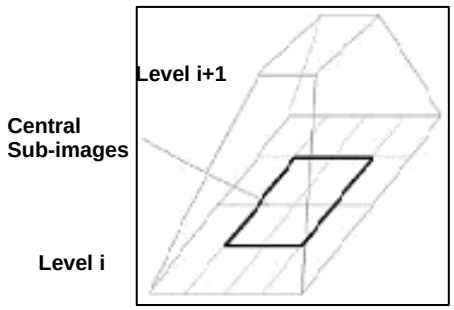


Figure 3.6. : Structure pyramidale de la transformée de Hough hiérarchique.

d) Transformée de Hough incrémental

Pour appliquer la TH dans les tâches de traitement d'image en temps réels, son calcul doit être le plus court possible. Habituellement la TH dans le cas particulier des droites, est définit par l'équation :

$$f(x,y,\rho,\theta) = \rho - x\cos\theta - y\sin\theta = \boldsymbol{0} \qquad (3.4)$$

Donc son calcul réclame l'utilisation des formules trigonométrique et des multiplications, ce qui nécessite un temps de calcul énorme. Pour remédier à ce problème, on utilise la transformée de Hough incrémental. Cette dernière utilise une autre expression de la TH, définit par des fonctions d'additions et des décalages [77].

3.2.1. Principe de la Transformée de Hough

Une droite est décrite dans le plan cartésien (xy) par l'expression suivante :

$$f(y,x,a,b) = y - ax - b = \boldsymbol{0} \qquad (3.5)$$

Sachant que ***a*** est la pente et ***b*** l'ordonnée à l'origine des abscisses.

Etant donné un ensemble de contours d'objets représentés par un ensemble de points discrets M_i, nous cherchons à déterminer si un ou plusieurs sous-ensembles de points M_i font partie d'une courbe dont les paramètres a et b restent à définir. Si nous cherchons à tester les *n* points M_i deux par deux, nous arriverons à un nombre exagéré d'itérations au moins supérieur à n^2.

Hough puis Rosenfield [66] et [67] ont proposé une méthode pour détecter les droites à l'aide des points du plan *(x, y).* Son principe est de calculer pour chaque point M_i de coordonnées (x_i , y_i), du contour d'un objet, l'ensemble des paramètres ***a*** qui vérifient l'équation $f(y_i , x_i , a , b) = 0$ avec ***b*** fixé.

Pour chaque point $M_i (x_i , y_i)$,de l'image, il y a un ensemble de valeurs possibles pour les paramètres ***a*** et ***b***. Cet ensemble forme une droite d'équation $b = -ax + y$ dans l'espace des paramètres *(ab)* appelé espace de Hough. Deux points p_i et p_j de cordonnées (x_i , y_i) et (x_j , y_j) respectivement, appartenant à la même droite, forment des droites dans l'espace des paramètres *(ab)*, qui se coupent au point ***N*** de coordonnées*(a', b')*. De cette façon tous les points qui appartiennent à la même droite forment des droites dans le plan des paramètres *(ab)* qui se coupent au même point. Ce concept est illustré dans les Figures 3.7.a et 3.7.b.

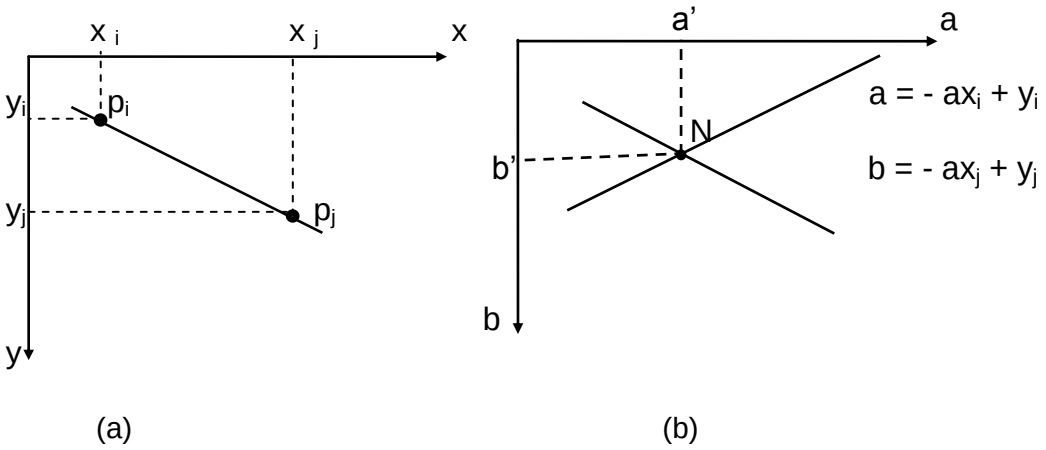


Figure 3.7. : Transformée de Hough.
(a) : Plan cartésien *(xy).*
(b) : Plan des paramètres *(ab).*

Le traitement de Hough consiste en une quantification du plan des paramètres en cellules accumulatrices sur la Figure 3.8 où (a_{min}, a_{max}) et (b_{min}, b_{max}) sont les valeurs limites de l'intervalle de la pente ***a*** et de l'ordonnée à l'origine des abscisses ***b***.

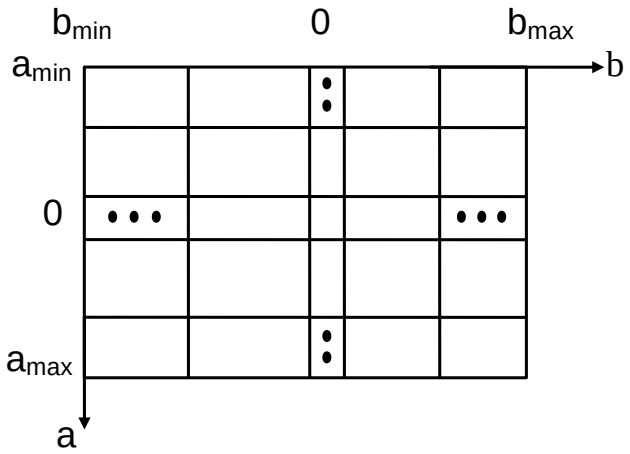


Figure 3.8. : Quantification du plan des paramètres (ab).

Chaque cellule de coordonnées (i, j) a une valeur accumulée $A(i, j)$ et correspond à la cellule de coordonnées (a_i, b_j) dans le plan des paramètres *(ab)*. Initialement, ces cellules sont mises à zéro. Pour chaque point de l'image de coordonnées (x_k, y_k) on calcule pour chaque valeur de ***a*** quantifié a la valeur a_p sur l'axe des ***a***, son correspondant ***b*** en utilisant l'équation suivante : $b = -ax_k + y_k$.

La valeur résultante résultat ***b*** est arrondie à la valeur la plus proche de ***b*** quantifiés b_q sur l'axe des ***b***. Si on obtient une valeur b_q suite à a_p choisie, on incrémente la valeur de la cellule correspondante: $A(p,q) = A(p,q) + 1$.

A la fin de cette procédure, la valeur ***n*** de $A(i, j)$ dans une cellule (i, j) correspond à ***n*** points dans le plan (xy) qui vérifient l'équation $y = a_i x + b_j$, donc il existe ***n*** points qui appartient à la droite de pente a_i et de l'ordonnée à l'origine des abscisses b_j.

L'inconvénient majeur de cette procédure réside dans son incapacité de détecter les droites verticales. Pour remédier à ce problème, un paramétrage polaire (ρ, θ) est plus satisfaisant. Ce paramétrage est illustré dans la Figure 3.9.

Une droite est alors définie par l'équation (3.4), avec ρ la distance perpendiculaire à la droite de l'origine du plan (xy) et θ l'angle entre cette distance et l'axe des *x*.

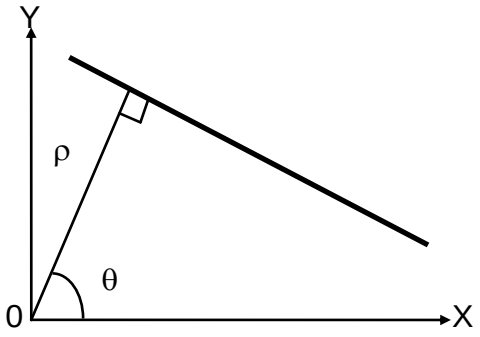


Figure 3.9. : Paramétrage polaire d'une droite.

L'utilisation de cette représentation dans la construction du tableau accumulateur est identique à celle développée précédemment (la représentation ab sur la Figure(3.10)).

On précisera que le choix de quantification de l'espace des paramètres (ρ,θ) doit porter sur les trois objectifs essentiels suivants:

1. Garantir une précision de détection aussi bonne que possible,
2. Diminuer la mémoire nécessaire au stockage des accumulateurs,
3. Accélérer les calculs.

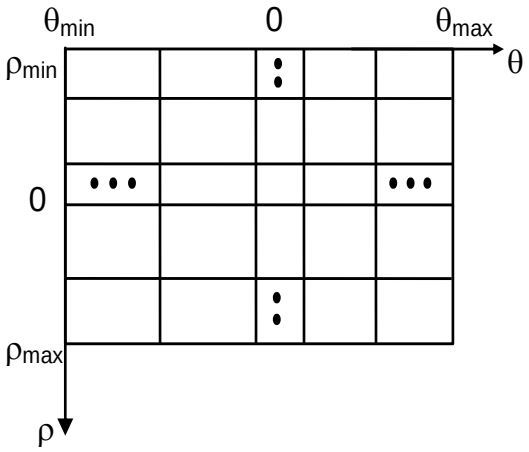
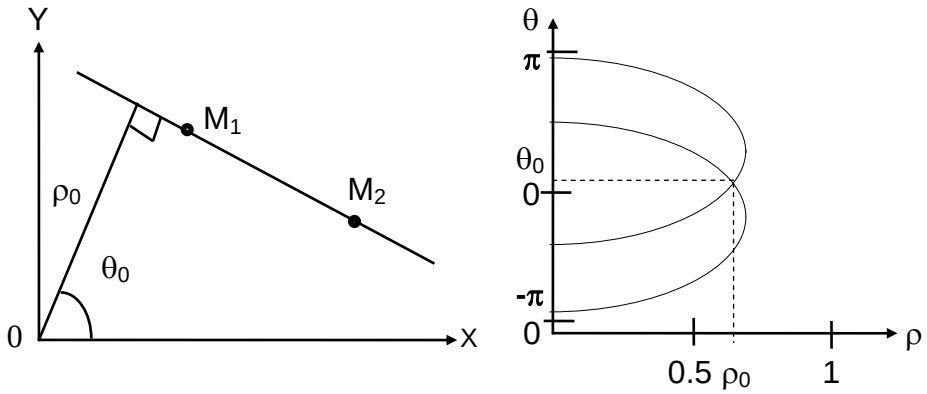


Figure 3.10. : Quantification du plan des paramètres *(ab).*

Chaque point M_i de coordonnées (x_i, y_i) d'une droite se transforme dans le plan des paramètres *(ab)* en une sinusoïde d'équation $\rho = x_i \cos\theta + y_j \sin\theta$. Donc une droite sera représentée par un ensemble de sinusoïdes qui se coupent en un seul point de coordonnées polaires (ρ_0, θ_0) caractéristique de cette droite dans le plan des paramètres (se référer aux Figures 3.11-a et 3.11-b).



(a) (b)

Figure 3.11. : Transformée de Hough.

(a) : Plan cartésien *(xy).*

(b) : Plan des paramètres *(ab).*

3.2.2. Dimension des paramètres θ et ρ

3.2.2.1. Champ de la dimension de θ

Le champ de la dimension de θ est $[0, 2\pi]$. Pour un angle θ appartenant à cet intervalle, toutes les droites s'expriment avec un ρ positif. Nous remarquons que les droites dont θ appartenant à l'intervalle $[\pi, 2\pi]$ peuvent être vue comme des droites à ρ négatif, donc l'intervalle de θ peut être réduit de moitié, c'est à dire le champ de θ sera $[0, \pi]$ et pour chaque valeur θ appartenant à cet intervalle toutes les droites s'expriment avec un ρ pas strictement positif. En effet si on considère la droite *D2* avec les paramètres polaires (ρ_2, θ_2) avec : $\theta_2 = \pi + \theta_1$

$$\rho_2 = x.\cos\theta_2 + y.\sin\theta_2$$
$$\rho_2 = x.\cos(\pi + \theta_1) + y.\sin(\pi + \theta_1)$$
$$\rho_2 = -(x.\cos\theta_1 + y.\sin\theta_1)$$
$$\rho_2 = -\rho_1$$

La droite *D2* de la Figure 3.12 est vue comme la droite *D1* mais avec un ρ opposé $(\rho_2 = -\rho_1)$.

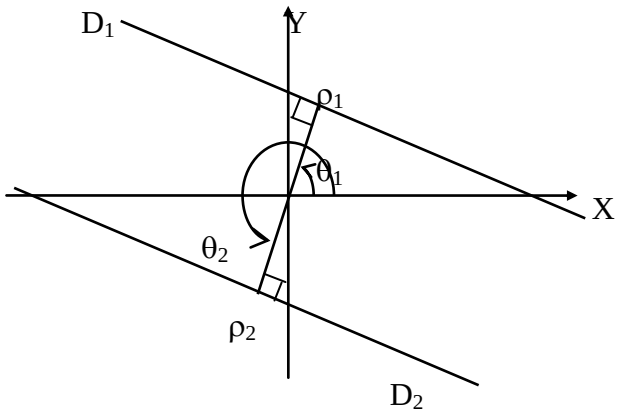


Figure 3.12. : Paramètres polaires de deux droites opposées.

3.2.2.2. Champ de dimension de ρ

Le champ de dimension de ρ est défini selon la taille de l'image pour une image carrée N*N. A partir de la Figure 3.13, on trouve que le champ de dimension de ρ est : [*0 , N√2*].

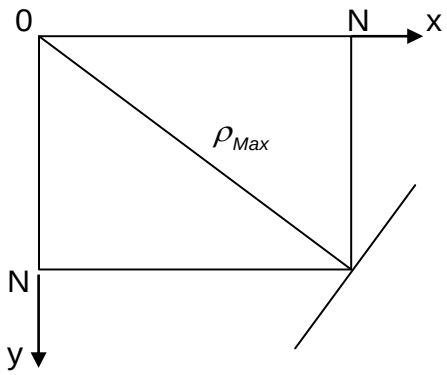


Figure 3.13. : Champ de la dimension de ρ.

Si nous déplaçons l'origine *O* du repère du plan *(x y)* au centre de l'image, puis nous examinons les valeurs ρ_{min} et ρ_{max} de la dimension de ρ .

- ρ_{max} est donnée par la plus grande distance de la droite par rapport à l'origine *O*. Cette droite est représentée par la droite *D2* dans la Figure 3.14 et $\rho_{\max} = Max/\sqrt{2}$.

- $\rho_{\min}$ est la plus petite distance négative de la droite par rapport à l'origine *O*. Elle est représentée par la droite *D1* et $\rho_{\min} = -Max/\sqrt{2}$.

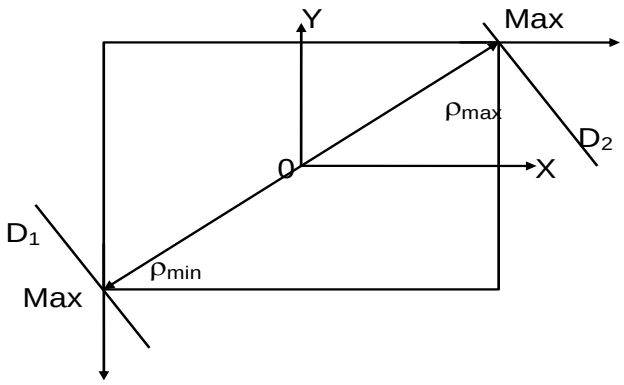


Figure 3.14. : Champ de la dimension de ρ.

3.2.2.3. Propriétés de la Transformée de Hough

- Un point du plan cartésien correspond à une sinusoïde dans le plan des paramètres.
- Un point du plan des paramètres correspond à une droite dans le plan cartésien.
- Les points appartenant à la même droite dans le plan cartésien correspondent au point d'intersection des courbes du plan des paramètres.
- Les points appartenant à la même courbe du plan des paramètres correspondent à des droites du même point du plan cartésien.

3.2.3. Utilisation de la Transformée de Hough

L'extraction des lignes droites utilisant le tableau accumulateur de Hough est une tâche qui nécessite la prise de certaines précautions car les ***n*** points, d'une cellule du tableau accumulateur, n'indiquent pas s'ils sont approchés ou dispersés. Ceci induit le problème d'extraction de fausses droites ou droites insignifiantes. Ce problème est accentué lorsqu'on a une image trop éclairée (création de zones d'ombre) et une image qui possède un nuage de points (une image trop bruitée).

Pour résoudre ce problème, on élimine d'abords tous les segments de droite ayant un nombre de points réduit dans le tableau accumulateur, ensuite on scrute les cellules accumulatrices de ce dernier en cherchant les droites significatives [78] et [79].

Ces droites sont des droites réelles qui existent dans l'image telles que les arrêtes d'un objet quelconque dans cette image. Une droite significative se présente par un pic (sommet d'une montagne).Lorsqu'on représente le tableau accumulateur en relief3D sur la Figure 3.15, une droite significative doit avoir les caractéristiques suivantes:

- Sa longueur ne doit pas être trop petite devant un seuil donné (seuil *H*).
- La longueur des segments de droite qui forment cette droite ne doit pas être inférieure à un seuil donné (seuil *C*), sinon ils seront éliminés.
- Les points qui forment ces segments de droite ne doivent pas être trop espacés, donc la distance séparant deux points ne doit pas être supérieure à un troisième seuil donné (seuil *D*).

Donc pour avoir une bonne détection des droites significatives, il faut faire un compromis entre les trois seuils seuil *H*, seuil *D* et seuil *C* suivant le domaine d'application utilisé.

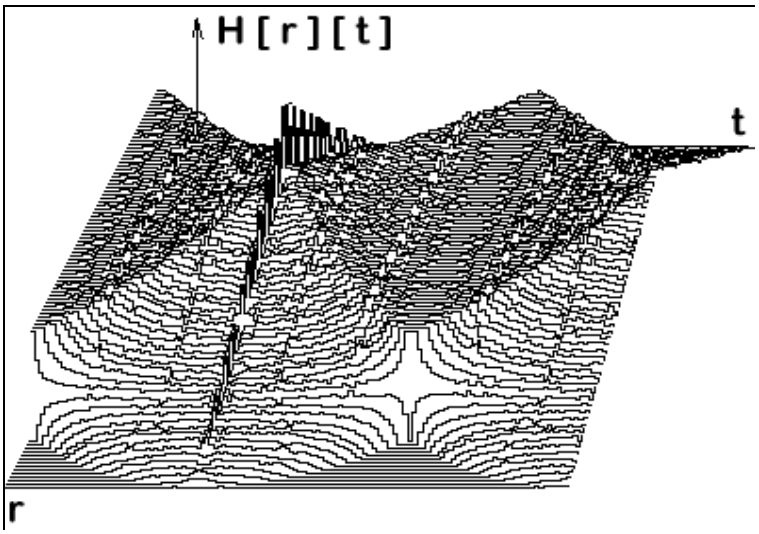


Figure 3.15. : Le plan de Hough en relief.

Nous remarquons aussi qu'il y a présence des vallées et des montagnes avec leurs pics. Les pics des montagnes correspondent aux droites réelles qui existent dans l'image, par contre les vallées ne correspondent en réalité à rien car elles sont les conséquences du calcul de la transformée de Hough.

3.2.4. Implémentation de la Transformée de Hough

La Transformée de Hough est un outil puissant dans l'analyse des formes et utilisée aussi pour l'extraction des traits globaux des formes dans l'image. Elle donne de bons résultats, même en présence de bruits dans l'image.

La Transformée de Hough (TH) opère sur des données binaires des points contours de l'image. Chaque point contour de coordonnées (x_i , y_i) de l'image est transformé en une sinusoïde dans le plan des paramètres $\rho\theta$.Elle contribue dans ce plan par l'incrémentation de la cellule (ρ_j , θ_j) du tableau accumulateur crée par cette transformation où θ_j est une valeur discrète suivant la résolution choisie de θ dans l'intervalle [0 , π]. Initialement les cellules du tableau accumulateur sont mises à zéro

[69], [80] et [81]. Les points contours colinéaires de l'image produisent des sinusoïdes dans le plan des paramètres qui se croisent en un point commun (ρ, θ).

Nous affectons dans la cellule (ρ, θ) du tableau accumulateur, avec ρ la distance normale à la droite qui porte ces points colinéaires de l'origine de l'image et θ l'angle entre cette distance avec l'axe des *x*, le nombre de croisés des courbes au point d'intersection (ρ, θ) dans le plan des paramètres. La quantification du plan des paramètres ρθ revient à quantifier l'intervalle $0 \leq \theta < \pi$ pour la dimension de θ et l'intervalle $-R \leq \rho \leq R$ pour la dimension de ρ, avec *R* la moitié de la diagonale de l'image. De plus, si nous prenons :

ρ_K et θ_K pour pas de quantification des dimensions de ρ et θ .

n_ρ et n_θ le nombre de valeurs discrètes dans les intervalles de ρ et θ.

Les valeurs de ρ et θ discrétisées s'écrivent comme suit :

$$\theta = t.\theta_k \tag{3.6}$$

avec : $\boldsymbol{0} \leq t \leq n_\theta$

$$\rho = -R + r.\rho_K \tag{3.7}$$

avec : $\boldsymbol{0} \leq r \leq n_\rho$

$$n_\theta = \frac{\pi}{\theta_k} \; et \; n_\rho = \frac{\boldsymbol{2}R}{\rho_k} \tag{3.8}$$

Pour chaque point contour de coordonnées *(x , y)* de l'image, nous calculons pour chaque valeur discrète de θ la valeur discrète de ρ suivant l'équation suivante :

$$\rho = x.\cos\theta + y.\sin\theta \tag{3.9}$$

Ensuite, nous incrémentons la cellule correspondante dans le tableau accumulateur. Nous faisons la même chose pour tous les points contours de l'image.

L'organigramme de la Figure 3.16 décrit ce calcul.

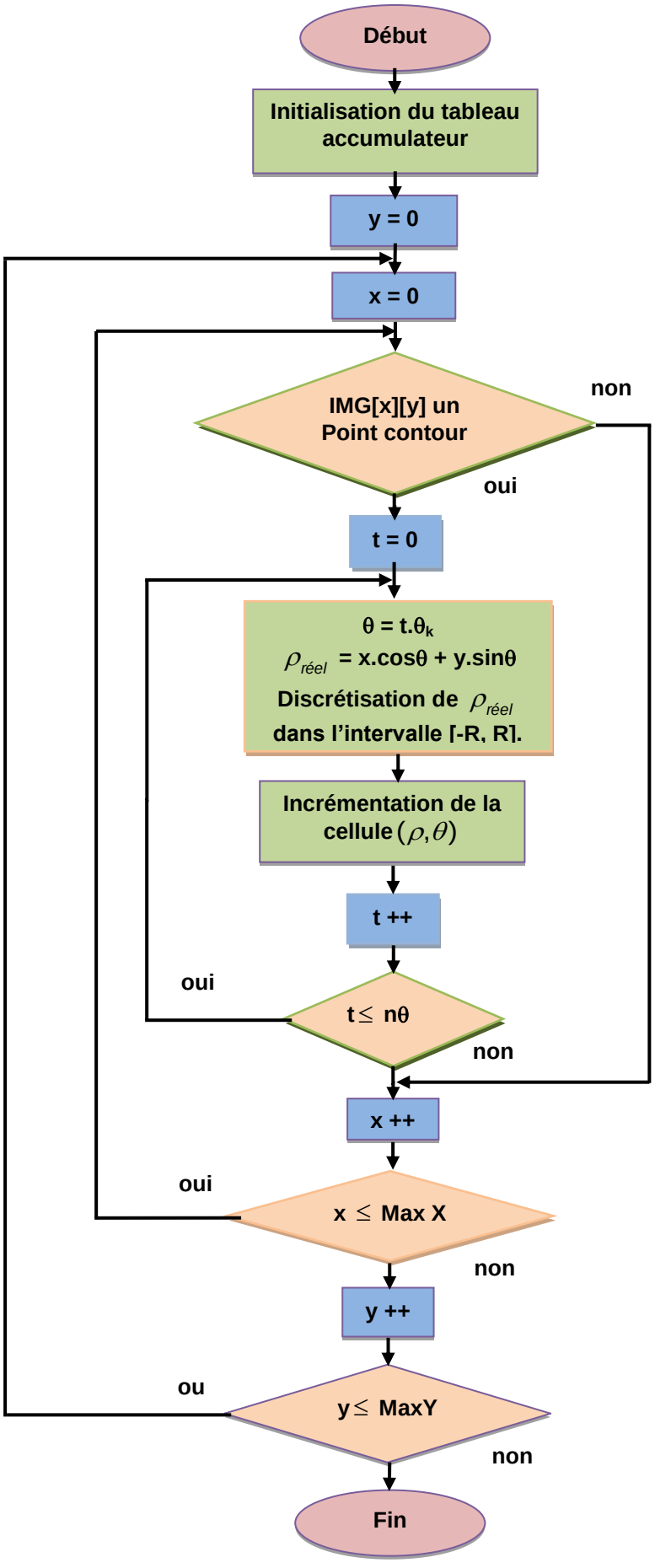


Figure 3.16. : Organigramme de la TH.

A la fin de ce calcul, une valeur M se trouvant dans une cellule quelconque (ρ , θ) du tableau accumulateur indique que *M* points de l'image appartiennent à une même droite dont les paramètres polaires sont ρ et θ. Mais, nous ne pouvons pas savoir que ces M points de l'image sont rapprochés ou dispersés.

C'est le problème d'existence de fausses droites ou droites insignifiantes sur la Figure 3.17.

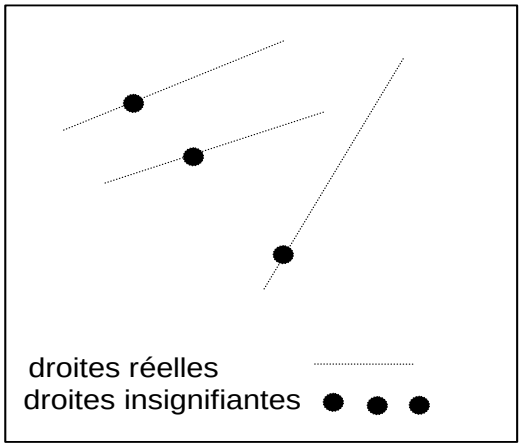


Figure 3.17. : Droites réelles et insignifiantes.

Donc pour avoir une bonne détection des droites significatives, il faut faire un compromis entre les trois seuils *seuil H*, *seuil D* et *seuil C* suivant le domaine d'application.

Par des tests d'expérimentation nous avons implémenté la Transformée de Hough (TH) sur la station Bull. DPX2000, en langage C sous Unix, et nous l'avons appliquée sur une image en niveaux de gris de dimension 256x256 (Figure 3.18).

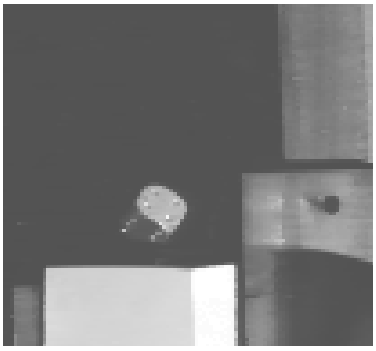


Figure 3.18. : Image en niveaux de gris.

La détection de contours est faite par l'opérateur "Roberts" avec un seuil de 30 sur la Figure 3.19.

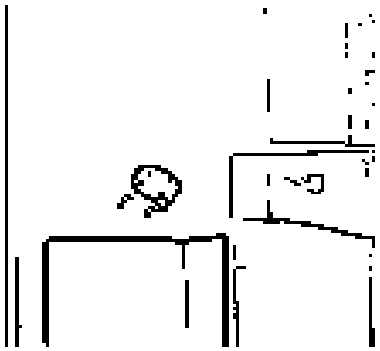
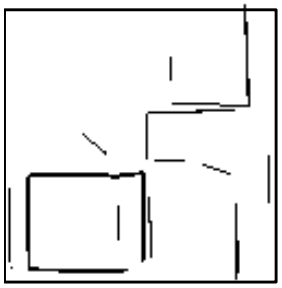


Figure 3.19. : Image contour.

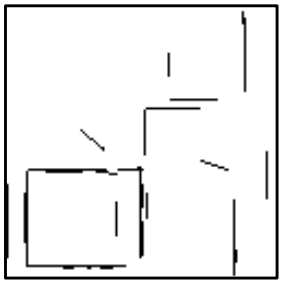
Si nous reprenons nos tests et que l'on désire comparer le temps de traitement " t_{Tr}" pour la détection des droites significatives, en d'autres termes plus équivoque entre le temps de traitement qui tient compte de toutes les cellules accumulatrices du tableau accumulateur, c'est-à-dire les cellules qui forment les vallées et les pics - (se référer aux images résultats *IM2, IM4, IM6* et *IM8* de la Figure 3.20) et le temps de traitement qui ne tient compte que des cellules accumulatrices qui forment les pics, après élimination des cellules accumulatrices qui forment les vallées- (images résultats *IMM2*, *IMM4*, *IMM6* et *IMM*8 pour les différentes valeurs du seuil D de la même Figure 3.20).

Nous constatons par conséquent que nous avons obtenu les résultats suivants:

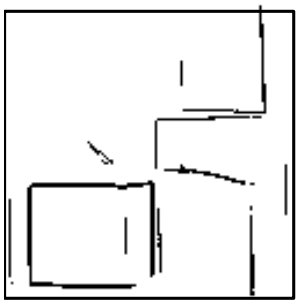
Par hypothèse, nous prenons le *seuil H* égale au *seuil C*, égaux tous les deux à 10 pixels, c'est à dire que nous ignorons toute droite ou segment de droite dont la longueur est supérieure à 10 pixels, les pas de quantification de ρ et θsont : $\rho_k=1$ et $\theta_k=10°$.



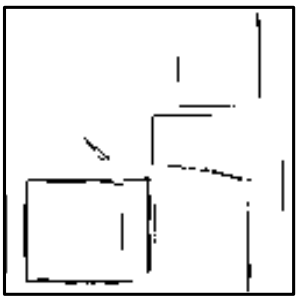
IM2
Seuil D = 2, t_{Tr} = 5s



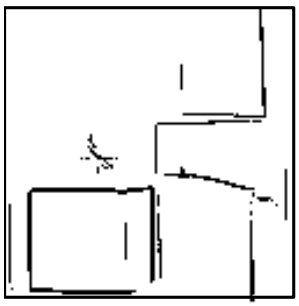
IMM2
Seuil D = 2, t_{Tr} = 2s



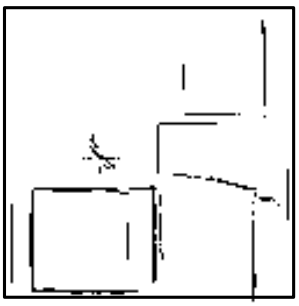
IM4
Seuil D = 4 t_{Tr} = 5s



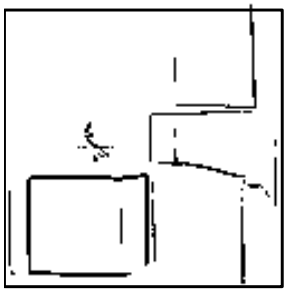
IMM4
Seuil D = 4, t_{Tr} = 2s



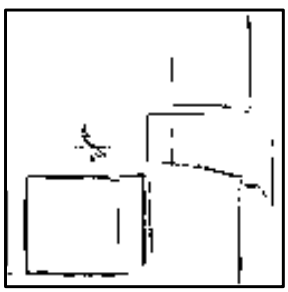
IM6
Seuil D = 6, t_{Tr} = 5s



IMM6
seuil D = 6, t_{Tr} = 2s



IM8
Seuil D = 8, t_{Tr} = 5s



IMM8
seuil D = 8, t_{Tr} = 2s

Figure 3.20. : Images Résultats t_{Tr} est le temps de traitement de la TH.

Nous remarquons aussi qu'en ignorant les cellules accumulatrices qui forment les vallées, le temps de traitement de la détection des droites significatives diminue considérablement. Ce temps de traitement diminue encore si nous augmentons le seuil H et le seuil C, éliminant ainsi les segments de droite de longueur moyenne.

3.2.5. Etat de l'art : Evaluation en ligne d'Algorithmes de la TH

3.2.5.1. Algorithme de H. Koshimizu et M. Numada [82]

L'algorithme de H. Koshimizu et M. Numada se présente comme suit :

$$\rho_{n+1} = \rho_n + \varepsilon\rho'_n \tag{3.10}$$

avec : $0 \le 1 \prec \frac{K}{2} - 1$

$$\rho'_{n+1} = \rho'_n - \varepsilon.\rho_{n+1} \tag{3.11}$$

avec : $\frac{K}{2} - 1 \le n \prec K$

Sachant que n et k sont respectivement l'indice et le nombre de divisions de l'axe θ dans l'espace des paramètres.

Dans le mode de traitement en ligne, à chaque cycle de calcul, les équations de récurrence sur les entrées et les sorties sont données par :

$$\rho_{n+1}[J] = \rho_{n+1}[J-1] + r_{n+1}[J].b^{-j} \tag{3.12}$$

$$\rho_n[J] = \rho_n[J-1] + r_n[J]b^{-j} \tag{3.13}$$

$$\rho'_{n+1}[J] = \rho'_{n+1}[J-1] + r'_{n+1}[J].b^{-j} \tag{3.14}$$

$$\rho'_n[J] = \rho'_n[J-1] + r'_n[J]b^{-j} \tag{3.15}$$

L'expression de la sortie ρ_{n+1} au pas J s'écrit :

$$\rho_{n+1}[J] = \rho_n[J] + \varepsilon\rho'_n[J] \tag{3.16}$$

En remplaçant $\rho_{n+1}[J]$, $\rho_n[J]$, $\rho'_n[J]$ et $\rho'_{n+1}[J]$ par les systèmes (3.12), (3.13), (3.14) et (3.15), on obtient le résidu partiel *Res* défini comme suit :

$$Res[J] = 2^{-j}.\left|2^{-p}.(\rho_n[J] + \varepsilon.\rho'_n[J] - \rho_{n+1}\right| \tag{3.17}$$

Pour la convergence de l'algorithme, il faut que :

$$2^{-2}.|Res[J]| \le 2^{-J}/2 \tag{3.18}$$

$$|Res[J]| \leq 1/2 \quad (3.19)$$

d'où :

$$Res[J] = 2.Res[J-1] + L[J] - r_{n+1}[J] \quad (3.20)$$

L'expression de l'accumulation partielle :

$$L[J] = 2^{-p} . (r_n[J] + \varepsilon . r'_n[J]) \quad (3.21)$$

Le calcul du résidu complet H[J] est alors défini comme suit:

$$H[J] = 2.Res[J-1] + L[J] \quad (3.22)$$

Où :

$$H[J] = 2.H[J-1] + L[J] - 2.r_{n+1}[J-1] \quad (3.23)$$

A partir de l'équation (3.19), on aura:

$$-3/2 \leq H[J-1] \leq 3/2 \quad (3.24)$$

Le problème est de définir la valeur minimale du retard p pour lequel le processus de calcul converge vers r_{n+1}. Pour cela, il est nécessaire qu'à chaque cycle de calcul, la condition d'arrondissement suivante soit vérifiée:

$$Max(2 . Res[J-1] + L[J]) \leq 3/2$$

$$\rho \geq log2(|2 . Max(r_n[J] + \varepsilon . r'_n[J])|)$$

avec : $Max(r_n[J]) = 1$ et $Max(r'_n[J]) = 1$

$$\varepsilon = \pi/100 \Leftrightarrow \varepsilon = 2^{-5} = 0,0313$$

on aura : $p \geq 1.0446 \Rightarrow p=2$

Le même procédé mathématique sera appliqué à l'équation

$$\rho'_{n+1}[J] = \rho'_n[J] - \varepsilon.\rho_{n+1}[J]$$

Donc l'expression du retard p est donné par :

$$\rho \geq log2(|2 . Max(r'_n[J] - \varepsilon . r_{n+1}[J])|) \Rightarrow p=2$$

Procédure de sélection :

A partir des équations (3.22) et (3.23), on peut déduire la procédure de sélection du bit résultat $r_{n+1}[J]$:

$$H[J] = 2.H[J-1] + L[J] - 2.r_{n+1}[J-1] \quad (3.24)$$

avec : $r_{n+1}[J] = S(H[J])$

3.2.5.2. Algorithme de S. Tagzout et al. [83]

Les nouvelles expressions de la TH incrémentale s 'écrivent comme suit :

$$\rho_{n+1} = \rho_n + \varepsilon.\rho_{n+K/2} \tag{3.25}$$

$$\rho_{n+1+K/2} = \rho_{n+K/2} - \varepsilon;\rho_n \tag{3.26}$$

avec : $\rho_{n+K/2} = \rho'_n$, $\rho_o = x$ et $\rho_{K/2} = \rho'_0 = y$

Dans le mode de calcul en ligne les expressions des entrées et sorties sont données par :

$$\rho_{n+1}[J] = \rho_{n+1}[J-1] + r_{n+1}[J].b^{-j} \tag{3.27}$$

$$\rho_n[J] = \rho_n[J-1] + r_n[J].b^{-j} \tag{3.28}$$

$$\rho_{n+K/2}[J] = \rho_{n+K/2}[J-1] + r_{n+K/2}[J].b^{-j} \tag{3.29}$$

$$\rho_{n+1+K/2}[J] = \rho_{n+K/2}[J-1] + r_{n+1+K/2}[J].b^{-j} \tag{3.30}$$

L'expression de la sortie ρ_{n+1} au pas J s'écrit :

$$\rho_{n+1}[J] = \rho_n[J] + \varepsilon.\rho_{n+K/2}[J] \tag{3.31}$$

Le résidu partiel *Res* est défini comme suit :

$$Res[J] = 2^{-j}.\left|2^{-p}.(\rho_n[J] + \varepsilon.\rho_{n+K/2}[J] - \rho_{n+1}\right| \tag{3.32}$$

Pour la convergence de l'algorithme, il faut que :

$$2^{-J}.|Res[j]| \le 2^{-j}/2 \tag{3.33}$$

avec: $|Res[J]| \le 1$

d'où:

$$Res[J] = 2.Res[J-1] + L[J] - r_{n+1}[J] \tag{3.34}$$

L'expression de l'accumulation partielle : $L[J] = 2^{-p}.(r_n[J] + \varepsilon.r_{n+K/2}[J])$.

Le calcul du résidu complet $H[J]$est :

$$H[J] = 2.Res[j-1] + L[J] \tag{3.35}$$

$$H[J] = 2.H[J-1] + L[J] - 2.r_{n+1}[J-1] \tag{3.36}$$

A partir de l'équation (3.33), on aura :

$$-3/2 \le H[J-1] \le 3/2$$

La valeur minimale du retard p pour lequel le processus de calcul converge vers r_{n+1} est définie à partir de la condition d'arrondissement suivante:

$$Max(\ H[J]\ ) \le 3/2$$

$$Max(\ 2.Res[J-1]+L[J]\ ) \le 3/2$$

$$p \ge log_2(\ |\ 2\ .Max(\ r_n[J]+\varepsilon\ .r_{n+K/2}[J])\ |\ )$$

avec $Max(\ r_n[J]\ ) = 1$ et $Max(\ r_{n+K/2}[J]\ ) = 1$

$\varepsilon = \pi/100 \quad \Leftrightarrow \quad \varepsilon = 2^{-5} = 0{,}0313$

On aura : $p \ge 1{,}0446 \quad \Rightarrow \quad p = 2$

Le même procédé mathématique sera appliqué à l'équation

$$\rho_{n+1+K/2} = \rho_{n+K/2} - \varepsilon\ .\rho_n$$

donc l'expression du retard *p* est donné par :

$$p \ge log_2\ (\ |\ 2\ .Max(\ r_{n+K/2}[J]-\varepsilon.r_n[J]\ )\ |\ ) \quad \Rightarrow \quad p = 2$$

Procédure de sélection :

À partir des équations (3.35) et (3.36) on peut déduire la procédure de sélection du bit résultat $r_{n+1}[J]$:

$$H[J] = 2\ .H[J-1]+L[J]-2\ .r_{n+1}[J-1] \qquad (3.37)$$

$$r_{n+1}[J] = S(\ H\ [J]\ )$$

3.2.5.3. Algorithme proposé [84].

Nous venons de voir dans les parties précédentes la modélisation de la TH incrémentale ainsi que l'algorithme de S. Tagzout et al dans le mode de traitement en ligne.

Nous venons de voir dans le paragraphe précédent, que de la TH incrémentale proposée dans [83], conduit à l'utilisation de la TH standard pour corriger les erreurs de calcul, ce qui nécessite l'utilisation des LUTs. De cela, on a essayé d'améliorer l'algorithme de S. Tagzout et a la fin d'optimiser les performances de calcul global de la TH pour l'utiliser dans les applications en temps réels. Tout le travail a été concentré pour trouver un algorithme qui n'utilise plus la TH standard et augmente le nombre de ρ générés pour des intervalles très réduit de θ, ce qui permet d'éliminer l'utilisation des LUTs et de ne plus effectuer des calculs trigonométriques au cours du

traitement, ainsi l'espace occupé et le temps globale de calcul seront réduits. L'idée de base de cet algorithme est de générer M valeurs de ρ dans un intervalle réduit de θ, où les approximations sur s*in* et c*os* n'engendrent pas d'erreurs significatives sur les résultats.

En utilisant les mêmes approximations sur c*os* et s*in* et les mêmes notations du paragraphe précédent, on obtient pour chaque valeur de θ_n, *M* valeurs de ρ générées en même temps dans l'intervalle $[0, \pi/M]$ de θ.

Notre algorithme est défini par l'expression générale suivante :

$$\begin{cases} 0 \le m \prec M \quad et \quad 1 \prec M \le K \\ \\ 0 \le n \prec K/M \\ \rho_{n+1+m.K/M} = \rho_{n+m.K/M} + \alpha.\varepsilon.\rho_{n+m.K/M+\alpha.K/2} \\ \rho_{mK/M} = x.cos\left(\frac{mK}{M}\varepsilon\right) + y.sin\left(\frac{mK}{M}\varepsilon\right) \\ \alpha = \begin{cases} 1 & si \quad m \prec M/2 \\ -1 & si \quad m \ge M/2 \end{cases} \end{cases} \tag{3.38}$$

avec :

- ε est la résolution de θ.
- M est le nombre de valeurs de ρ générées en même temps.
- K représente le nombre de division de θ.
- ρ_i est la valeur de ρ obtenue pour un angle θ_i de l'axe de θ.

Cette expression peut être réarrangée pour aboutir au système d'équations suivant:

$$\begin{cases} 0 \le m < \frac{M}{2} \ , \ 1 < M \ < K \ \ et \ \ 0 \le n < \frac{K}{M} \\ \\ \rho_{n+1+\frac{m.K}{M}} = \rho_{n+\frac{m.K}{M}} + \varepsilon.\rho_{n+\frac{m.K}{M}+\frac{K}{2}} \\ \rho_{n+1+\frac{m.K}{M}+\frac{K}{2}} = \rho_{n+\frac{m.K}{M}+\frac{K}{2}} - \varepsilon.\rho_{n+\frac{m.K}{M}} \\ \\ \rho_{\frac{m.K}{M}} = x.cos\left(\frac{m.K}{M}.\varepsilon\right) + y.sin\left(\frac{m.K}{M}.\varepsilon\right) \\ \rho_{\frac{m.K}{M}+\frac{K}{2}} = y.cos\left(\frac{m.K}{M}.\varepsilon\right) - x.sin\left(\frac{m.K}{M}.\varepsilon\right) \end{cases} \tag{3.39}$$

Pour cette nouvelle expression, on note que :

Les valeurs initiales sont calculées par les deux équations $\left(\rho_{\frac{mK}{M}}\right)$ et $\left(\rho_{\frac{mK}{M}+\frac{K}{2}}\right)$.

Pour calculer la prochaine valeur de ρ , (ρ_{n+1}) dans l'intervalle :

$$\left[\frac{m.K}{M},\frac{(m+1).K}{M}\right]$$

On a besoin de l'ancienne valeur de ρ , (ρ_n)calculée dans le même l'intervalle et l'ancienne valeur de $\rho\left(\rho_{n+\frac{K}{2}}\right)$ calculée dans l'intervalle $\left[\frac{m.K}{M}+\frac{K}{2},\frac{(m+1).K}{M}+\frac{K}{2}\right[$ en même temps et vice versa.

Les valeurs de ρ calculés dans les intervalles :

$$\left[\frac{m.K}{M},\frac{(m+1).K}{M}\right[\text{ et } \left[\frac{m.K}{M}+\frac{K}{2},\frac{(m+1).K}{M}+\frac{K}{2}\right[$$

Sont générées en même temps ce qui réduit le temps de calcul global de la TH.

La valeur de *M* doit être paire et la valeur de $\frac{K}{M}$ doit être entière.

3.3. Intérêt de l'hybridation

On retrouve dans les recherches sur les systèmes hybrides l'ambivalence des approches possibles en résolution de problèmes: certains veulent construire de meilleurs outils informatiques pour résoudre des problèmes, d'autres veulent construire de meilleurs modèles cognitifs et beaucoup ont des démarches intermédiaires en utilisant des idées provenant des sciences cognitives pour obtenir de meilleurs outils informatiques [61] et [85].

Que ces recherches soient plutôt informatiques ou plutôt cognitives, elles ont un point commun important: elles sont toutes motivées par l'incapacité actuelle d'un paradigme donné (connexionniste, symbolique ou autre) à résoudre, à lui seul, des plusieurs types de problèmes. Etant donné qu'il est très difficile d'inventer de toutes pièces un nouveau paradigme plus satisfaisant, il serait avantageux de tirer parti des points forts de plusieurs paradigmes et réaliser des systèmes ou modèles hybrides.

Certains chercheurs vont encore plus loin en donnant des justifications et des motivations cognitives à la réalisation de modèles hybrides [60], [61] et [86] :

- Il est indispensable de disposer de plusieurs modes complémentaires d'expression des connaissances, de manière à pouvoir représenter le savoir-que et le savoir-faire concernant un problème donné.
- Les systèmes hybrides sont une nécessité, en raison de la complexité des systèmes cognitifs. Quand un chercheur examine les différents aspects d'un problème cognitif et sélectionne les niveaux de description les plus utiles, le modèle qui émergera sera un patchwork de composants de divers types, essentiellement symboliques et connexionnistes, donc un modèle hybride.

Du point de vue de l'ingénieur des connaissances ou cogniticien, les arguments en faveur de l'utilisation conjointe des techniques développées dans le cadre des approches symbolique et connexionniste ne manquent pas [87]. Face aux problèmes posés par la modélisation de la cognition humaine (et a fortiori, de l'expertise), il faut constater :

- L'échec de l'option 'tout-symbolique' tel que l'a révélé l'impossibilité de tenir les promesses du programme de l'IA définies par les pères fondateurs.
- Le peu de chances d'aboutir, même à moyen terme, à des résultats probants si l'option 'tout-connexionniste' est choisie.

Hatzilygeroudis I., Prentzas J. [62] définissent deux sous-classes de l'approche représentationnelle.

- Orientée connexionnisme : cette catégorie donne la prééminence aux concepts connexionnistes (en offrant moins de capacités de modularité) comme c'est le cas des systèmes experts connexionnistes.
- Orienté symbolique : cette catégorie donne la prééminence aux concepts symboliques (en offrant moins de capacité de généralisation) comme c'est le cas des systèmes experts connexionnistes.

Boudjemaa et al. [88] proposent une contribution à l'identification automatique de personnes à partir de l'analyse texturale de l'image de l'iris de l'œil. La technique utilisée consiste à extraire, tout d'abord, l'image de l'iris à partir de l'image globale de l'œil saisie par caméra en utilisant la transformation de Hough combinée avec l'opérateur de Canny. Après une transformation de cette image en coordonnées polaires étirées, le filtre de Gabor est, ensuite, appliqué à l'image rectangulaire obtenue.

Notre étude nous a amené à mieux comprendre la tendance vers les systèmes hybrides neuro-mimétiques ainsi que l'inexistence, du moins à notre connaissance à travers la littérature, d'une orientation explicite vers ces approches dans un domaine tel que celui de la Reconnaissance de formes et plus particulièrement en vision robotique.

3.4. Conclusion

Dans ce chapitre nous avons passé en revue les méthodes d'estimation robuste le plus utilisées en vision robotique. L'utilisation de ces méthodes est nécessaire afin de réaliser des tâches en environnement réel. Le prix à payer est un temps de calcul un peu plus élevé et une vitesse de convergence réduite. Si les techniques de vote (Hough, Ransac) sont très efficaces, le temps de calcul est souvent trop élevé pour assurer une utilisation des algorithmes de vision à une cadence compatible avec la commande d'un robot.

Finalement, cette technique est capable de segmenter les données en plusieurs populations qui vérifient le modèle de référence. Toutefois, la Transformée de Hough est très rarement utilisée seule en vision robotique car pour des problèmes qui nécessitent l'estimation de plus de trois ou quatre paramètres, les temps de calculs deviennent prohibitifs. Pour parer à cela, une hybridation avec les RNA's représente un bon compromis entre robustesse et efficacité algorithmique.

En effet, la Transformée de Hough, ainsi soit-elle est une méthode de vote très robuste. La version originale de la méthode proposée par Hough a été modifiée par [21].

Depuis plusieurs variantes ont été proposées [73]. Cette approche repose sur une discrétisation de l'espace des paramètres. On obtient alors des hyper-cubes dans l'espace d'état auquel sont associés des accumulateurs. Pour un jeu de données de

taille minimale, les paramètres recherchés sont estimés et l'accumulateur correspondant de l'hyper-cube est incrémenté. Ce processus est itéré jusqu'à considérer toute les combinaisons possibles des données à disposition. L'accumulateur ayant la valeur la plus importante correspond alors à la meilleure estimation des paramètres.

La Transformée de Hough est bien adaptée aux problèmes ayant un nombre important de données par rapport aux nombre des paramètres à estimer. En effet, si les données et les inconnues sont de taille équivalente, il est difficile de trouver un accumulateur prépondérant par rapport aux autres. En plus, dû à la discrétisation et au bruit, il est possible que l'optimum soit délocalisé. La Transformée de Hough est très robuste car elle effectue une recherche globale et exhaustive.

L'implémentation de la Transformée de Hough sur un système informatique a l'avantage de présenter un large éventail de tâches à exécuter en parallèle. Ce dernier peut exécuter d'autres algorithmes en plus de la Transformée de Hough et être capable de donner une performance aux calculs antérieurs tels qu'une segmentation ou aussi en amont une détection de contours.

Dans ce chapitre, nous avons présenté l'évaluation de la Transformée de Hough en utilisant le mode de calcul en ligne. L'algorithme élaboré permet de générer tous les paramètres de la Transformée de Hough d'une manière très rapide et ceci grâce au mode de présentation des données en série, poids forts en tête. Les performances de l'architecture pipeline résident essentiellement dans la manière de génération des bits résultats r_{n+1} et $r_{n+1+k/2.}$

Pour une meilleure illustration, nous avons implémenté notre algorithme sur un circuit FPGA de Xilinx, plus précisément la famille Virtex2, qui contient un module de génération des paramètres (ρ,θ) ainsi que le module nécessaire pour le processus de vote.

Cette dernière dispose de ressources dédiées aux opérations arithmétiques et aussi dans le domaine de traitement de signal. Le circuit donne de très bons résultats en termes de performances temporelles.

L'idée d'utiliser la TH est due à sa robustesse dans l'analyse de l'extraction des primitives d'objets et aussi à ses propriétés d'implémentation hardware.

La Transformée de Hough Incrémentale Généralisée (THIG) permet d'optimiser en temps et en espace le calcul global de la TH, ce qui justifie notre choix de l'étudier et de l'implémenter dans notre travail de recherche. Par conséquent, dans le chapitre

qui suit, nous présentons un système de reconnaissance de formes (Rdf) moyennant un processus hybride basé sur une approche connexionniste associant la TH. Cette étape est très importante pour notre expérimentation .en vision robotique car elle solutionne une des problématiques de la perception et de la compréhension automatiques de l'environnement.

Une autre façon de justifier l'utilisation conjointe des techniques et concepts développés dans le cadre des approches hybrides émerge de l'étude de l'expertise et plus généralement de la cognition humaine. Un être humain est hybride dans la mesure où ses concepts sont hybrides. Ce constat a suscité notre intérêt en associant une approche neuronale à la Transformée de Hough que nous commenterons dans nos expérimentations au chapitre 4.

CHAPITRE 4

Application à la segmentation d'images en vision robotique

Nous présentons dans ce dernier chapitre en l'occurrence le quatre, l'application directe de notre expérimentation qui prend en charge le calcul d'orientation des primitives, donnée importante dans le processus de modélisation du milieu dans des tâches de localisation et de navigation de notre robot mobile que ce soit l'ATRV2 où le CESA évoluant dans un environnement de bureau où d'atelier. L'objectif est de fournir des informations suffisamment pertinentes et concises afin de faciliter les traitements ultérieurs notamment l'aide à la décision. En effet, nous définissons en amont le système de perception utilisé par le robot mobile, ensuite, nous définissons la représentation des mesures issues de ce système de perception, en justifiant d'abord le choix du modèle de l'environnement de navigation utilisé tout au long de ce travail, ainsi que le modèle géométrique des capteurs du système de perception.

Par suite, une méthode de segmentation utilisée dans notre processus de reconnaissance de formes pour la vision en robotique mobile est illustrée dans ce chapitre, par la même que la validation des images résultats obtenus.

4.1. Modèle de l'environnement du système mobile

4.1.1. Système de perception utilisé.

Le type d'environnement dans lequel devra se déplacer le système mobile détermine le type de capteurs à utiliser pour la perception : des caméras, des proximètres, et/ou des capteurs tactiles. Aucun d'entre eux ne peut à lui seul rendre compte de tout ce que le système mobile doit connaître sur le monde qui l'entoure.

La perception de chacun d'eux est partielle et entachée d'erreurs. La solution consistera donc, à faire coopérer un grand nombre de capteurs. Cette coopération est très intéressante, voire indispensable pour la réalisation des objectifs de la perception. Elle peut se faire entre des capteurs de nature différente (vision et capteurs télémétriques, par exemple) permettant la perception de caractéristiques totalement différentes des objets, ou de même type placés à différents endroits diminuant ainsi les problèmes dus à l'occultation, de la résolution ou du champ de vue. Dans ce qui suit, nous décrivons le système de perception du robot mobile *ATRV2* à savoir les capteurs à ultrasons et le banc stéréoscopique.

4.1.1.1. Capteurs à ultrasons [89]

Le capteur à ultrasons présente l'avantage de donner directement une information de distance mais cette mesure est assez imprécise. En effet, l'angle d'ouverture (allant de 20° à 30°) introduit un facteur d'incertitude concernant la direction dans laquelle se situe l'obstacle perçu. Pour une mesure *d* donnée par le capteur, en considérant un espace à deux dimensions et en négligeant les incertitudes sur la mesure, la région correspondant aux positions possibles de l'obstacle détecté prend la forme d'un arc centré sur le capteur, avec un rayon égal à d.

Les dimensions de l'objet sont aussi masquées par cette propriété du capteur à ultrasons : il peut s'agir d'un mur ou seulement d'un objet de petite taille.

La Figure 4.1 illustre la situation où deux obstacles possibles produisent la même mesure.

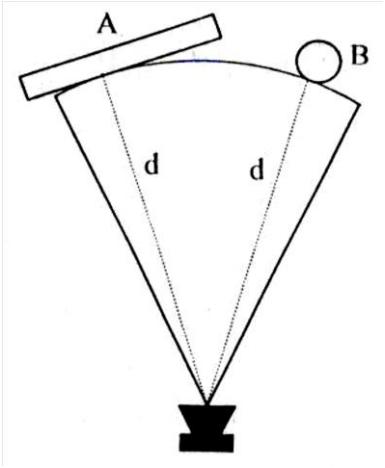


Figure 4.1.: Deux obstacles renvoyant la même mesure.

Une incertitude sur la distance mesurée, appelée également incertitude radiale, est aussi présente. Elle provient des phénomènes atmosphériques (température, courants d'air, etc.) pouvant modifier la vitesse de l'onde. Pour les capteurs utilisés comme émetteur et récepteur tels que ceux que nous utilisons, une distance minimale détectable est définie. Cette distance est déduite du temps nécessaire pour que la membrane du capteur puisse se stabiliser après l'émission de l'onde (06 cm pour le cas des capteurs US du mobile ATRV2). Quand une onde acoustique heurte un objet, l'écho détecté représente seulement une petite partie du signal original. L'énergie restante est réfléchie dans des directions dispersées et peut être absorbée par la cible ou être passée à travers elle, ceci dépendant de la surface de l'objet et de l'angle d'incidence du rayon (Figure 4.2).

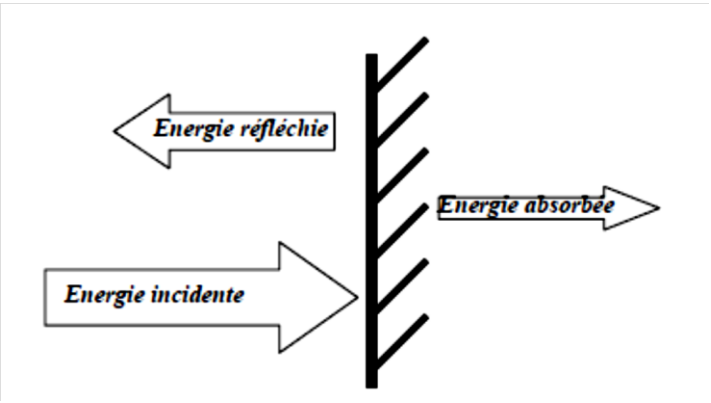


Figure 4.2.: Répartition des énergies.

Par ailleurs, plus la distance parcourue par l'onde est grande plus cette dernière est atténuée. Ainsi, une distance maximale correspondant au minimum d'énergie détectable est définie comme limite de la portée du capteur (04 m pour l'ATRV2).

Dans le domaine de perception du capteur à ultrasons, des phénomènes tels que la spécularité, les réflexions multiples ou la diaphonie peuvent intervenir, au demeurant nous décrivons les causes et les effets.

→ **a.1. La spécularité :** dite effet miroir peut provoquer la non détection des obstacles. En effet, la fiabilité des mesures ultrasonores est très dépendante de la texture de la surface réfléchissante de l'objet. Les surfaces rugueuses retournent l'onde ultrasonore produite par le capteur quel que soit l'angle d'incidence, contrairement aux surfaces lisses qui ne renvoient que les ondes ayant un angle d'incidence proche d'un angle droit (Figure 4.3).

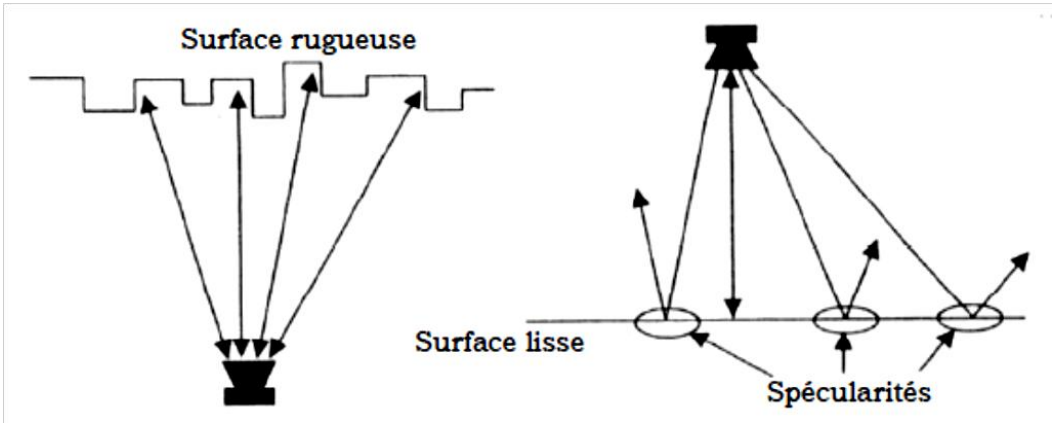


Figure 4.3.: Phénomène de spécularité.

Si une spécularité intervient, la mesure issue du capteur concerné est inexploitable, ce qui peut être gênant puisque nous n'avons que 12 mesures de capteurs à ultrasons pour décrire l'environnement autour du système mobile ATRV2. Si l'onde ne se réfléchit pas sur un autre obstacle pour revenir sur le capteur, la mesure sera annulée; par contre, si la spécularité est suivie d'une ou plusieurs réflexions sur les obstacles, nous avons alors des réflexions multiples.

→ **a .2. Les réflexions multiples :** Le phénomène de spécularité peut se produire plusieurs fois, il s'ensuit des réflexions multiples. Le signal ultrasonore peut donc se réfléchir sur plusieurs surfaces avant de retourner au capteur. Par ce fait, l'obstacle paraît plus loin que son emplacement réel (Figure 4.4). Les mesures issues de réflexions multiples ne correspondent pas à un obstacle de l'environnement et risquent de fausser la localisation en associant la mesure à un autre obstacle situé plus loin.

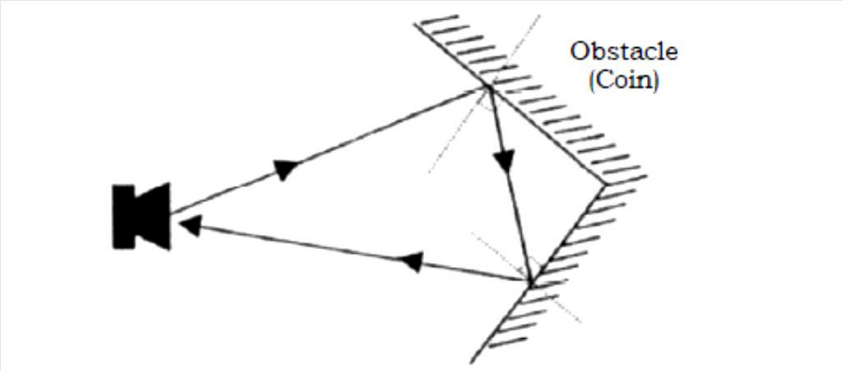


Figure 4.4.: Exemple de réflexions multiples.

→ **a .3. La diaphonie :** La diaphonie intervient quand un capteur perçoit l'onde émise par un autre. Supposons qu'un capteur *x* émette une onde ultrasonore, le cône émis est renvoyé vers le système mobile. Si le capteur *y* se trouve en mode de réception, il interprétera l'onde émise par le capteur *x* et renvoyée par l'obstacle comme étant la sienne, et percevra un objet plus proche qu'il ne devrait. Ce phénomène est appelé diaphonie par chemin critique direct (Figure 4.5.a). Le capteur *y* peut aussi recevoir une onde issue de réflexions multiples, il s'agit alors de diaphonie par chemin critique indirect (Figure 4.5.b).

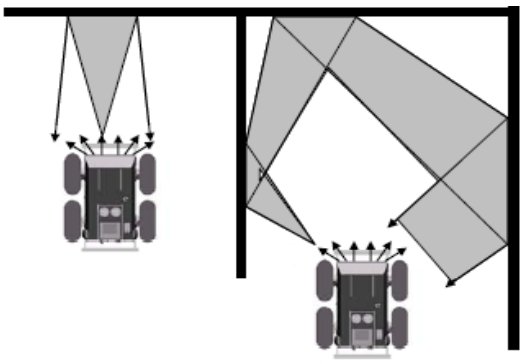


Figure 4.5.: Exemple de diaphonie.

Là encore ce type de mesures représente des obstacles fictifs et peut fausser la localisation des objets; le regroupement des capteurs sous forme de nœuds vise à limiter la diaphonie en évitant d'activer simultanément des capteurs voisins. La spécularité, les réflexions multiples et la diaphonie sont des phénomènes parasites plus ou moins imprévisibles contrairement aux incertitudes angulaire et radiale qui sont toujours présentes dans la mesure. En conséquence, le modèle que nous proposons représente les incertitudes angulaires et radiales mais ne tient pas compte des autres phénomènes. Il est toutefois important de connaître leur existence afin de prendre en considération la possibilité de mesures aberrantes dont il faudra limiter l'influence lors de la reconstruction de l'environnement.

Le système mobile ATRV2 (Figure 4.6.a) est équipé de douze capteurs à ultrasons montés en horizontal autour du système (six à l'avant, deux à l'arrière et deux sur chaque côté). La portée de leur mesure s'étend sur une distance allant de six centimètres (6cm) jusqu'à quatre mètres (4m). Le lobe de la fonction de sensibilité est contenu dans un angle d'ouverture de 30°. Grâce à ces capteurs, l'ATRV2 peut acquérir rapidement une perception panoramique complète à 360° (Figure 4.6.b).

(a) (b)

Figure 4.6.: (a): Système mobile ATRV2, (b): Perception panoramique.

Cependant, tous ces capteurs sont actionnés en même temps et cela peut provoquer une diaphonie importante entre capteurs (causée par une résolution angulaire faible, des erreurs dues aux réflexions multiples ou des réflexions spéculaires loin du capteur).

4.1.1.2. Système de vision

Les systèmes de vision en robotique mobile sont basés une (2d) où plusieurs caméras CCD (3d), très performants en termes de portée, de précision et de quantité d'informations exploitables. En revanche, l'inconvénient majeur sur de tels systèmes de perception se situe au niveau de la gestion du flux important de données exploitables: traiter une image demeure une opération délicate et surtout coûteuse en temps de calcul. Le système de vision du système mobile ATRV2 est composé de deux caméras CCD couleurs montées sur une tourelle ''Pan Tilt'' (Figure 4.7). Leur utilisation a pour avantage de fournir des informations très riches sur l'environnement. A partir de ces informations tels que des points et segments de droite extraits de l'image de la scène observée, on peut contrôler le déplacement relatif et la position du système mobile vis à vis d'éléments caractéristiques de l'environnement.

Figure 4.7.: Système de vision (ATRV2).

4.1.2. Représentation des mesures issues des capteurs embarqués

a) Modèle de l'environnement

Le modèle de l'environnement que nous avons utilisé est le modèle métrique (grille de certitude) basé sur l'algorithme *HIMM* (Histogramic In Motion Mapping) présenté dans [90]. Il utilise une grille histogramme à deux dimensions pour la représentation des obstacles. Chaque cellule de cette grille possède une valeur de certitude CV qui représente la confiance de l'algorithme dans l'existence d'un obstacle à cet endroit. Ce modèle est exploitable dans tout type d'environnement

sans nécessiter d'aménagements particuliers, et peut être utilisé à partir des mesures issues de plusieurs types de capteurs.

Chaque capteur construit sa carte locale qui va ensuite être intégrée à une carte locale du système mobile servant à établir la carte globale de l'environnement.

La carte locale qui est centrée dans le système mobile et sa taille est ajustée sur la plus grande mesure du système de perception embarqué. A l'arrêt, les cellules occupées par le système mobile sont considérées libres. Lorsque le système mobile se déplace, il utilise sa position, son orientation et les mesures de son système de perception embarqué pour construire sa carte locale.

La carte globale de l'environnement de navigation définit l'espace ou l'endroit dans lequel le système mobile va se mouvoir. La valeur *CV* de ses cellules est initialisée à zéro. Au cours des déplacements du système mobile, chaque obstacle détecté incrémente la valeur *CV* des cellules qu'occupe cet obstacle, ainsi, les obstacles seront repérés ce qui permet de les éviter. Toutes les cartes locales sont intégrées pour former une seule carte (carte globale). Chaque carte locale est intégrée à l'ancienne carte globale pour donner une nouvelle carte globale pour une mise à jour de l'environnement.

En partant de l'idée de n'avoir aucune information complémentaire sur la mesure ultrasonore, notre contribution a consisté à considérer tous les points de l'arc sont équivalents pour une position possible de l'obstacle, il nous paraît logique d'attribuer la même valeur à chacune des cellules représentant le même état (libre ou occupé).

Notre modèle utilise cette idée en décrémentant de 1 les valeurs *CV* des cellules correspondant à l'espace libre jusqu'à un seuil minimum défini (*Min_Vote_Seuil*=0), et en incrémentant de 3 les valeurs *CV* des cellules de l'arc de cercle représentant l'obstacle jusqu'à un seuil maximum défini (*Max_Vote_Seuil*).

Les valeurs des cellules restantes ne sont pas modifiées. Le modèle du capteur à ultrasons ainsi obtenu et sa grille correspondante sont représentés par la Figure 4.8.

La règle de mise à jour des valeurs *CV* des cellules de la grille selon la Figure 4.8, est exprimée comme suit :

$$CV(X) = \begin{cases} CV(X)+3 & si \quad X \in B \\ CV(X)-1 & si \quad X \in A \\ CV(X) & si \quad X \in C \end{cases}$$

$$CV_{Min} \leq CV(X) \leq CV_{Max}$$

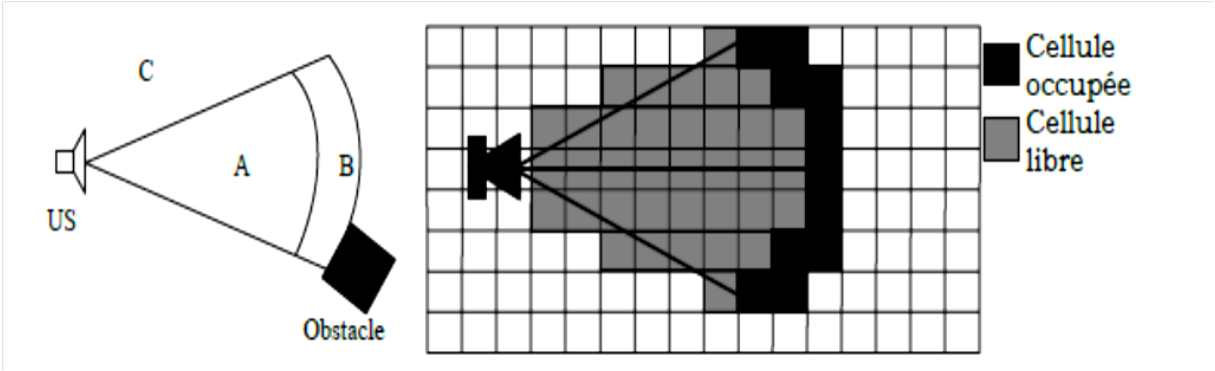


Figure 4.8.: Système de vision (ATRV2).

a.1.Calcul des paramètres du modèle géométrique d'une caméra : Méthode choisie.

Une caméra réalise une transformation ponctuelle qui fait passer d'un point physique de l'espace réel 3D à un point 2D sur le plan image de celui-ci. Ce qui revient à une transformation mathématique de R^3 vers R^2. On suppose généralement que la transformation réalisée est une projection centrale par rapport au centre de la caméra : c'est le modèle *Pinhole* ou modèle *Sténopé*.

Le modèle sténopé se caractérise par un plan de projection (le plan image) et un centre de projection (le centre optique *F*).

La Figure 4.9 montre qu'un point *B* de la scène se projette sur le plan image en un point *b* qui est l'intersection de la droite (*FB*) avec le plan image.

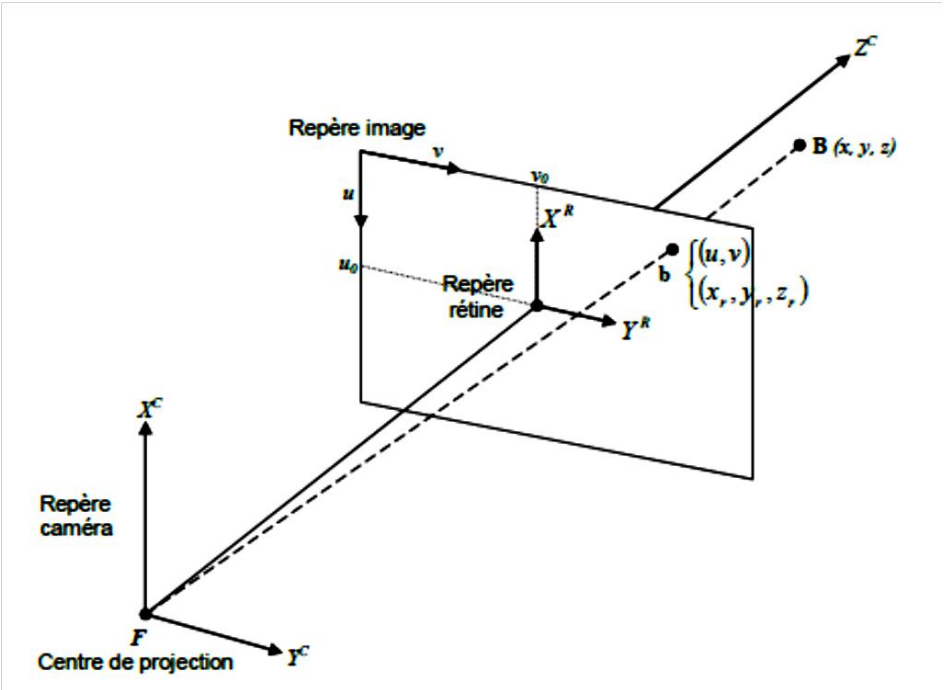


Figure 4.9.: Modèle sténopé d'une caméra.

Avec ce modèle, le processus d'obtention d'une image peut être décrit de manière synthétique par la matrice de projection perspective (ou encore matrice de transformation *3D-2D*). Il s'agit en fait de la matrice qui décrit la transformation qui permet de passer des coordonnées de B exprimées, par exemple, en millimètres dans un repère absolu tridimensionnel, à sa projection *b* sur le plan image décrite par ses coordonnées exprimées en pixels dans un repère bidimensionnel lié au plan image. Si les coordonnées de B dans le repère absolu sont (x,y,z) alors les coordonnées (u,v) de sa projection *b* dans le repère image sont obtenues par une transformation linéaire en coordonnées homogènes qui s'écrit sous la forme matricielle suivante :

$$\begin{pmatrix} su \\ sv \\ s \end{pmatrix} = M \begin{pmatrix} x \\ y \\ z \\ \boldsymbol{1} \end{pmatrix} \qquad (4.1)$$

avec : $\begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{pmatrix}$

Calibrer une caméra va consister à estimer la matrice de projection M de l'équation (4.1). Il suffit de disposer d'un ensemble de points de référence $P_i\ (x_i, y_i, z_i)$ appelés aussi points d'intérêt appartenant au repère absolu et de leur projection (u_i , v_i) dans l'image numérisée [91].

D'après l'équation (4.1), chaque correspondance donne un système de deux équations :

$$u_i = \frac{m_{11}x_i + m_{12}y_i + m_{13}zi + m_{14}}{m_{31}x_i + m_{32}y_i + m_{33}z_i + m\boldsymbol{34}}$$

$$v_i = \frac{m_{21}xi + m_{22}y_i + m_{23}z_i + m_{24}}{m_{31}x_i + m_{32}y_i + m_{33}z_i + m_{34}} \qquad (4.2)$$

Ces équations sont linéaires par rapport aux coefficients m_{ij} de la matrice de projection M, donc au moins six (06) points non coplanaires suffisent pour les

déterminer. L'équation (4.2) peut se réécrire comme une combinaison linéaire des m_{ij}:

$$\begin{aligned} &m_{11}x_i + m_{12}y_i + m_{13}z_i + m_{14} - u_i m_{31}x_i - u_i m_{32}y_i - u_i m_{33}z_i = u_i m_{34} \\ &m_{21}x_i + m_{22}y_i + m_{23}z_i + m_{24} - v_i m_{31}x_i - v_i m_{32}y_i - v_i m_{33}z_i = v_i m_{34} \end{aligned} \tag{4.3}$$

On obtient donc $2n$ équations pour n points et on peut écrire ces équations sous forme matricielle :

$$\begin{pmatrix} & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \\ x_i & y_i & z_i & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -u_i x_i & -u_i y_i & -u_i z_i \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & x_i & y_i & z_i & \mathbf{1} & -v_i x_i & -v_i y_i & -v_i z_i \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{14} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{24} \\ m_{31} \\ m_{32} \\ m_{33} \end{pmatrix} = \begin{pmatrix} \\ \\ \\ u_i m_{34} \\ v_i m_{34} \\ \\ \\ \end{pmatrix} \tag{4.4}$$

ou, sous forme plus condensée : $Kx = u$

Le système précédent étant homogène (*M* est défini à un facteur d'échelle près), il faut utiliser une contrainte supplémentaire. Nous utilisons pour cela la méthode de calibrage décrite par Faugeras et Toscani [91] et [92].

Si l'on écrit M sous la forme suivante :

$$M = \begin{pmatrix} m_1 & m_{14} \\ m_2 & m_{24} \\ m_3 & m_{34} \end{pmatrix}$$

où : $m_i = \begin{pmatrix} m_{i1} & m_{i2} & m_{i3} \end{pmatrix}$

Et si l'on pose la contrainte suivante : $\|m_3\| = 1$

Alors le système d'équations (4.4) peut s'écrire sous la forme suivante :

$$Bx_9 + Cx_3 = \mathbf{0}$$

où :

$$B = \begin{pmatrix} & & & & & & & & \\ & & & & & & & & \\ x_i & y_i & z_i & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -u_i \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & x_i & y_i & z_i & \mathbf{1} & -v_i \\ & & & & & & & & \\ & & & & & & & & \end{pmatrix}, C = \begin{pmatrix} & & \\ & \cdot & \\ & \cdot & \\ & \cdot & \\ -u_i x_i & -u_i y_i & -u_i z_i \\ -v_i x_i & -v_i y_i & -v_i z_i \\ & \cdot & \\ & \cdot & \\ & \cdot & \end{pmatrix} \qquad (4.5)$$

avec : $x_9 = \left(m_1 \quad m_{14} \quad m_2 \quad m_{24} \quad m_{34}\right)^t$ et $m_3 = m_3^t$.

Le critère à minimiser étant : $\left\|Bx_9 + Cx_3\right\|^2$

Sous la contrainte : $\left\|m_3\right\|^2 = \mathbf{1}$

Après calculs, l'estimation des coefficients de la matrice de projection M est donnée par les étapes suivantes [93] :

1. *Construction d'une matrice D telle que :*

$$D = C^t C - C^t B(B^t B)^{-1} B^t C$$

2. *Calcul des valeurs et vecteurs propres de D.*
3. *X_3 est le vecteur propre correspondant à la plus petite valeur propre.*
4. *Normalisation de x_3 :*

$$x_3 \leftarrow \frac{x_3}{\left\|x_3\right\|}$$

5. *Calcul de x_9 :*

$$x_9 = -\left(B^t B\right)^{-1} B^t C x_3$$

6. *Si* $m_{34} \le \mathbf{0}$ *alors* $M \leftarrow (-M)$

A partir des coefficients m_{ij} de la matrice de projection M et grâce aux équations de Transformation repère absolu/repère image, on peut donc déterminer les paramètres intrinsèques et extrinsèques de la caméra. Calibrer le banc stéréoscopique en mode test du système *ATRV2* revient à calibrer chaque caméra indépendamment en utilisant la même mire. La mire de calibrage fournit des points

de calibrage dont la position doit être connue avec une très grande précision dans un repère absolu.

La Figure 4.10 montre les mires de calibrage utilisées : la première pour le calibrage fort est constituée d'un ensemble de petits carrés noirs et blancs de 2.6 cm de longueur pour chacun de leur côté, et la seconde pour le calibrage faible est constituée d'un ensemble de petits cercles noirs de 1.0 cm de diamètre et espacés de 05 cm. Ces motifs ont été choisis pour que leur projection dans l'image numérisée puisse être mesurée avec une grande précision.

Le calibrage commence par le placement en différentes positions de la mire pour qu'elle soit perçue à la fois par les deux caméras. Le calibrage fort a été réalisé sur des paires d'images de 160 x 120 pixels de résolution, tandis que le calibrage faible a été réalisé sur des paires d'images de 600 x 560 pixels de résolution.

Les corrections de la distorsion radiale des deux caméras n'ont pas été prises en compte pendant le processus de calibrage.

(a) **(b)**

Figure 4.10.: Mise en œuvre du calibrage du banc stéréoscopique du systèmeATRV2.
(a) : en utilisant la première mire pour le calibrage fort.
(b) : en utilisant la deuxième mire pour le calibrage faible.

Les résultats issus du calibrage sont présentés en annexe. Pour valider les résultats du calibrage du banc stéréoscopique du système *ATRV2*, nous avons reconstruit les points 3D des deux mires utilisées et comparer les valeurs des points reconstruits avec leur vraie valeur acquise lors du calibrage (Figure 4.11). Nous remarquons qu'il y a une petite différence entre la position des points réels et les points calculés. Cela peut s'expliquer par le mode manuel de prélèvement des points 3D/2D des deux mires utilisées pour l'estimation des paramètres intrinsèques et extrinsèques des caméras gauche et droite du banc stéréoscopique.

Les points 3D de l'environnement obtenus lors des acquisitions sont filtrés avant d'être sauvegardés dans le modèle de l'environnement choisi. Le filtrage se fait comme suit :

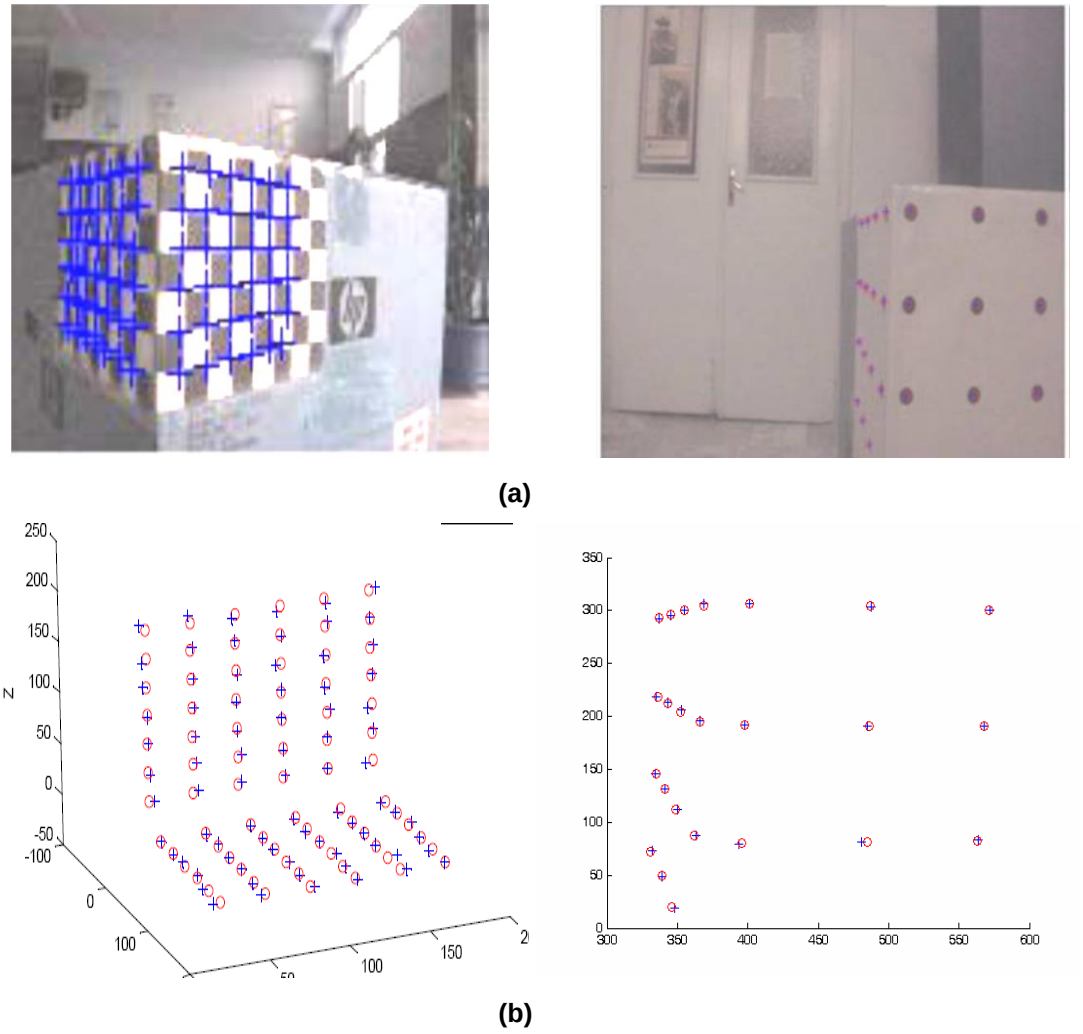


Figure 4.11.: Reconstruction 3D de la mire.
(a) : Deux mires utilisées.
(b) : Reconstruction 3D des deux mires dans le repère caméra gauche.

Calibrer le banc stéréoscopique du système mobile *ATRV2* revient à calibrer chaque caméra indépendamment en utilisant la même mire.

La Figure 4.12 montre les mires de calibrage utilisées : la première pour le calibrage fort est constituée d'un ensemble de petits carrés noirs et blancs de 2.6 cm de longueur pour chacun de leur côté, et la seconde pour le calibrage faible est constituée d'un ensemble de petits cercles noirs de 01cm de diamètre et espacés de 05 cm. Ces motifs ont été choisis pour que leur projection dans l'image numérisée puisse être mesurée avec une grande précision.

(a) **(b)**

Figure 4.12.: Mise en œuvre du calibrage du banc stéréoscopique de l'ATRV2

(a) : en utilisant le calibrage fort.

(b) : en utilisant le calibrage faible.

b) Fusion des données capteurs

Les systèmes qui utilisent plusieurs types de capteurs sont de plus en plus présents dans la littérature. Ils permettent une précision accrue, une plus grande portée de détection et une fiabilité supérieure. C'est pour cette raison que l'usage de différents types de capteurs est maintenant fréquent dans des applications comme la robotique mobile.

Ces systèmes comportent de nombreux capteurs qui acquièrent de l'information complémentaire ou redondante afin d'obtenir une perception plus complète et plus fiable de leur environnement. Mais la façon de gérer l'information provenant de tous ces capteurs demeure une préoccupation.

A cet effet, le système de perception du système *ATRV2* (banc stéréoscopique et douze capteurs) a nécessité une fusion. La méthode de fusion choisie est une méthode basée sur la théorie des possibilités.

Ce choix est justifié par le fait que le modèle de l'environnement de navigation choisi n'utilise pas des fonctions de probabilité pour la reconstruction.

4.2. Expérimentations de Vision

4.2.1. Problématique et solution

La problématique générale en vision se réduit dans la plupart des cas à faire déplacer le robot dans un environnement connu ou inconnu, tout en évitant d'éventuels obstacles fixes ou mobiles, pour réaliser une tâche prescrite. Il en

découle qu'il faut pouvoir définir une stratégie de mouvement (planification), puis exécuter le déplacement prescrit.

La valeur ajoutée de notre travail trouve son intérêt pour le roboticien qui va cherché à exécuter un mouvement planifié par le développement de lois de commande pour le suivi de trajectoire des robots mobiles non holonomes. Ainsi dans notre cas, nous nous sommes intéressés seulement aux robots mobiles de type uni-cycle CESA correspondant à celui qu'on a élaboré au centre de développement des technologies avancées (CDTA) et aussi celui du système ATRV2 (acquis des USA à des fins d'expérimentation).Le CESA dispose d'une motorisation électrique (dynamique rapide) et d'un faible encombrement (faible inertie). Par conséquent, un asservissement en vitesse est suffisant pour le commander. De ce fait, les modèles considérés pour développer les lois de commandes sont basés sur le modèle cinématique du robot mobile de type uni-cycle.

Différentes approches existent pour la stabilisation ou le suivi de trajectoire de ce type de robots. Le problème de la stabilisation est délicat du fait que la condition de Brockett n'est pas vérifiée, et il ne peut donc pas exister de retour d'état stabilisant de type continu et stationnaire pour ces modèles. De nombreux auteurs proposent alors une solution partielle en assurant un suivi de trajectoire à validité locale et ne prenant pas en compte les éventuels retards sur les mesures.

Les planificateurs globaux se basent sur les techniques dites «Roadmaps» /Spong, 2006/, /LaValle, 2006/, /Laumond, 1998/ et nécessitent la connaissance globale de l'espace de travail du robot. Les planificateurs locaux, comme les techniques d'évitement d'obstacles utilisant la stratégie du champ de potentiel /Khatib 1986/, n'ont besoin quant à eux que d'une connaissance locale de l'espace de travail.

Le concept de la localisation et de la cartographie simultanée (SLAM) est aussi apparu. Les phases de SLAM comprennent l'estimation d'état, la reconnaissance des repères et les mises à jour appropriées.

Des solutions communes informatiques à la problématique SLAM induisent le filtre de Kalman et celui de Kalman étendu (EKF). Bien que SLAM se réfère généralement au processus de création de cartes géométriquement cohérentes, il en est également des approches de SLAM topologique qui ont été utilisées pour assurer la cohérence globale de des algorithmes métriques de SLAM [94].

Il existe différents problèmes quant à la commande du robot mobile :

- le suivi de chemin où l'objectif est qu'un point lié au robot suive une courbe prédéterminée en imposant au robot une vitesse donnée ;
- la stabilisation de trajectoires consistant à prendre en compte la dimension temporelle : la trajectoire de référence dépend du temps et la vitesse du robot n'est plus fixée à l'avance ;

On a débuté nos tests en utilisant la vision stéréoscopique car cela nous permet de calculer la géométrie tridimensionnelle 3d d'une scène observée par une ou deux caméras. La première étape du calcul s'est consolidée par une mise en correspondance des couples d'entités (primitives) liées, extraits des deux images d'une paire stéréoscopique. L'appariement dans notre cas par le réseau de Hopfield (cf. Figure 4.12) revenait à établir une relation biunivoque entre primitive d'une image et leurs homologues dans l'autre image. Cette tâche représente l'un des problèmes les plus importants et les plus difficiles de la vision robotique du fait qu'une primitive donnée dans une image peut être à première vue associée à plusieurs primitives dans la seconde image ayant les mêmes propriétés.

Se pose alors le problème du choix du bon candidat. Il est aussi possible que quelques primitives visibles dans une image soient absentes dans l'autre image à cause de la position de la caméra par rapport à la scène. Afin de lever les ambiguïtés et de permettre de dégager un ensemble d'appariements corrects, un certain nombre de contraintes ont été proposés [95] :

- Les contraintes globales, pour résoudre le problème de l'appariement de primitives appelées loi d'unicité et loi de continuité. La première se traduit par le fait qu'une primitive dans une image a au plus un homologue dans l'autre image. La seconde s'appuie sur le fait que le monde physique est constitué de surfaces continues, elle permet à partir d'un appariement initial de prédire d'autres appariements qui viennent confirmer le premier.
- Les contraintes locales qui concernent l'orientation, la longueur de la primitive, son intensité, etc.
- La contrainte épipolaire, qui permet une fois qu'une primitive d'une image est sélectionnée, de réduire l'espace de recherche de son homologue dans l'autre image. La recherche ne se fait plus dans toute l'image, mais sur la droite épi-polaire.

Dans cette approche, nous utilisons la primitive ''segment de droite'' car elle présente un meilleur compromis parmi les différents type de primitives en terme de robustesse, d'invariance, temps de traitement, etc.

L'appariement, dans notre cas, consiste à définir pour chaque segment de droite d'une image, une fenêtre de recherche dans l'autre image. Cette fenêtre contient tous les candidats potentiels du segment de départ et un coefficient de similitude est calculé pour chacun d'eux. L'homologue est retenu pour une valeur maximum de ce coefficient.

4.2.2. Planification du robot ATRV2

L'approche de navigation proposée appelée DVFF combine deux algorithmes existants, l'algorithme de planification de chemin optimal D* pour le calcul de la direction globale du chemin permettant au système mobile ATRV2 de s'orienter vers sa destination, et l'algorithme de navigation *non-stop* basé sur le principe des champs de forces virtuelles VFF (Virtual Force Field en anglais) pour l'évitement d'obstacles.

La principale motivation de cette combinaison est de développer un algorithme hybride de navigation robuste comportant les avantages de ces deux algorithmes, et en même temps, éliminant plusieurs de leurs inconvénients.

L'algorithme de navigation à base des champs de forces virtuelles est un algorithme de navigation *non-stop* n'utilisant que les données courantes des capteurs, et non des données provenant d'un modèle interne, pour décider de l'action à effectuer.

Cet algorithme dérive des méthodes dites à base de champs de potentiels réalisant l'association perception-action à l'aide d'une fonction générant des chemins cherchant à minimiser un critère donné (la fonction potentiel).

La méthode à base des champs de forces virtuelles (VFF) a été développée par Borenstein et Koren [96]. Son but est de permettre à un système mobile rapide de contourner un obstacle imprévu détecté par ses capteurs embarqués.

En ce qui concerne l'algorithme de planification de chemin optimal basé sur D*, il est inspiré essentiellement de celui présenté dans [97] et modifié afin de bien s'adapter au type de représentation de l'environnement choisi.

La Figure 4.13 ci-dessous illustre l'organigramme de cet algorithme et montre comment le système mobile combine ces deux directions (locale et globale) pour une navigation en toute sécurité à partir de sa position initiale vers son point d'arrivée:

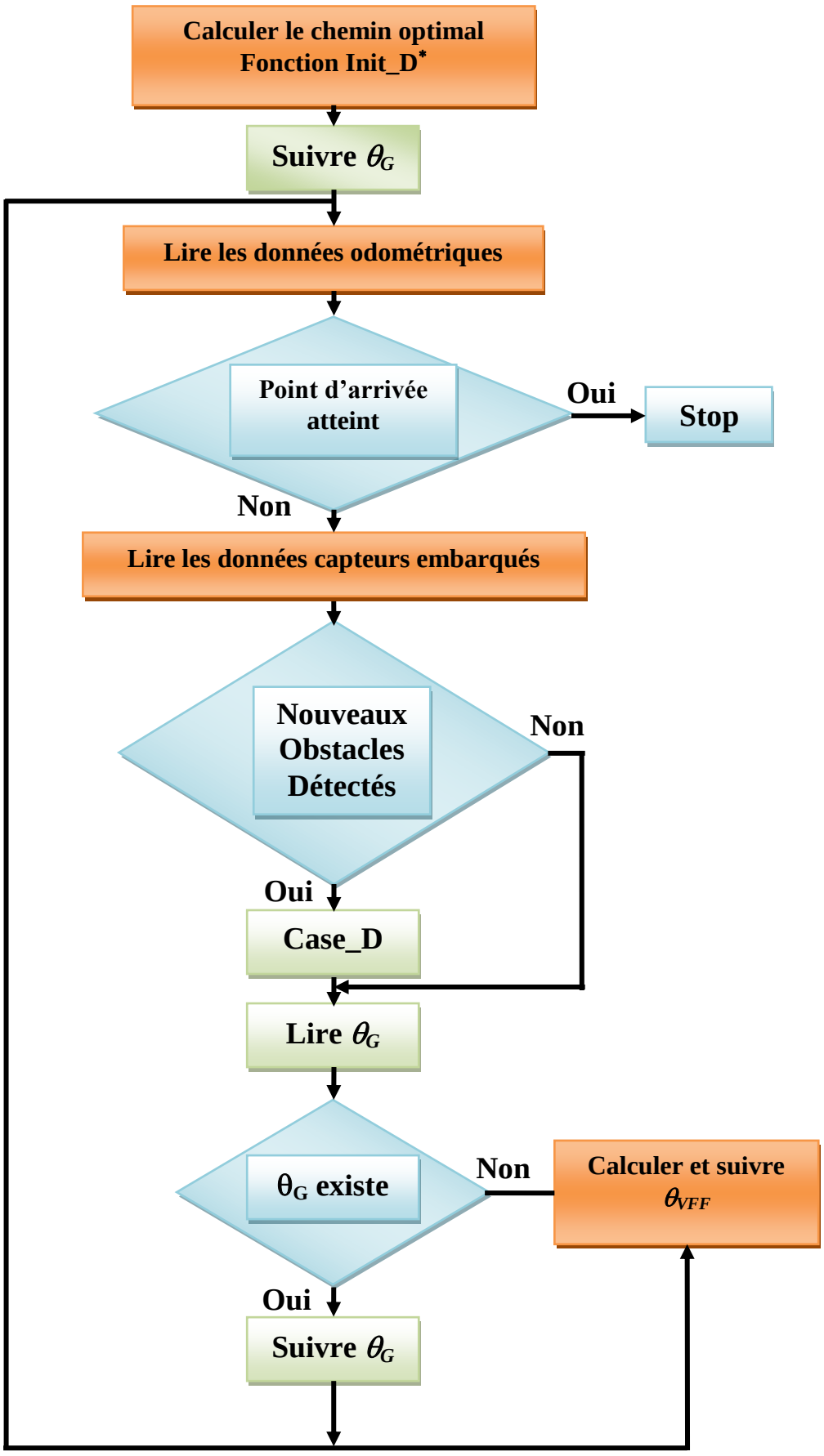


Figure 4.13.: Algorithme de planification de chemin optimal D*.

En combinant l'algorithme de planification de chemin optimal D* et l'algorithme de navigation non stop basé VFF, nous proposons une nouvelle approche de navigation nommée DVFF qui fait déplacer et de manière robuste un système mobile dans un environnement quelconque peu connu.

Le diagramme de cette approche est détaillé dans la Figure 4.14. Il comporte principalement deux modules:

→ Module VFF: Génère le vecteur local F_{VFF} avec l'orientation θ_{VFF}. Il est pris en compte lorsque le système mobile est à proximité des obstacles. En suivant la direction θ_{VFF} le système mobile est protégé d'éventuelles collisions avec des obstacles qui lui sont proches.

→ Module D^+: Génère le vecteur global F_D avec l'orientation θ_G avec $\theta_G$est la direction globale du chemin permettant au système mobile de se diriger vers sa position d'arrivée.

Le module D* peut être facilement utilisé avec le module VFF parce qu'ils génèrent tous les deux une direction. Ces directions (ou vecteurs) peuvent être utilisées séparément ou combinées pour donner une seule direction. La direction sélectionnée sera utilisée pour calculer la commande du mouvement permettant au système mobile d'éviter les obstacles et atteindre son point d'arrivée.

En suivant cette direction, le système mobile perçoit, à travers ses capteurs embarqués, son environnement en découvrant d'éventuelles divergences avec l'environnement initial (détection d'un nouvel obstacle par exemple). Une fois ses divergences constatées, le système mobile replanifie son chemin optimal, en appelant la fonction Case_D*, pour rejoindre son point d'arrivée sans risque de collisions.

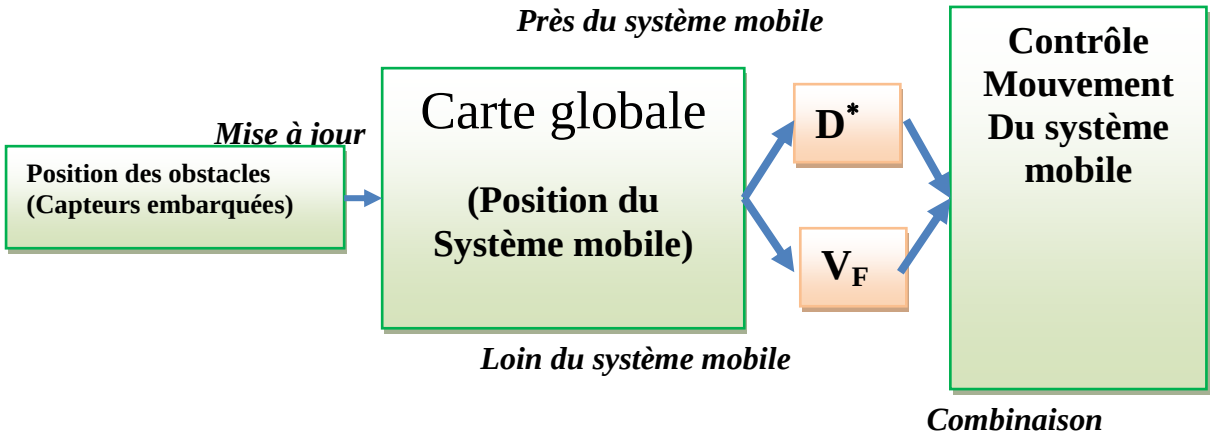


Figure 4.14.: Diagramme de l'approche de navigation D*VFF.

4.2.3 Résultats expérimentaux

Les deux algorithmes DVFF développés ci-dessus ont été implémentés sur le système mobile ATRV2 et testés à partir de plusieurs situations réelles montrant l'avantage de la combinaison entre l'algorithme de planification de chemin optimale D* et l'algorithme de navigation non-stop basée VFF.

(a)

(b)

(c)

(d)

(e)

Figure 4.15.: Chemins de navigation du robot mobile ATRV2.

On constate que d'après la Figure 4.15, le système mobile se déplace en ligne droite et quand il entre dans la zone à risques près de l'obstacle, il l'évite et suit sa trajectoire.

Par conséquent, il apparait que le système mobile ATRV2 réagit parfaitement et sort de son minimum local grâce aux nouvelles directions globales du chemin optimal recalculé quand il a détecté les obstacles frontaux.

Dans cette partie du chapitre, nous avons présenté en premier les caractéristiques spécifiques du système mobile ATRV2, aussi nous avons défini son architecture matérielle et logicielle, ainsi que sa géométrie. La cinématique et le caractère non holonome du système mobile sont des caractéristiques extrêmement importantes pour le développement de notre travail. Nous avons développé ensuite la méthode de localisation basée sur un algorithme utilisant la position fournie par les odomètres

et la position fournie par la mise en correspondance des grilles pour déterminer une estimation de la position réelle du système mobile.

Puis, nous avons décrit en détail l'approche de navigation que nous avons développée, appelée DVFF. Elle combine deux méthodes existantes, l'algorithme de planification de chemin global D* pour calculer la direction globale du chemin permettant au système mobile de se diriger vers son point d'arrivée, et l'algorithme de champ de forces virtuelles (VFF) pour l'évitement de nouveaux obstacles détectés.

Les résultats expérimentaux de l'algorithme DVFF implémenté sur le système mobile ATRV2 avec des configurations réelles de l'environnement ont été acceptables, toutefois, les problèmes inhérents à l'utilisation des capteurs ultrasonores tels que les réflexions multiples et la diaphonie mènent à des situations de blocage ou à des réactions imprévues du système mobile.

4.2.4. Notre processus Rdf

Dans les contextes réels, les objets ne sont pas isolés au milieu de nulle part : il y a toujours d'autres objets, un environnement autour d'eux. C'est ce que l'on appellera dans notre cas le ‘’bruit’’.

Ignorer le bruit permet ainsi de reconnaitre un objet en partie caché, ou tout simplement d'ignorer ce qui entoure l'objet et qui pourrait perturber sa reconnaissance. Les réseaux de neurones sont généralement optimisés par des méthodes d'apprentissage de type statistique, si bien qu'ils sont placés d'une part dans la famille des applications statistiques, qu'ils enrichissent avec un ensemble de paradigmes permettant de générer de vastes espaces fonctionnels, souples et partiellement structurés, et d'autre part dans la famille des méthodes de l'intelligence artificielle qu'ils enrichissent en permettant de prendre des décisions s'appuyant davantage sur la perception que sur le raisonnement logique formel. En modélisation des circuits biologiques, ils permettent de tester les hypothèses fonctionnelles issues de la neurophysiologie ou de tester les conséquences de ces hypothèses afin de les comparer aux réseaux réels. En robotique mobile et particulièrement en vision robotique, les techniques d'apprentissage qui utilisent la vision artificielle représentent le plus souvent l'image par un ensemble de descripteurs visuels. Ces descripteurs sont extraits en utilisant une méthode fixée à l'avance ce qui compromet les capacités d'adaptation du système à un environnement visuel changeant.

Par conséquent, nous proposons une méthode permettant de décrire et d'apprendre des algorithmes de vision de manière globale, depuis l'image perçue jusqu'à la décision finale. Le système de reconnaissance de formes (Rdf) proposé pour nos expérimentations est illustré ainsi sur la Figure (4.16):

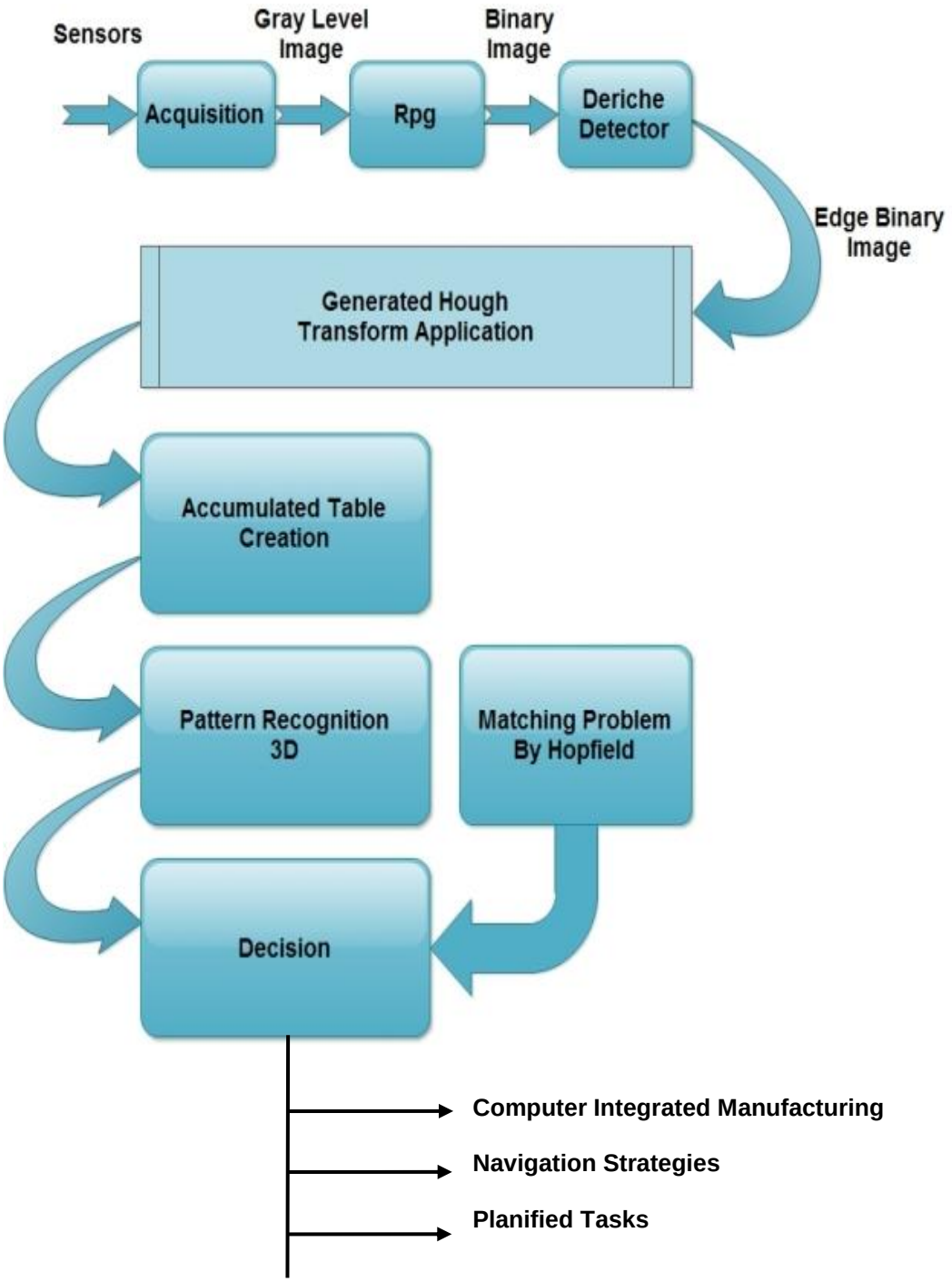


Figure 4.16.: Segmentation de notre processus de reconnaissance de formes.

L'objectif de l'organigramme dans un premier temps est de permettre au robot mobile (CESA, puis l'ATRV) de mieux comprendre l'environnement perçu. Ceci engendrera une meilleure perception des attributs (bonne détection des balises parmi les amers) et par la même reconnaitre leurs formes. La mise en correspondance par le réseau de Hopfield est bien adaptée pour qu'il puisse comparer avec sa base de données et en prendre en final la bonne décision.

Les segments de droites sont obtenus par l'application de la Transformée de Hough sur les images contours de la paire stéréoscopique. Elle engendre deux listes de segments de droites, où chaque segment est représenté par : sa distance perpendiculaire (ρ) par rapport au repère de l'image, son orientation (θ) dans ce repère et les coordonnées de ses extrémités. Ensuite, chaque segment de ces deux listes est étiqueté adjacent (voisin) lorsqu'il vérifie un critère de proximité qui sera défini par la suite [98].

4.2.5. Implémentation Neuronale

Les méthodes neuronales sont nombreuses, chaque méthode a ses propres caractéristiques incluant l'architecture, la convergence, le temps d'exécution et ses résultats dans divers domaines. Cependant ces réseaux de neurones à plusieurs couches, de connexions modifiables, sont confrontés à un problème qui s'énonce de la manière suivante : Comment répercuter sur chacune des connexions le signal d'erreur qui n'a été mesuré que sur la couche de sortie, après avoir traversé plusieurs étapes ?

Cette problématique a été levée par l'algorithme de rétro-propagation du gradient [99] et [100]. Cette technique neuronale est implémentée dans notre processus de traitement. Elle a le mérite d'avoir deux aspects essentiels: l'aspect parallèle en temps réel .et les non linéarités du processus concernant la commande.

Après l'étape de calibration de notre système d'acquisition (senseurs), nous obtenons des images en niveaux de gris (gray level image). Ensuite, nous utilisons l'Algorithme Rpg qui est la rétro-propagation du gradient. En effet, notre réseau de neurones est un MLP (perceptron multicouches), recommandé dans les études de reconnaissance de formes, car il nous fournit des résultats précis, Il permet une meilleure généralisation pour l'image entière à l'entrée du réseau.

a) Architecture du réseau

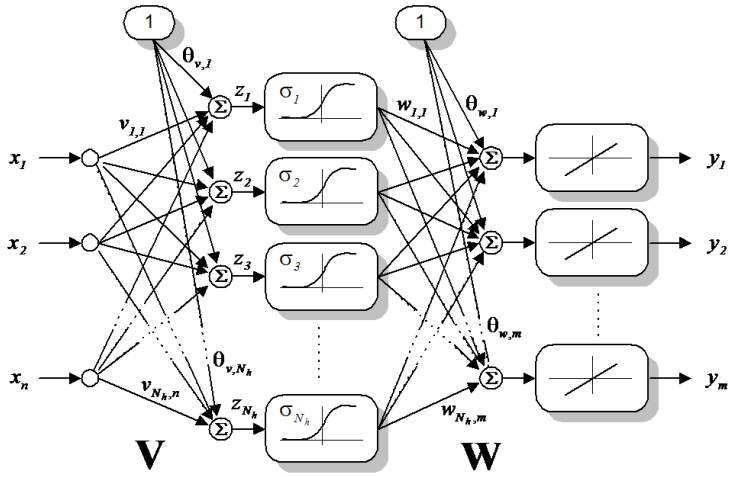


Figure 4.17.: Architecture du réseau.

L'architecture (9-6-3-1) dans la Figure (4.17) repose sur la correction de l'erreur quadratique par une approximation d'une descente du gradient, ainsi que la fonction sigmoïde sans biais :

$$f(x) = 1/\,1+\exp(-x).$$

Afin de reproduire notre image ou de détecter les objets d'une scène, il est nécessaire de développer un ''classifieur neuronal''. Ce dernier consiste en une fenêtre d'un réseau multicouche à rétro-propagation du gradient. Des essais multiples effectués en laboratoire avec des fenêtres de tailles différentes ont montré que la fenêtre (3*3) était la plus adéquate.

La Figure (4.18) ci-dessous illustre la pertinence de notre choix, [101].

Figure 4.18.: Image originale d'objets de formes cylindrique et polyédrique.

Figure 4.19.: Résultat comparatif.
(a) : Pour un réseau (25-15-5-1), T = 0.125.
(b) : Pour un réseau (9-6-3-1), T = 0.125.

Il apparait dans la Figure 4.19 un résultat important qui est indicatif puisque il nous démontre directement par l'expérimentation que le réseau (9-6-3-1) donne une meilleure généralisation pour l'image entière à l'entrée du réseau comparativement au(25-15-5-1).

Après ce choix pertinent, notre réseau de neurones devient un MLPusant le Rpg pour l'apprentissage, et il possède une couche cachée, son nombre d'entrée N dépend directement de la taille de l'image. N est donc défini de la manière suivante :

$N=(T/P)*3$, T représentant la taille de l'image, P un nombre carré (1^2, 2^2, 3^2…) représentant la précision, et 3 correspondant aux trois couleurs de base d'une image (RGB).

S le nombre de neurones de sortie est égal au nombre d'objets que l'on veut enregistrer dans notre base de données. Ainsi chaque neurone de sortie correspondra à un objet différent. Concernant le nombre de neurones composant la couche cachée C de notre réseau, il est compris entre N et S ainsi $C \in [N,S]$.Les neurones d'entrée recevront eux l'image en couleur découpée de la manière suivante :

On prend l'image d'origine et on prend un nombre de pixels en carré dépendant de la précision, et on fait la moyenne de chacune des couleurs des pixels pour chaque zone ainsi définie. Les couleurs de chaque zone sont ensuite mises en entrée dans notre réseau de neurones.

Par exemple, pour la reconnaissance d'un objet, on entre une image dans le réseau de neurones en la soumettant au même découpage que les images mises en entrée lors de l'apprentissage, ensuite on regarde tous les neurones de sortie du

réseau, et celui qui a la valeur la plus élevée correspond à l'objet reconnu. Si la valeur est inférieure à un certain pallier, aucun objet n'a été reconnu dans l'image.

b) Règle Delta Généralisée

La règle delta généralisée étant une description mathématique du Rpg est en fait l'algorithme d'apprentissage de notre réseau. Les étapes de l'algorithme sont décrites ci-dessous :

1) Appliquer un vecteur d'entrée au réseau et calculer les valeurs des sorties correspondantes.
2) Comparer les sorties actuelles avec les sorties réelles, puis déterminer l'erreur.
3) Rétro-propager l'erreur de toutes les couches intermédiaires cachées et calculer les erreurs pour toutes les unités de ces couches.
4) Déterminer la variation des poids avec laquelle sera ajusté chaque poids de connexion du réseau.
5) Appliquer les corrections aux poids des connexions du réseau.
6) Répéter les étapes de 1 à 5 avec tous les vecteurs d'exécution jusqu'à ce que l'erreur de tous les vecteurs soit réduite à une valeur acceptable.

c) Momentum

Le concept de Momentuma été introduit pour réaliser un compromis entre un coefficient d'apprentissage faible et un temps d'apprentissage acceptable [102] et [103].

Le Momentum agit comme un filtre passe bas sur le terme de variation de poids, puisqu'il renforce la tendance générale et diminue le risque d'oscillation, cela permet un apprentissage plus rapide avec un coefficient d'apprentissage faible. L'équation de variation du poids est modifiée de telle sorte qu'une partie de la variation de poids est réintroduite par itération.

$$\Delta W_{ji}^{(c)}(t+1) = Lcof(t+1).e_j^{(c)}(t).x_i^{(c-1)}(t) + Mom(t+1).\Delta W_{ji}^{(c)}(t) \quad (4.6)$$

Il est essentiel d'améliorer la règle de la modification des poids dans l'algorithme de la rétro-propagation par l'adjonction de la règle :

$$\Delta W(t+1) = \Delta W(t) + 0{,}9W \quad (4.7)$$

Cette modification majeure améliore considérablement la fiabilité de l'algorithme. Les meilleurs résultats sont obtenus avec un momentum de valeur 0.001 et un coefficient d'apprentissage de valeur 0.015. Notre objectif est d'évaluer aussi la

performance du classifieur neuronal utilisé qui prend en entrée l'image en niveaux de gris. L'erreur a pu être réduite jusqu'à l'ordre de 1/1000 [84]. Le seuil T a été calculé avec la méthode de calcul de la moyenne arithmétique.

4.2.6. Extraction des chaînes de points de contour

Tout détecteur de contour est considéré comme étant un triplet (algorithme, paramètres, pré-condition). La pré-condition est représentée par le contexte dans lequel le couple (algorithme, paramètres) s'exécute correctement.

Par conséquent et au vu de l'expérimentation et paradoxalement notre cas, il apparait que le détecteur de Deriche semble être le mieux approprié en adaptation pour la mise en évidence des contours peu nets et bruités (Figures 4.20 et 4.21).

Ce dernier est un opérateur de détection de contour optimal basé sur le critère de Canny qui est implémenté récursivement. D'ailleurs il nous en sortie fournit des contours mieux localisés, et il engendre ainsi une plus faible erreur de localisation et d'omission. Pour l'utilisation rationnelle de la TH dans notre système de vision, il est impératif d'avoir ce type d'images contours binaires. En effet, on procède au filtrage du bruit de l'image binaire Rpg, qu'elle soit en 2d via le robot mobile CESA ou en 3d moyennant l'ARTV2, nous obtenons en sortie des images contours binaires (icb).

Ceci nous permet d'obtenir une forte atténuation de bruit assez importante qui réduit considérablement les erreurs de reconnaissance à postériori. Les résultats le confirment.

4.2.6.1 Explication

A l'issue du filtrage (du type gradient ou Laplacien), on est en présence d'une image dont on veut extraire et organiser les points de contraste le long de chaînes de points de contour qui marquent les frontières de régions à gradient plus ou moins homogène. Il s'agit donc de déterminer ces chaînes de points de contour en suivant les maxima ou les zéros de la sortie du filtre. La technique de seuillage par hystérésis a prouvé son utilité dans le cas du filtrage de type gradient, en présence de bruit : il s'agit de ne commencer à suivre une chaîne de contour que lorsque le gradient a une valeur supérieur à un seuil et d'abandonner le chaînage lorsque le gradient tombe au-dessous d'un seuil [104]. Cette technique a l'avantage d'éliminer de nombreux maxima du signal dus au bruit. Notre choix s'est porté sur le détecteur de Deriche [37] pour ses performances pratiques.

Le filtre Deriche s'écrit :

$$s_4(x) = k.(\alpha|x| + 1)^{-\alpha|x|} \quad (l) \tag{4.8}$$

k est choisi de manière à obtenir un filtre discret normalisé soit :

$$\sum_{-\infty}^{+\infty} s_4(x) = 1 \Leftrightarrow k = \frac{\left(1 - e^{-\alpha}\right)^2}{1 + 2.\alpha.e^{-\alpha} - e^{-2\alpha}}$$

Le filtre optimal de dérivation s'écrit comme suit :

$$d_4(x) = -k.e^{-\alpha|x|} \tag{4.9}$$

$$k = \frac{(1 - e^{-a})^2}{e^{-\alpha}} \tag{4.10}$$

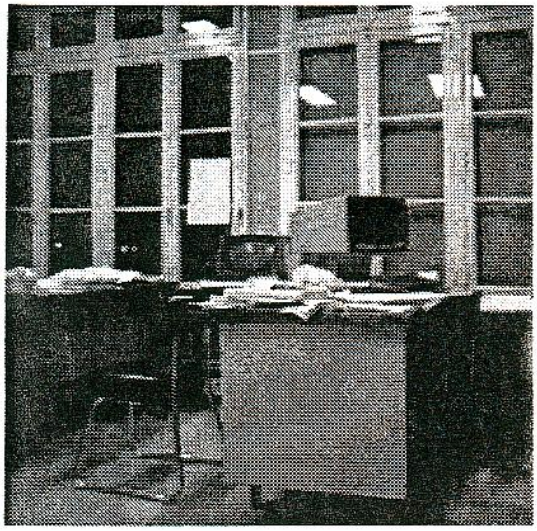


Figure 4.20.: Image originale d'une scène d'intérieur de Laboratoire.

Figure 4.21.: Détection de contours par Deriche (α=1.5).

4.3. Application de la Transformée de Hough dans l'appariement des images

Nous avons vu au chapitre précédent que l'extraction des segments de droites de l'image contours se fait donc par une recherche des segments de droites portés sur chaque droite significative détectée (ρ_i, θ_j) dans le tableau accumulateur Figure(4.22).

Une droite significative dans l'image contours se caractérise par un pic dans le tableau accumulateur et pour chaque pic détecté on élimine l'effet des points appartenant à ce pic dans le tableau accumulateur. Le résultat de l'extraction est une liste de segments de droites où chaque segment est stocké avec la liste d'attributs suivants :

- Index : un entier qui caractérise sa position dans la liste des segments de droites,

- ρ: sa distance perpendiculaire par rapport au repère de l'image,

- θ: son orientation dans ce repère, (x_1 , y_1) et (x_2 , y_2) : les coordonnées de ses extrémités.

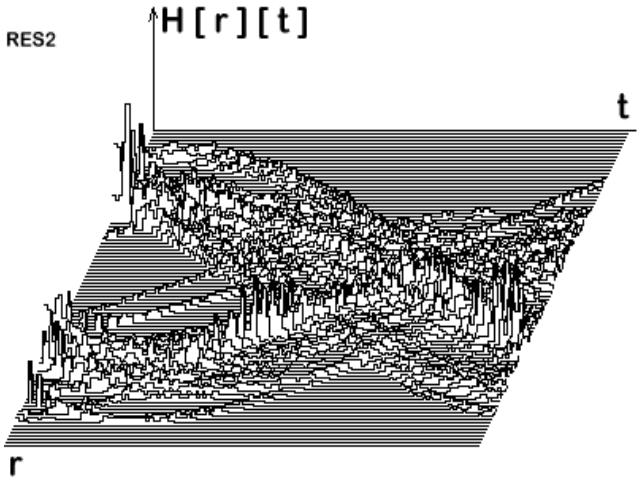


Figure 4.22. : Tableau accumulateur.

Des relations de voisinage entre segments viennent s'ajouter aux attributs locaux cités ci-dessus. Elles permettent de lier tous les segments de la scène entre eux. En effet, un ensemble de segments voisins entre eux peuvent être issus d'un même objet. Pour retrouver le voisinage des segments de droites, nous avons opté pour la méthode appliquée par [94]. Elle consiste à découper l'image en un ensemble de

fenêtres de forme carré de dimension fixe, puis à rechercher pour chaque segment de droite l'ensemble des fenêtres qu'il intersecte. Deux segments de droites sont dits voisins s'ils ont au moins une fenêtre en commun. Les attributs qui vont s'ajouter aux attributs locaux sont :

- vois [lv] : liste de lv segments voisins dont on connaît les ρ, θ, (x_1, y_1) et (x_2, y_2) ainsi que leur Index dans la liste des segments de droites.

Nous avons donc créé une structure de données qui regroupe tous les segments de droites d'une image avec leurs propriétés locales ainsi que leurs voisinages. L'image est donc représentée par un graphe d'adjacence dont les nœuds sont des segments de droites auxquels sont attachés des propriétés géométriques et dont les arêtes définissent des relations de voisinage entre segments.

4.3.1. Algorithme d'appariement

L'initialisation de notre algorithme d'appariement se fait par tous les appariements entre les segments de droites qui répondent aux contraintes locales suivantes :

a) Contrainte de voisinage

Cette contrainte remplace la contrainte epipolaire. Elle définit pour chaque segment S_i d'une image une fenêtre de recherche $F(S_i)$ de taille dynamique (ses dimensions dépendent de la taille du segment courant S_i et de la valeur du seuil Tfen) sur l'autre image [105].

Les segments de droites qui appartiennent à cette fenêtre sont des candidats potentiels pour la mise en correspondance. La Figure 4.23 montre bien la forme de la fenêtre de recherche $F(S_i)$.

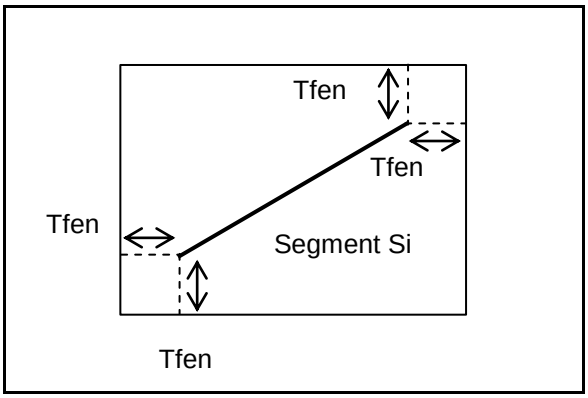


Figure 4.23.: Construction de la fenêtre de recherche $F(S_i)$.

b) Contrainte d'orientation :

Deux segments sont dits d'orientation similaire si la différence de leur orientation est inférieure à un certain seuil (seuil-teta).

c) Contrainte de longueur :

Deux segments sont dits de longueur similaire si la différence de leur longueur est inférieure à un certain seuil (seuil-long).

d) Contrainte de distance :

Deux segments sont dits similaires si la différence de distance de leur milieu est inférieure à un certain seuil (seuil_dis). Pour chaque segment d'une image, nous aurons un ensemble de couples de segments potentiels. Nous affectons à chaque couple de segments potentiels un coefficient "Q" qui met en œuvre une comparaison des attributs de ce couple, et qui sont : la longueur, l'orientation et la distance reliant les deux points milieu des segments du couple. La valeur de ce coefficient est calculée comme suit :

Pour un couple de segments (S_i, S_j) :

$$Q_{ij} = \frac{long\left|S_i - S_j\right|}{seui - long} + \frac{orient\left|S_i - S_j\right|}{seuil - teta} + \frac{dist\left|m_{Si} - m_{Sj}\right|}{seuil - dis}, Sim_{ij} = 1/Q_{ij} \qquad (4.11)$$

long $|S_i\text{-}S_j|$, orient $|S_i\text{-}S_j|$ et dist $|m_{SI}\text{-}m_{SJ}|$ représentent la différence des valeurs des attributs : longueur, orientation et la distance entre les deux points milieu des segments S_i et S_j.

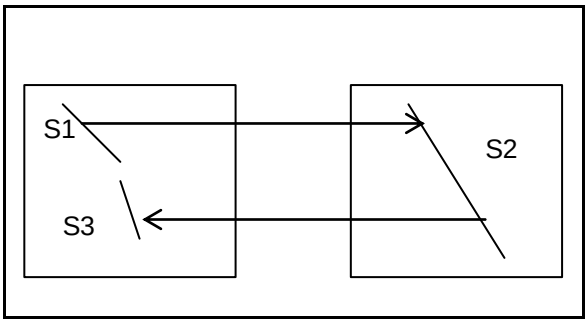
Seuil_long, seuil_teta et seuil_dis sont les erreurs sur les seuils permises sur les valeurs des attributs du segment S_j homologue du segment S_i respectivement pour la longueur, l'orientation et la distance entre les deux points milieu des deux segments S_i et S_j.

Enfin, nous en retenons que le couple de segments potentiels dont la valeur du coefficient *"Q"* est minimale (la similitude "Sim" est maximale), et les autres couples des segments potentiels seront rejetés. Le segment S_k dont la valeur du coefficient *"Q"* est minimale est le segment homologue du segment S_i. Cette procédure d'appariement permet d'obtenir pour chaque segment d'une image, zéro ou un correspondant (homologue) dans l'autre image. Ce processus est appliqué deux fois

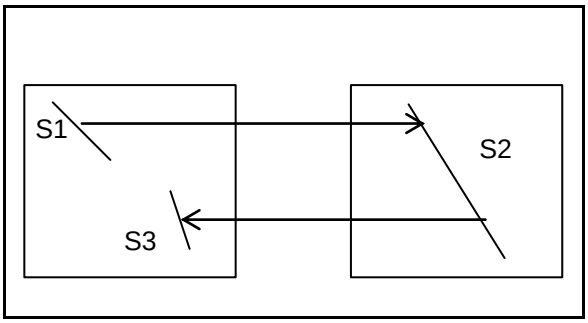
en inversant les rôles respectifs des deux images de la paire stéréoscopique. On dispose alors de deux listes de segments appariés résultats de l'appariement gauche-droite et droite-gauche. Mais ces deux listes ne sont pas cohérentes, car nous pouvons avoir un segment de droite d'une image apparié à plusieurs segments de droites dans la seconde image ayant les mêmes propriétés géométriques. Afin de lever les ambiguïtés et de permettre de dégager un ensemble d'appariements corrects, un certain nombre de contraintes ont été proposés :

a) Contrainte de compatibilité : Etant donné un appariement (S_1, S_2) :

- si S_2 est apparié à S_1 (appariement gauche-droite) et si S_2 est apparié à S_3 (appariement droite-gauche) avec S_3 voisin à S_1, donc (S_1, S_2) est validé sur la figure 4.24-a.
- sinon (S_1, S_2) est rejeté sur la figure 4.24-b.



(a) : Cas compatible.



(b) : Cas non compatible.

Figure 4.24.:Contrainte de compatibilité entre les appariements gauche-droite et droite-gauche.

Soulignons ici que les appariements gauche-droite et droite-gauche résultent de deux processus d'appariement totalement indépendants. Le test de la contrainte de

compatibilité mutuelle des résultats obtenus offre donc une garantie supplémentaire de fiabilité des appariements finalement retenus.

b) Contrainte d'unicité :

Cette contrainte impose que chaque primitive d'une image ait un seul correspondant dans l'autre image. Ceci interdit donc les recouvrements entre segments de droites (cf. Figure 4.25).

Dans le cas d'un recouvrement entre deux segments de droites, nous supprimons celui dont l'appariement est de qualité la moins bonne.

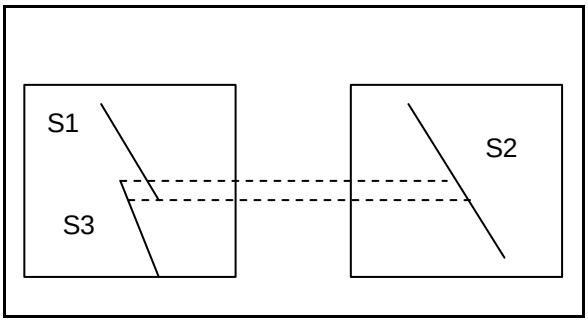


Figure 4.25. : Cas de recouvrement entre deux segments de droites.

4.3.2. Mise en correspondance par la modèle de Hopfield

Vu l'état de l'art, nous avons opté pour le modèle de Hopfield qui s'avère le plus compatible pour notre application. On montre que le problème de mise en correspondance peut être formulé comme une tâche d'optimisation où ou il faudra satisfaire les contraintes de correspondance. Une fonction d'énergie représentant ces contraintes sur la solution est réduite au minimum [106] et [107]. Puis la matrice de connexion se déduit pour faire évoluer le réseau vers son état stable [108]. La Figure 4.26 montre la structure de Hopfield à deux dimensions qui peut être assimilé à une matrice. Ce réseau est employé pour trouver la correspondance entre les primitives de l'image gauche et ceux de l'image droite. L'état (activé ou non) de chaque neurone dans le réseau représente une possibilité d'appariement entre un segment dans l'image gauche avec son homologue dans l'image droite. La fonction de Lyapunov pour un réseau binaire bidimensionnel de Hopfield [109] et [110] est donnée par:

$$E = -\frac{1}{2}\sum_{u,l}\sum_{v,m} T_{ulvm} V_{ul} V_{vm} - \sum_{u,l} I_{ul} V_{ul} \qquad (4.12)$$

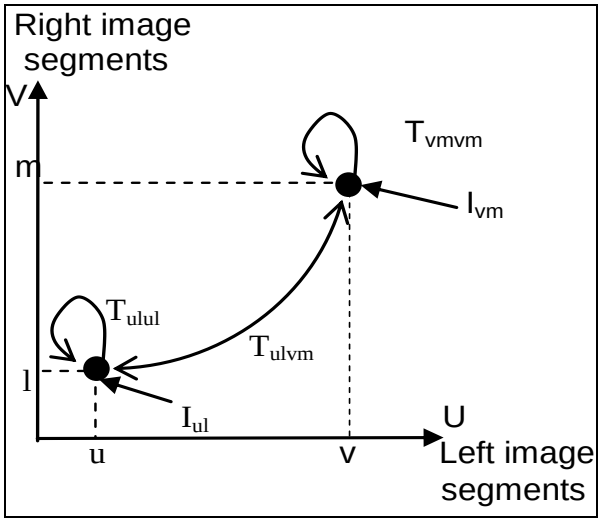


Figure 4.26. : Structure du Réseau de Hopfield.

V_{ul} et V_{vm} représentent respectivement les états binaires (sorties) des neurones *(u,l)* et *(v, m)*, qui peuvent être à l'état 1 (actif) ou 0 (inactif). T_{ulvm} est le poids de connexion entre les deux neurones. Cette connexion est symétrique $T_{ulvm} = T_{vmul}$. Il est démontré que pour une stabilité du réseau, il faudrait que chaque neurone n'ait pas de connexion sur lui-même autrement dit $T_{ulul} = 0$ et I_{ul} est l'entrée initiale à chaque neurone. Dans ce réseau nous avons employé d'une part les contraintes d'intensité et de disparité, et d'autre part la loi d'unicité et celle de l'ordonnancement.

4.4. Résultats expérimentaux

L'algorithme d'appariements stéréoscopiques implémenté en langage C++ sous Unix sur une station de travail, a été testé sur plusieurs images. Nous présentons ici les résultats de notre méthode appliquée sur deux paires stéréoscopiques. Les contours binaires de chaque image sont extraits par l'opérateur de Deriche. Les segments de droites sont détectés par la transformée de Hough (TH) avec 10 degrés pour pas de quantification de la dimension de θ et (01) pixel pour pas de quantification de la dimension de ρ.

La TH est indépendante de la phase suivante d'appariement et a été implantée sur une puce FPGA.. Les résultats de notre méthode d'appariement ont été effectués avec un jeu de paramètres déterminé expérimentalement, et qui sont :

- *Tfen* = 20, - *seuil_dis* = 1, - *seuil_long* = 10, - *seuil_teta* = 10°.

Nous donnons dans le tableau ci-dessous (Tableau 4.1) les principaux résultats d'appariement obtenus sur nos 2 exemples des deux Figures 4.27 et 4.28. Près de 67% de segments de droites ont été appariés avec un taux d'erreurs inférieur à 1%.

Tableau 4.1. : Résultats d'appariements.

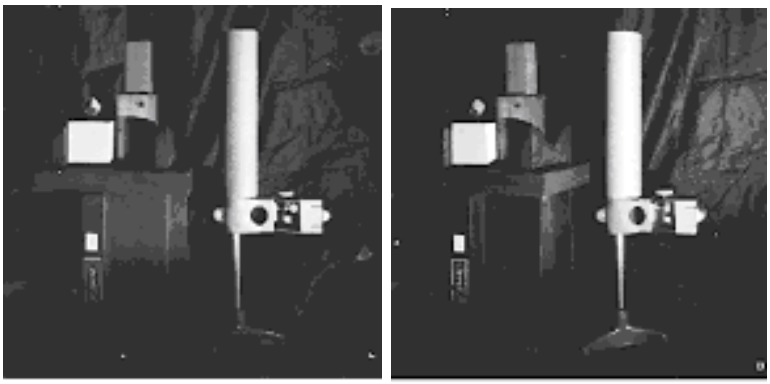
Exemples d'images	3dlab	Ball
Nombre de segments gauches	736	752
Nombre de segments droites	845	734
Nombre de segments appariés gauche-droite	502	584
Nombre de segments appariés gauche-droite après les contraintes de compatibilité et l'unicité	466	524
Nombre de segments appariés droite-gauche	580	562
Nombre de segments appariés droite-gauche après les contraintes de compatibilité et l'unicité	498	486

Dans les figures 4.27 et 4.28, la partie supérieure en (a) représente les images gauche et droite fournies par les caméras. Au-dessous en (b), nous avons représenté les segments de droites détectés par la Transformée de Hough à partir des images contours gauche et droite. Enfin, dans la partie inférieure en (c) nous présentons les appariements gauche-droite et droite-gauche résultats de la méthode d'appariement élaborée. Un petit nombre de segments ne sont pas appariés parce qu'ils n'ont pas de véritable homologue.

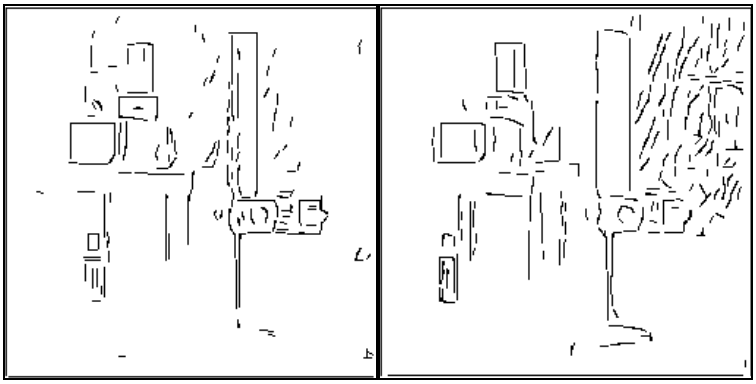
Tableau 4.2. : Résultats obtenus par le logiciel.

Images	3DLAB_D	3DLAB_ G	Objects
Size	256*256	256*256	256*256
ThH	3	3	3
Figures	4.31c	4.30c	4.33d
N_{seg}	210	208	267
N_{peak}	98	94	122

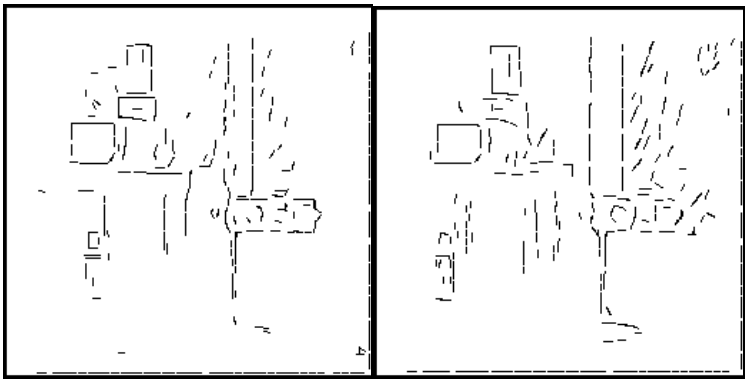
Le tableau illustre les résultats obtenus par notre logiciel sur les images 4.29a, 4.30a et 4.31a. Npeak, Nseg, and ThH sont respectivement, le nombre de pics détectées de l'espace de Hough, le nombre de segments de droites détectés et la valeur de longueur de seuil pour la détection des segments de droites.



(a) : Paire stéréo en niveaux de gris.

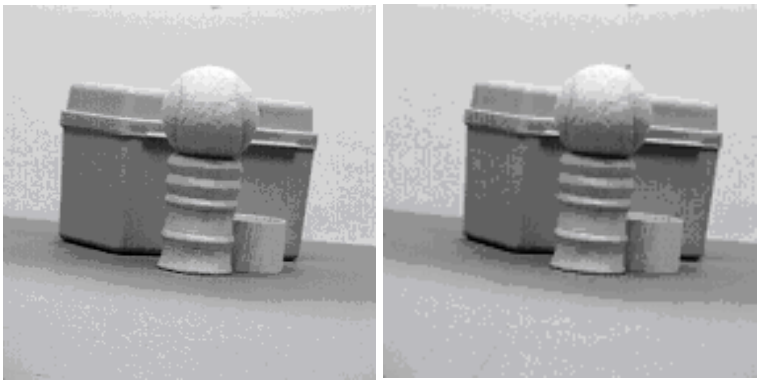


(b) : Résultats de la segmentation de la paire stéréo.

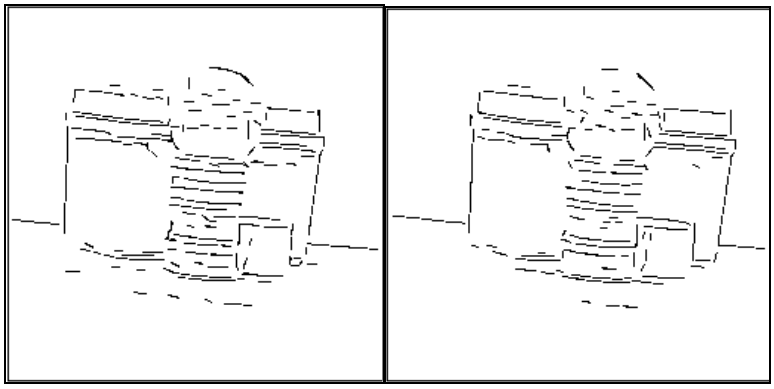


(c) : Résultats des appariements gauche-droite et droite-gauche.

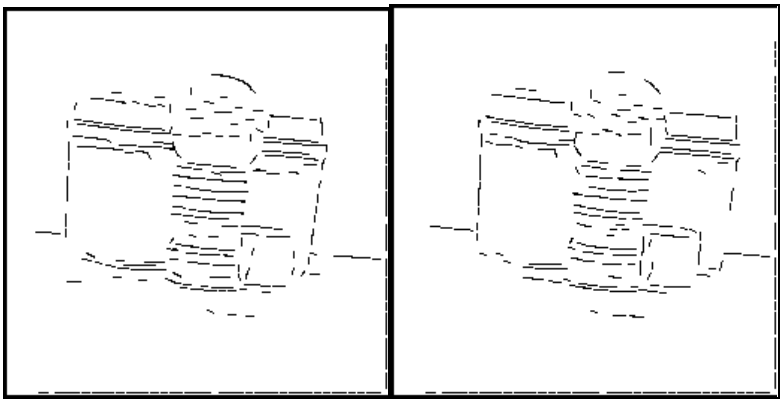
Figure 4.27. : Paire stéréo 3D Labo.



(a) : Paire stéréo en niveaux de gris.



(b) : Résultats de la segmentation de la paire stéréo.



(c) : Résultats des appariements gauche-droite et droite-gauche.

Figure 4.28. : Paire stéréo Ball.

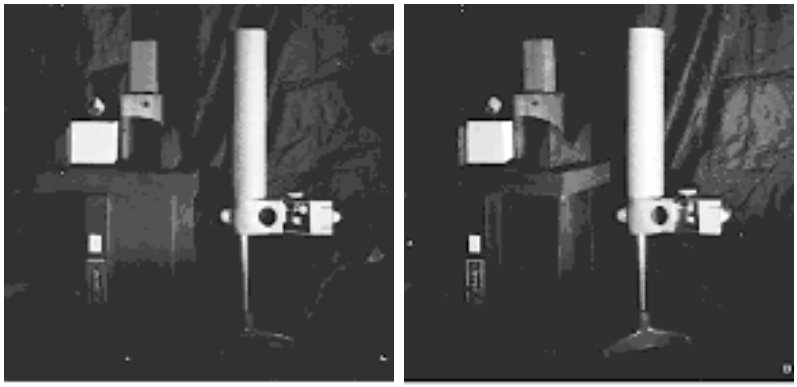
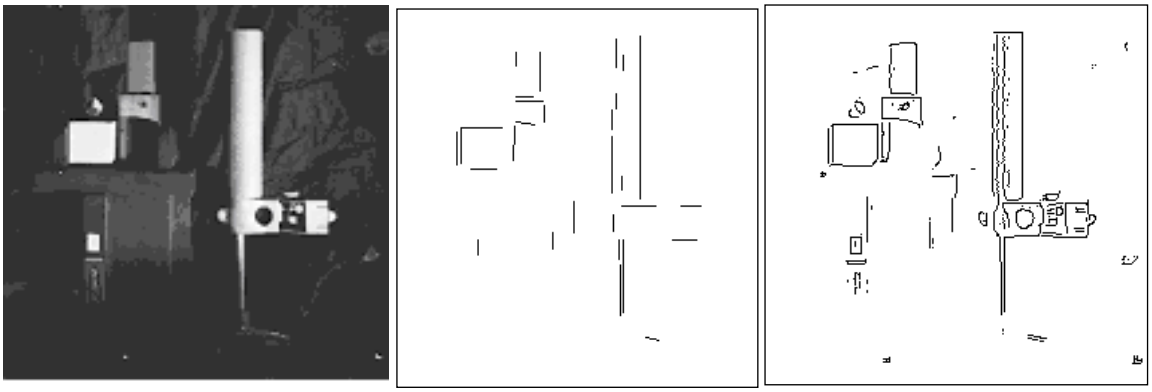
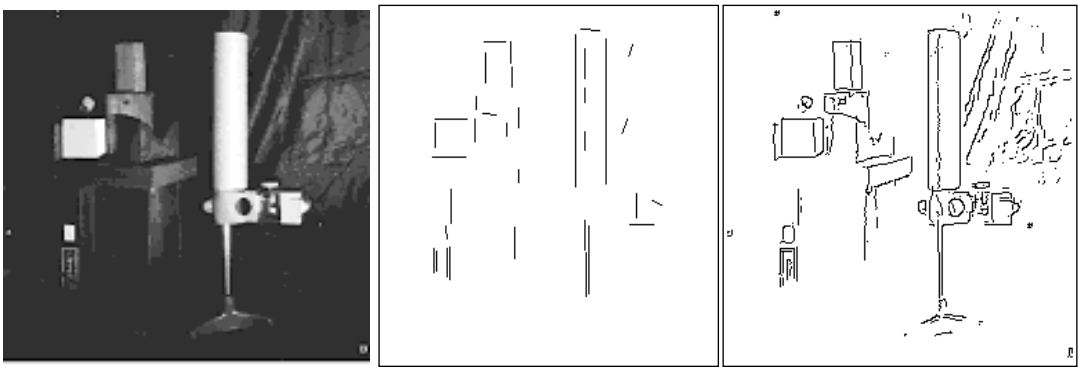


Figure 4.29. : Couple stéréoscopique, images originales 3D Labo.



(a) (b) (c)

Figure 4.30. : Résultats de l'Algorithme appliqué sur l'image 3D gauche du Labo.
(a) : Image initiale; **(b)** : Image contour binaire; **(c)** : Résultat de la reconnaissance.



(a) (b) (c)

Figure 4.31. Résultats de l'Algorithme appliqué sur l'image 3D droite du Labo.
(a) : Image initiale; **(b)** : Image contour binaire; **(c)** : Résultat de la reconnaissance.

Une autre contribution est à mentionner illustrant le CESA en mode vision.

Différentes balises expérimentales de différentes formes (polyédrique, cylindrique et rectangulaire) ont été posées dans l'atelier du laboratoire. Il apparait que dans toutes les expériences, l'algorithme converge et le système mobile reconnait pratiquement bien au regard des images résultats obtenues sur la Figures 4.32.

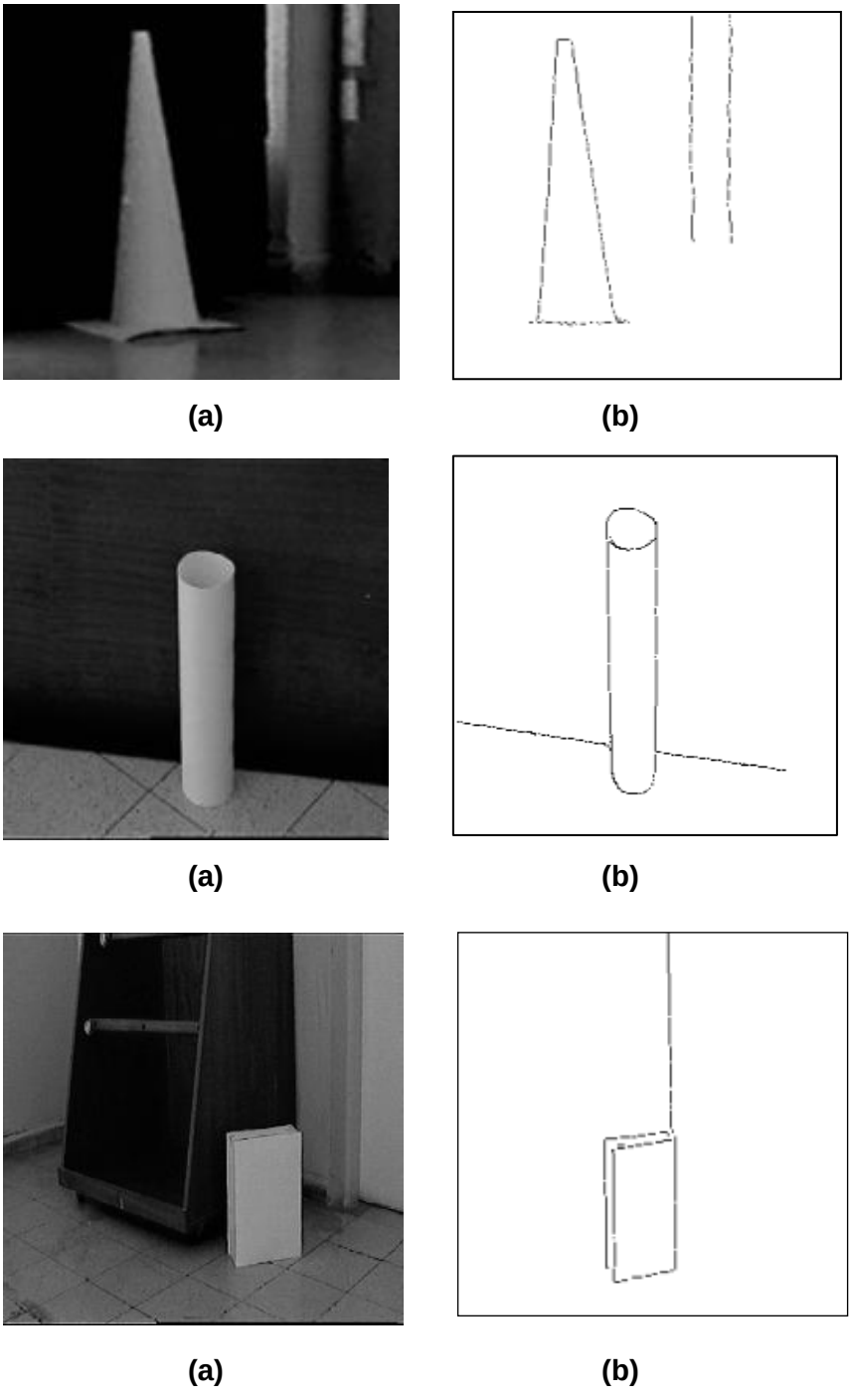
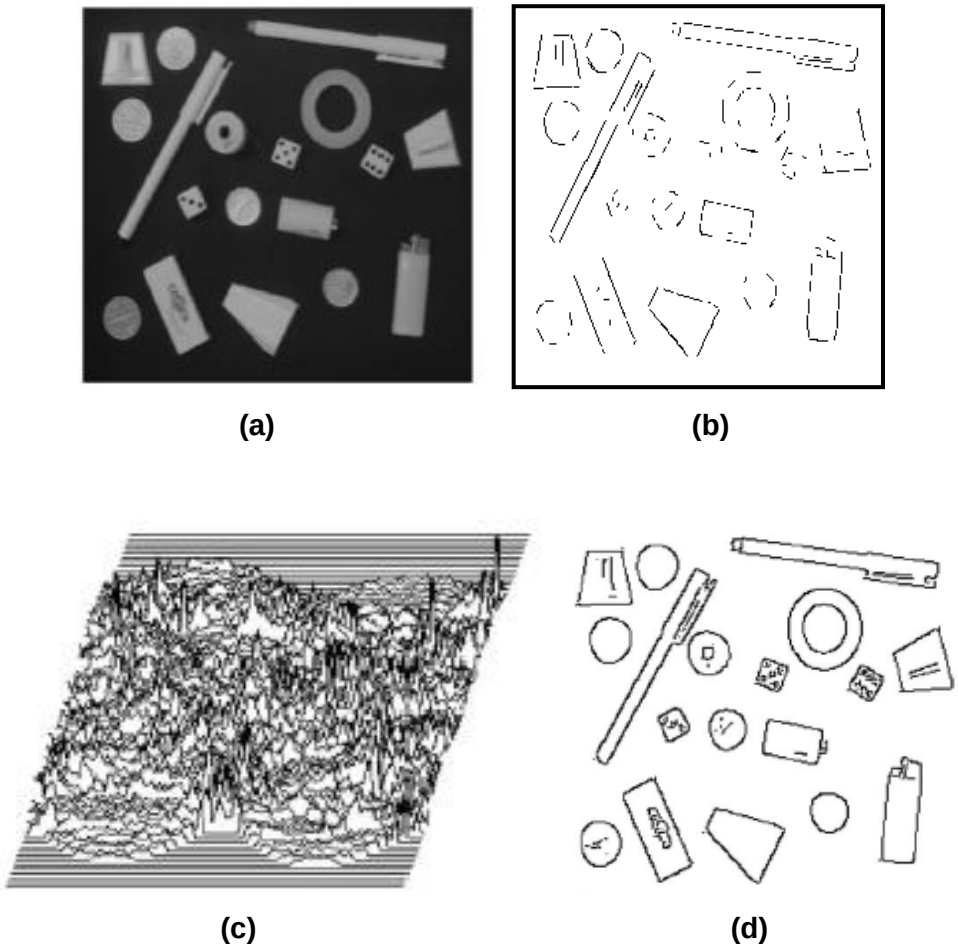


Figure 4.32. Application de l'Algorithme sur des balises (Polyédrique, Cylindrique et Rectangulaire)
(a): Image originale
(b): Image résultat

La robustesse est testée au vu d'une autre expérimentation plus complexe avec le robot mobile *ATRV2* sur des objets multiformes (stylos, cerceau, briquet, anneau, etc.).



(a) **(b)**

(c) **(d)**

Figure 4.33.:Résultats de l'Algorithme appliqué sur des objets multiformes
(a) : Image initiale; **(b)** : Extraction des segments
(c) : Espace de Hough; **(d)** : Résultat de la reconnaissance.

Le classifieur hybride a été évalué sur la base de plusieurs tests. La Figure 4.34 nous montre l'évolution du taux de reconnaissance en fonction du seuil d'ambiguïté et du seuil de confusion sur les objets utilisés dans la base de données. On peut constater que plus le seuil de confusion n'est élevé, plus le taux de reconnaissance n'augmente.

Le taux de reconnaissance croît aussi avec le seuil d'ambiguïté. Les Figures 4.30c, 4.31c et 4.33d montrent l'efficacité de notre logiciel de traitement sur des objets toutes formes confondues. Au vu du traitement, le taux de reconnaissance est estimé dans une moyenne approximant le taux de 97.8 % [84].

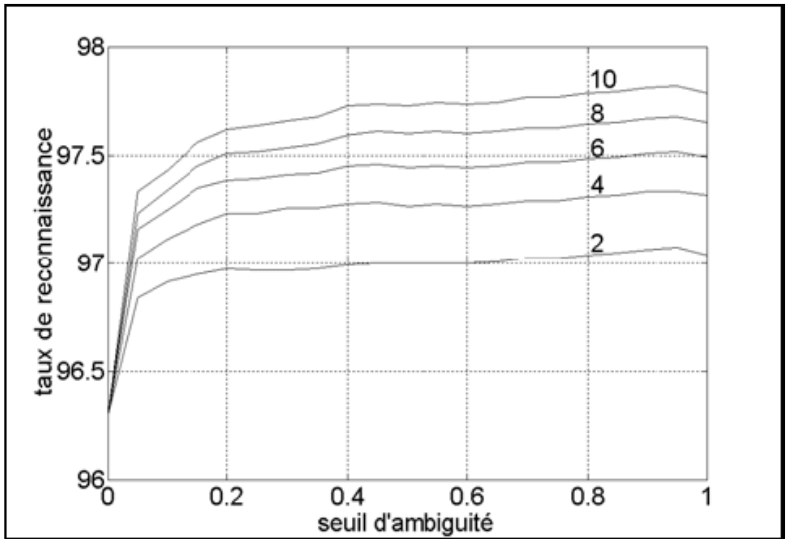


Figure 4.34. : Taux de reconnaissance en % du classifieur hybride en fonction des seuils de confusion et d'ambiguïté

4.4.1. Implémentation sur FPGA

La Figure 4.35 représente le circuit FPGA obtenu après l'implémentation de l'architecture de notre algorithme sur le circuit Xc250-5fg456C de Virtex.II de Xilinx.

L'occupation de l'espace FPGA est donnée par le tableau 4.3.

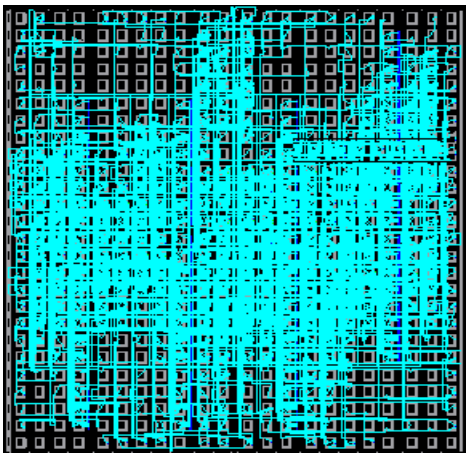


Figure 4.35.: Circuit FPGA réalisé

Tableau 4.3. : L'occupation de l'espace FPGA.

Slices (CLBs) occupés	618/1536	Occupation de 37%
IOBs occupés	138/200	Occupation de 69%
RAM	19/24	Occupation de 81%

→ La fréquence de fonctionnement de ce circuit peut aller jusqu'à 606.081 MHz.

→ Quatre paramètres ρ_n correspondant à quatre θ_n sont générés chaque 11,852 ns.

4.5. Discussion et Interprétation

Dans ce chapitre nous avons présenté un système de reconnaissance des formes par la construction d'informations tridimensionnelles d'une scène à partir des couples de régions homologues déterminés. La technique d'extraction d'attributs par les réseaux de neurones (seuillage par la Rpg) a permis une maximalisation de la corrélation entre l'image originale et l'image binaire produite. Ce qui est un des buts recherchés vu la robustesse de notre algorithme.

On observe aussi dans l'étape de détection de contours par le filtre de Deriche, (basé sur le critère de Canny récursif) une nette réduction du bruit qui a atténué les erreurs de reconnaissance. Les images résultats valident ce constat.

Parmi nos contributions, notons celle qui a permis de résoudre le problème de l'appariement des segments de droites des images stéréoscopiques. Les points forts de notre approche résident d'une part sur l'utilisation de plusieurs indices visuels (régions, contours,..) afin de bénéficier d'informations plus riches, et d'autre part sur le renforcement de la segmentation à chaque niveau du système.

En effet, la méthode comprend l'application de contraintes locales pour cerner les appariements les plus probables en utilisant une fenêtre de recherche et un critère de compatibilité des appariements basé sur le graphe d'adjacence de segments de droites contenus dans l'image. Un coefficient de similitude est calculé pour chaque couple potentiel et nous retenons pour ''homologue'' qu'un seul couple dont la valeur de ce coefficient est maximum.

Cette méthode a été implémentée sur une station de travail en langage C++ sous Unix appliquée sur des paires d'images stéréoscopiques. Les résultats de cette technique sont probants. Cette dernière se révèle très robuste et efficace, qualités que l'on peut renforcer encore en ajoutant d'autres critères tels que la luminance.

La méthode d'appariements stéréoscopiques, présenté dans ce travail a été utilisée (après implémentation de la TH sur un circuit FPGA) pour des tâches de localisation et aussi de navigation et autres moyennant la vision 3D pour le robot mobile autonome développé dans notre laboratoire (CESA). Lors de son test de déplacement et malgré la présence de bruits d'origine diverse (reflets dus à l'éclairage, etc.), notre robot mobile s'est accoutumé à l'environnement, ce qui montre encore une fois les bonnes règles qui régissent son apprentissage. Il a pu différencier entre des balises et les amers dans la scène (pilier, reflet). Ce qui est un des indicateurs de la robustesse de notre algorithme. En fait notre contribution a

montré les limitations d'un traitement de bas-niveau sur les images et la nécessité d'inclure la compétence de traitements de plus haut niveau. La méthode de stéréovision que nous avons développée par suite en est un exemple.On observe avant de faire entrer notre robot mobile dans la salle du laboratoire que nous lui modélisons le couloir par ses deux arêtes inférieures pour que ce dernier puisse se localiser. En se référant à sa position, le robot doit aussi générer une trajectoire lui permettant de se déplacer de sa position vers une position destination - le milieu des deux arêtes détectées qui forment le couloir en fixant le bout du couloir. La détection des deux arêtes du couloir se fait par la recherche de la plus grande valeur accumulée dans les cellules du tableau accumulateur.

L'arête gauche se distingue de l'arête droite par la valeur de sa variable θ dans l'intervalle [0, π]. L'arête gauche a un θ compris entre 0° et 90° et l'arête droite a un θ compris entre 90° et 180°. La mauvaise détection des arêtes de l'image de la scène observée influe négativement sur la bonne génération de la trajectoire car le robot mobile suppose que c'est un espace libre. Il peut donc heurter le mur de la salle de laboratoire ou un éventuel obstacle. Ce qui nous fait remarquer que cette approche donne de bons résultats si nous avons une bonne acquisition de l'image de la scène observée et si le milieu d'évolution est bien éclairé.

Il en ressort de l'expérimentation d'affirmer que notre travail présente des performances meilleures par comparaison à d'autres travaux édités dans les mêmes conditions d'environnement (taille d'images, types d'objets) car elle de type coopératif et dirigé, et la modification des attributs est effectuée par une règle probabiliste [102], [111], [112], [113] et [114].

Par exemple la référence [102] présente une approche complexe qui combine une «appearance based technic» avec un réseau de neurones suivi d'une technique «multi-résolution fusion décision»» qui permet d'assurer une robustesse du système au prix d'un temps de calcul additif. Contrairement à notre travail qui nous a permis d'avoir un bon compromis entre robustesse et temps de calcul qui sont d'ailleurs les indicateurs les plus importants de performances pour des systèmes travaillant en temps réel [77], [84], [115], [116] et [117].

4.6. Conclusion

Une des contributions apportées à notre système mobile ATRV2 dans un environnement qui peut être peu connu a priori. Notre motivation a été de rendre ce système capable de percevoir et d'utiliser ses données perceptuelles pour augmenter son autonomie de déplacement tout en évitant au cas des obstacles imprévus, pouvoir agir et réagir face à son environnement et accomplir ses tâches de traitements en vision d'une manière autonome.

Cet objectif requiert des capacités de traitement pour effectuer la mission demandée, en particulier, il est nécessaire de modéliser l'environnement à partir de ses données perceptuelles issues des capteurs embarqués. Cette étape est primordiale puisqu'elle est à la base des décisions futures sur laquelle reposera le traitement pour établir une localisation et par la suite gérer son déplacement dans l'environnement.

Pour construire le modèle de l'environnement, il nous a fallu développer des méthodes d'analyse et d'affichage intégrant tous les algorithmes et procédures développés telles que la lecture des données en provenance des différents capteurs embarqués (US, odomètre et banc stéréoscopique), les calibrages fort et faible pour le banc stéréoscopique, les changements de repères entre les différents capteurs embarqués avec celui du système mobile, et la fusion des données capteurs US-Banc stéréoscopique appartenant à leur champ de vision commun.

Pour le mode de représentation des mesures et de l'environnement, nous avons utilisé des grilles de certitude, car elles sont utilisables dans tout type d'environnement sans nécessiter d'aménagements particuliers, et peuvent être utilisées à partir des mesures issues de plusieurs types de capteurs. Chaque capteur construit sa carte locale qui va ensuite être intégrée à une carte locale du système mobile servant à établir la carte globale de l'environnement. La précision et la robustesse du modèle de l'environnement ainsi obtenu ont été testées dans l'étape de localisation.

L'algorithme de navigation autonome, appelé DVFF, que nous avons développé est un algorithme hybride qui combine à la fois deux méthodes de navigation existantes, l'algorithme de planification de trajectoire globale basé sur l'algorithme D*, et l'algorithme de navigation réactive non stop basé sur le principe des champs de forces virtuelles VFF.

Il utilise les valeurs courantes des capteurs ultrasonores embarqués, et les données provenant de son modèle interne, pour décider de l'action à effectuer. Il permet au système mobile ATRV2 de s'orienter vers sa position destination tout en évitant les collisions avec les obstacles de son environnement.

L'autre contribution à nos yeux essentielle a été de concevoir une version améliorée en relation avec l'état de l'art à savoir la mise en œuvre d'une méthode hybride de segmentation d'images pour la reconnaissance de formes (Rdf). Cette dernière a été implantée pour une application de vision robotique en temps réel.

Pour ce faire, nous avons adapté un modèle étiqueté basé sur les Rna's et la Transformée de Hough.

Lors de l'application avec l'ATRV2 en stéréovision, notre solution a été de mettre en exergue une interaction entre les traitements mis en œuvre. Notre système non loin d'être un système expert a pu permette des échanges dynamiques, contrôler le processus, avoir une base de connaissances, et un moteur d'inférence. En pis, il a pu être organisé en modules qui ont été les suivants : extraction des primitives, segmentation, appariement, calcul des informations 3D, recalage scène pour l'identification, localisation, navigation, apprentissage. La perception de l'environnement par notre robot mobile a té un problème riche et étendu, dont nous l'avons proposé en approche sémantique mais de nombreuses autres voies demeurent ouvertes.

Cependant la force du paradigme neuronal dans le domaine de la Rdf résulte principalement da la distribution des connaissances. En effet les formes apprises sont mémorisés à travers les poids des connexions du réseau, ce qui confère à ce dernier des capacités remarquables de généralisation et de résistance au bruit. Cependant des reproches peuvent être faits aux RNa's liés au caractère empirique de la conception de leur topologie.

La vision par ordinateur, discipline dans laquelle s'insère notre étude, est un sujet vaste et passionnant et notre contribution à réaliser un système de stéréovision et à lever un des aspects de la problématique de la reconnaissance de formes reste modeste.

Au demeurant, la méthode RANSAC (Random Sample Consensus) apparait comme étant une méthode de vote probabiliste qui semble être intéressante au paradigme vision-compréhension pour réduire les temps de calcul [23].

Cependant seulement son insertion (Ndlr. Ransac) dans un processus hybride pourrait nous éclairer d'avantage quant aux performances de cette méthode.

Ceci nous permis de passer en revue les méthodes d'estimation robuste les plus utilisées en vision robotique. L'utilisation de techniques de vote telles que la Transformée de Hough où la méthode de consensus de Ransac, représente un bon compromis entre robustesse et efficacité algorithmique avec une vitesse de convergence réduite.

Conclusion Générale

L'objectif principal de cette thèse est la conception d'un système permettant d'adapter les capacités du robot en fonction du contexte visuel.

Dans notre approche, ce terme de contexte visuel regroupe les informations liées à la structure de l'environnement dans lequel évolue le robot et celles de plus bas niveau liées à l'aspect des objets dans cet environnement (texture, illumination, etc.) jusqu'au haut niveau (reconnaissance d'objets).

La grande diversité de ces informations visuelles et des manières de les exploiter nous a poussés à nous intéresser aux techniques d'évolution artificielle, bien adaptées pour gérer des représentations à structure variable et explorer de grands espaces d'état. Cette thèse s'inscrit donc directement dans le cadre de la vision robotique. La prise en compte du contexte visuel implique la nécessité d'inclure toute la chaîne de vision dans le processus d'apprentissage, et plus particulièrement la vision bas niveau. La plupart des travaux présentés précédemment font en réalité une utilisation très restreinte de la vision.

Toute la partie concernant l'extraction d'informations depuis l'image est fixée, et l'apprentissage s'effectue uniquement sur l'utilisation qui est faite de cette information extraite. Ce parti pris est très réducteur par rapport à l'utilisation qui est faite de la vision par les animaux, où tout le processus est intimement lié au contexte visuel et à la tâche effectuée. Nous pensons qu'une approche plus globale de la vision est indispensable pour développer de vraies capacités d'apprentissage et d'adaptation sur des robots mobiles.

Comme nous l'avons indiqué précédemment, les travaux utilisant la vision en robotique traitent généralement des images de quelques pixels seulement. Cela s'explique aisément d'un point de vue computationnel, nous voulons en effet prendre en compte un maximum d'indices visuels, et notamment des informations de texture qui ne sont disponibles qu'en haute résolution.

Heureusement, nous bénéficions de ce point de vue du développement continu de la puissance de calcul disponible, ce qui rend envisageable des expériences qui restaient impensables il y a dix ans. Notons toutefois que ce que nous entendons par “haute résolution” reste limité, puisqu’il s’agit ici d’images de 320 x 160 pixels.

Cependant le coût computationnel reste élevé et les expériences présentées dans cette thèse durent généralement des dizaines de jours.

Un des inconvénients majeurs est que cela rend impossible toute analyse statistique digne de ce nom sur la variabilité des résultats obtenus.

Plusieurs algorithmes ont été développés pour résoudre cet aspect de la problématique de la vision en robotique mobile entre autres les algorithmes développés par [28, 29] pour restaurer les niveaux de gris des images afin d’obtenir des meilleures images de pointe et de meilleure segmentation.

1) Une des contributions majeures de cette thèse a été de concevoir une structure extensible pour représenter des algorithmes de vision, et de développer les outils permettant de les faire évoluer automatiquement. Ces concepts sont réutilisables pour beaucoup d’autres applications, potentiellement toutes celles qui utilisent la vision. Les modifications nécessaires pour cela consistent à déterminer les types de données, la base de primitives et les données d’entrée et de sortie adaptés à l’application visée.

2) Une autre contribution essentielle rapportée aussi à notre travail de thèse et qui vient en amont a concerné la localisation et une navigation spécifique de notre système mobile ATRV2 dans un environnement qui peut être peu connu a priori.

Notre motivation a été de rendre ce système capable de percevoir et d’utiliser ses données perceptuelles pour augmenter son autonomie de déplacement tout en évitant au cas des obstacles imprévus, pouvoir agir et réagir face à son environnement et accomplir ses tâches de traitements en vision d’une manière autonome.

Cet objectif requiert des capacités de traitement pour effectuer la mission demandée, en particulier, il est nécessaire de modéliser l’environnement à partir de ses données perceptuelles issues des capteurs embarqués. Cette étape est primordiale puisqu’elle est à la base des décisions futures sur laquelle repose le traitement pour établir une localisation et par la suite gérer son déplacement dans l’environnement.

Pour construire le modèle de l'environnement, il nous a fallu développé des méthodes d'analyse et d'affichage intégrant tous les algorithmes et procédures développés telles que la lecture des données en provenance des différents capteurs embarqués (US, odomètre et banc stéréoscopique), les calibrages fort et faible pour le banc stéréoscopique, les changements de repères entre les différents capteurs embarqués avec celui du système mobile, et la fusion des données capteurs US-Banc stéréoscopique appartenant à leur champ de vision commun.

Pour le mode de représentation des mesures et de l'environnement, nous avons utilisé des grilles de certitude, car elles sont utilisables dans tout type d'environnement sans nécessiter d'aménagements particuliers, et peuvent être utilisées à partir des mesures issues de plusieurs types de capteurs. Chaque capteur construit sa carte locale qui va ensuite être intégrée à une carte locale du système mobile servant à établir la carte globale de l'environnement. La précision et la robustesse du modèle de l'environnement ainsi obtenu ont été testées dans l'étape de localisation.

L'algorithme de navigation autonome, appelé DVFF, que nous avons développé est un algorithme hybride qui combine à la fois deux méthodes de navigation existantes, l'algorithme de planification de trajectoire globale basé sur l'algorithme D*, et l'algorithme de navigation réactive non-stop basé sur le principe des champs de forces virtuelles VFF.

Il utilise les valeurs courantes des capteurs ultrasonores embarqués, et les données provenant de son modèle interne, pour décider de l'action à effectuer. Il permet au système mobile ATRV2 de s'orienter vers sa position destination tout en évitant les collisions avec les obstacles de son environnement.

3) L'autre contribution à nos yeux essentielle a été de concevoir une version améliorée d'un processus de reconnaissance de formes (Rdf) appliquée à la vision en robotique mobile et en temps réel. Pour ce faire, nous avons adapté un modèle étiqueté basé sur les Rna's et la Transformée de Hough. Nous visons en particulier un système embarqué de type capteurs de vision "intelligents" programmable sur un circuit FPGA. Celle-ci présente l'intérêt de pouvoir à la fois acquérir et traiter l'image en consommant extrêmement peu d'énergie.

En effet, dans le manuscrit, nous avons passé en revue les méthodes d'estimation robuste le plus utilisées en vision robotique. L'utilisation de ces méthodes est nécessaire afin de réaliser des tâches en environnement réel. Le prix à payer est un temps de calcul un peu plus élevé et une vitesse de convergence réduite. Si les techniques de vote (Hough, Ransac) sont très efficaces, le temps de calcul est souvent trop élevé pour assurer une utilisation des algorithmes de vision à une cadence compatible avec la commande d'un robot.

Toutefois, la Transformée de Hough est très rarement utilisée seule en vision robotique car pour des problèmes qui nécessitent l'estimation de plus de trois ou quatre paramètres, les temps de calculs deviennent prohibitifs. Pour parer à cela, une hybridation avec les Rna's représente un bon compromis entre robustesse et efficacité algorithmique.

Depuis plusieurs variantes ont été proposées [70]. Cette approche repose sur une discrétisation de l'espace des paramètres. On obtient alors des hyper-cubes dans l'espace d'état auquel sont associés des accumulateurs. Pour un jeu de données de taille minimale, les paramètres recherchés sont estimés et l'accumulateur correspondant de l'hyper-cube est incrémenté. Ce processus est itéré jusqu'à considérer toute les combinaisons possibles des données à disposition.

L'accumulateur ayant la valeur la plus importante correspond alors à la meilleure estimation des paramètres. La Transformée de Hough est bien adaptée aux problèmes ayant un nombre important de données par rapport aux nombre des paramètres à estimer. En effet, si les données et les inconnues sont de taille équivalente, il est difficile de trouver un accumulateur prépondérant par rapport aux autres. En plus, dû à la discrétisation et au bruit, il est possible que l'optimum soit délocalisé. La Transformée de Hough est très robuste car elle effectue une recherche globale et exhaustive.

L'implémentation de la Transformée de Hough sur un système informatique a l'avantage de présenter un large éventail de tâches à exécuter en parallèle. Ce dernier peut exécuter d'autres algorithmes en plus de la Transformée de Hough et être capable de donner une performance aux calculs antérieurs tels qu'une segmentation ou aussi en amont une détection de contours.

Nous avons présenté l'évaluation de la Transformée de Hough en utilisant le mode de calcul en ligne. L'algorithme élaboré permet de générer tous les paramètres

de la Transformée de Hough d'une manière très rapide et ceci grâce au mode de présentation des données en série, poids forts en tête. Les performances de l'architecture pipeline résident essentiellement dans la manière de génération des bits résultats r_{n+1} et $r_{n+1+k/2}$.

Pour une meilleure illustration, nous avons implémenté notre algorithme sur un circuit FPGA de Xilinx, plus précisément la famille Virtex2, qui contient un module de génération des paramètres (ρ,θ) ainsi que le module nécessaire pour le processus de vote. Cette dernière dispose de ressources dédiées aux opérations arithmétiques et aussi dans le domaine de traitement de signal. Le circuit donne de très bons résultats en termes de performances temporelles.

L'idée d'utiliser la TH est due à sa robustesse dans l'analyse de l'extraction des primitives d'objets et aussi à ses propriétés d'implémentation hardware. Au demeurant, elle nous permet entre autre d'optimiser en temps et en espace tout le calcul global.

Une autre façon de justifier l'utilisation conjointe des techniques et concepts développés dans le cadre des approches hybrides émerge de l'étude de l'expertise et plus généralement de la cognition humaine. Un être humain est hybride dans la mesure où ses concepts sont hybrides. Ce constat a suscité notre intérêt en associant une approche neuronale à la Transformée de Hough que nous avons commenté dans nos expérimentations antérieures.

Lors de l'application avec l'ATRV2 en stéréovision, notre solution a été de mettre en exergue une interaction entre les traitements mis en œuvre. Notre système non loin d'être un système expert a pu permette des échanges dynamiques, contrôler le processus, avoir une base de connaissances, et un moteur d'inférence. En plus, il a pu être organisé en modules qui ont été les suivants : extraction des primitives, segmentation, appariement, calcul des informations 3D, recalage scène pour l'identification, localisation, navigation, apprentissage. La perception de l'environnement par notre robot mobile a té un problème riche et étendu, dont nous l'avons proposé en approche sémantique mais de nombreuses autres voies demeurent ouvertes.

Cependant la force du paradigme neuronal dans le domaine de la Rdf résulte principalement da la distribution des connaissances. En effet les formes apprises

sont mémorisés à travers les poids des connexions du réseau, ce qui confère à ce dernier des capacités remarquables de généralisation et de résistance au bruit. Cependant des reproches peuvent être faits aux Rna's liés au caractère empirique de la conception de leur topologie.

La vision par ordinateur, discipline dans laquelle s'insère notre étude, est un sujet vaste et passionnant. Les problèmes inhérents à la vision et particulièrement en robotique mobile sont complexes et multiformes. Notre contribution à réaliser un système de stéréovision et à essayer de lever un des aspects de la problématique de la perception et de la compréhension automatiques de l'environnement de la reconnaissance de formes reste modeste.

Au demeurant, la méthode RANSAC (Random Sample Consensus) apparait comme étant une méthode de vote probabiliste qui semble être intéressante au paradigme vision-compréhension pour réduire les temps de calcul [21, 115].

Cependant seulement son insertion (Ndlr. Ransac) dans un processus hybride pourrait nous éclairer d'avantage quant aux performances de cette méthode. Ceci nous a permis de passer en revue les méthodes d'estimation robuste les plus utilisées en vision robotique. L'utilisation de techniques de vote telles que la Transformée de Hough où la méthode de consensus de Ransac, représente un bon compromis entre robustesse et efficacité algorithmique, avec une vitesse de convergence réduite.

En perspective, notre travail peut être poursuivi dans un projet de recherche qui consisterait à inclure le système mobile ATRV2 comme un agent autonome dans un système de surveillance prédéfinie. La tâche de surveillance serait restreinte à la détection des intrus dans un environnement structuré. Le système mobile pourrait éventuellement ajuster son mouvement en fonction de sa propre interprétation de l'environnement proche et de recueillir les informations reçues de ses capteurs embarqués et les transmettre à un superviseur humain. Ce dernier recevrait ces data-informations sur une interface graphique et commanderait au demeurant le système mobile par le biais d'un lien radiofréquence.

ANNEXES

ANNEXE A

Calculs liés à la géométrie du flux optique.....164

ANNEXE B

Mobile Robot ATRV2.....171

ANNEXE C

Transformée de Hough.....174

ANNEXE A

Calculs liés à la géométrie du flux optique

D'un point de vue géométrique, le flux optique est la projection sur l'image du mouvement des objets dans le référentiel de la caméra. Pour les applications robotiques, on considère généralement que la caméra est fixée sur un robot qui se déplace et que les autres objets de l'environnement sont immobiles. Le mouvement apparent est donc uniquement causé par le déplacement du robot.

Nous allons détailler dans cette annexe A, les calculs permettant d'obtenir les relations décrivant le flux optique dans différents types de projection.

A.1 : Définition du flux optique dans le repère de la caméra

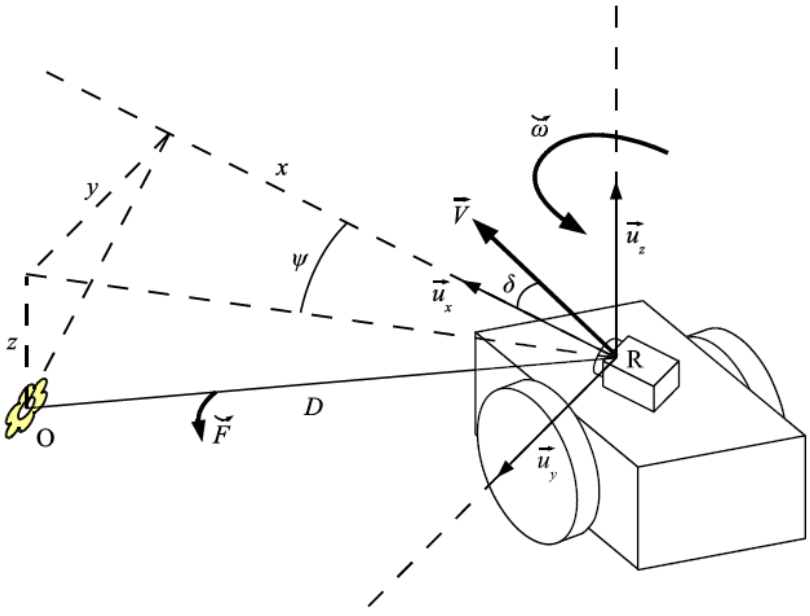


Figure. A.1 : Déplacement du robot générant le flux optique.

Nous reprenons en figure A.1 la représentation du mouvement générant le flux optique. Pour exprimer le déplacement du point O dans le repère $((R,\vec{u}_x,\vec{u}_y,\vec{u}_z))$, nous utilisons les relations suivantes déduites de cette figure lorsqu'on considère uniquement le mouvement de translation $\vec{V}$:

$$\dot{x} = -V\,coc\delta$$
$$\dot{y} = -V\,sin\,\delta$$
$$\dot{z} = 0$$

Lorsqu'on considère au contraire uniquement le mouvement de rotation, on a :

$$\dot{\Psi} = -(\omega + \dot{\delta})$$
$$\dot{z} = \mathbf{0}$$
$$\dot{D} = \mathbf{0}$$

Exprimons à présent x et y en fonction de Ψ :

$$x = \sqrt{D^2 - z^2}\ cos\Psi$$
$$y = \sqrt{D^2 - z^2}\ sin\Psi$$

En dérivant ces expressions, on obtient :

$$\begin{aligned} \dot{x} &= \sqrt{D^2 - z^2}\ sin\Psi\,\dot{\Psi} \\ &= (\omega + \dot{\delta})y \\ \dot{y} &= \sqrt{D^2 - z^2}\ cos\Psi\,\dot{\Psi} \\ &= -(\omega + \dot{\delta})x \end{aligned}$$

D'où les relations finales prenant en compte la translation et la rotation :

$$\dot{x} = -V\,cos\delta + (\omega + \dot{\delta})y$$
$$\dot{y} = -V\,sin\delta - (\omega + \dot{\delta})x$$
$$\dot{z} = 0$$

Il sera également utile pour la suite de calculer $\dot{D}$. On réalise cela en utilisant la relation :

$$D^2 = x^2 + y^2 + z^2$$

En la dérivant, on obtient :

$$\mathbf{2}D\dot{D} = \mathbf{2}x\dot{x} + \mathbf{2}y\dot{y} + \mathbf{2}z\dot{z}$$
$$\dot{D} = \frac{x}{D}(-V\cos\delta + (\omega + \dot{\delta})y) + \frac{y}{D}(-V\sin\delta - (\omega + \dot{\delta})x)$$

D'où finalement :

$$\dot{D} = \frac{V}{D}(x\cos\delta + y\sin\delta)$$

A.2 : Projection sphérique

Nous reprenons dans la figure A.2 la représentation du flux optique lorsqu'il est projeté en coordonnée sphériques. Notons que les angles α et β ne sont pas disponibles directement sur une image sphérique, il est nécessaire de les calculer à partir des coordonnées x_s et y_s avec les formules suivantes :

$$\alpha = x_S^2 + y_S^2$$
$$\beta = arctan\frac{y_S}{x_S}$$

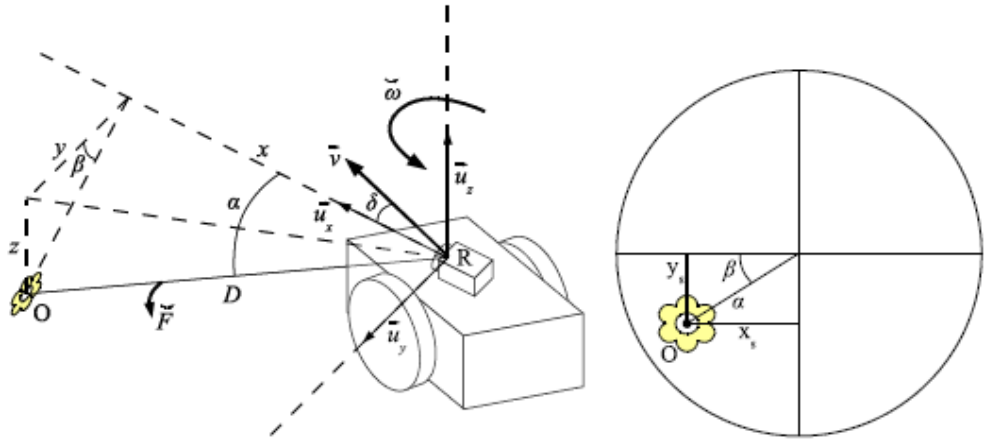


Figure. A.2 : Représentation de l'image perçue en coordonnées sphériques.

On obtient par géométrie à partir de la figure A.2 les relations suivantes :

$$x = D\ cos\,\alpha$$
$$y = D\ sin\,\alpha\, cos\,\beta$$

Ainsi que les relations :

$$cos\,\alpha = \frac{x}{D}$$
$$tan\,\beta = \frac{z}{y}$$

Pour obtenir la représentation du flux optique en coordonnées sphériques, il suffit de dériver ces deux dernières expressions :

$$\sin\alpha\,\dot{\alpha} = \frac{\dot{x}}{D} - \frac{x\dot{D}}{D^2}$$
$$= \frac{-V\cos\delta + (\omega+\dot{\delta})y}{D} + \frac{xV(x\cos\delta + y\sin\delta)}{D^3}$$
$$= \frac{V}{D}((\cos^2\alpha - 1)\cos\delta + \cos\alpha\cos\beta\sin\delta) + \sin\alpha\cos\beta((\omega+\dot{\delta})$$

D'où finalement :

$$\dot{\alpha} = \frac{V}{D}(\cos\delta\sin\alpha - \cos\alpha\cos\beta\sin\delta) - \cos(\omega+\dot{\delta})$$

On procède de même pour déterminer $\dot{\beta}$:

$$\dot{\beta} = \frac{\sin\beta}{D\sin\alpha}(V\sin\delta + D\cos(\omega+\dot{\delta}))$$
$$= \frac{V\sin\delta\sin\beta}{D\sin\alpha} + \frac{Sin\beta}{\tan\alpha}(\omega+\dot{\delta})$$

A.3 : Projection polaire

Nous reprenons dans la figure A.3 la représentation du flux optique lorsqu'il est projeté en coordonnées polaire. On obtient de cette figure les relations :

$$x = D\cos\theta\cos\Psi$$
$$y = D\cos\theta\sin\Psi$$
$$\tan\Psi = \frac{y}{x}$$
$$\sin\theta = \frac{z}{D}$$

Ici aussi, nous allons dériver les deux dernières expressions pour obtenir la représentation du flux optique en coordonnées polaires :

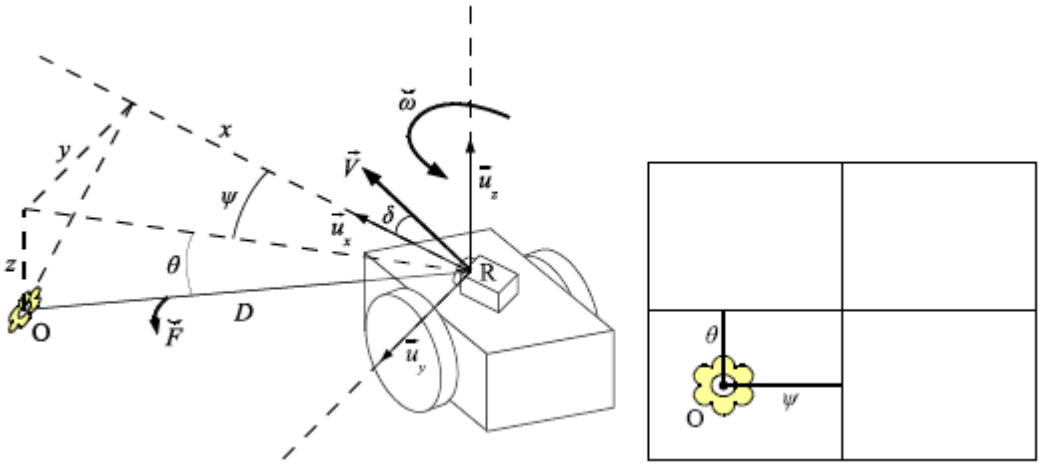


Figure A.3 : Représentation de l'image perçue en coordonnées polaires.

$$\frac{\dot{\Psi}}{\cos^2\Psi} = -\frac{y\dot{x}}{x^2} + \frac{\dot{y}}{x}$$
$$= -\frac{\tan\Psi}{x}(-V\cos\delta + (\omega+\dot{\delta})y) + \frac{1}{x}(-V\sin\delta - (\omega+\dot{\delta})x)$$
$$= \frac{V}{x}(\tan\Psi\cos\delta - \sin\delta) - (\omega+\dot{\delta})\tan^2\Psi + 1)$$

On en déduit :

$$\dot{\Psi} = \frac{V\cos^2\Psi}{x}(\tan\cos\delta - \sin\delta) - (\omega+\dot{\delta})$$
$$= \frac{V}{D\cos\theta}(\sin\Psi\cos\delta - \cos\Psi\sin\delta) - (\omega+\dot{\delta})$$
$$= \frac{V\sin\Psi-\delta)}{D\cos\theta} - (\omega+\dot{\delta})$$

On procède de même pour déterminer $\dot{\theta}$:

$$\cos\theta\,\dot{\theta} = \frac{\dot{z}}{D} - \frac{z\dot{D}}{D^2}$$
$$= \frac{zV}{D^3}(x\cos\delta + y\sin\delta)$$
$$= \frac{V\sin\theta}{D^2}(D\cos\theta\cos\Psi\cos\delta + D\cos\theta\sin\Psi\sin\delta)$$

D'où finalement :

$$\dot{\theta} = \frac{V\sin\theta}{D}(\cos\Psi\cos\delta + \sin\Psi\sin\delta)$$
$$= \frac{V\sin\theta\cos(\Psi-\delta)}{D}$$

A.4 : Projection plane

Nous reprenons dans la figure A.4 la représentation du flux optique lorsqu'il est projeté sur un plan image. On obtient de cette figure les relations :

$$x_i = -f\frac{y}{x}$$

$$y_i = f\frac{z}{x}$$

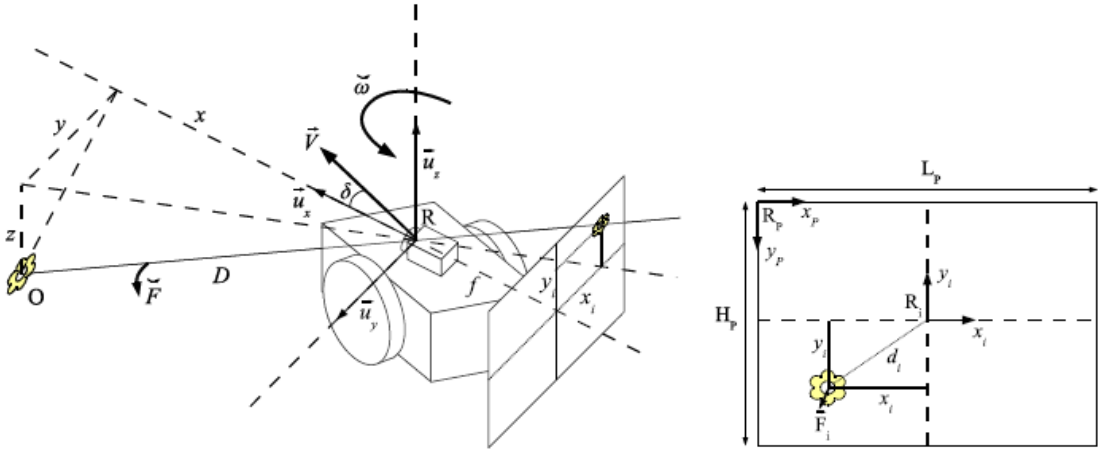


Figure. A.4 : Représentation du flux optique projeté sur le plan image.

Nous allons dériver ces deux expressions pour obtenir la représentation du flux optique dans l'image.

$$\dot{x}_i = \frac{fy\dot{x}}{x^2} - \frac{f\dot{y}}{x} \qquad \text{A.1}$$

$$= \frac{-x_i}{x}(-V\cos\delta + (\omega+\dot{\delta})y) - \frac{f}{x}(-V\sin\delta - (\omega+\dot{\delta})x) \qquad \text{A.2}$$

$$\frac{V}{x}(x_i\cos\delta + f\sin\delta) + (\omega+\dot{\delta})(\frac{x_i^2}{f} + f) \qquad \text{A.3}$$

$$= \frac{fV}{x}(\frac{x_i\cos\delta}{f} + \sin\delta) + f(\omega+\dot{\delta})(\frac{x_i^2}{f^2} + \mathbf{1}) \qquad \text{A.4}$$

On procède de même pour déterminer $\dot{y}_i$:

$$\dot{y}_i = -\frac{fz\dot{x}}{x^2} + \frac{f\dot{z}}{x}$$

$$= -\frac{y_i}{x}(-V\cos\delta + (\omega+\dot{\delta})y)$$

$$= \frac{Vy_i\cos\delta}{x} + (\omega+\dot{\delta})\frac{x_i y_i}{f}$$

Ces relations sont exprimées dans le repère image centré sur R_i. Il est donc nécessaire avant tout d'effectuer la transformation des coordonnées en pixel vers ces coordonnées image. Cela est effectuer à l'aide des relations suivantes obtenues de la figure A.4 et où η est l'angle de vue horizontal :

$$x_i = \left(\frac{2x_p}{L_p} - 1 \right) f \tan(\eta / 2)$$

$$y_i = -\left(\frac{2y_p}{H_p} - 1 \right) f \frac{H_p \tan(\eta / 2)}{L_p}$$

$$\dot{x}_i = \frac{2 f \tan(\eta / 2)}{L_p} \dot{x}_p$$

$$\dot{y}_i = -\frac{2 f \tan(\eta / 2)}{L_p} \dot{y}_p$$

ANNEXE B

Mobile Robot ATRV2

A. Mobile Robot Specifications

Length 105 cm, Width 80 cm, Height 65 cm, Clearance 7 ½ cm,

Weight: 118 kg

Body Formed and welded aluminium

Speed 0 – 1.5 m/sec

Payload 100 kg

Run Time 4 to 6 hours, terrain dependent

Drive 4-wheel, PWM

Steering Skid Steering

Turn Radius Zero (turns on center)

Tires 31.75 cm pneumatic knobby

Batteries 4, 24V, 672 Watt/hr

Aux. Voltages Reg. +5, +12, & system battery voltage of 18–27V

B. Aux. Power Ports

Motion Control IROBOT rFLEX System

Motors 2 high torque, 24V DC servo motors

Computer Single or dual Pentium Lv computer

Software IROBOT Mobility Robot Infrastructure

I/O Ports Ethernet, RS-232, Joystick

Sensors Sonar: 12 (6 front, 2 each side, 2 rear)

Safety Motor enable key switch plus four emergencies kill buttons

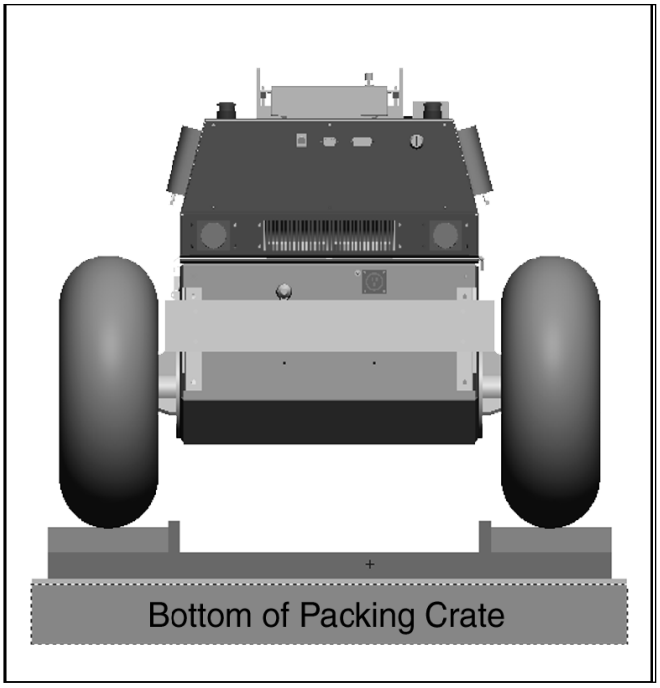


Figure B.1 : ATRV Mobile Robot On Packing Crate, Rear View.

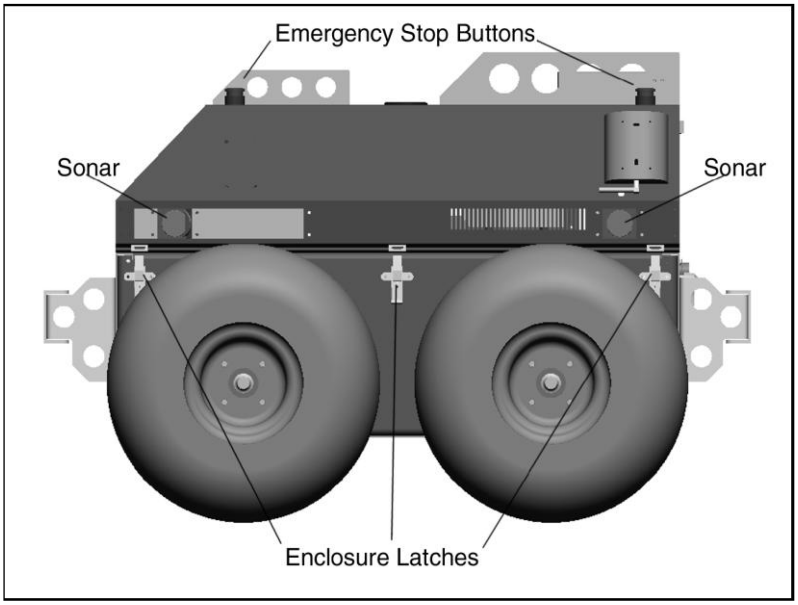


Figure B.2 : ATRV Mobile Robot Side View.

C. Accessories

Choice of Pentium-IV® Computer Systems

PR Triclops 3D Vision System

High performance vision system

Computerized pan-tilt unit

Camera and frame grabber options

Wireless communication (1-3 Mb/s Ethernet)

Integrated SICK proximity laser scanner

ASCII-to-speech interface

Computerized navigation compass

Global Positioning System (GPS) receiver

Digital Global Positioning System (DGPS) receiver

Inertial sensors

Tactile sensing bumpers

PTZ (Pan-tilt-zoom) camera

Absolute Position Sensor

6-Axis Inertial Sensor

(a) (b)

Figure B.3 : Robot mobile utilisant la stéréovision.

(a) : à l'arrêt

(b) : en déplacement

ANNEXE C

Transformée de Hough

I. Evaluation des différents algorithmes de la transformée de Hough dans le mode de calcul en ligne.

Avizienis [119] a introduit l'écriture des nombres dans un système redondant de représentation. Plus tard furent élaborés des algorithmes de calcul pour des opérations élémentaires et des fonctions plus complexes.

Ces algorithmes sont basés sur la circulation des opérandes dans l'opérateur, bit par bit, le bit le plus significatif (MSB) en tête [120].

Les propriétés du mode de traitement en ligne sont définies de la manière suivante :

→ Les opérandes sont introduits, à chaque cycle, bit par bit, le bit de poids fort en tête.

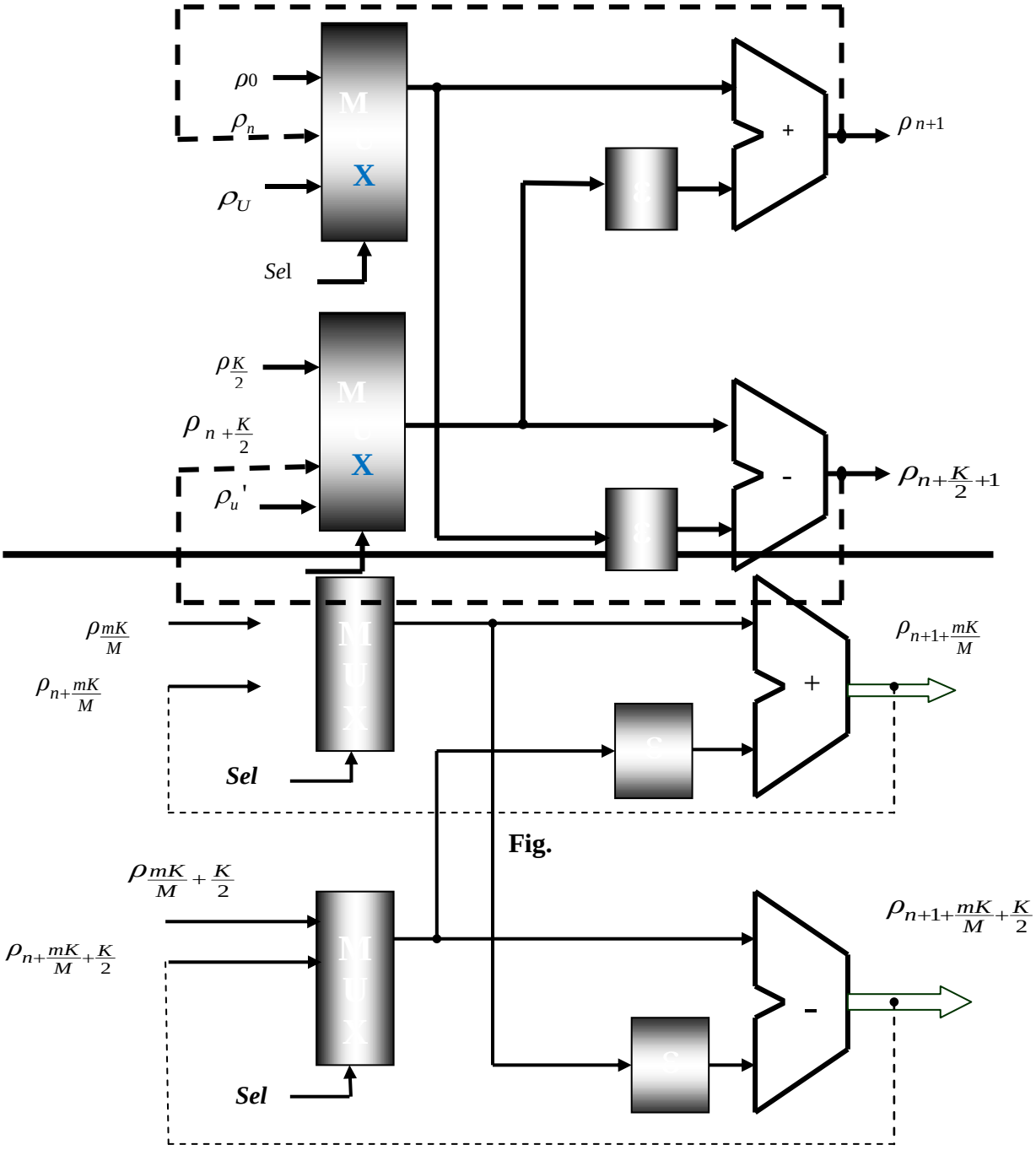
→ Les résultats sont obtenus de la même manière, bit par bit, poids fort en tête, mais avec un retard p, tel qu'au pas j, alors que les $J^{(ième)}$ digits des opérandes sont introduits, le $(J-p)^{(ième)}$ digit du résultat est généré.

→ Le retard p est très petit devant la taille des opérandes d'entrée. Les résultats et les opérandes d'entrées sont représentés dans un système redondant d'écriture des nombres.

Le calcul en mode semi-line (Half-line) présente une grande similitude avec celui en mode en ligne. La seule différence consiste en l'introduction de certains opérandes avec tous leurs digits, et ce, dès le lancement de l'opération. De tels opérandes peuvent être considérés comme les paramètres de l'opération alors ceux restants (variables) sont introduits comme dans le mode enligne, bit par bit de manière séquentielle.

II. Implémentation de la Transformée de Hough.

La Transformée de Hough est difficile à implémenter à cause du nombre énorme de calcul nécessaire pour générer tous les ρ_n correspondant à tous les θ_n, pour chaque point contour de l'image. Les implémentations proposées par S. Tagzout & al. sont données par les Figures ci-dessous *C(1) et C(2).*



Figures C(1) et C(2) : Architectures de S. Tagzout et al.

Nous avons estimé l'utilisation du mode de calcul en ligne afin d'accroître la vitesse de calcul des paramètres ρ_n et qui consiste à utiliser une architecture pipeline. Ceci est particulièrement intéressant lorsqu'il s'agit d'évaluer un grand nombre de paramètre ρ_n.

Dans cette partie, nous présenterons les caractéristiques et les performances d'un circuit pouvant recevoir un flux de pixels dont les cordonnées sont codées sur 10 bits. L'implémentation de cette nouvelle Transformée de Hough par le mode de calcul en ligne implique l'implémentation des deux expressions du résidu complet H [J] et H' [J].

La structure générale de l'implémentation de la Transformée de Hough en ligne (voir figure 9) sera divisée en deux parties:

→ Bloc de normalisation (calcul de l'accumulation partielle L [J] et L' [J]).

→ Bloc de sélection (Calcul des résidus complet H [J] et H' [J] et génération du bit résultat r_{n+1} [J] et $r_{n+k/2+1}$ [J].

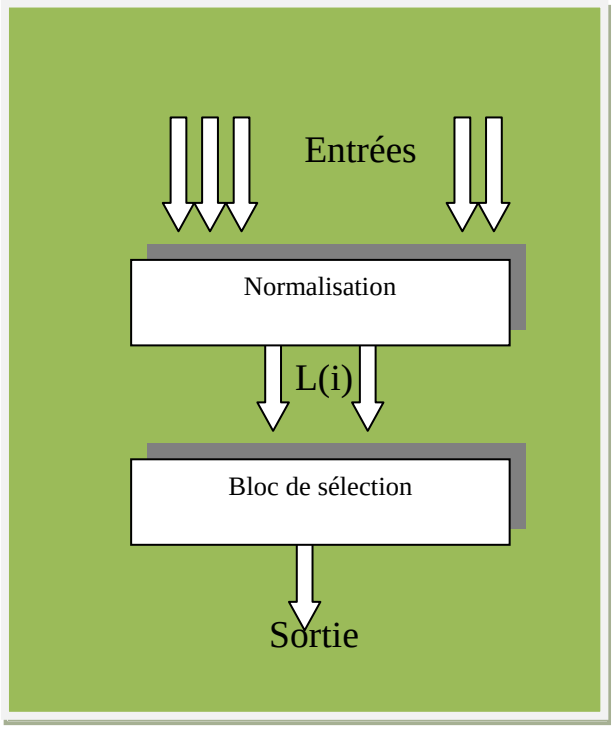


Figure C(3) : Structure générale de l'architecture en ligne.

Comme montré sur la Figure C(3), il existe deux blocs principaux dans l'architecture en ligne :

→ Bloc de normalisation : Calcul de l'accumulation partielle L(j).

→ Bloc de sélection : Calcul de l'accumulation complète H et génération du bit résultat à partir de la fonction de sélection.

II.1 : Bloc de normalisation.

Le rôle de ce bloc est l'évaluation de l'expression de l'accumulation partielle donnée par : $L(j)= 2^{-p} (r_n (j) +\varepsilon\ r_{n+k/2} (j)) – 2\ r_{n+1}(j)$

On remarque que $L_b (j)$ dépend du retard p**,** de la constante ε, des entrées $r_n (j)$ $r_{n+k/2}(j)$ et de la sortie $r_{n+1}(j)$**.** Pour réaliser cette évaluation nous allons utiliser une ROM dans laquelle nous stockerons toutes les valeurs possibles de $L_b (j)$.

Tableau C(1) : Toutes les valeurs possibles de $L_b(j)$

r_n^+	r_n^-	$r_{n+k/2}^+$	$r_{n+k/2}^-$	r_{n+1}^+	r_{n+1}^-	L_b (J) en base10	L_b (j) en base2
1	1	1	1	0	1	-2.257825	101,1011111
1	1	0	0	0	1	-2.25	101,1100000
1	1	0	1	0	1	-2.272175	101,1100001
0	0	1	1	0	1	-2.007825	101,1111111
0	0	0	0	0	1	-2	110,0000000
0	0	0	1	0	1	-1.992175	110,0000001
0	1	1	1	0	1	-1.757825	110,0011111
0	1	0	0	0	1	-1.75	110,0100000
0	1	0	1	0	1	-0.1742175	110,0100001
1	1	1	1	0	0	-0,257825	111,1011111
1	1	0	0	0	0	-0,25	111,1100000
1	1	0	1	0	0	-0.242175	111,1100001
0	0	1	1	0	0	-0,007825	111,1111111
0	0	0	0	0	0	0	000,0000000
0	0	0	1	0	0	0.007825	000,0000001
0	1	1	1	0	0	0,242175	000,0011111
0	1	0	0	0	0	0,25	000,0100000
0	1	0	1	0	0	0.257825	000,1000001
1	1	1	1	1	1	1,742175	001,1011111
1	1	0	0	1	1	1,75	001,1100000
1	1	0	1	1	1	1,757825	001,1100001
0	0	1	1	1	1	1,992175	001,1111111
0	0	0	0	1	1	2	010,0000000
0	0	0	1	1	1	2,007825	010,0000001
0	1	1	1	1	1	2,242175	010,0011111
0	1	0	0	1	1	2,25	010,0100000
0	1	0	1	1	1	2,257825	010,0100001

Pour sélectionner la case mémoire voulue, on utilise un décodeur qui fournit 27 sorties. Sa table de vérité est donnée par le tableau suivant :

Tableau C(2) : Table de vérité du décodeur d'adresses.

r_n^+	r_n^-	$r_{n+k/2}^+$	$r_{n+k/2}^-$	$r_{n+1}+$	$r_{n+1}-$	L_b (J) en base10	L_b (j) en base 2	L_b(j) normalisé	adresses
1	1	1	1	0	1	-2.257825	101,1011111	1011011111	111101
1	1	0	0	0	1	-2.25	101,1100000	1011100000	110001
1	1	0	1	0	1	-2.272175	101,1100001	1011100001	110101
0	0	1	1	0	1	-2.007825	101,1111111	1011111111	001101
0	0	0	0	0	1	-2	110,0000000	1100000000	000001
0	0	0	1	0	1	-1.992175	110,0000001	1100000001	000101
0	1	1	1	0	1	-1.757825	110,0011111	1100011111	011101
0	1	0	0	0	1	-1.75	110,0100000	1100100000	010001
0	1	0	1	0	1	-0.174275	110,0100001	1100100001	010101
1	1	1	1	0	0	-0,257825	111,1011111	1111011111	111100
1	1	0	0	0	0	-0,25	111,1100000	1111100000	110000
1	1	0	1	0	0	-0.242175	111,1100001	1111100001	110100
0	0	1	1	0	0	-0,007825	111,1111111	1111111111	001100
0	0	0	0	0	0	0	000,0000000	0000000000	000000
0	0	0	1	0	0	0.007825	000,0000001	0000000001	000100
0	1	1	1	0	0	0,242175	000,0011111	0000011111	011100
0	1	0	0	0	0	0,25	000,0100000	0000100000	010000
0	1	0	1	0	0	0.257825	000,1000001	0001000001	010100
1	1	1	1	1	1	1,742175	001,1011111	0011011111	111111
1	1	0	0	1	1	1,75	001,1100000	0011100000	110011
1	1	0	1	1	1	1,757825	001,1100001	0011100001	110111
0	0	1	1	1	1	1,992175	001,1111111	0011111111	001111
0	0	0	0	1	1	2	010,0000000	0100000000	000011
0	0	0	1	1	1	2,007825	010,0000001	0100000001	000111
0	1	1	1	1	1	2,242175	010,0011111	0100011111	011111
0	1	0	0	1	1	2,25	010,0100000	0100100000	010011
0	1	0	1	1	1	2,257825	010,0100001	0100100001	010111

II.2 : Bloc de sélection.

Ce bloc est la partie cruciale de tout algorithme en ligne, il s'occupe de la réalisation des opérations suivantes :

→ Evaluation du résidu complet H (j)

→ Evaluation de l'intervalle d'appartenance de H (j)

→ Génération des bits résultats $r_{n+1}(j)$

II.2.1 : Calcul du résidu complet H(j).

Le calcul du résidu complet est détaillé dans la Figure ci-dessous.

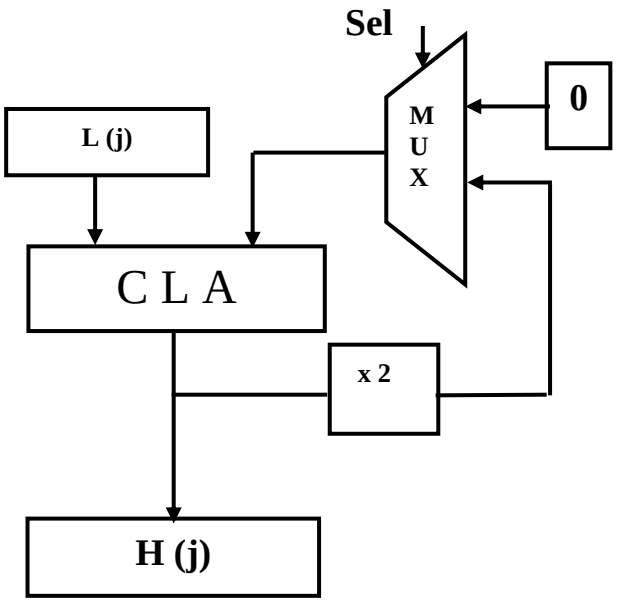


Figure C(4) : Partie de l'architecture calculant H (j).

Les cellules (x2) réalisent la multiplication par deux de H (j-1) cela signifie un décalage à gauche de l'opérande.

II.2.2 : Taille du bus

Pour trouver la taille du bus à l'entrée du CLA on commence par calculer le nombre de bits de L_b (j) sortant de la ROM :

$L(j)=2^{-2}(r_n(j) + \varepsilon\, r_{n+k/2}(j)) - 2\, r_{n+1}(j)$

r_n (j) et $r_{n+k/2}$ (j) étant sur deux bits chacun et $\varepsilon = 2^{-5}$

Le calcul de la largeur du bus est régi par la valeur de L(j), celle-ci contient la valeur de ε qui est codée sur 6 bits, la valeur du délai égale à 2, et elle est codée sur 3 bits, l'ensemble est codé sur 9 bits et le bit de signe de la soustraction du bit résultat augmente la taille de L(j) à 10 bits. Le schéma suivant illustre le processus :

r_n (j)				,							
$r_{n+k/2}$ (j)				,							
ε $r_{n+k/2}$ (j)				,							
r_n (j) +ε $r_{n+k/2}$ (j)				,							
-2 r_{n+1}(j)				,							
L (j)				,							

Nous savons qu'en mode en ligne les bits résultats ne sont générés qu'après un délai p c'est pour cette raison qu'on a utilisé des multiplexeurs à l'entrée de chaque ROM, ils sont synchronisés par rapport à ce délai, ils travaillent comme suit :

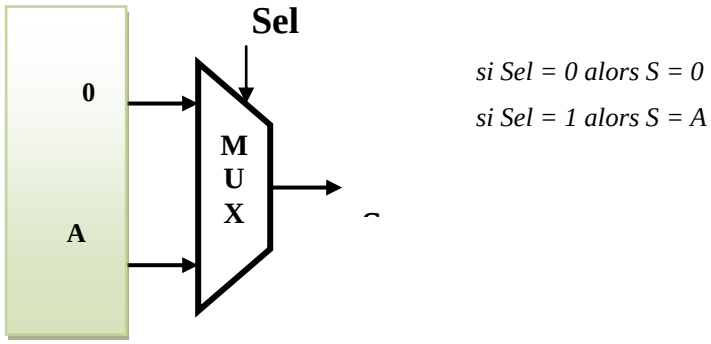


Figure C(5) : Schéma du multiplexeur de la ROM.

La sélection est une horloge qui est initialement à 0 et devient à 1 au $p^{ième}$ top.

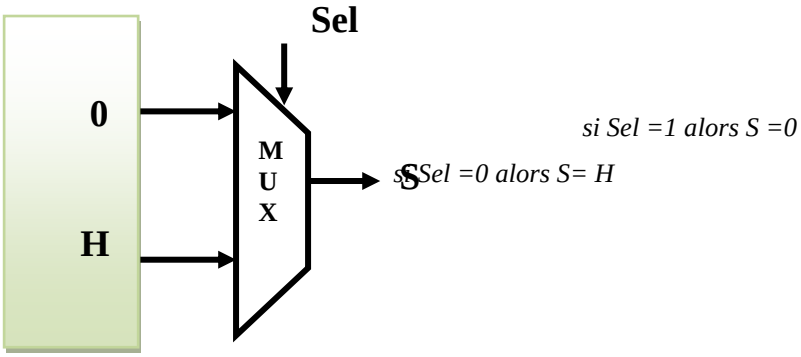


Figure C(6) : Schéma du multiplexeur de H(j).

II.2.3 : Procédure de sélection

Pour définir de façon plus rapide l'appartenance de H à l'un des intervalles [-3/2,-1/2], [-1/2,1/2] où [1/2, 3/2], les trois premiers bits des dix bits du CLA ont été utilisés afin de générer le bit de sélection.

Le problème se pose lors de la détermination du bit résultat $r_{n+1}(j)$. En effet, ce dernier est obtenu après un nombre de comparaisons de la valeur de H par rapport aux différents intervalles :

$$r_{n+1}(j) = \begin{cases} -1 & \text{si} \quad -3/2 \le H(j) < -1/2 \\ 0 & \text{si} \quad -1/2 \le H(j) < 1/2 \\ +1 & \text{si} \quad 1/2 \le H(j) < 3/2 \end{cases}$$

Le nombre moyen de tests dans ce cas est 3/2, c'est-à-dire de façon générale il est égal à (card. G)/2.Vu que H (j) est compris entre *(-3/2)* et *(3/2)*, les trois bits de H(j) qui peuvent nous renseigner sur l'appartenance de celui-ci à l'un des trois intervalles sont :

- → bit de signe h_1
- → bit de poids 0 h_0
- → bit de poids -1 h_{-1}

Tableau C(3) : Valeurs possibles de r_{n+1} à partir des trois bits de H(j).

h_1	h_0	h_{-1}	$r_{n+1}(j)$	f_1	f_2
0	0	0	0	0	0
0	0	1	+1	0	1
0	1	0	+1	0	1
0	1	1	-	-	-
1	0	0	-	-	-
1	0	1	-1	1	0
1	1	0	-1	1	0
1	1	1	0	0	0

À partir de ce tableau, nous arrivons facilement aux fonctions $f_1$et f_2 :

$f_1 = h_1\, h_0\, h_{-1} + h_1\, h_0\, h_{-1} = h_1\, (h_0 \text{ x or } h_{-1})$

$f_2 = h_1\, h_0\, h_{-1} + h_1\, h_0\, h_{-1} = h_1\, (h_0 \text{ x or } h_{-1})$

Remarque :

Vu la symétrie qui caractérise l'architecture générale, un schéma bloc similaire à celui qui est chargé de générer les bits r_{n+1} est dupliqué afin d'obtenir les bits $r_{n+k/2+1}$

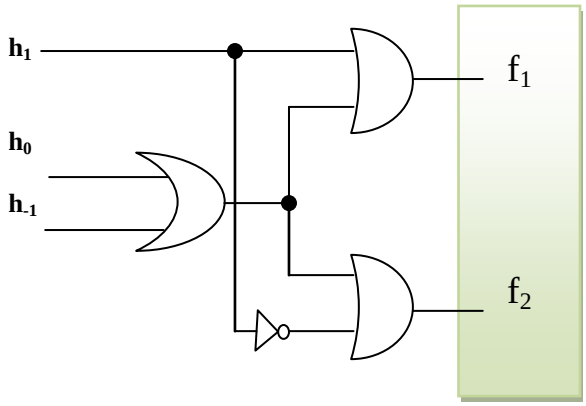


Figure C(7) : Circuit de génération des fonctions f_1 et f_2 (noté circuit f_1 f_2).

II.2.4 : Architecture générale d'un étage j

La Figure ci-après représente l'architecture globale du calcul de la Transformée de Hough par le mode de calcul en ligne.

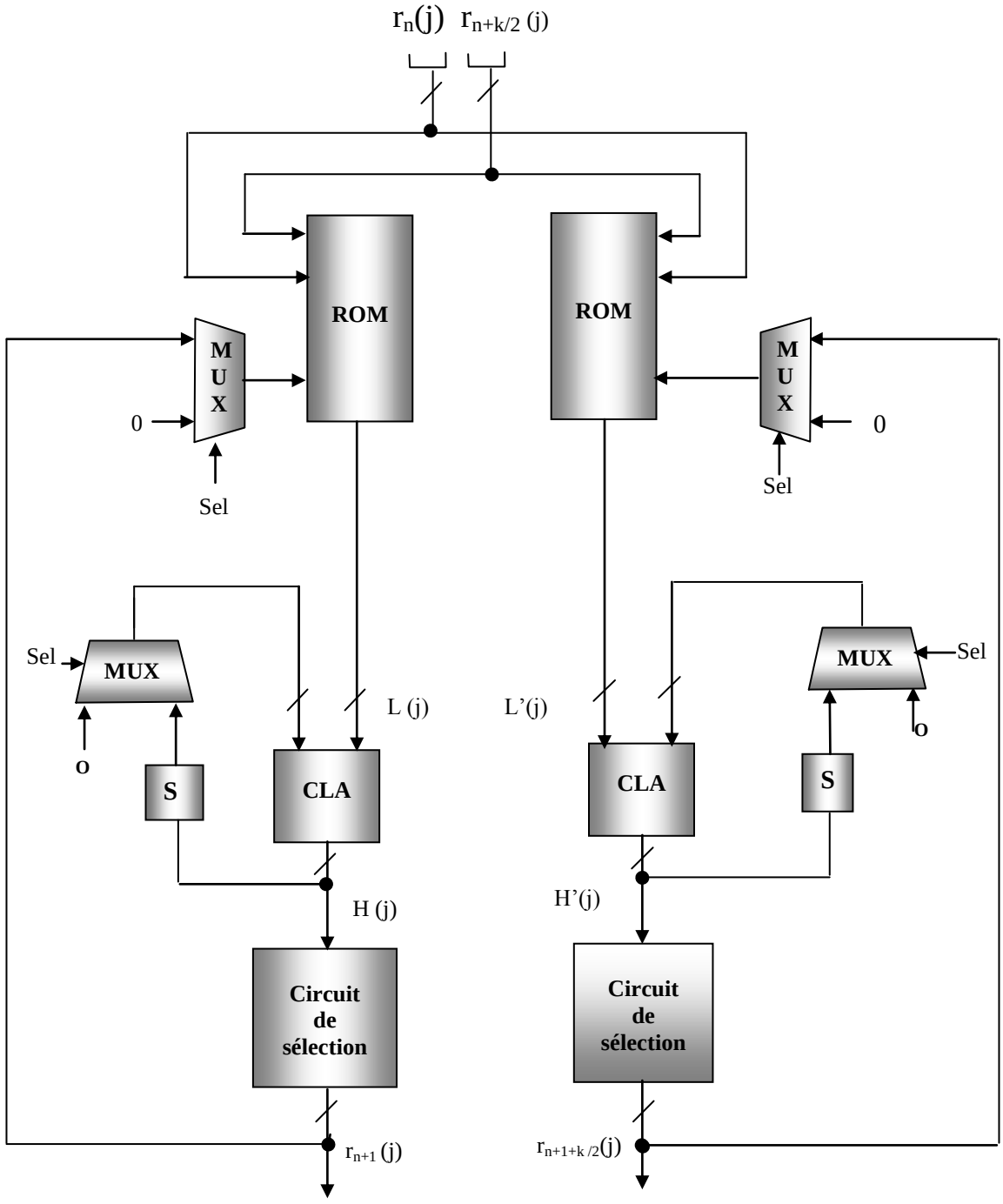


Figure C(8) : Architecture générale d'un étage j.

III Implémentation de la Transformée de Hough sur FPGA de Xilinx

III.1 : Simulation VHDL

L'arbre et les résultats de simulation de notre architecture sont donnés dans l'annexe. Les blocs sont conçus d'une manière modulaire et hiérarchique pour faciliter leur lecture et leur modification.

Pour valider l'efficacité de l'algorithme et les résultats de la simulation, un programme sous Matlab 7.1 a été développé pour confirmer le bon fonctionnement du circuit.

III.2 : Evaluation des résultats d'implémentation

Dans notre application, nous avons choisi d'implémenter notre conception sur le circuit Xilinx XC2V80. Ce dernier dispose de ressources dédiées aux opérations arithmétiques et permet une grande densité d'intégration [121], [122] et [123].

L'intérêt de l'utilisation du pipeline nous a permis de faire un recouvrement des étages sur quatre niveaux avec une latence inter étage de trois bits. Pour un nombre de ρ égal à 50, le nombre de cycles global du traitement est estimé à 180 cycles, la période minimale de génération d'un paramètre ρest de 8,5 ns.

Dans notre implémentation, chaque point contour est codé sur 10 bits, par contre ρ est codé sur 12 bits, dû au retard de l'algorithme p égal à deux.

Le calcul en mode linéaire d'un couple de paramètre (ρ_{n+1}, $\rho_{n+1+K/2}$) est de 105ns, en utilisant le pipeline, cette valeur est descendu à 26,654ns, soit une performance avoisinant les 90%.

III.3 : Traitement d'une Image

Le temps moyen de calcul d'un couple (ρ_{n+1}, $\rho_{n+1+K/2}$) est de 26,654ns. Pour effectuer un traitement en temps réel d'une image vidéo dont le temps de défilement est de 40 ms, on peut traiter 40ms/(26,654ns*50) = 30014 points, donc une image de 128*128= 16384 points est traité en 18.3 ms.

REFERENCES BIBLIOGRAPHIQUES

[1] **Ungerleider L. G. & Haxby J. V.,** «'What' and 'where' in the human brain». Current Opinion in Neurobiology, vol. 4, no. 2, pages 157–165, 1994.

[2] **Desimone R. & Duncan J.,** «Neural Mechanisms of Selective Visual Attention». Annual Reviews in Neuroscience, vol. 18, pages 193–222, 1995.

[3] **Lettvin J. Y., Maturana H. R., McCulloch W. S. & Pitts W. H.,** « What the Frog's Eye Tells the Frog's Brain». Proceedings of the IRE, vol. 47, no. 11, pages 1940 – 1951, 1959.

[4] **Chaumette F. & Hutchinson S.,** «Visual Servo Control, Part I: Basic Approaches». IEEE Robotics and Automation Magazine, vol. 13, no. 4, pages 82 – 90, 2006.

[5] **Davison A. J.,** «Real-time Simultaneous Localization and Mapping with a Single Camera». In Proceedings of the Ninth IEEE International Conference on Computer Vision, volume 2, pages 1403–1410, 2003.

[7] **Espiau B., Chaumette F., Rivers P.,** A new approach to visual servoing in robotics. IEEE Trans. on Robotics and Automation, 8(3) : 313-326, June 1992.

[6] **DeSouza G. N. & Kak A. C.,** «Vision for Mobile Robot Navigation: A Survey». IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 2, pages 237–267, 2002.

[8] **Corke P. I.** ``High-Performance Visual Closed-Loop Robot Control'' PhD thesis U of Melbourne July 1994.

[9] **Chen J., Dixon W.E., Dawson D.M. & McIntyre M.,** «Homography-Based Visual Servo Tracking Control of a Wheeled Mobile Robot». IEEE Transactions on Robotics, vol. 22, no. 2, pages 406–415, 2006.

[10] **Mariottin G.L. i, Oriolo G. & Prattichizzo D.,** «Image-Based Visual Servoing for

Nonholonomic Mobile Robots Using Epipolar Geometry». IEEE Transactions on Robotics, vol. 23, no. 1, pages 87–100, 2007.

[11] **Vidal R., Shakernia O. & Sastry S.,** «Distributed Formation Control with Omnidirectional Vision-Based Motion Segmentation and Visual Servoing». IEEE Robotics and Automation Magazine, vol. 11, no. 4, pages 14–20, 2004.

[12] **Dissanayake M., Newman P., Clark S., Durrant-Whyte H.F. & Csorba M.,**« A Solution to the Simultaneous Localization and Map Building (SLAM) Problem». IEEE Transactions on Robotics and Automation, vol. 17, no. 3, pages 229–241, 2001.

[13] **Pollefeys M., Koch R. & Van Gool L.,** «Self-Calibration and Metric Reconstruction Inspite of Varying and Unknown Intrinsic Camera Parameters». International Journal of Computer Vision, vol. 32, no. 1, pages 7–25, 1999.

[14] **Davison A. J., Reid I.D., Molton N. D. & Stasse O.,** «MonoSLAM : Real-Time Single Camera SLAM». IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 6, pages 1052–1067, 2007.

[15] **Welch G. & Bishop G.,** «An Introduction to the Kalman Filter». Rapport technique, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 1995.

[16] **Montemerlo M., Thrun S., Koller D. & Wegbreit B.,** «FastSLAM : A Factored Solution to the Simultaneous Localization and Mapping Problem». In Proceedings of the Eighteenth National Conference on Artificial intelligence, pages 593–598, 2002.

[17] **Angeli A.**, **Doncieux S., Meyer J.A. & Filliat D.,** « Real-Time Visual Loop-Closure Detection». In. Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 1842–1847, 2008.

[18] **Shi J. & Tomasi C.,** (1994). Good features to track. In Computer Vision and Pattern Recognition, (June 1994). Proceedings.

[19] **Lowe D.G.,** «Distinctive Image Features from Scale-Invariant Keypoints». International Journal of Computer Vision, Vol. 60, no. 2, pp. 91–110, 2004.

[20] **Hough P.V.C.,** «Machine analysis of bubble chamber pictures». In Int. Conf. on High Energy Accelerators and Instrumentation, pages 554–556, CERN, 1959.

[21] **Duda R.O. & Hart PE.**, «Use of the Hough transformation to detect lines and curves in pictures».Communication of the ACM, 15:11-15, January 1972.

[22] **Illingworth J. &Kittler J.,** «A survey of the Hough Transform». Computer Vision, Graphics, and Image Processing, 44(1):87–116, 1988.

[23] **Fischler N. & Bolles R.C.,** «Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography». Communication of the ACM, 24(6):381–395, June 1981.

[24] **Ebner M. & Zell A.,** «Evolving a task specific image operator». In Proceedings of the First european workshop on evolutionary image analysis, signal processing and telecommunications (evoiasp), pages 74–89, 1999.

[25] **Trujillo L. & Olague G.,** «Synthesis of interest point detectors through genetic programming». In Proceedings of the 8th annual conference on Genetic and evolutionary computation, pages 887–894. ACM Press, 2006.

[26] **Trujillo L., Olague G., Lutton E. &de Vega F.F.,** «Multiobjective design of operators that detect points of interest in images». In. Proceedings of the 10th annual conference on Genetic and evolutionary computation, GECCO'08, pp. 1299–1306. ACM Press, 2008.

[27] **Louchet J., Guyon M., Lesot M.J. & Boumaza A.,** «Dynamic flies: a new pattern recognition tool applied to stereo sequence processing». Pattern Recognition Letters, vol. 23, no. 1-3, pages 335–345, 2002.

[28] **Pauplin O., Louchet J., Lutton E. & De La Fortelle A.,** «Evolutionary Optimization for Obstacle Detection and Avoidance in Mobile Robotics». Journal of Advanced Computational Intelligence and Intelligent Informatics, vol. 9, no. 6, pages 622–629, 2005.

[29] **Olague G. &Puente C.,** «Parisian evolution with honeybees for three dimensional reconstruction». In Proceedings of the 8th annual conference on Genetic and evolutionary computation, pages 191–198. ACM Press New York, NY, USA, 2006.

[30] **Mian A. S.**, **Bennamoun M. & Owens R.,** «Three-dimensional model-based object recognition and segmentation in cluttered scenes». IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(10), pp.1584–1601, 2006.

[31] **Boutarfa A.,** «A New approach to beacons detection for a mobile robot using a neural network model». RoMoCo'04, Proceedings of the fourth international workshop on robot motion and control, June 17-20, Puszczykowo, Poland, pp. 197-202, 2004.

[32] **AloimonosY.,** «Is visual reconstruction necessary? Obstacle avoidance without passive ranging». J. Robotic Syst. 9, pp. 843– 858,1992.

[33] **Achour K. & Belaiden R.,** «A new stereo matching using multiscale algorithm». in: Proc. Int. Conf. on Recent Advances in Mechatronics, Istambul (Turkey), 1995.

[34] **Wrobel-Dautcourt B.,** «Perception de la distance par mise en correspondance de régions dans des images stéréoscopiques». Thèse de doctorat, Institut National Polytechnique de Lorraine, 1988.

[35] **Lux A. & Souvignier V.**, " PVV: un système de vision appliquant une stratégie de prédiction vérification ", 4ème Congrès de recon. Des formes et I.A., Paris, **pp** 223-234, 1984.

[36] **Binford Thomas O.,** «Inferring Surfaces from Images». Artif. Intell. 17(1-3): 205-244 (1981).

[37] **Deriche R. & Cocquerez J. P.,** «Extraction de composantes connexes basée sur une détection optimale des contours». Actes du Congrès MARI-87, Paris, pages 1-9, Tome 2, 1987.

[38] **Zucker Steven W.,** Region growing : Childhood and adolescence».Computer Graphics and Image Processing, vol. 5, no. 3, pages 382–399, septembre 1976.

[39] **Horowitz E. & Sahni S.,** «Fundamentals of Computer Algorithms», Computer Science Press, Potomac, MD, 1978.

[40] **Gambotto J. P. & Monga O.,** A parallel and hierarchical algorithm for region growing_ In IEEE Conference on Vision and Pattern Recognition. San Francisco. June 1985.

[41] **Haralick R.M. & Shapiro L.**, "Survey": Image Segmentation Techniques. Computer Vision. Graphies and Image Processing, 29:10G- 132, 1985.

[42] **Pavlidis T.,** A Vectorizer and Feature Extractor for Document Recognition *Computer Vision, Graphics, and Image Processing,* **35**. (1986), pp. 111-127. **GC: 170**

[43] **Pavlidis T.,** «Structured Pattern Recognition». Springer Verlag, Berlin, 1977.

[44] **Morris J. H. & Andrewal.**, «A Distributed Personal Computing Environment,» Communications of the ACM, 29(3), pp. 184-201, (March 1986).

[45] **Aho A.**, **Hopcroft J.**, **Ullman J.**, «Structures de données et algorithmes», IIA InterEditions, p. 94-102, 1987.

[46] **Monga O. & Wrobel B.,** Segmentation d'images : vers une méthodologie, Traitement du Signal, 1987, vol. 4, n° 3, p. 169-193.

[47] **Wrobel B.,** «Segmentation d'images», Rapport CRIN 85-R-103. Centre de recherche en informatique de Nancy, 1985.

[48] **Demazeau Y.,** «Niveaux de représentation pour la vision par ordinateur : indices d'image et indices de scène», Thèse Doctorat, INPG, Laboratoire LIFIA/IMAG, Grenoble, décembre 1986.

[49] **Wrobel B., Monga O.**, «Segmentation d'images naturelles: coopération entre un détecteur contour et un détecteur région»» Actes du 11ème colloque sur le traitement du signal et des images (GRETSI), Nice, pp 539-542, 1987.

[50] **Deriche R.,** Using Canny's Criteria to Derive a Recursively Implemented Optimal Edge Detector. Computer Vision, 1(2):167, 1987.

[51] **Zaroli F.,** Réalisation d'un système d'analyse et d'interprétation d'images tridimensionnelles. These de 3ème Cycle, Université de Nancy 1, 1983.

[52] **Monga O. & Keskes N.,** A hierarchical algorithm for the segmentation of 3-d images. In Proc. of eighth International Conference on Pattern Recognition, Paris, October 1986.

[53] **Bouthemy P. & Santilla Rivero J.**, Hierarchical likelihood Aproach for region segmentation According to motion-based criteria. In Proc. First Intern, Conference on computer vision Iccv87, London, UK, 8-11 June 1987 p.463-467.

[54] **Boutarfa A**, « Reconnaissance de formes 3D par approche neuronale associant la transformée de Hough en robotique mobile application à la productique ». Thèse de doctorat de l'université Hadj-Lakhdar Batna, 2006.

[55] **Davalo E. & Naim P.,** «Des réseaux de neurones», Eyrolles, 1993.

[56] **Dreyfus G & Al.**, «Réseaux de neurones, méthodologie et applications». Eyrolles, 2004.

[57] **Herault J. & Jutten C.,** «Réseaux neuronaux et traitement du signal». Hermès, 1994.

[58] **Jadouin JF.,** «Les réseaux de neurones : Principes et définitions». Hermès, 1994.

[59] **Jadouin JF., «** Les réseaux neuromimétiques : Modèles et Applications». Hermès, 1994.

[60] **Osorio F.,** INSS, un système hybride neuro-symbolique pour l'apprentissage automatique constructi. Thèse de Doctorat en Informatique, Lab. LEIBNIZ-IMAG, INPG-Grenoble, France, 1998.

[61] **Orsier B.**, «Etude et Applications des Systèmes Hybrides Neuro-Symboliques». Thèse de Doctorat en Informatique, Lab. LIFIA-IMAG, UJF-Grenoble, France, 1995.

[62] **Hatzilygeroudis I. & Prentzas J.**, «Neuro-symbolic approach for knowledge representation in expert system». IJHIS, International Journal of Hybrid Intelligent Systems, Vol. 1, No. 3-4, pp. 111-126, 2004.

[63] **Dierks T.**, **Brenner B. & Jagannathan S.,** Neural network-based optimal control of mobile robot formations with reduced information exchange more.IEEE Trans. Control Syst. Technol., 21(4), pp. 1407-1415, 2013.

[64] **Twickel von A., Hild M., Siedel T., Patel V., & Pasemann F.,** Neural control of a modular multi-legged walking machine: Simulation and hardware. Robotics and Autonomous Systems, vol. 60, no. 2, pp. 227 - 241, 2012.

[65] **Bishop C. M.,** « Neural Networks for Pattern Recognition ». Clarendon Press, Oxford, 1995.

[66] **Hough PVC.,** «Methods and Means for Recognizing Complex Patterns». No.3, December 1962.

[67] **Rosenfield A.,** «Picture Processing by Computer». Academic, New York, 1969.

[68] **Offen RJ.,** «VLSI Image processing». In McGraw Hill Book Company New York, St Louis, San Francisco, pp. 65-68, 1985.

[69] **Tzvi Ben., Sandler M.,** «Analogue Implementation of the Hough Transform». In. IEEE Proceeding-G, Vol. 138, No. 4, August 1991.

[70] **Shah S.,** «An adaptative model of information processing in the primative retina». Master's, Me Gill University 1993.

[71] **Kirsch J. C., Loudin J. A., & Gregory D.A.,** « Hybrid Modulation Properties of the Epson LCTV», Proc. of SPIE, vol. 1558, pp. 432-441, 1991.

[72] **Berger M.O.,** «Les contours actifs: modélisation, comportement et convergence». PhD thesis, Institut Polytechnique de Lorraine, Février 1991.

[73] **Kitter J. & Illing J.,** «A survey of the Hough Transform". In computing Vision, Graphics and Image Processing 44: (1), pp. 87-116, 1988.

[74] **Airiau R., Gerge JM., & Olive V.,** «Circuit Synthesis with VHDL3». Kluwer Academic Publishers, 1994.

[75] **Murgai R., Brayton R. & Sangiovanni-Vincentelli A.,** «Logic Synthesis for Field Programmable Gate Array". Kluwer Academic Publishers, 1995.

[76] **Lotufo RA., Dagless EL., Milford DJ., Morgan AD. & Morrissey JF.,** «Hough Transform for transporters arrays». In: Proceedings of the third international conference on image processing and its applications, IEEE proceedings, London, pp. 122–33, 1994.

[77] **Achour K., Djekoune O.,** «Incremental Hough transform: an improved algorithm for digital device implementation». Real Time Imaging, Elsevier, 2004.

[78] **Atiquzzaman M.,** «Multiresolution Hough transforms—an efficient method of detecting pattern in images". IEEE Transactions on Pattern Analysis and Machine Intelligence; 14(11):1090–5, 1992.

[79] **Marion A.,** "Acquisition et Visualisation des Images». Eyrolles, 1997.

[80] **Primentel K. & Teixeria K.,** «Virtual Reality Through the new look-ing Glass», ISBN 0070501688,2ième edition, McGraw Osborne Media, Emery ville, 1994.

[81] **Del Bimbo A., Vicario E. & Zingoni D.**, « A Spatial logic for symbolic description of image contents ». Journal of visual languages and computing. 5(3), 267-286,1994.

[82] **H. Koshimizu and M. Numada** : FIHT2 algorithm, a fast incremental - Hough transform, IEICE Trans., Vol. E 74, No.10, pp.3389-3393 (1991)

[83] **Tagzout S., Achour K. & Djekoune O.,** «Hough transform for FPGA implementation".Elsevier Journal, Signal Processing; 81(6):1295–301, 2001.

[84] **Hamada M., & Boutarfa A.**, « A new method using Hough transform (HT) in robotic vision » **J**ournal of Electrical Engineering: Volume 14 Edition: 2, **Article 14.2.2** pp. 8-18**,** 2014**.**

[85] **Markovic Ivan & Petrovic Ivan,** «Bayesian Sensor Fusion Methods for Dynamic Object Tracking—A Comparative Study». Automatika, Journal for Control, Measurement, Electronics, Computing and Communications, 55(4), pp. 386–398, 2014.

[86] **Sun R.,** «An introduction to hybrid connectionist-symbolic models». In: Sun R. & Alexander F. (Eds). Connectionist-Symbolic Integration: From Unified to Hybrid Approaches. Chapter 1, Lawrence Erlbaum Associates, 1997.

[87] **Giacometti A.,** «Modèles Hybrides de l'Expertise». Thèse de Doctorat en Informatique et Réseaux, Lab. LIFIA-IMAG, Grenoble / ENST Paris, France, 1992.

[88] **Boudjemaa F., Djedjiga B. & Diaf M.,** «Identification automatique de personnes par l'iris de l'œil utilisant la Transformée de Hough et les filtres de Gabor». Ciia05, Congrès International en Informatique Appliquée, Bordj-Bou-Arréridj, 19-21 Novembre, pp. 282-287, 2005.

[89] **Courcelle Alain,** Localisation d'un robot mobile : Application à l'aide à la mobilité des personnes handicapées moteur. Doctorat de l'Université de Metz, France, janvier 2000.

[90] **Djekoune O.,** ''Localisation et guidage du robot mobile Atrv2 dans un environnement naturel''. Thèse de Doctorat en Electronique, Option: Contrôle de processus et robotique, USTHB, Alger, 2010, N° d'ordre : 06/2010-D/EL.

[91] **Crouzil Alain,** « Perception du relief et du mouvement par analyse d'une séquence stéréoscopique d'images », Thèse de doctorat, Université Paul Sabatier de Toulouse, Septembre 1997.

[92] **Faugeras OD., Ayache N., & feverjon B.,** « Building visual Maps by Combing Noisey Stereo Measurements ».IEEE International conference on Robotics and Automation, San Francisco, Cal., April, 1986.

[93] **Horaud Radu & Oulivier Monga,** «Vision par ordinateur », 1993(first edition) [ISI 98] ISIMA 1988, 1999. Introduction a` C++ builder

[94] **Šter B.,** « Selective recurrent neural network». Neural Processing Letters, 38(1), pp.1-15, 2013.

[95] **Marr D. & Poggio T.,** «Cooperative computation of stereo disparity». Science, Vol. 194, pp. 283-287, 1976.

[96] **Borenstein J. & Koren Y.,** «High-speed obstacle avoidance for mobile robots», IEEE Transactions on Systems, Man, and Cybernetics, vol. 19,n°5, août 1988, pp. 1179-1187.

[97] **Stephen R. Marschner & Richard J. Lobb.,** « An evaluation of reconstruction filters for volume rendering». In R. Daniel Bergeron and Arie E. Kaufman , editors, Proceedings of Visualization '94, pages 100--107. IEEE, October 1994.

[98] **Ayache N., Faugeras OD., Faverjon B. &Toscani G.,** « Mise en correspondance de cartes de profondeur obtenues par stéréoscopie passive ». AFCET, 1985.

[99] **Nekovei Resa. & Sun Ying.,** «Back-propagation network audits configuration for blood vessel detection in angiograms». IEEE Transactions on neural networks. Vol. 6, No. 1, January 1995.

[100] **Freeman James A., & Skapura D.,** "Neural Networks". Algorithms, Applications and programming Techniques. CNS : Computation and Neural Systems Series, 1991.

[101] **Boutarfa A., Bouguechal N. & Abdessemed Y.,** «An approach to beacons recognition for a mobile robot using a neural network model». Revue des Sciences et Technologie B, ISSN 1111-5041, Université de Constantine, Juin 2005.

[102] **Fnaiech F., Sayadi M. & Najim M.,** «Factored and fast algorithms for training feed forward neural networks». ESST de Tunis, 1997.

[103] **Boutarfa A.,** « An approach to beacons detection for a mobile robot using a neural network ». Proceedings of the 18th International Conference Modeling and Simulation, Montreal, Quebec, Canada, May 30, pp. 118-124, 2007.

[104] **Giraudon G.,** «Chaînage efficace de contours», Rapport de recherche n°605 INRIA, Mars 1987.

[105] **Robert A.,** «Perception et modélisation de l'environnement d'un robot mobile : Une approche par stéréovision». Thèse de Doctorat au L.A.A.S, Grenoble, Novembre 1986.

[106] **Achour K. & Benkhelif M.,** «3D Reconstruction using four references points without camera calibrate for a mobile robot». International Conference on Intelligent Robots and Systems, IROS'98, Victoria, Canada, October 13-17, 1998.

[107] **Medioni G. & Nevatia R.,** «Segment-Based Stereo Matching ». Computer Vision, Graphics and Image Processing, No.31, pp 2-18, 1985.

[108] **Chaumette F., Bouthemy P. & Juvin D**. "Mise en correspondance de segments de droites dans une séquence d'image". I.N.R.I.A, Rapport de Recherche, No. 1792, Novembre 1992.

[109] **Hopfield J.,** «Neural networks and physical systems with emergent collective computational abilities». Proc .Nat .Acad. Science, Vol. 79, pp. 2554-2558, April 1982.

[110] **Young S., Scott P. & Nasrabadi NM.,** «Object recognition multilayer Hopfield Neural Network». IEEE Transations on image processing, Vol. 6, No. 3, Marsh 1997.

[111] **Baluja S.,** «Evolution of an artificial neural network based autonomous land vehicle controller». IEEE Transactions on Systems, Man and Cybernetics, Part B, Vol. 26, No. 3, pp. 450 – 463, June 1996.

[112] **Caplier A., Luthon F. & Dumantier C.,** «Real time implementations of an mrf-based motion detection algorithm, special issue on real-time motion analysis». Journal of real time imaging, Vol. 4, No. 1, pp. 41-54, February, 1998.

[113] **Song Qing, Jizhong Xiao. & Soh Yeng Chai,** «Robust back propagation training algorithm for multilayered neural tracking controller». IEEE Transactions on Robotics and Automation, Vol. 10, No.5, September 1999.

[114] **Zaatri A., Tuytelaars T., Waarsing R., Van Brussel H. & Gool L.,** «Supervised intelligent function». In: Proceedings of IEEE, Conference on Robotics and Automation, ICRA, pp. 3707–3712, 2000.

[115] **Tuytelaars T. & Van Gool L.,** «Matching widely separated views based on affinity invariant neighbourhoods». International Journal on Computer Vision, July, 2003.

[116] **Boutarfa A., Bouguechal N., Abdessemed Y. & Redarce T**. "Pattern Recognition in computer integrated manufacturing".International Journal of Engineering, Vol.57, No 1, pp. 28-35, ISSN 1335-3632, 2006.

[117] **Markovic Ivan, & Petrovic Ivan.,** «Bayesian Sensor Fusion Methods for Dynamic Object Tracking—A Comparative Study». Automatika, Journal for Control, Measurement, Electronics, Computing and Communications, 55(4), pp. 386–398, 2014.

[118] **Zuliani M.,** «Ransac for Dummies With examples using the RANSAC toolbox for & Octave and more». http://vision.ece.ucsb.edu/~zuliani esearch/RANSAC/docs/RANSAC4Dummies.pdf, july 2014.

[119] **Avizienis** "Signed-Digit Number Representation for Fast Parallel Arithmetic".IRE Trans. Electron. Comput. Vol. EC-10, pp. 389-400, September 1961.

[120] **Ercegovac MD.**, "On- line Arithmetic for Recurrence Problems". SPIE, Vol. 1556, Adv. Sig. Proc. Alg. Arch. and Imp. II, 1991.

[121] **Xilinx.** "Synthesis and simulation Design Guide". Xilinx INC, USA.1998.

[122] **Xilinx.** "Virtex^{TM}-II Platform FPGAs: Introduction and Overview". advance product specification, DS031-1, Vol. 9, September 26, 2002.

[123] **Xilinx.** "Tutorial ISE Foundation". Lilian Boussuet, September 2002.

VALORISATION

DE MES TRAVAUX DE RECHERCHE

1. A. Boutarfa, Hamada M., Bouguechal N., Emptoz H.
   "An Improved Approach to Beacons Detection for a Mobile Robot using a Neural Network Model".
   International Journal of Engineering, JEE.ro, Vol.7, No 2, pp. 28-35, 2007
   ISSN 1582-4594.

2. A. Boutarfa, M. Hamada, H. Emptoz.
   "A Novel Feature for Dimensional and Functional Inspection of Industrial Parts in Computer Integrated Manufacturing",
   International Conference on Applied Informatics, ICAI'09, November 15-17, 2009, Bordj-Bou-Arreridj (Algérie).

3. A. Boutarfa, M. Hamada, H. Emptoz
   "A method for segmenting and recognizing a vehicle licence plate from a road image".
   Proceedings of International Conference on Computer Vision Theory and Applications, VISAPP, Vol.2, pp. 413-419, 17-21 May, 2010, Angers, France,
   ISBN: 978-989674028-03

4. M. Hamada, A. Boutarfa.
   "A new method combining neural networks and Hough Transform (HT) in robotic vision". *International Journal of Computer Science Issues (IJCSI), Vol. 10, Issue 2, No 3, March 2013.*
   ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784

5. M. Hamada, A. Boutarfa.
   "A new method using Hough Transform (HT) in robotic vision".
   International Journal of Electrical Engineering (Jee.rom), Vol. 14, Edition 2, pp.1-10, 2014. ISSN: 1582-4594