Titre: Title:	Construction et reconnaissance de vues panoramiques pour la localisation dans un environnement inconnu
Auteur: Author:	Véronique Béranger
Date:	1996
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Béranger, V. (1996). Construction et reconnaissance de vues panoramiques pour la localisation dans un environnement inconnu [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/8995/

UNIVERSITÉ DE MONTRÉAL

CONSTRUCTION ET RECONNAISSANCE

DE VUES PANORAMIQUES POUR LA LOCALISATION

DANS UN ENVIRONNEMENT INCONNU

BÉRANGER Véronique

DÉPARTEMENT DE GÉNIE ÉLECTRIQUE ET INFORMATIQUE

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION

DU DIPLÔME DE MAÎTRE ÈS SCIENCES APPLIQUÉES (M.Sc.A.)

GÉNIE INFORMATIQUE

Novembre 1996

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE

Ce mémoire intitulé:

CONSTRUCTION ET RECONNAISSANCE

DE VUES PANORAMIQUES POUR LA LOCALISATION

DANS UN ENVIRONNEMENT INCONNU

présenté par: BÉRANGER Véronique

en vue de l'obtention du diplôme de: Maître ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de:

M. BRAULT Jean-Jules, Ph.D., président

M. HERVÉ Jean-Yves, Ph.D., membre et directeur de recherche

M. COHEN Paul, Ph.D., membre

À mon chum.

Remerciements

Je tiens tout particulièrement à exprimer ma reconnaissance à mon directeur de recherche Jean-Yves Hervé, qui m'a guidée et motivée tout au long de ma recherche. J'aimerais aussi remercier le directeur du Groupe de Recherche en Perception et Robotique Paul Cohen, qui m'a accueillie au sein du laboratoire. Par ailleurs, je dois des remerciements aux organismes qui ont financé ce projet : le CRSNG (OGP0171210) et la chaire CRSNC-Noranda en automatisation minière. Je voudrais encore adresser mes chaleureux remerciements à mes amis du GRPR qui m'ont fait généreusement bénéficier de leur soutien, de leurs conseils et de leurs compétences au cours de ma maîtrise. En particulier, je citerais Christian Zanardi, Frédéric Labrosse, Françoys Labonté et Philippe Debanné. Enfin, je remercie le professeur Jean-Jules Brault pour sa précieuse aide au cours de mes recherches sur la reconnaissance par réseau de neurones.

Résumé

Ce projet s'inscrit dans le cadre des recherches sur l'exploration et la modélisation de l'environnement. Notre étude s'applique à un environnement tel qu'une exploitation minière, constitué de cavités reliées par des galeries. Dans une mine, certaines zones du terrain peuvent se révéler dangereuses à cause de la fragilité de la roche ou tout simplement à cause de précédentes extractions de minerai qui ont affaibli sa structure. Dans un souci d'optimisation dans l'exploitation des ressources, il peut s'avérer rentable d'envoyer des robots mobiles intelligents effectuer le travail d'extraction du minerai (ou de remblaiement de cavités). Cette solution évite de risquer la vie des mineurs. Mais, par contre, cette solution soulève immédiatement le problème de la navigation autonome et de la localisation du robot (construction de carte) dans cet environnement qui est inconnu, dynamique et non-structuré. Plus précisément, nous cherchons comment permettre à un robot mobile de se repérer visuellement dans l'environnement.

Les difficultés sont liées en particulier aux mauvaises conditions d'illumination de la scène qui ne s'amélioreront pas à l'avenir car la plupart des projets de "mine du

futur" prévoient d'enlever complètement les sources lumineuses dans la mine. Nous supposerons donc que la seule source lumineuse provient du robot. Étant données ces conditions, nous avons sélectionné les intersections comme étant les points d'intérêt les plus faciles à détecter et les plus significatifs dans un tel environnement. Notre problème revient donc à détecter les intersections dans les séquences d'images et à reconnaître une intersection précédemment rencontrée quel que soit le chemin par lequel le robot y est arrivé.

Pour détecter les intersections, nous proposons une méthode basée sur l'analyse du sol. En effet, la forme du sol est caractéristique de la topologie de l'environnement. Supposant que le sol est localement plan, nous pouvons prédire le déplacement des points appartenant au sol à partir d'une image à l'instant t et par conséquent identifier les points appartenant réellement au sol à l'instant $t + 1$. De plus, l'analyse des occlusions permet de détecter les corridors dans la scène. Nous approximons le contour du sol par des segments et leur attribuons une étiquette "Mur Visible", "Mur Occulté", "Limite de Visibilité" ou "Occlusion". Nous proposons une méthode simple et efficace pour déterminer le nombre de corridors dans la scène. S'il y a plus d'un corridor, c'est que le robot fait face à une intersection. Nous introduisons la notion d'intersection et de super-intersection qui permettent de modéliser des intersections simples et complexes. Et nous proposons un modèle topologique. Sur ce modèle, nous construisons un modèle tridimensionnel de l'intersection sur lequel nous appliquons la texture de la scène. Ainsi, notre représentation ne dépend pas du point de vue de

l'observateur. Nous générons les parties du modèle manquantes (vues occluses dans la scène) par la méthode des fractales. Enfin, nous générons la vue panoramique sur 360° de l'intersection que le robot verrait s'il était placé au centre de l'intersection. Cette vue panoramique contient tous les éléments qui permettent d'effectuer la reconnaissance à savoir à la fois la géométrie de la scène et son apparence contenue dans la texture. Nous proposons une approche connexionniste pour effectuer l'apprentissage et la reconnaissance des intersections à partir de leurs représentations panoramiques. Nous proposons un réseau de neurones de type perceptron multi-couches qui apprend à associer des vues panoramiques (en entrée) aux identificateurs des intersections (en sortie). Grâce aux propriétés d'invariance en rotation de notre représentation panoramique, le réseau peut identifier une intersection quel que soit le chemin par lequel le robot est arrivé.

Nous avons implanté notre système de construction et de reconnaissance des vues panoramiques sur un système PowerPC. Les résultats montrent que le système est capable de modéliser correctement une intersection à partir du contour du sol segmenté et étiqueté dans l'image. Par ailleurs, notre système est capable de reconnaître une intersection quel que soit le chemin d'arrivée du robot avec 62% de réussite. Le robot est aussi capable de distinguer une intersection inconnue avec 75% de réussite.

Abstract

This project is related to research on exploration and environment modelling. Our study applies to an environment such as a mining complex that is composed of rooms linked by galleries. In a mine, some areas may prove to be dangerous, either because of rock weakness or simply because of previous extractions that have weakened the structure. In order to optimize resource exploitation, it may be profitable to send intelligent mobile robots to do the work of extraction (or cavity filling). This solution avoids risking human lives. It also immediately raises the problem of autonomous navigation and localization of the robot (map building) in this environment that is unknown, dynamic, and unstructured. More precisely, we are looking for means to allow a mobile robot to locate itself in its environment, using visual information.

Difficulties are linked in particularly to bad illumination conditions of the scene, which will not improve in the future, since most "mine of the future" projects consider removing completely light sources throughout the mine. We will therefore assume that the headlights of the robot are the only light source. Given these conditions, we have selected intersections as being the most easy-to-detect and the most significant

interest points in such an environment. Therefore, our problem becomes that of detecting intersections in image sequences and of recognizing an already met intersection whichever way the robot comes from.

To detect intersections, we propose a method based on ground analysis. Indeed, the topology of the environment is characterized by the shape of the ground. Assuming that the ground is locally flat, we can predict ground points displacements from an image at time t and consequently identify points actually belonging to the ground at time $t+1$. Furthermore, occlusion analysis allows to detect corridors in the scene. We approximate the ground contour by segments and attribute each of these segments a label: "Visible Wall," "Occluded Wall," "Visibility Limit," or "Occlusion." We propose a simple and efficient method to determine the number of corridors in the scene. If there is more than one corridor, this means that the robot has encountered an intersection. We introduce the notions of intersection and super-intersection, which allow to model simple and complex intersections as well. We also propose a topological model that is used to build the intersection's tridimensional model, on which we map the texture of the scene. Therefore, our representation does not depend on the point of view of the observer. We generate missing parts of the model (occluded views in the scene) using a fractal-based method. Finally, we generate the 360° panoramic view of the intersection that the robot would see if it were positioned at the center of the intersection. This panoramic view contains all clues that allow recognition, that is to say the geometry of the scene as well as its appearance, contained in the

texture. We propose a connectionist approach to learn and recognize intersections from panoramic representations. We propose a neural network of type multilayer perceptron that learns to associate panoramic views (as input) to intersections identifiers (as output). Owing to rotational invariance proprieties of our representation, the neural network can identify an intersection whichever way the robot came from.

We have implemented our panoramic view building and recognition system on a PowerPC system. Results show that the system can correctly model an intersection from the segmented and labeled ground contour in an image. Besides, our system can recognize an already encountered intersection, whichever way the robot comes from, with a 62% success rate. The robot can also distinguish an unknown intersection with a 75% success rate.

Table des matières

Liste des tableaux

Liste des figures

Liste des annexes

Introduction

Ce projet s'inscrit dans le cadre des problèmes de navigation autonome par un robot mobile guidé par la vision se déplaçant dans un environnement inconnu et dynamique. Notre étude s'applique en particulier à des exploitations minières, composées de corridors et de salles dans lesquels des véhicules miniers doivent se diriger. Dans l'objectif d'exploiter une mine de façon plus efficace, les sociétés minières cherchent les moyens d'accéder à certaines zones "à risque", que ce soit pour extraire le minerai ou pour remblayer des cavités fragilisant la structure de la mine. Pour éviter de risquer la vie des mineurs en les envoyant travailler dans ces zones dangereuses, une solution consiste à employer des robots mobiles autonomes, ce qui soulève le problème de la navigation autonome et de la représentation (sous forme d'une carte) que le robot doit construire de son environnement. Les conditions d'opération dans un tel environnement sont délicates pour les senseurs. En particulier, les conditions d'illumination sont très mauvaises. Il est peu probable que cela s'améliore à l'avenir car la plupart des projets de "mines du futur" proposent de retirer les sources lumineuses dans les corridors. Par conséquent, pour un robot se déplaçant dans une

mine, la seule source lumineuse éclairant la scène sera probablement fournie par les phares du véhicule lui-même. Ces conditions, ainsi que l'absence de repères faciles à détecter nous amènent à utiliser des techniques de reconstruction sophistiquées.

En général, la solution préconisée pour effectuer des tâches de navigation implique la représentation de l'environnement par une carte. À cause du manque de précision des systèmes odométriques, un des problèmes classiquement rencontrés lors de la construction de cartes est celui de relocaliser le robot. En d'autres termes, le robot doit être en mesure de déterminer sa position sur la carte, et en particulier de reconnaître les endroits qu'il a déjà rencontrés. Comme le robot ne peut pas mémoriser toutes les scènes qu'il rencontre, nous devons sélectionner des points d'intérêt. Nous avons choisi les intersections de tronçons de galeries comme étant les repères les plus significatifs ou encore les plus facilement identifiables dans un tel environnement. Finalement, notre problème revient à :

- détecter les intersections dans une séquence d'images et les représenter ;
- identifier une intersection par laquelle le robot serait déjà passé (éventuellement par un autre chemin).

Nous avons choisi de représenter les intersections par des vues panoramiques sur 360°. Cette représentation permet en effet de prendre en compte à la fois la géométrie des intersections et leur apparence, contenue dans la texture de la scène. Un modèle purement géométrique serait trop simple car il ne permettrait pas de distinguer deux intersections ayant la même géométrie. Par ailleurs, devant la complexité de l'envi-

ronnement, il n'est pas évident de sélectionner des critères fiables pour différencier deux intersections possédant la même géométrie. Une représentation panoramique permet de conserver toute la richesse de l'information contenue dans la texture de la scène sans stocker d'éléments redondants. L'intérêt de cette représentation réside en particulier dans ses propriétés d'invariance en rotation. Il s'agit en effet d'une représentation cylindrique dont le centre est défini comme étant le centre de l'intersection. Les figures 0.1, 0.2, 0.3 et 0.4 illustrent les étapes de la méthode que nous proposons pour construire une vue panoramique à partir d'une séquence d'images prises par une caméra montée sur le véhicule.

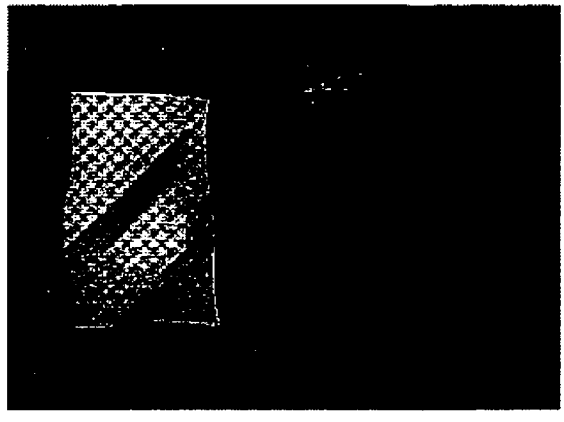
Pour reconnaître les intersections, nous avons choisi une approche connexionniste plutôt qu'une méthode basée sur la mise en correspondance de caractéristiques à cause de l'absence de caractéristiques stables dans le type d'environnement dans lequel nous travaillons. Les réseaux de neurones sont bien adaptés à ce type de problème qui met en jeu des mécanismes d'apprentissage et de reconnaissance. De plus, le temps de réponse des réseaux de neurones est rapide pour la reconnaissance. L'apprentissage peut être long mais peut être réalisé plus tard, une fois que les scènes ont été enregistrées.

La figure 0.5 décrit le système global.

Le robot doit détecter et identifier les intersections qu'il rencontre à partir d'une séquence d'images. Le système se décompose en deux modules :

- détection d'une intersection et classification ou représentation d'une intersec-

(a) (b)



(c)

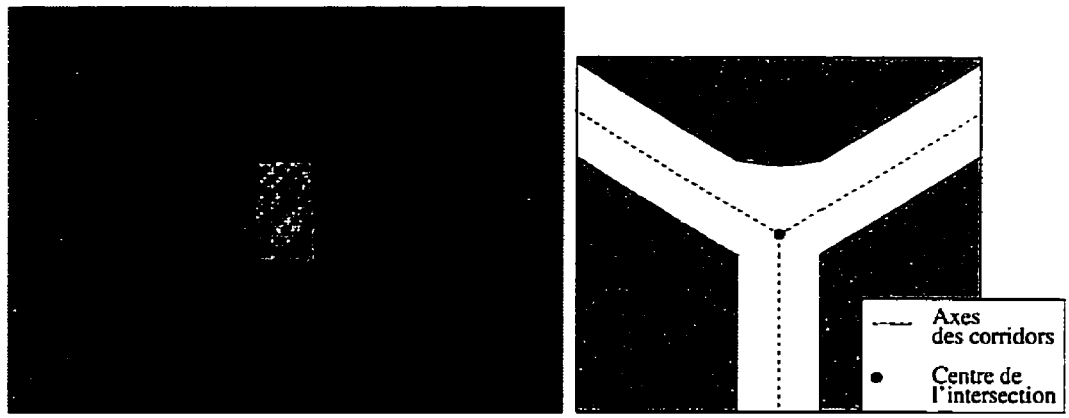
Figure 0.1 : Une partie de la séquence d'images prise par le robot

tion ;

- reconnaissance d'une intersection déjà rencontrée.

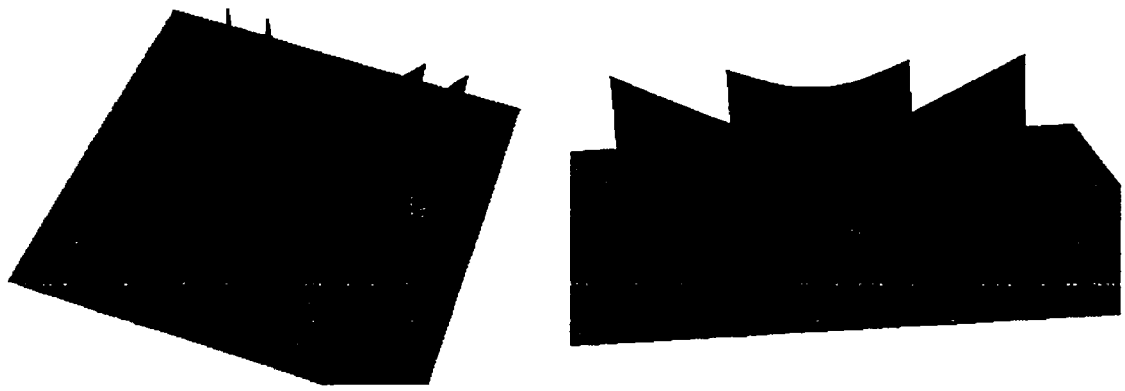
Une base de données regroupe les représentations des intersections rencontrées. Au cours de ses déplacements, le robot peut donc :

- ajouter la représentation d'une nouvelle intersection ;
- compléter les informations relatives à une intersection qu'il reconnaît.



(a) (b)

Figure 0.2 : (a) Détection du contour du sol sur la première image – (b) Modèle topologique de l'intersection



(a) (b)

Figure 0.3 : (a) Modèle géométrique tridimensionnel de l'intersection – (b) Modèle géométrique sur lequel la texture reconstruite à partir de l'image a été appliquée

Nous proposons une méthode basée sur l'analyse du sol pour détecter les intersections. En effet, la forme du sol est caractéristique de la topologie de l'environnement. De plus, l'observation du flux de mouvement facilite la détection en révélant des discontinuités de profondeur, donc des ouvertures dans les parois du couloir c'est-à-dire

Figure 0.4 : Vue panoramique générée

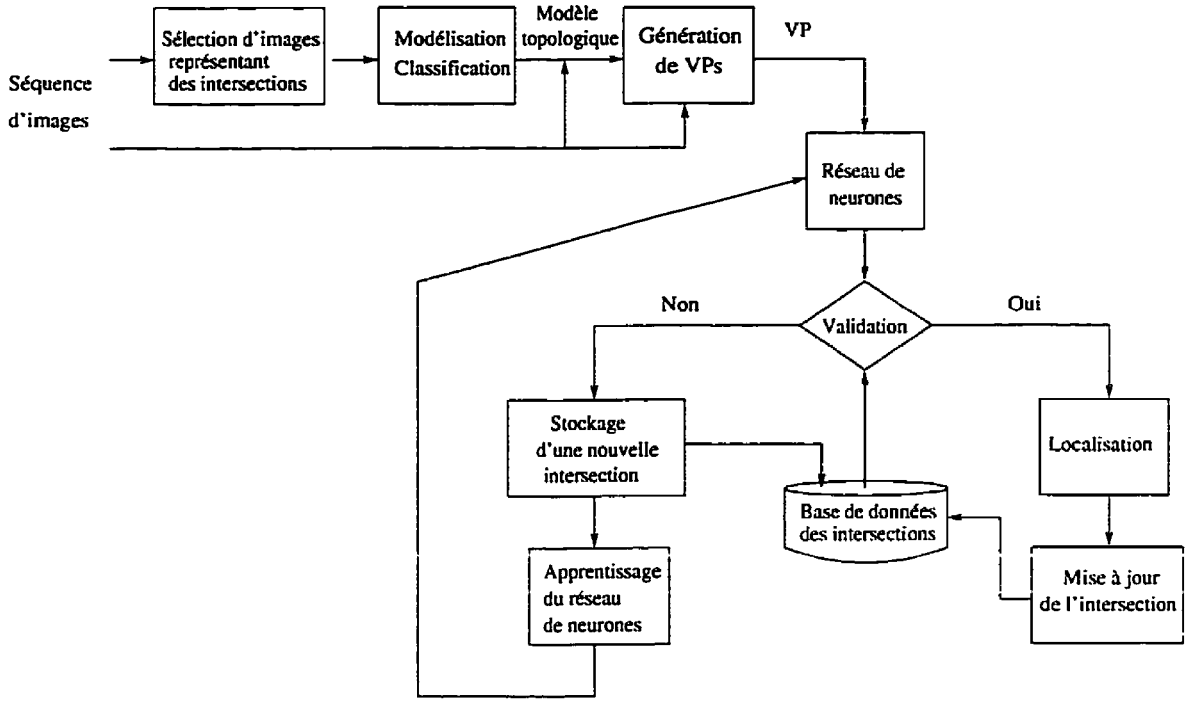


Figure 0.5 : Description du système global

des intersections. Une première analyse du contour du sol dans l'image nous permet de déterminer si la scène perçue par le robot comporte une intersection et fournit le cas échéant, le nombre de corridors composant l'intersection. Les intersections sont classées dans la base de données selon ce critère décisif. À partir du contour du sol, nous déterminons le modèle topologique (bidimensionnel) d'une intersection

sur lequel nous nous basons pour construire le modèle géométrique tridimensionnel. Nous extrayons la texture de la séquence d'images pour l'appliquer sur ce modèle et nous générons la vue panoramique qui serait perçue par le robot s'il était placé au centre de l'intersection. Cette vue panoramique est présentée à un réseau de neurones qui l'associe à son identificateur dans la base de données si l'intersection a déjà été rencontrée. Si le réseau de neurones identifie l'intersection rencontrée, le système procède à la localisation de l'intersection dans l'environnement et à la mise à jour de sa représentation dans la base de données. Si, au contraire, l'intersection rencontrée est inconnue, sa représentation est ajoutée dans la base de données et sa vue panoramique correspondante est apprise au réseau de neurones.

Dans le chapitre 1, une revue des méthodes existantes dans les domaines de la représentation de l'environnement, de la localisation de l'environnement et de la détection d'intersections est proposée. Dans le chapitre 2, nous décrivons le modèle que nous proposons pour représenter une intersection et la méthode que nous avons conçue pour détecter une intersection dans une séquence d'images. Nous montrons aussi comment construire le modèle de l'intersection. Le chapitre 3 est consacré au problème de la reconnaissance d'intersections. Nous montrons comment nous construisons une vue panoramique à partir du modèle topologique de l'intersection et de la séquence d'images. Nous présentons aussi le réseau de neurones que nous avons développé pour mémoriser les vues panoramiques et les reconnaître.

Chapitre 1

Revue bibliographique

Nous présentons ici les conclusions de notre analyse bibliographique des domaines pertinents pour ce projet : modélisation de l'environnement, relocalisation dans l'environnement et détection d'intersections. La revue bibliographique détaillée, approche par approche, est donnée dans l'annexe A.

1.1 Modélisation de l'environnement

Une des principales difficultés rencontrées en robotique mobile est liée à la complexité de l'environnement. Un robot mobile doit faire face à une grande variété d'objets tant sur le plan qualitatif que quantitatif, tels que des obstacles à éviter ou des objets à manipuler. Considérons par exemple le cas d'un véhicule minier chargé d'opérer dans des zones à risques d'une mine. Comment le rendre capable de se situer dans cet environnement qui lui est inconnu ? Il est essentiel pour cela

de doter le robot mobile des moyens de construire et d'utiliser des modèles de son environnement. Ce problème de modélisation de l'environnement est sous-jacent aux problèmes de planification de chemin, de navigation, de planification et d'exécution de tâches spécifiques, etc.

Une difficulté majeure du problème vient de la complexité de l'environnement. Il y a une grande diversité d'éléments à traiter. Comment modéliser des éléments aussi différents que des obstacles à éviter, des objets à manipuler ou encore des évènements (par exemple une porte qui se ferme) ? Une réponse à cette question est fournie dans (Labrosse et al. 1996*a*, Labrosse et al. 1996*b*). Ces travaux décrivent la modélisation d'une mine par un modèle générique d'objets. Par ailleurs, et c'est ce que nous évoquerons plus en détail, dans un environnement non-structuré, il est très difficile de définir des points d'intérêt sur des objets. En général, les objets ne sont pas polyédriques mais ont des formes courbes. Le problème consiste à construire et à mettre à jour un modèle (des objets et de l'espace), qui reste consistant tout au long de l'exploration du robot, aussi bien lorsqu'il explore de nouvelles régions que lorsqu'il retourne dans des régions déjà visitées. Un dernier aspect du problème est la localisation du robot. La principale difficulté réside dans l'incertitude sur les mesures fournies par les capteurs qui peut engendrer une accumulation d'erreurs importante sur de grandes distances, tant sur la position des objets que sur la position du robot lui-même.

Nous avons dégagé trois approches pour la modélisation de l'environnement. L'ap-

proche "métrique" consiste à représenter l'environnement sous forme de carte de type géographique. L'approche "probabiliste" prend en compte les incertitudes sur les mesures. L'espace est décomposé en régions probablement vides ou probablement occupées. Enfin, une troisième approche, qualifiée de "qualitative", se base sur l'idée que les humains reconnaissent des repères caractéristiques. Le modèle qualitatif est un modèle topologique. L'environnement est représenté par des graphes de places distinctives reliées par des chemins décrits comme des stratégies de contrôle. Ces approches sont décrites plus en détail dans l'annexe A.1.

Ces modèles abordent les difficultés liées à la complexité de l'environnement, aux erreurs de mesure et à la consistance du modèle de façon différente.

Le modèle métrique prend un modèle du monde simplifié : les objets ont une forme polyédrique. Pour ce qui est des erreurs, le robot s'appuie sur des repères connus qui lui permettent d'ajuster sa position. En revanche le modèle ne prend pas en compte des modifications éventuelles de l'environnement, par exemple le cas d'une porte qui peut être ouverte ou fermée. Ce problème est traité par l'utilisation d'un système à ultrasons qui permet d'éviter des obstacles proches.

Le modèle probabiliste suppose que l'environnement est non structuré. Par contre, il n'y a aucun moyen d'ajuster la position du robot et les erreurs s'accumulent. Enfin, un point positif : le modèle est continuellement mis à jour en fonction de la situation présente.

Le modèle qualitatif travaille aussi dans un environnement non structuré. Mais

il s'agit d'un environnement simple, tel que Kuipers et Byun (1987) le montrent sur les exemples. En effet, les caractéristiques des places distinctives et des stratégies de contrôle sont peu élaborées. En effet, un surplus de caractéristiques complexifie le modèle et rend la méthode inapplicable. Par contre, s'il n'y en a pas assez, il est impossible de se repérer dans un environnement complexe. D'autre part, la position du robot n'étant pas définie dans un référentiel, elle est peu précise mais la marge d'erreur est fixe. Enfin, les modifications de l'environnement ne sont pas prises en compte.

Ces approches ont donc chacune leurs forces et leurs faiblesses. Dans chaque modélisation, les auteurs ont été obligés de faire des hypothèses simplificatrices qui diminuent leur capacité de généralisation.

1.2 Relocalisation dans l'environnement

Le problème de la localisation d'un robot mobile s'inscrit dans le cadre de la navigation autonome. En effet, pour naviguer efficacement, un robot autonome doit être capable de déterminer rapidement et précisément sa position courante dans l'environnement. Deux lignes d'approche peuvent être dégagées selon que l'environnement est connu ou inconnu. En général, dans un environnement connu, la méthode consiste à chercher à mettre en correspondance des caractéristiques ou "repères" extraits de l'image avec des caractéristiques extraites d'un modèle de l'environnement. Une difficulté importante réside dans l'incertitude sur le modèle et sur les mesures fournies

par les capteurs. Dans le cas d'un environnement inconnu, le problème se ramène à l'apprentissage et à la reconnaissance de scènes de l'environnement, ce qui rejoint le problème de la modélisation de l'environnement. Deux approches ont été abordées : l'une basée sur la mise en correspondance de vues panoramiques et l'autre basée sur les réseaux de neurones. Ces approches sont présentées dans l'annexe A.2.

Les approches basées sur une carte de l'environnement s'appliquent dans le cas d'environnements connus alors que les méthodes n'utilisant pas de carte s'appliquent dans le cas d'environnements inconnus. L'avantage des approches avec carte est que la scène peut être prédite à partir d'un modèle de l'environnement, ce qui facilite le problème de la mise en correspondance de caractéristiques. Certaines approches supportent de légères modifications de l'environnement. Ainsi dans (Kosaka et Kak 1992), tout objet non modélisé sera considéré comme un obstacle. La difficulté dans cette approche consiste à relever les repères pertinents dans l'image et à les mettre en correspondance avec les caractéristiques du modèle. Or dans ce processus, il se peut des repères soient confondus les uns avec les autres ou que des repères imprévisibles apparaissent (notamment dans des environnements dynamiques) ou encore tout simplement que selon le point de vue, des repères ne soient pas visibles. Généralement, les repères proposés sont basés sur les arêtes qui sont difficiles à détecter avec précision en vision. Les erreurs sur les mesures induisent des incertitudes sur la localisation du robot. Et la mise en correspondance des repères souffre de l'accumulation d'erreurs sur la localisation du robot. Par ailleurs, dans ces approches, l'environnement est

assez simple. En effet, il faut pouvoir déterminer les repères dans le modèle.

Dans les approches sans carte, l'avantage est que le robot construit son propre modèle de l'environnement, directement à partir de ce qu'il perçoit. L'idée est que le robot se repère par rapport à des caractéristiques de l'environnement, donc sur un plan plus qualitatif que quantitatif, ce qui permet d'éviter les accumulations d'erreur sur la localisation du robot. En revanche, la faiblesse de cette approche est qu'elle suppose que l'environnement est statique. En effet, dans ces méthodes, le robot apprend au fur et à mesure de son exploration. Si l'environnement est modifié, il ne retrouvera plus ses repères.

1.3 Détection d'intersections

Nous avons relevé deux approches pour la détection d'intersections dans des séquences d'images. Dans la première approche, les points de la route sont détectés à partir de leur couleur et une méthode est proposée pour modéliser les intersections dans l'image. La deuxième approche utilise un réseau de neurones pour apprendre des intersections. Ces approches sont décrites dans l'annexe A.4.

Le modèle proposé par Crisman et Thorpe (1993) pour les intersections présente l'avantage d'être basé directement sur l'image. Dans cette méthode, il n'y a pas besoin de repasser dans l'espace tridimensionnel. En revanche, le modèle montre des faiblesses quant à ses capacités de généralisation. D'une part, il ne permet pas de modéliser certains types d'intersections qui pourtant risquent classiquement d'être

rencontrées (par exemple, le cas d'une intersection en T). Et d'autre part, ce modèle n'est pas adapté à des intersections complexes c'est-à-dire à des intersections dont le nombre de branches est supérieur à trois. En effet, ce modèle suppose que les axes des routes se coupent en un point unique, ce qui n'est pas nécessairement vrai dès qu'une intersection comporte plus de deux branches. Enfin, la méthode proposée risque d'échouer si le nombre de branches composant l'intersection augmente. En effet, comme le nombre de branches n'est pas connu, les auteurs utilisent une méthode incrémentale dont la fiabilité risque de diminuer au fur et à mesure que l'intersection se complexifie. L'approche de Jochem et al. (1995) suppose que le monde est connu et modélisé, contrairement à la méthode précédente. Cette approche pose donc plusieurs problèmes : les problèmes de la modélisation de l'environnement et de la localisation du robot dans l'environnement. Le principal avantage de cette méthode réside dans le mécanisme de reconnaissance de l'intersection. En effet, les réseaux de neurones possèdent cette capacité de généralisation qui leur permettent de reconnaître des formes même légèrement déformées.

1.4 Positionnement de notre projet

Notre objectif est d'élaborer un système qui permette à un robot mobile de se localiser dans un environnement inconnu, non-structuré et dynamique. Ce problème nous a d'une part conduit à certains choix et d'autre part nous a permis d'apporter des solutions nouvelles aux problèmes plus généraux de modélisation de l'environnement

et de reconnaissance de scènes.

Dans le cadre de notre problème, nous cherchons à localiser le robot de manière globale dans une carte de l'environnement. Autrement dit, nous ne recherchons pas des données métriques qui permettraient au robot de se repérer localement et avec précision mais nous cherchons plutôt à identifier des repères qui permettront au robot de se situer globalement dans l'environnement. Par conséquent, nous représentons l'environnement qualitativement, sous forme d'un graphe dont les nœuds sont les repères que le robot peut identifier, et les arcs, les chemins reliant les repères.

Nous avons vu que le problème de la localisation dans un environnement inconnu est essentiellement de déterminer les repères de façon fiable. En effet, il s'agit de choisir les repères qui seront visibles quel que soit le point de vue du robot, aisément détectables et identifiables, le tout dans un environnement dynamique. Les approches proposées ne permettent pas de résoudre le problème tel que nous venons de le poser. En effet, l'approche par mise en correspondance de vues panoramiques est fortement dépendante de la trajectoire suivie par le robot. Or dans le contexte de notre problème, il n'est pas réaliste d'imposer au robot de suivre toujours la même trajectoire. Dans l'approche neuronale, la méthode suppose que l'environnement est figé. Cela pose problème naturellement dans le cas d'évènements imprévus (par exemple, le passage d'un véhicule) mais surtout, cela signifie que l'environnement ne peut pas être modifié dans sa structure. Autrement dit, dans le cas d'une mine, cette approche ne pourrait pas prendre en compte l'ouverture de nouveaux corridors ou de

nouvelles salles, ou encore le remblaiement de cavités. C'est pourquoi nous proposons une solution qui est indépendante de la trajectoire du robot et qui permet de prendre en compte des modifications de l'environnement. Nous avons choisi les intersections comme moyen de repère le plus significatif et le plus stable dans l'environnement que nous considérons.

Nous avons donc cherché un modèle d'intersection plus général que le modèle proposé par Crisman et Thorpe (1993) et notamment une méthode pour déterminer le nombre de branches composant une intersection directement à partir de l'image. Nous avons aussi cherché une méthode de reconnaissance des intersections, qui prenne en compte les caractéristiques de l'environnement dans lequel nous travaillons, c'est-à-dire non-structuré et dynamique, et qui soit aussi indépendante de la trajectoire du robot. Nous avons cherché des techniques nouvelles pour représenter l'environnement de façon panoramique. Et nous avons retenu une approche connexionniste pour le processus de reconnaissance des intersections à cause des propriétés de généralisation des réseaux de neurones.

Chapitre 2

Détection et Modélisation d'une Intersection

2.1 Introduction

Nous proposons dans ce chapitre un modèle pour représenter des intersections et les moyens permettant de les détecter dans un environnement non-structuré. Nous montrons en particulier comment répondre à la question primordiale : le robot est-il en face d'une intersection ? Mais l'analyse permet aussi de déterminer de quel type est l'intersection rencontrée. Plus précisément, nous montrons comment le système fournit les éléments qui permettent l'élaboration d'un modèle topologique de l'intersection.

Une intersection est communément définie comme le lieu de rencontre de deux

courbes, de deux surfaces ou de deux volumes. Nous nous plaçons dans le cadre de la modélisation de l'environnement et tout particulièrement dans la perspective de se repérer dans l'environnement, pour préciser la définition d'intersection. Dans une première approche, nous dirons qu'une intersection est le lieu de rencontre de deux corridors ou plus. Nous en arrivons ensuite au modèle de l'intersection. Nous avons choisi de représenter la géométrie d'une intersection sous la forme d'une vue topologique. Il s'agit tout d'abord de modéliser un corridor. En ce qui concerne le modèle de l'intersection proprement dit, la principale difficulté consiste à s'assurer de son unicité dans sa détermination. Il faut choisir un repère local pour l'intersection avec un point de référence indépendant du point de vue du robot. Et surtout, il faut construire un modèle qui ne dépende pas du corridor par lequel le robot arrive. En effet, il faut tenir compte du fait que le robot peut n'avoir qu'une vue partielle de l'intersection et qu'il peut par conséquent omettre des corridors. Nous introduisons la notion de super-intersection pour résoudre ce problème.

La façon dont nous détectons les intersections dans les séquences d'images est intimement liée au modèle que nous voulons en construire. Pour détecter une intersection, nous exploitons les propriétés géométriques de l'environnement, et en particulier du sol, qui sont directement liées à la topologie de l'environnement. La forme du sol suffit en effet pour déterminer s'il y a ou non une intersection et pour en construire, le cas échéant, une vue topologique. Il s'agit donc en premier lieu de détecter le sol dans l'image. Plus précisément, cela revient à chercher dans l'image les pixels qui

appartiennent au sol. Nous avons sélectionné une approche parmi celles proposées dans la littérature pour traiter ce problème. Mais il ne suffit pas de détecter les pixels appartenant au sol. Encore faut-il pouvoir en déduire la forme du sol dans l'image et plus encore la forme du sol dans l'espace tridimensionnel. Nous avons donc développé des méthodes pour déterminer le contour du sol dans l'image et l'approximer par des segments de droite ou des arcs de conique. À partir des pixels de sol dans l'image et de la largeur du corridor connu, nous déterminons un modèle du contour du sol dans l'image que le système devrait détecter s'il n'y avait pas d'intersection. La comparaison de ce modèle avec le contour du sol mesuré dans l'image permet de détecter la rencontre d'un ou plusieurs corridors. Finalement, ayant estimé le contour du sol par des segments de droite ou des arcs de conique et les ayant étiquetés en fonction de leurs propriétés, nous proposons une méthode qui permet de calculer le nombre de corridors dans la situation rencontrée. Nous en déduisons ainsi facilement s'il y a ou non intersection. Enfin, concrètement, nous exposons comment construire le modèle topologique de l'intersection à partir de l'approximation du sol dans l'image.

Ainsi, le système est capable de détecter la présence ou non d'intersections dans l'environnement et de déterminer complètement leur modèle.

2.2 Définition d'une intersection

Nous avons tout d'abord besoin de préciser ce qu'est une intersection. Il faut considérer la notion d'intersection dans la perspective de repère dans l'environnement ou

encore de "place distinctive" au sens de Kuipers et Byun (1987). Nous cherchons à identifier et à modéliser des lieux qui permettront à un robot de se diriger dans l'environnement. Notre système de détection et de modélisation d'intersection s'achemine vers l'élaboration d'une représentation de l'environnement sous la forme d'un graphe non-orienté dont les nœuds seraient des intersections et les arcs reliant les nœuds, des corridors. Ainsi le robot pourrait à la fois planifier son chemin à partir du graphe de l'environnement et garder la possibilité d'explorer des zones inconnues c'est-à-dire de rajouter des nœuds et des arcs au graphe. C'est pourquoi nous décomposons l'environnement en un ensemble de corridors connectés, définis comme des cylindres généralisés de section rectangulaire et d'axe droit ou courbe (voir la figure 2.1). En fait, dans l'environnement considéré, les parois des corridors ne sont pas planes. La section réelle des corridors varie le long de l'axe mais ces variations sont de faible amplitude et peuvent être modélisées sous forme stochastique, par exemple par un modèle fractal. Pour notre modèle, nous prendrons la section médiane des corridors.

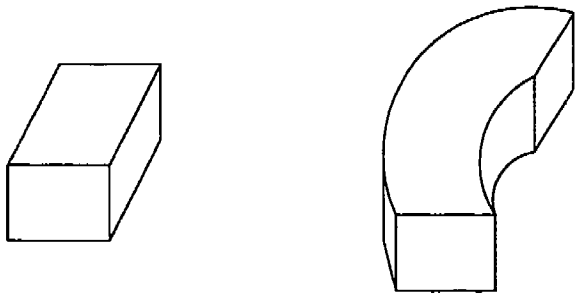


Figure 2.1 : Exemples de corridors

Nous pouvons maintenant préciser la définition d'une intersection : *une intersec-*

tion est le lieu de rencontre de deux corridors ou plus. Par conséquent, un corridor droit ou un corridor courbe ne sont pas des intersections. En revanche, le lieu de rencontre d'un corridor droit avec un corridor courbe en est une. Les figures 2.2, 2.3 et 2.4 représentent des types de situations classiques de façon idéalisée en projection perspective et en vue de dessus.

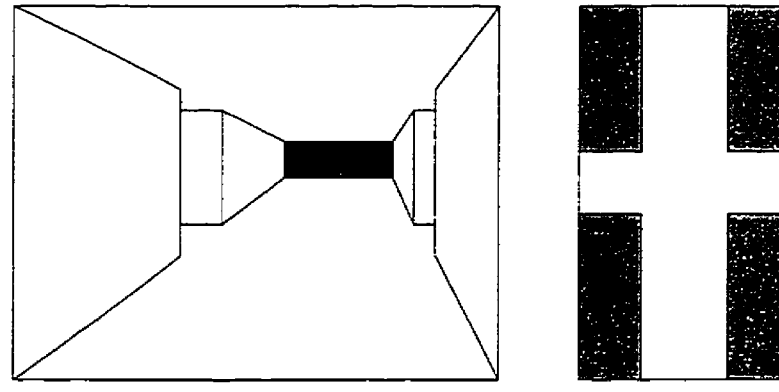


Figure 2.2 : Intersection en croix : (a) projection perspective – (b) vue topologique

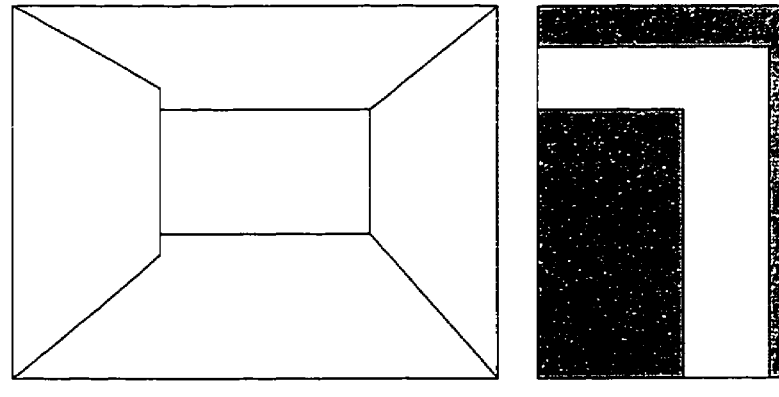


Figure 2.3 : Tournant à gauche : (a) projection perspective – (b) vue topologique

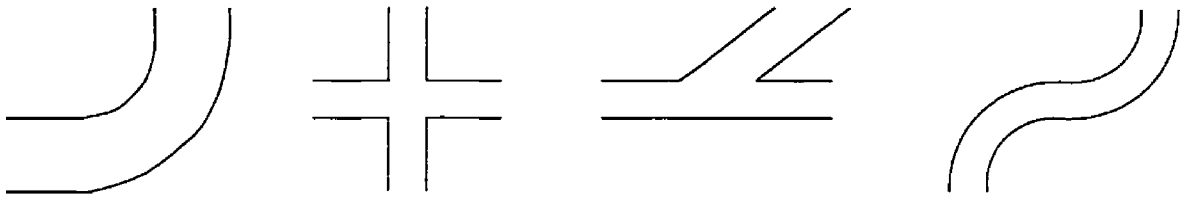


Figure 2.4 : Intersections idéalisées en vue de dessus

Il faut bien voir que la définition d'une intersection que nous proposons est intimement liée au problème de la détection des intersections dans des séquences d'images. Ceci découle de notre objectif général qui consiste en la reconnaissance de points d'intérêt. Un lieu ne saurait être "intéressant" ou, autrement dit, ne pourrait servir de point de repère s'il ne se distinguait pas par des caractéristiques qu'il nous est possible de détecter visuellement. Le type d'environnement considéré ici nous a amenés à opter pour des critères géométriques globaux dans notre définition d'une intersection. D'autres critères tels que le relief, la configuration de la texture ou la couleur locale nous semblent en effet moins robustes car ils dépendent davantage du point de vue de l'observateur et de la source lumineuse utilisée. Nous présentons ici le module que nous avons développé pour la détection et la modélisation d'une intersection telle que nous l'avons définie c'est-à-dire d'un point de vue géométrique et en tant que point d'intérêt dans l'environnement. Dans le cas éventuel où une autre définition de points d'intérêt et un module permettant de les détecter seraient proposés, le module de reconnaissance que nous proposons au chapitre 3 resterait applicable.

La figure 2.5 montre un plan de mine vue de dessus idéalisée et la figure 2.6 sa

représentation sous forme d'un graphe que l'on pourrait réaliser avec notre définition d'une intersection.

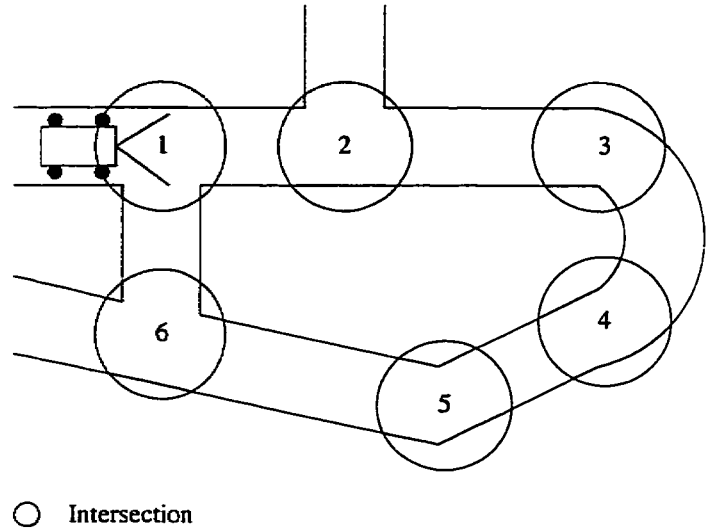


Figure 2.5 : Plan topologique d'une mine

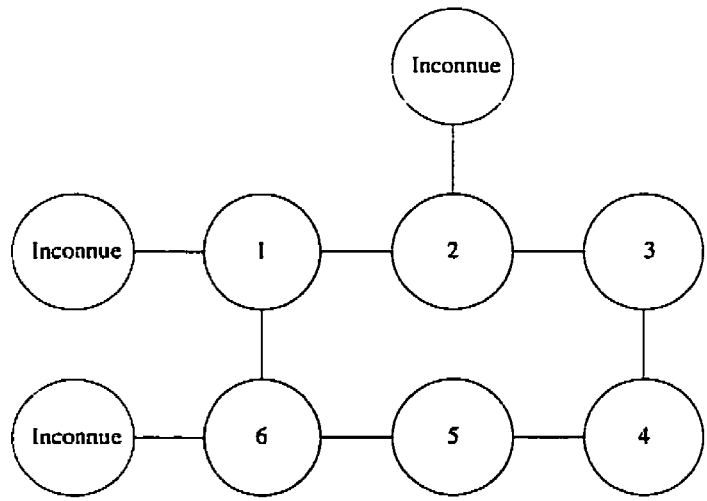


Figure 2.6 : Représentation de la mine de la figure 2.5 sous forme de graphe

2.3 Modèle topologique d'une intersection

2.3.1 Modèle topologique d'un corridor

Une intersection est composée de corridors droits ou courbes que nous modélisons par une représentation axiale ou encore plus précisément par des rubans. *Un ruban* est défini par une courbe appelée "axe" et par une figure géométrique telle un disque ou un segment de droite appelée "générateur" qui glisse le long de l'axe et dont la taille peut varier. Les extrémités du générateur décrivent les bords du ruban. Plus précisément, le générateur contient un point de référence unique, P, qui est par exemple le centre du disque ou le milieu du segment de droite. Une copie du générateur est placée à chaque point M de l'axe de telle façon que le point de référence du générateur P coïncide avec M. L'union de toutes ces copies est la forme générée. Différentes représentations axiales de la forme ont été proposées, notamment par Blum, Brooks et Brady et comparées par Rosenfeld (1986).

Il ne faudrait pas qu'une représentation puisse engendrer des formes qui ne sont pas des rubans. Pour cela, Rosenfeld définit les critères suivants :

- les générateurs ne devraient pas pouvoir s'intersecter eux-mêmes (voir la figure 2.7) ;
- les générateurs ne devraient pas être contenus les uns dans les autres.

La symétrie des générateurs tend à rendre un ruban symétrique localement mais cela n'implique aucun type de symétrie globale.

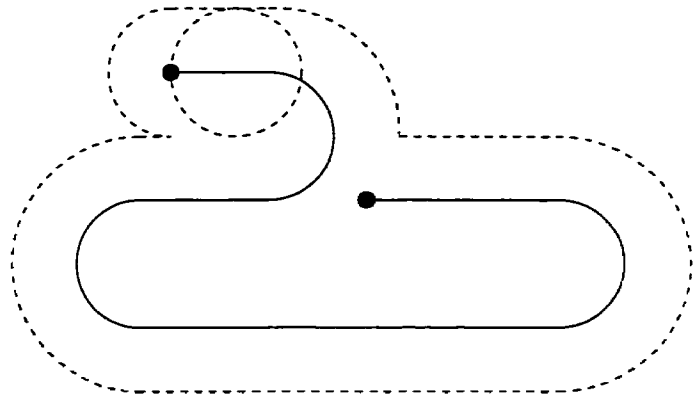


Figure 2.7 : Exemple de ruban dont les générateurs s'intersectent

Rosenfeld passe en revue les représentations de Blum, Brooks et Brady sous deux aspects. Dans le premier cas, l'axe et les générateurs sont donnés et l'objectif consiste à générer la forme du ruban (problème de la génération). Le deuxième aspect est le problème inverse qui consiste à déterminer l'axe et les générateurs d'un ruban donné (problème de la reconstruction). De notre point de vue, les deux problèmes nous intéressent. Nous voulons effectivement pouvoir modéliser les corridors à partir de leur forme dans l'image (problème de la reconstruction) mais aussi reconstituer la forme d'un corridor à partir de son modèle (génération).

Dans le ruban de Blum, le générateur est un disque dont le centre est sur l'axe (voir la figure 2.8). Rosenfeld démontre qu'avec cette représentation, l'axe et les générateurs d'un ruban peuvent être déterminés de façon unique. En revanche, cette représentation est limitée en ce qu'elle n'accepte pas des points de courbure hautement convexes. De plus, elle ne permet pas de représenter des tournants aigus et il est très difficile de définir une règle qui permette de remplir le critère de non-intersectabilité

des générateurs. Rosenfeld définit la classe des L-rubans à laquelle appartiennent les rubans de Brooks et de Brady, comme des représentations axiales dont les générateurs sont des segments de droite. La longueur et l'orientation du segment par rapport à l'axe peuvent varier. Ces rubans ne présentent pas les limitations précédentes mais peuvent avoir des formes qui ne sont pas considérées comme des rubans. De plus, la même forme peut être générée de différentes manières ce qui fait que la reconstruction n'est pas unique (voir la figure 2.9).

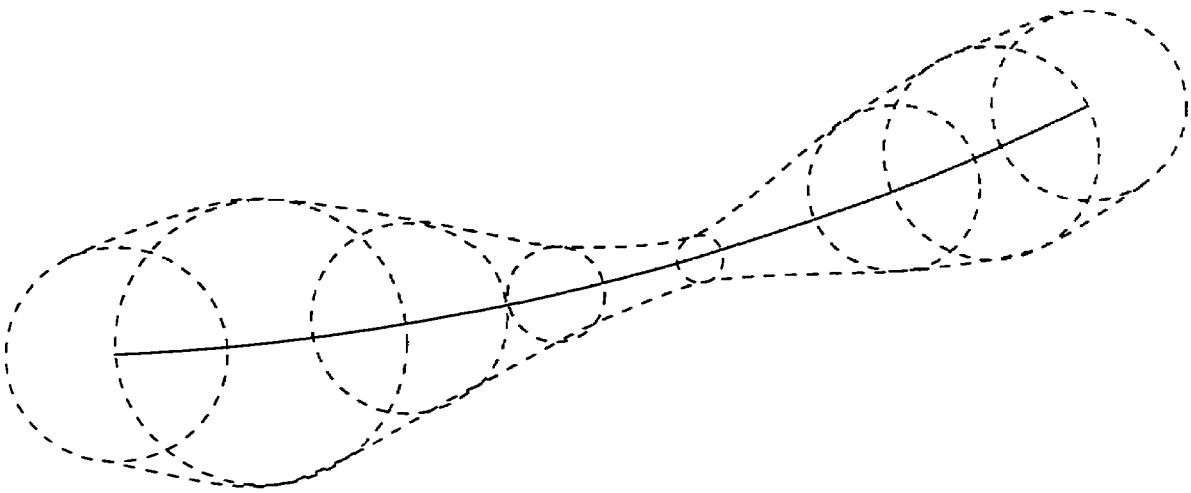


Figure 2.8 : Ruban de Blum

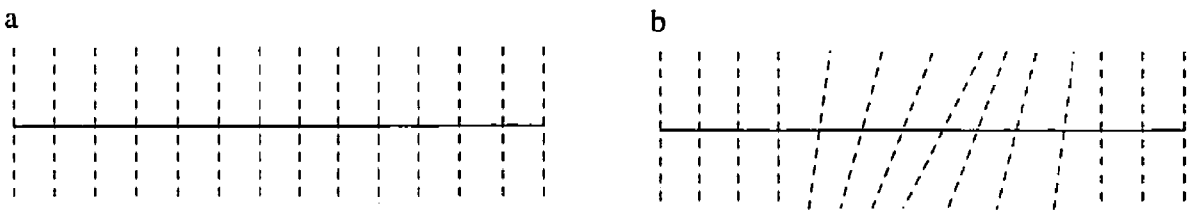


Figure 2.9 : Un L-ruban généré de deux manières

Brooks introduit une restriction à la définition de ce type de ruban qui permet de réduire les ambiguïtés (voir la figure 2.10).

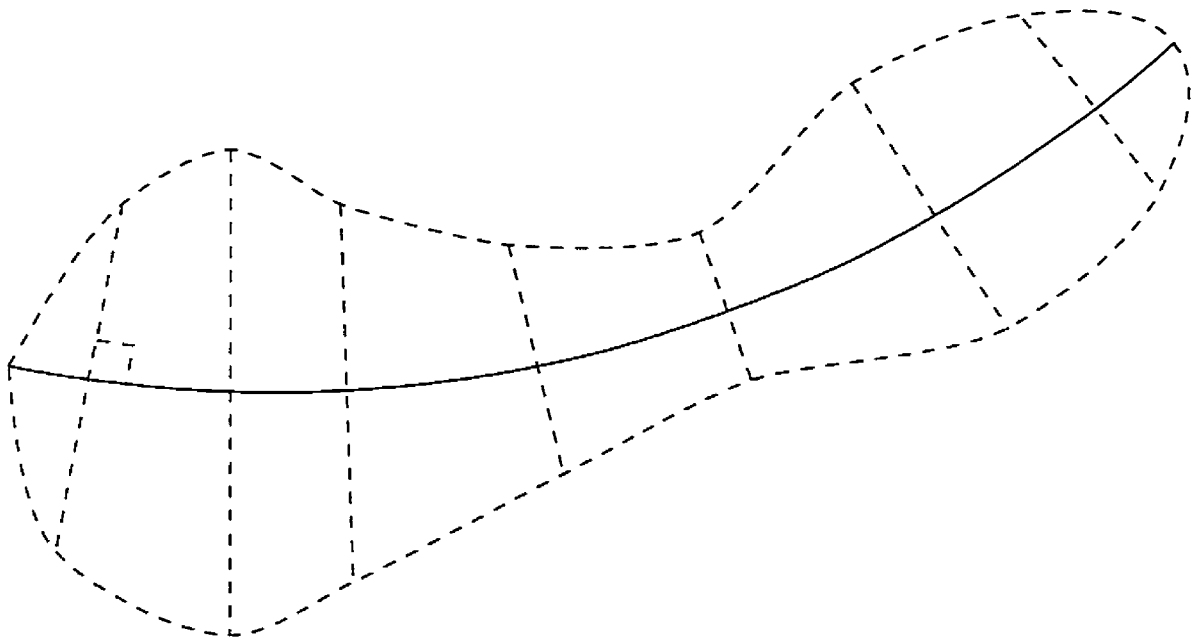


Figure 2.10 : Ruban de Brooks

Dans un ruban de Brooks, l'angle du générateur avec l'axe est fixe. Cette contrainte limite la capacité du ruban à posséder des tournants aigus mais permet d'observer des propriétés intéressantes. En particulier, si les bords du ruban de Brooks R sont droits et parallèles, alors l'axe et les générateurs de R sont déterminés de façon unique ; l'axe est la droite parallèle aux bords et à mi-chemin entre eux. Brady propose une autre restriction à la définition d'un L-ruban : le générateur fait toujours des angles égaux avec les bords du ruban (voir la figure 2.11).

Un ruban de Brady possède la propriété de l'unicité de la reconstruction dans le cas où un bord est droit mais le problème de la génération est très délicat pour ce

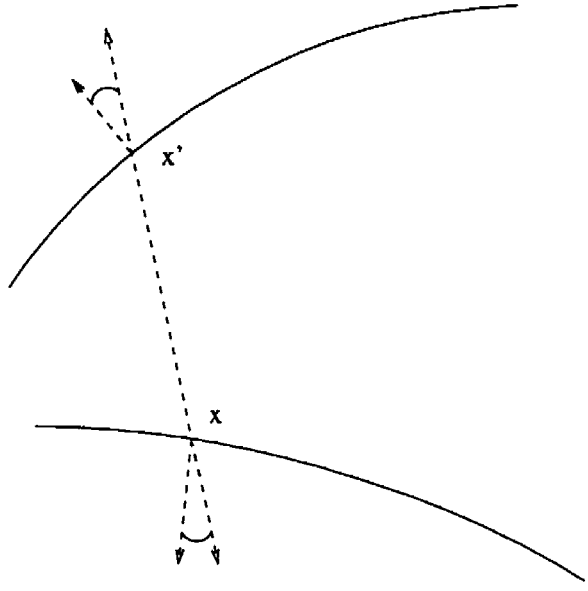


Figure 2.11 : Ruban de Brady

type de ruban.

Finalement, ces trois modèles aboutissent à des définitions de la notion de ruban ayant des possibilités plus ou moins intéressantes en matière de flexibilité de la forme, de génération et de reconstruction. Dans le cas très particulier où les axes des rubans sont droits, Rosenfeld démontre qu'un ruban de Blum est un ruban de Brooks qui est lui-même un ruban de Brady. Ces modèles sont donc très semblables. Nous choisissons le modèle de Brooks comme étant le plus approprié à notre problème (voir la figure 2.12). En effet, nous avons une relation directe entre le générateur (segment de droite) et la largeur du corridor ce qui simplifie le problème de modélisation. Nous ajoutons une contrainte supplémentaire qui est que la longueur du générateur (largeur du corridor) est constante. De plus, le ruban de Brooks a des propriétés particulières

de génération et de reconstruction. La génération n'est en effet pas évidente dans les modèles de Blum et de Brady. Par ailleurs, la reconstruction d'un ruban de Brooks dans les cas particuliers d'axes droits ou courbes est directe. En effet, Rosenfeld démontre que : si les bords du ruban de Brooks R sont droits et parallèles, alors l'axe et les générateurs de R sont déterminés de façon unique ; l'axe est la droite parallèle aux bords et à mi-chemin entre eux. De la même manière, on peut montrer que : si les bords du ruban de Brooks R sont des arcs de cercle de même centre C et de rayons de courbure constants R_1 et R_2, alors l'axe et les générateurs de R sont déterminés de façon unique. L'axe de R est l'arc de cercle de centre C et de rayon $(R_1 + R_2)/2$. Le générateur est le segment de droite perpendiculaire à l'axe et de longueur $\mid R_2 - R_1 \mid$.

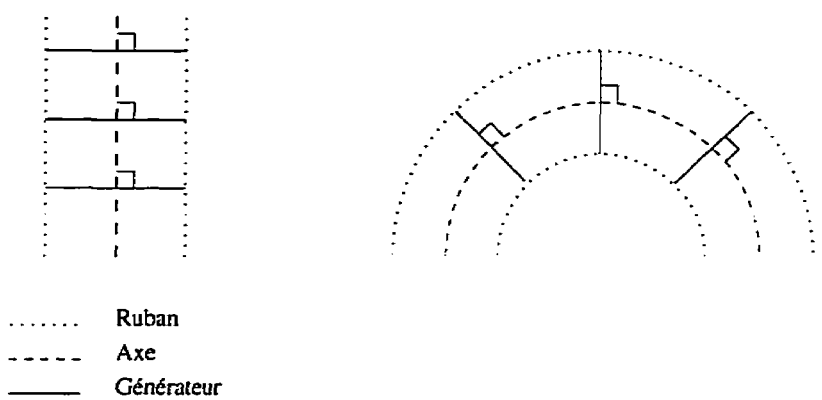


Figure 2.12 : Modèle de Brooks pour un corridor droit et un corridor courbe

La figure 2.13 illustre la façon dont nous modélisons un ensemble de corridors droits et courbes connectés.

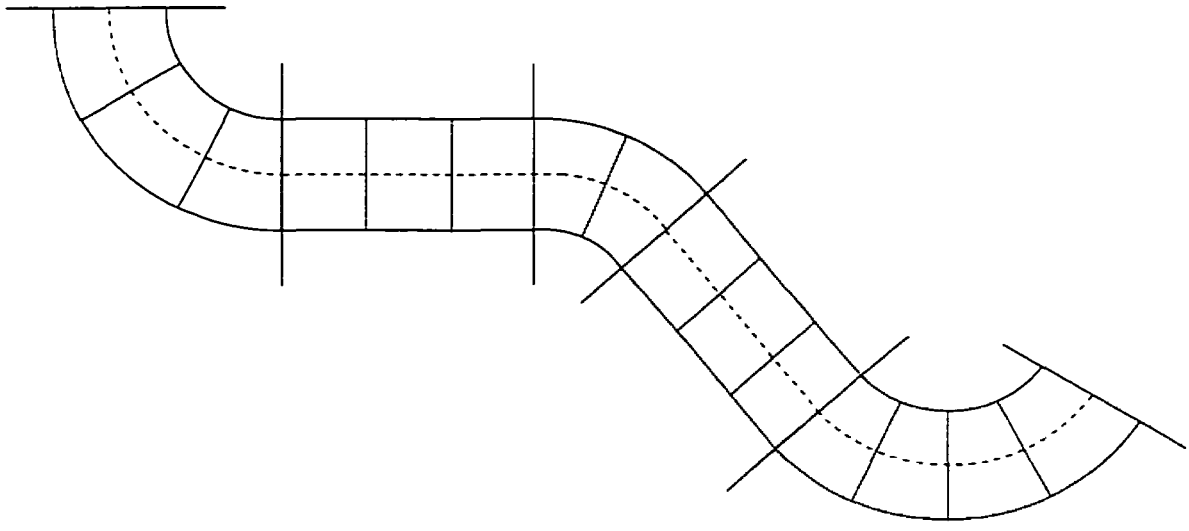


Figure 2.13 : Modèle de corridors droits et courbes connectés

2.3.2 Centre de l'intersection

Pour construire notre modèle, nous aurons besoin d'un point de référence, appelé *centre de l'intersection*. Nous définissons le centre de l'intersection, P, comme le centre de gravité d'une région R, commune aux corridors composant l'intersection. Nous avons envisagé différentes définitions de cette région qui sont représentées sur la figure 2.14.

Nous proposons une première définition, représentée sur la figure 2.14(a), où la région R est délimitée par des segments joignant les arêtes des corridors adjacents. La deuxième solution que nous avons envisagée est représentée sur la figure 2.14(b). Elle consiste à prolonger les rubans modélisant les corridors et à considérer leur intersection. Enfin, nous proposons une troisième solution, que nous illustrons sur la figure 2.14(c), qui consiste à restreindre la section des corridors. La figure 2.14(d) est

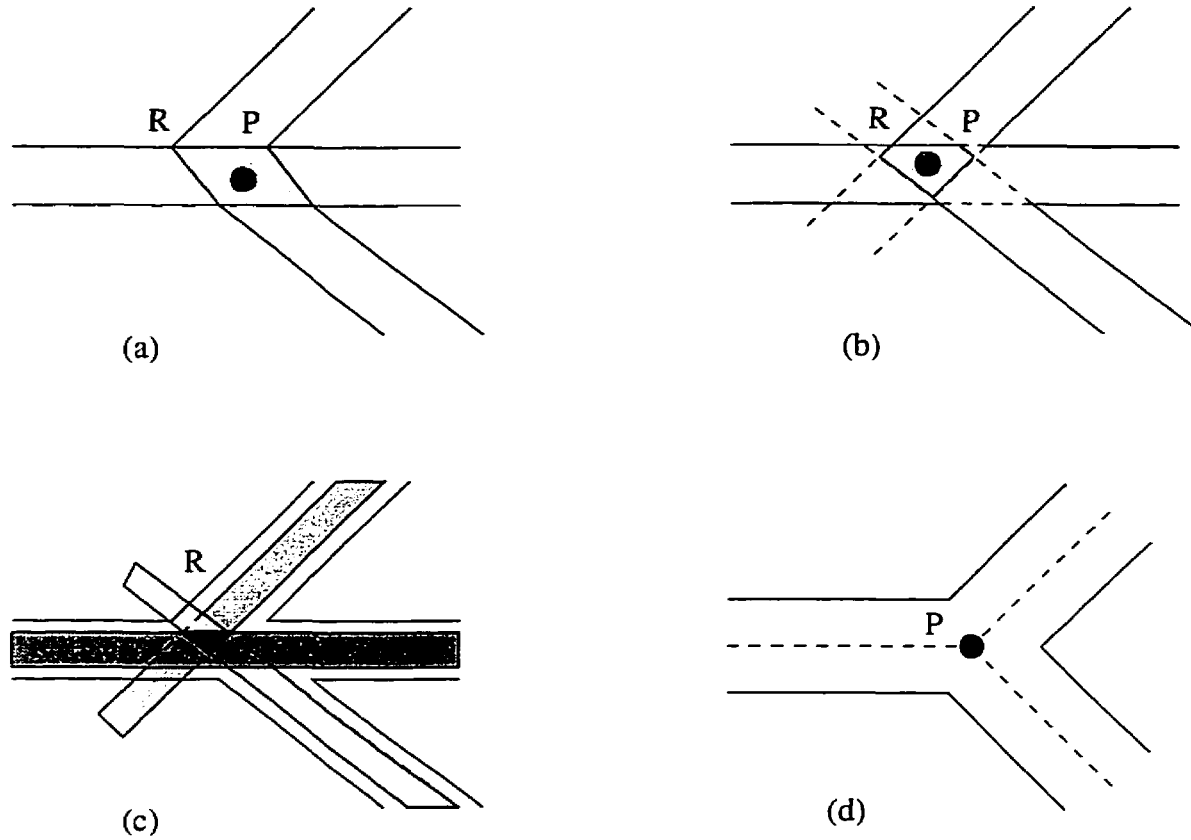


Figure 2.14 : Définition du centre d'une intersection

un cas particulier de chacun de ces trois cas. La région R se restreint au point P, point intersection des axes des corridors.

Notre choix se base essentiellement sur un critère de visibilité. En effet, nous considérons que la position optimale serait celle pour laquelle *le robot placé en ce point aurait la meilleure "visibilité" sur les corridors s'ouvrant devant lui, autrement dit les images qu'il reçevrait comporteraient le minimum de parties cachées ou d'occlusions*. La figure 2.15 met en lumière les inconvénients liés aux définitions (a) et (b). Ainsi la définition (a) ne garantit pas que, placé au centre de l'intersection, le

robot "voie" complètement tous les corridors composant l'intersection. Dans la figure 2.15(a), une partie du corridor (1) serait occultée. Au contraire, les définitions (b) et (c) garantissent cette propriété qui constitue un atout supplémentaire pour le processus de reconnaissance. Par rapport à la solution (b), la définition (c) présente l'avantage d'obtenir un point "également proche" des axes des couloirs. Ainsi avec la définition (b), dans la figure 2.15(b), P est excentré par rapport à l'axe du corridor (3). Avec la définition (c), P est également proche des axes des trois corridors (voir la figure 2.15(c)). Cela implique nécessairement que P est plus éloigné des axes (1) et (2) qu'avec la définition (b).

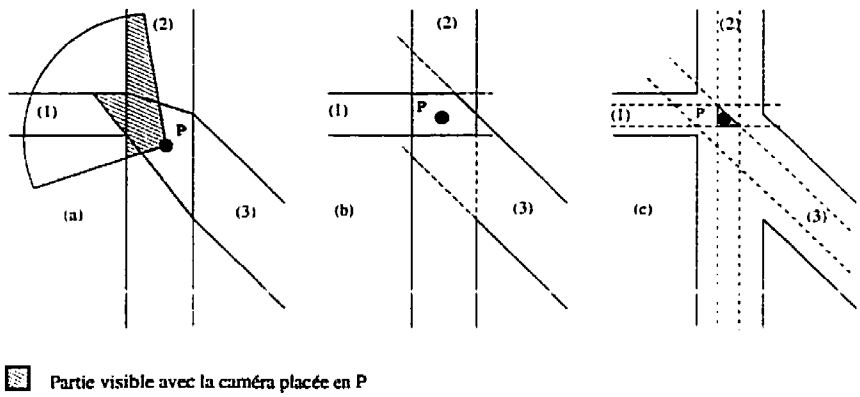


Figure 2.15 : Inconvénients des différentes solutions pour la définition d'un centre d'une intersection

Nous choisissons la solution (c) qui permet de construire le modèle le plus exploitable dans l'objectif de reconnaître des intersections.

2.3.3 Repère d'une intersection

La représentation d'une intersection nécessite la définition préalable d'un repère associé à cette intersection. Nous proposons donc un repère $P(x, y)$ orthonormé direct tel que :

- l'origine du repère P est confondue avec le centre de l'intersection ;
- l'axe des abscisses est parallèle à la direction d'observation du robot quand le robot arrive à l'intersection et dans le sens de déplacement du robot.

La figure 2.16 montre le repère d'une intersection en T.

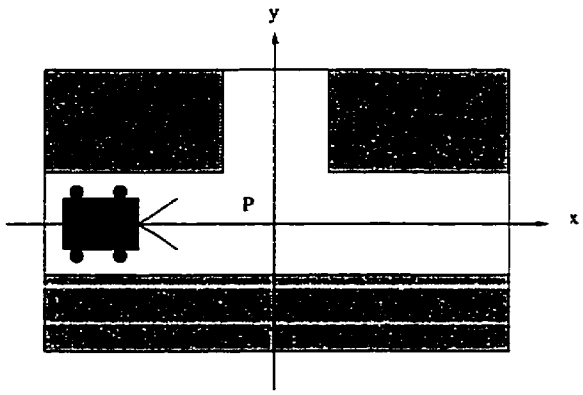


Figure 2.16 : Exemple : repère d'une intersection en T

Dans ce repère, nous utilisons la paramétrisation de Hough $r = x\cos\theta + y\sin\theta$ (voir la figure 2.17) pour représenter les axes des corridors de l'intersection.

Le paramètre r représente la distance euclidienne de l'origine à l'axe. Le paramètre θ représente l'angle de la perpendiculaire à l'axe par rapport à l'axe de référence x. En fait, l'axe d'un corridor n'est pas une droite, mais une demie-droite. Cette observation

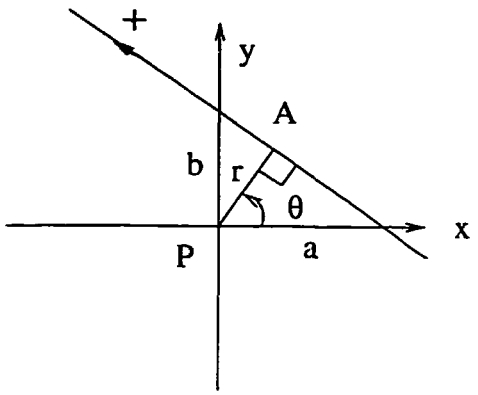


Figure 2.17 : Paramétrisation de Hough

nous contraint à introduire une nouvelle notion qui permet de définir complètement l'axe du corridor. Nous proposons de définir une origine A et un sens pour l'axe d'un corridor. Le sens positif que nous avons choisi est indiqué sur la figure 2.17. La figure 2.18 montre les paramètres des corridors d'une intersection à trois branches.

Remarquons que si la région de visibilité R est réduite à un point, pour tous les axes, $r = 0$.

Le modèle topologique d'une intersection sera donc constitué de :

- le repère de l'intersection ;
- un ensemble de rubans caractérisés par leur axe, leur sens et la longueur de leur générateur (largeur du couloir).

La représentation que nous proposons présente l'avantage d'être simple et efficace. En outre, elle donne directement les angles entre les axes des corridors ce qui est un avantage non négligeable pour notre modèle.

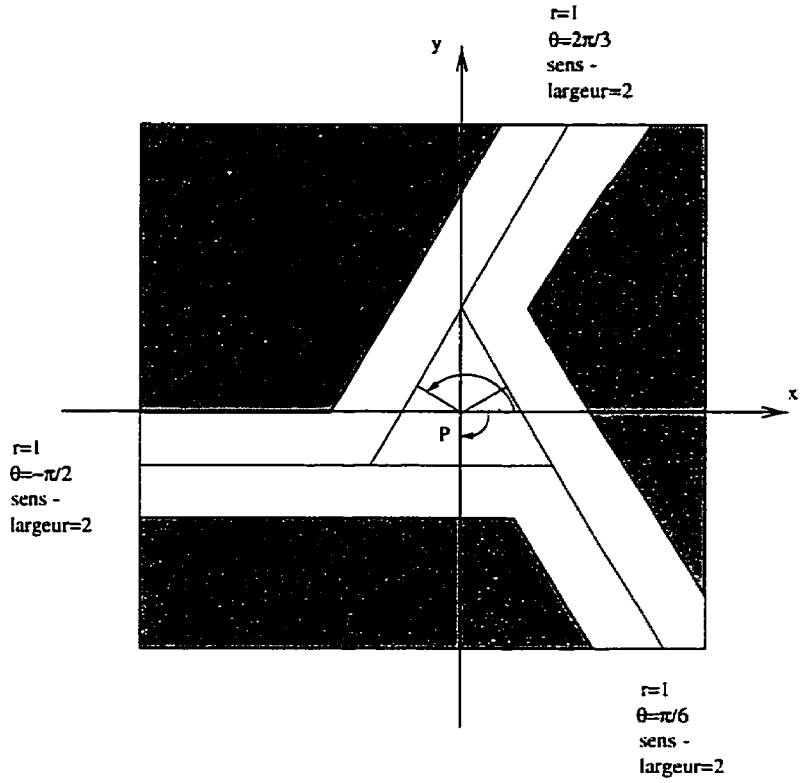


Figure 2.18 : Paramétrisation sur un exemple d'intersection à trois branches

2.3.4 Notion de super-intersection

Le modèle que nous proposons sera établi directement à partir de l'image. Or dans la plupart des cas, le robot n'aura qu'une vue partielle de l'intersection, ce qui impose des contraintes de souplesse sur notre modèle de façon à :

- permettre l'identification à partir d'informations partielles,
- être complété par des informations plus fiables.

Le cas présenté à la figure 2.19 montre quels peuvent être les problèmes qui peuvent se poser. Devons-nous considérer cette configuration comme une ou deux intersec-

tions ? Sur quels critères se baser : largeur de la section restreinte conduisant à une région de "visibilité" R ou non, seuil d entre les intersections entre les axes ?

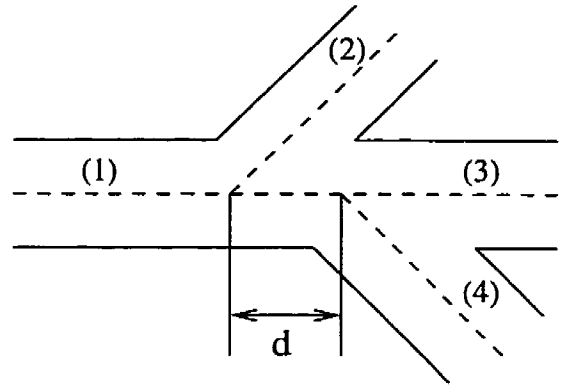


Figure 2.19 : Exemple de configuration délicate

Ces critères sont très contestables car la décision dépendant d'un seuil est par conséquent arbitraire. Si nous décomposons cette configuration en plusieurs intersections, ce qui revient à prendre un seuil d petit, alors nous perdons le bénéfice de la richesse de l'information fournie par la complexité de la situation. Si au contraire, nous regroupons le maximum de corridors dans une intersection, ce qui revient à choisir un seuil d élevé, alors nous courons le risque de construire des modèles différents selon le corridor d'arrivée du robot. En effet, selon le corridor d'arrivée du robot et selon sa trajectoire, certaines parties de l'intersection peuvent être ou non cachées. Considérons par exemple le cas de la figure 2.20. Arrivant du corridor (1), le robot détecte tous les corridors, mais arrivant de (2), à moins de poursuivre sa route dans le corridor (4), il ne verra pas le corridor (3).

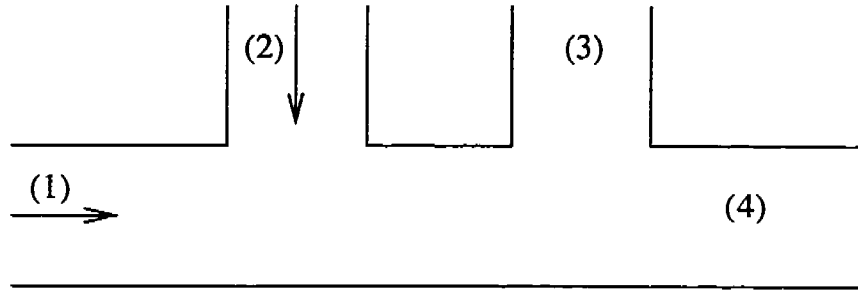


Figure 2.20 : Intersection en double T

Nous proposons un nouveau concept qui permet de représenter des intersections complexes. Autrement dit, par ce concept, nous avons trouvé le moyen de modéliser des intersections de façon générale. La notion de super-intersection que nous introduisons présente en particulier l'avantage de tenir compte des problèmes de visibilité sans négliger l'intérêt que représentent de telles configurations.

Définitions Nous proposons ici quelques définitions qui nous permettront de définir la notion de super-intersection.

Nous appelons *domaine de visibilité*, la région formée par l'intersection des rubans au lieu de rencontre de deux corridors ou plus.

Nous dirons que le robot est capable de *détecter un corridor* s'il existe dans la séquence d'images des informations révélant la présence d'un corridor. En pratique, cela signifie qu'il existe dans l'image des points qui appartiennent à ce corridor. Nous verrons plus loin comment extraire cette information de l'image.

Nous définissons une *intersection* comme le lieu de rencontre de l'ensemble des

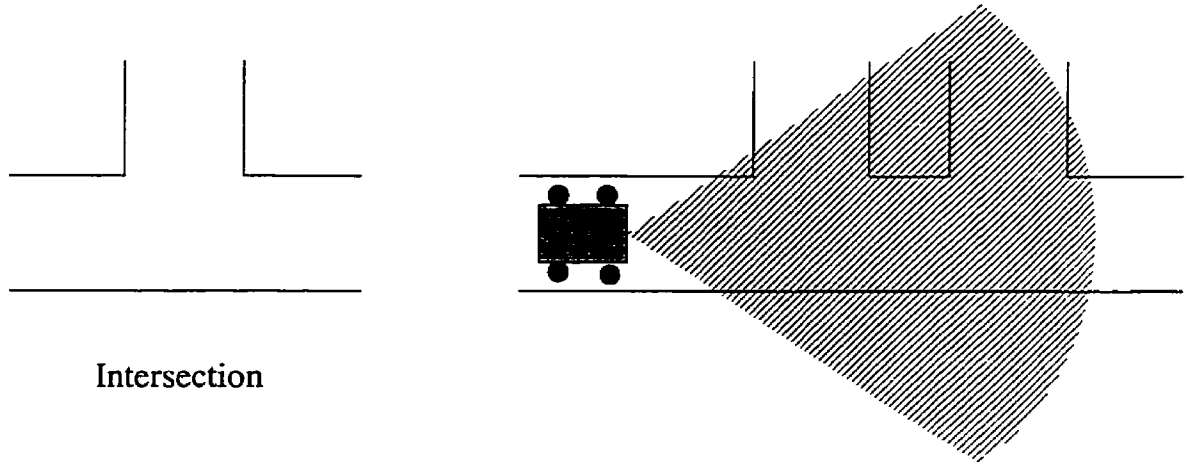
corridors ayant un domaine de visibilité commun et tel qu'arrivant de n'importe quel corridor, le robot est capable de détecter tous les autres corridors composant l'intersection. Autrement dit, quel que soit le chemin d'arrivée, il existe des points de l'image qui appartiennent à chacun des corridors composant l'intersection.

Nous définissons une *super-intersection* S comme un regroupement de N intersections $I_k, 1 \leq k \leq N$ qui satisfont le critère suivant. Étant donnée une intersection I_k appartenant à S et composée de M_k corridors C_{kj} : $I_k = \{C_{kj}, 1 \leq j \leq M_k\}$, il existe une intersection I_l appartenant à S différente de I_k $(l \neq k)$: $I_k = \{C_{lj}, 1 \leq j \leq M_l\}$ telle que : au moins un corridor de I_k peut être détecté d'un corridor de I_l et au moins un corridor de I_l (qui peut être différent) peut être détecté de I_k. Par conséquent, un même corridor peut éventuellement appartenir à deux intersections composant une super-intersection. Les figures 2.21 et 2.22 montrent des exemples d'intersection et de super-intersection.

(a) (b)

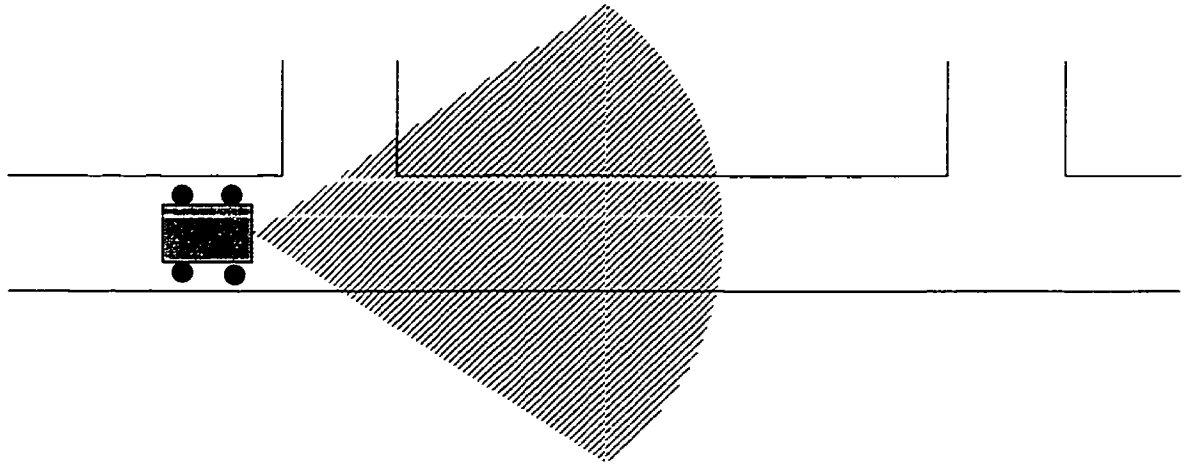
Figure 2.21 : (a) Intersection - (b) Super-intersection

Nous définissons *l'arête commune* de deux corridors C_1 et C_2 comme l'arête physique dans l'espace tridimensionnel qui est l'intersection, si elle existe, d'un des



Intersection

Super-intersection



Deux intersections

Figure 2.22 : Un cas limite

deux murs de C_1 avec l'un des deux murs de C_2.

La figure 2.23 illustre les définitions ci-dessus.

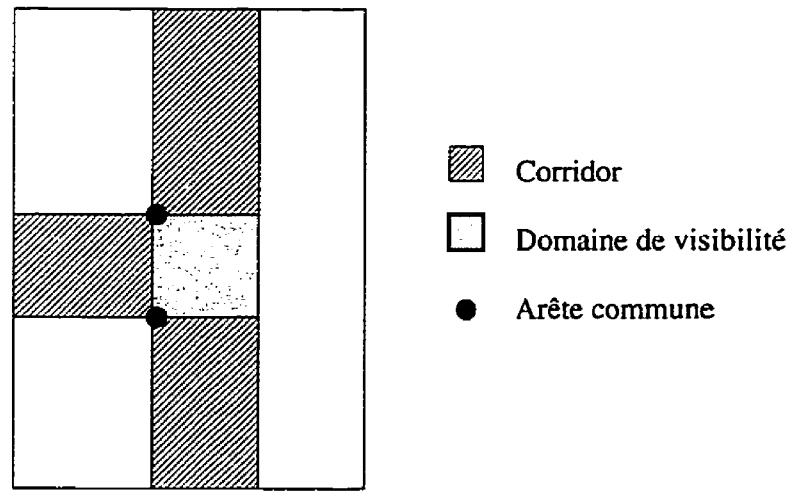


Figure 2.23 : Définitions liées au concept de super-intersection

Repère d'une super-intersection Nous avons précédemment défini un repère lié à chaque intersection. Pour représenter une super-intersection, nous avons besoin d'un repère qui lui est associé. Nous choisissons le repère $\Re$(S,x,y) lié à une super-intersection comme étant le repère d'une des intersections composant la super-intersection. Les repères de chaque intersection seront définis relativement au repère de la super-intersection.

Décomposition en intersections et en super-intersections Un ensemble de corridors forme une intersection si :

1. Les corridors partagent un domaine de visibilité commun : l'intersection de leurs rubans existe.

2. A partir de n'importe quel corridor, le robot est capable de détecter tous les autres corridors. Pour cela, il faut tester tous les corridors deux à deux. **Un corridor C_1 est détectable d'un autre corridor C_2 si l'une des conditions suivantes est remplie :**

   (a) C_1 et C_2 n'ont pas d'arête commune. Par exemple, sur la figure 2.24(a), les corridors (1) et (2) ainsi que (3) et (4) ont une arête commune mais les corridors (2) et (4) ou (1) et (3) n'en ont pas.

   (b) C_1 et C_2 ont une arête commune et leurs arêtes fermant les corridors forment un angle strictement inférieur à π (voir les exemples de la figure 2.24).

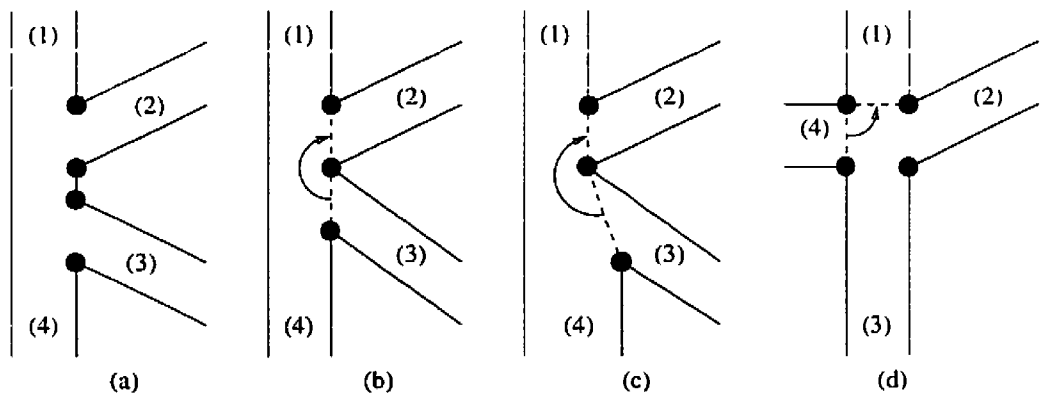


Figure 2.24 : (a) (2) et (4) n'ont pas d'arête commune – (b)(c) (2) et (3) ont un angle supérieur à π – (d) (1) et (4) forment un angle inférieur à π

Si deux corridors satisfont la condition (1) mais pas la condition (2), ils appartiennent à la même super-intersection. Le problème consiste maintenant à décomposer une super-intersection en intersections.

Ayant déterminé les couples de corridors susceptibles de former une intersection, nous cherchons maintenant à les regrouper. La méthode est incrémentale et consiste à former des regroupements de trois corridors, puis de quatre corridors à partir des triplets formés et ainsi de suite jusqu'à ce qu'il n'y ait plus de regroupement possible. Les couples, triplets, quadruplets, etc. n'ayant pu être regroupés constituent la super-intersection.

Exemple Nous décrivons cette méthode à partir de l'exemple de la figure 2.25. Notons toutefois que la procédure décrite ci-dessous ne constitue pas un algorithme applicable mais a seulement pour but de comprendre le principe de la méthode. En effet, comme nous le verrons par la suite, ce problème de décomposition est un problème NP-complet dont une solution est décrite dans la section 2.3.5.

Le principe de la procédure de regroupement est la suivante :

1. Parmi les $\begin{pmatrix} 2 \\ 5 \end{pmatrix} = \frac{5!}{2!(5-2)!} = 10$, les couples de corridors satisfaisant aux conditions décrites ci-dessus sont les suivants : (1,2) (1,3) (1,5) (2,3) (2,4) (2,5) (3,5) (4,5).

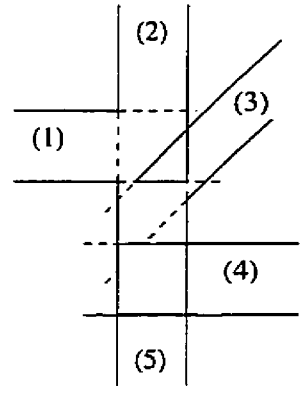


Figure 2.25 : Exemple de super-intersection : $S = \{(1,2,3,5),(2,4,5)\}$

2. Les couples sont regroupés pour former les triplets réalisables parmi les

$$\begin{pmatrix} 3 \\ 5 \end{pmatrix} = \frac{5!}{3!(5-3)!} = 20$$

combinaisons possibles : (1,2,3) (1,2,5) (1,3,5) (2,3,5) (2,4,5).

3. Les triplets sont regroupés pour former les quadruplets réalisables parmi les

$$\begin{pmatrix} 4 \\ 5 \end{pmatrix} = \frac{5!}{4!(5-4)!} = 5$$

combinaisons possibles : (1,2,3,5). Le triplet (2,4,5) ne peut pas être regroupé au sein d'un quadruplet. Il constitue donc une intersection.

4. Il ne reste plus de regroupement réalisable. La super-intersection est donc composée de deux intersections : $S = \{(1,2,3,5),(2,4,5)\}$.

2.3.5 Décomposition d'une super-intersection en intersections

Notion de graphe Nous utilisons la terminologie définie par Evans et Minieka (1992). Un *graphe* G est un ensemble de *sommets* notés $1, 2, ..., i, j, ...$ et d'arêtes reliant certains sommets. Une arête reliant le sommet i au sommet j est notée (i, j). Un sommet ne peut pas être relié à lui-même. Autrement dit, l'arête (i, i) n'est pas autorisée. Un graphe est dit *non-orienté* si les arêtes reliant les sommets ne sont pas ordonnés c'est-à-dire $(i, j) = (j, i)$. Un graphe dans lequel chaque paire de sommets est connectée par une arête est appelé un *graphe complet*. Un graphe g est un *sous-graphe* d'un graphe G si tous les sommets et les arêtes de g sont dans G. Deux sommets sont dits *adjacents* s'ils sont reliés par une arête. Un graphe peut être représenté par sa *matrice d'adjacence des sommets* A. Soit N le nombre de sommets du graphe. A est une matrice $N \times N$ dans laquelle $a_{ij} = 1$ si les sommets i et j sont adjacents, 0 sinon. Pour un graphe non-orienté, A est symétrique c'est-à-dire $a_{ij} = a_{ji}$. Un graphe est *complet* si $a_{ij} = 1, \forall i \neq j$.

Graphe de la super-intersection Nous représentons la super-intersection par un graphe dont les sommets sont occupés par les corridors. Deux corridors C_1 et C_2 sont reliés par une arête si les deux corridors partagent un domaine de visibilité commun et vérifient le critère de détectabilité réciproque. Ce graphe sera représenté par sa matrice d'adjacence $A = (a_{ij})_{1 \leq i,j \leq N}$, avec N le nombre de corridors de la super-intersection.

Domaine de visibilité de deux corridors Soit S la région occupée par le sol dans l'environnement E perçu par le robot. Soit D la région constituée par l'intersection des rubans des deux corridors considérés dans E. Le domaine de visibilité V de deux corridors est l'intersection de D avec S :

$$V = D \cap S.$$

La figure 2.26 montre un exemple de domaine de visibilité entre deux corridors.

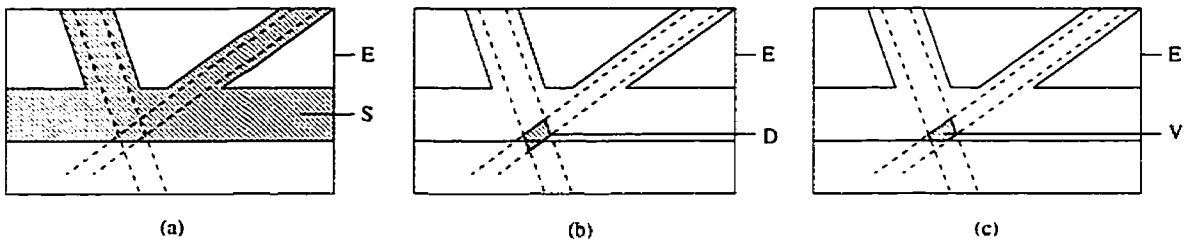


Figure 2.26 : Domaine de visibilité et grilles d'occupation

Nous discrétisons l'espace (X, Z) et le représentons par une grille d'occupation $O(i, j)$ dont les valeurs sont :

$$\begin{cases} 1 & \text{si } (i,j) \text{ est occupée} \\ 0 & \text{sinon} \end{cases}$$

Les opérations logiques s'appliquent sur les grilles d'occupation. Le "et logique" (&) correspond à une intersection des régions de E. Le domaine de visibilité V de deux corridors est donc représenté par sa grille d'occupation V_o qui est l'intersection

des grilles d'occupation des régions D et S. Ainsi, considérons R_1 et R_2 les régions occupées par les deux corridors, R_{o1} et R_{o2} leurs grilles d'occupation. Si D_o et S_o sont les grilles d'occupation correspondant aux régions D et S, alors le domaine de visibilité V de deux corridors est défini par :

$$V : V_o = D_o \& S_o$$

$$V : V_o = R_{o1} \& R_{o2} \& S_o.$$

Si $V_o = \emptyset$, alors les deux corridors R_1 et R_2 n'ont pas de domaine de visibilité commun. Par conséquent, ils ne peuvent pas appartenir à la même intersection.

Critère de détectabilité réciproque La deuxième condition que nous avons posée pour que deux corridors C_1 et C_2 appartiennent à la même intersection stipulait que C_1 soit "détectable" de C_2 et réciproquement. Nous montrons ici comment vérifier le critère de détectabilité réciproque.

Un corridor C_1 est détectable d'un corridor C_2 si la condition 1(a) ou la condition 1(b) est remplie. La condition 1(a) dit que C_1 et C_2 ne doivent pas avoir d'arête commune. Nous avons présenté le modèle de l'intersection comme un ensemble de corridors représentés par leur axe, leur largeur et leur sens dans le repère de l'intersection. Nous pouvons maintenant préciser que ces corridors sont ordonnés de façon à ce que deux corridors placés côte à côte dans l'espace tridimensionnel se retrouvent consécutifs dans le modèle de l'intersection. Nous verrons plus loin comment nous pouvons déterminer l'ordre des corridors directement à partir de l'image avant même

de construire le modèle topologique de l'intersection. La figure 2.27 illustre cette méthode sur une intersection à quatre branches.

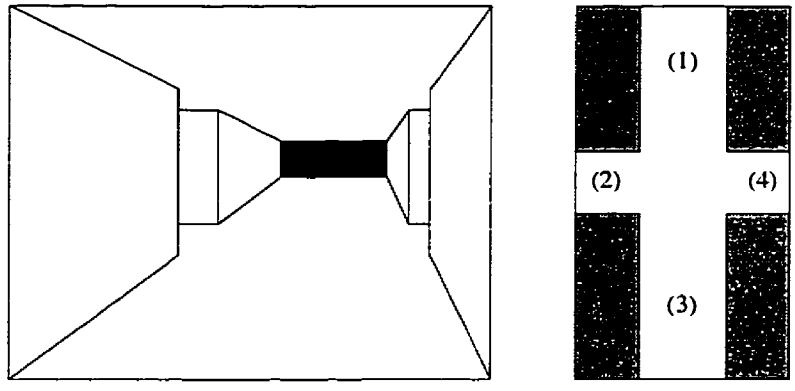


Figure 2.27 : Intersection en croix : (a) projection perspective – (b) vue topologique. Les corridors sont ordonnés dans le sens trigonométrique dans le plan du sol.

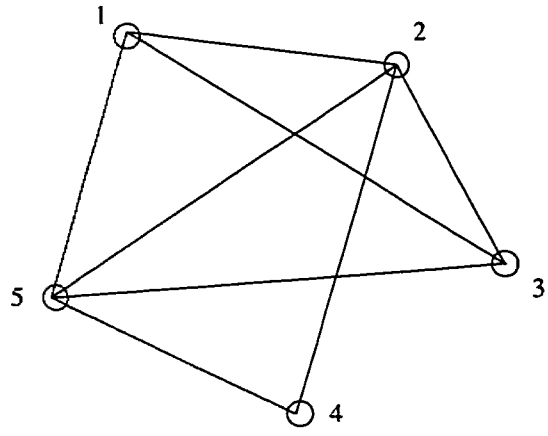
La condition 1(a) revient donc à dire que les corridors C_1 et C_2 ne sont pas consécutifs dans l'image.

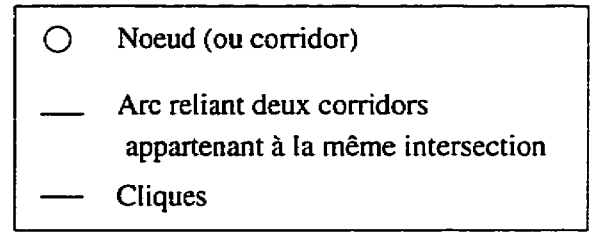
Si les corridors C_1 et C_2 ne sont pas consécutifs (condition 1(a) non vérifiée) mais l'angle formé par les arêtes les fermant est strictement inférieur à π (condition 1(b) vérifiée) alors ils appartiennent à la même intersection. Pour que cette condition soit vérifiée, il suffit de calculer la position des arêtes communes dans le plan du sol ce qui se calcule facilement d'après la définition d'une arête commune, le modèle des corridors et leur ordonnancement. Les arêtes fermant les corridors C_1 et C_2 et leur angle en découlent.

Analyse du graphe de la super-intersection Un *sous-graphe maximal complet* ou *clique* est un sous-graphe complet qui ne peut pas être contenu dans un autre sous-graphe complet. Autrement dit, une clique est un sous-graphe complet auquel on ne peut pas rajouter de sommet. Le problème de la décomposition d'une super-intersection en intersections revient donc à chercher les cliques dans le graphe de la super-intersection. La figure 2.28 montre le graphe de la super-intersection étudiée précédemment (voir la figure 2.25) et les cliques (ou intersections) de ce graphe.

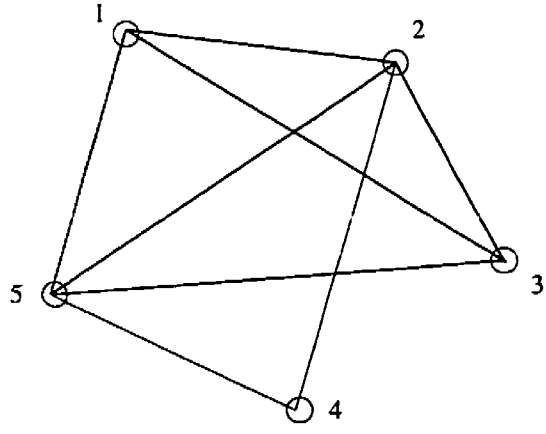
Montrons que ce problème est un problème NP-complet. Soit un graphe G non-orienté comportant V sommets. Soit k un nombre entier tel que $k \leq V$. Comment trouver une clique C dans G telle que le nombre de sommets de C, $\mid C \mid \geq k$? Even (1979) montre que ce problème, appelé recherche de la clique maximum est un problème NP-complet. Nous voyons que si nous arrivons à trouver une clique C telle que $\mid C \mid \geq k$, alors nous pouvons trouver toutes les cliques C de G. Par conséquent, le problème de trouver toutes les cliques d'un graphe non-orienté est un problème NP-complet.

Bron et Kerbosch (1973) proposent un algorithme qui permet de trouver toutes les cliques d'un graphe non-orienté. L'algorithme utilise une technique "branch-and-bound" pour couper les branches qui ne peuvent pas conduire à une clique. L'algorithme tend à minimiser le nombre de branches à examiner et génère les cliques dans un ordre aléatoire. La méthode sort les plus grandes cliques en premier et génère séquentiellement les cliques qui ont la plus grande intersection en commun. Chaque

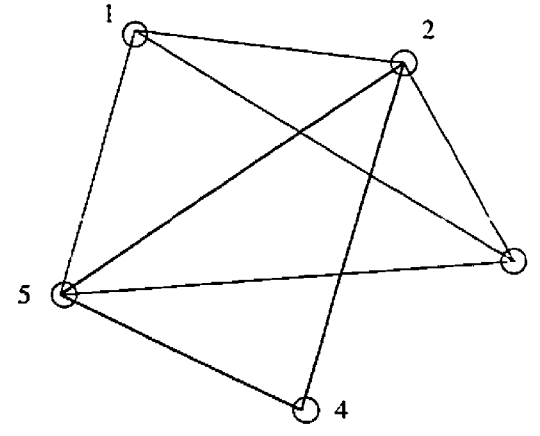




Graphe de la super-intersection



Clique (1,2,3,5) du graphe



Clique (2,4,5) du graphe

Figure 2.28 : Cliques d'un graphe. Le graphe représente la super-intersection de la figure 2.25 et ses cliques (ou intersections).

clique obtenue correspond à une intersection appartenant à la super-intersection. Ainsi cet algorithme nous permet de décomposer une super-intersection en intersections.

2.4 Détection du contour du sol

Pour détecter une intersection, nous nous basons sur la détection du sol dans l'image. Plusieurs approches sont proposées dans la littérature pour résoudre des problèmes de navigation et en particulier le problème de détection d'une route. Une première approche est basée sur la couleur des pixels de l'image. Une seconde approche consiste à détecter les bords de la route dans l'image en les approximant par des droites. Une dernière approche est basée sur la prédiction du déplacement des points dans la séquence d'images. Elles sont présentées dans l'annexe A.5.

2.4.1 Approche retenue

Les considérations suivantes expliquent notre décision dans le choix de la méthode. Nous avons écarté d'emblée une approche basée sur la couleur. En effet, dans une mine, les couleurs sont très uniformes. En particulier, alors que la couleur d'une route est très différente du reste de l'environnement, dans une mine, les couleurs du sol, des murs et du plafond sont très semblables. Un critère basé sur la couleur ne peut donc pas être retenu. Les approches basées sur la détection de lignes et celles basées sur la prédiction du déplacement des points sont plus intéressantes pour notre problème.

L'environnement considéré présente des particularités qui font que :

- La détection de lignes dans l'image est plus délicate que sur une image représentant une route.
- La texture du sol est différente de la texture des murs ou du plafond.

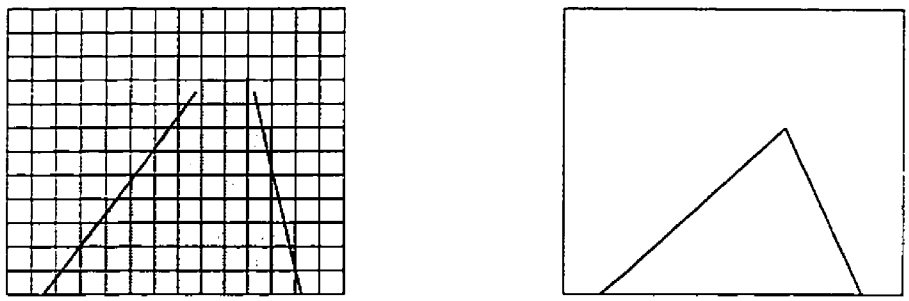
Il nous a donc paru plus fiable de nous baser sur la détection de régions appartenant au sol plutôt que sur la détection du contour du sol dans l'image. C'est pourquoi nous avons choisi une approche telle que celle proposée par Chioiu et Hervé (1996).

Cette méthode est basée sur la prédiction du déplacement des points du sol entre deux images consécutives, connaissant le déplacement de la caméra. Elle consiste à génèrer l'image virtuelle (prédiction de l'image suivante de la séquence) dans laquelle tous les pixels appartiennent au sol. Dans cette image virtuelle, les régions qui ne correspondent pas effectivement à des points du sol vont être affectés par des distorsions de parallaxe. L'algorithme évalue le degré de similarité entre les régions équivalentes de l'image réelle et de sa prédiction en calculant pour chaque pixel de l'image un coefficient de corrélation entre des fenêtres rectangulaires centrées sur le pixel considéré. Chaque pixel est alors étiqueté comme appartenant ou non au sol en fonction de la valeur de son coefficient de ressemblance.

2.4.2 Approximation du contour du sol

En entrée de ce module, le système reçoit deux images binaires, l'une indiquant les pixels correspondant aux occlusions dans la scène, l'autre indiquant les pixels

appartenant au sol (voir la figure 2.4.2(a)).



(a) (b)

Figure 2.29 : (a) Points du sol dans l'image – (b) Modèle du contour du sol

Le problème consiste à extraire de cette image les paramètres des segments (droits ou courbes) délimitant cet espace de sol. Pour ce faire, nous utilisons une méthode basée sur la transformée de Hough (Hough 1962, Illingworth et Kittler 1988), que nous avons adaptée aux cas particuliers des projections de rubans droits ou courbes tracés sur un plan horizontal d'altitude connue. Nous exploitons les propriétés géométriques de la projection perspective, comme par exemple le point de fuite observé pour un tronçon de couloir droit (voir la figure 2.4.2(b)) pour réduire la dimension de l'espace de recherche. Une description détaillée de la méthode est fournie dans l'annexe B.

En sortie de ce module, nous obtenons le contour du sol approximé par des segments de droite ou des arcs de conique dans le repère de l'image (voir la figure 2.30).

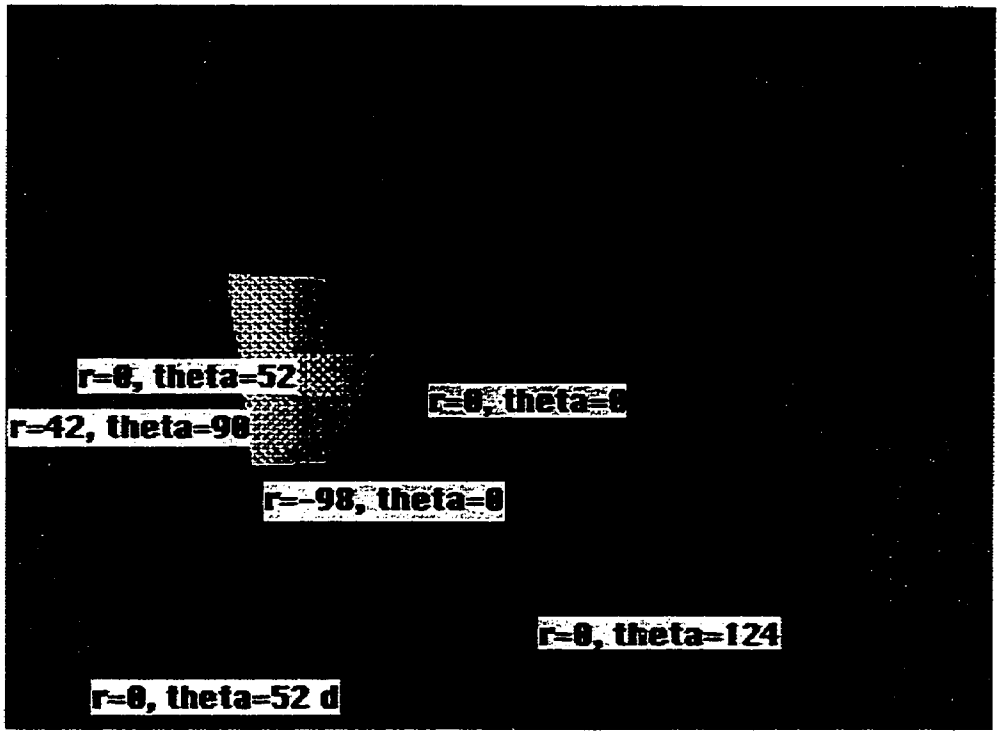


Figure 2.30 : Approximation du contour du sol

2.4.3 Étiquetage des segments

Pourquoi avons-nous besoin d'étiqueter les segments du contour du sol ? Il ne nous suffit pas d'effectuer la projection perspective inverse sur les segments obtenus. En effet, ces segments ne contiennent pas une information complète sur la topologie de la scène, simplement parce que certaines parties peuvent être occultées. Il s'agit donc d'interpréter ce qui est visible, autrement dit les segments de sol que nous venons de déterminer, pour en déduire ce qui ne l'est pas. Les étiquettes apposées aux segments vont nous permettre d'effectuer cette interprétation. Par exemple, si nous savons qu'un segment correspond à un bord de corridor, nous pouvons en déduire l'axe du

corridor.

Une importance toute particulière doit être accordée aux occlusions qui nous signalent la présence de corridors débouchant dans le corridor principal. Pour un point de vue donné de l'observateur, une occlusion correspond à une discontinuité de profondeur dans la scène observée. Si la scène est structurée et composée d'objets identifiables, il est théoriquement possible, en utilisant des techniques d'analyse d'image de haut niveau, de détecter les occlusions. Par exemple, en reconnaissant l'objet en avant-plan et l'objet en arrière-plan, on peut déterminer qu'une partie de ce dernier est cachée et donc en *déduire* la présence d'une occlusion.

Dans le type d'environnement qui nous intéresse ici, il n'est pas possible de détecter et *a fortiori* de reconnaître des objets dans la scène. L'utilisation de méthodes d'analyse de haut niveau est donc par conséquent exclue. Il est par contre possible d'exploiter la parallaxe de mouvement ou la disparité stéréoscopique pour détecter les discontinuités de profondeur (Pilon 1996). Par exemple, dans la scène d'intersection représentée à la figure 2.31, la discontinuité de profondeur observée au point C se traduit par une ligne d'occlusion dans l'image (voir la figure 2.32(a)) le long de laquelle le champ de mouvement dans l'image exhibe une discontinuité en magnitude (mais pas en direction).

Nous proposons d'étiqueter les segments selon les catégories suivantes :

- "Mur Visible" MV,
- "Occlusion" O,

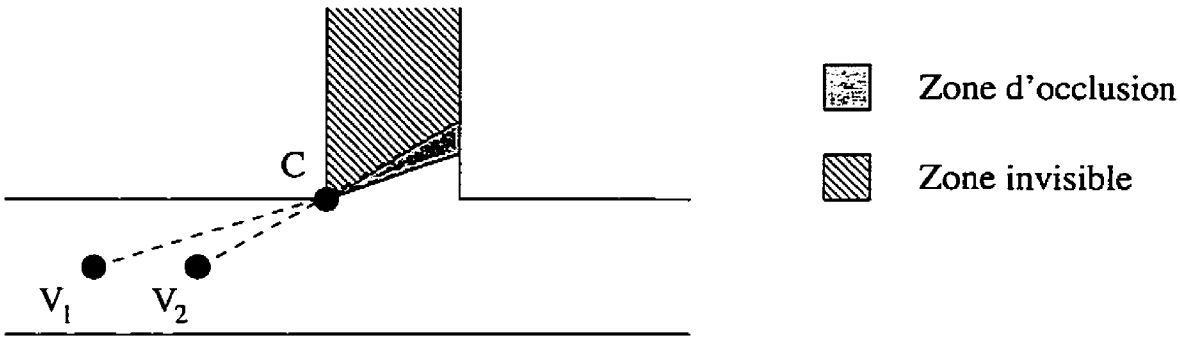


Figure 2.31 : Définition d'une occlusion

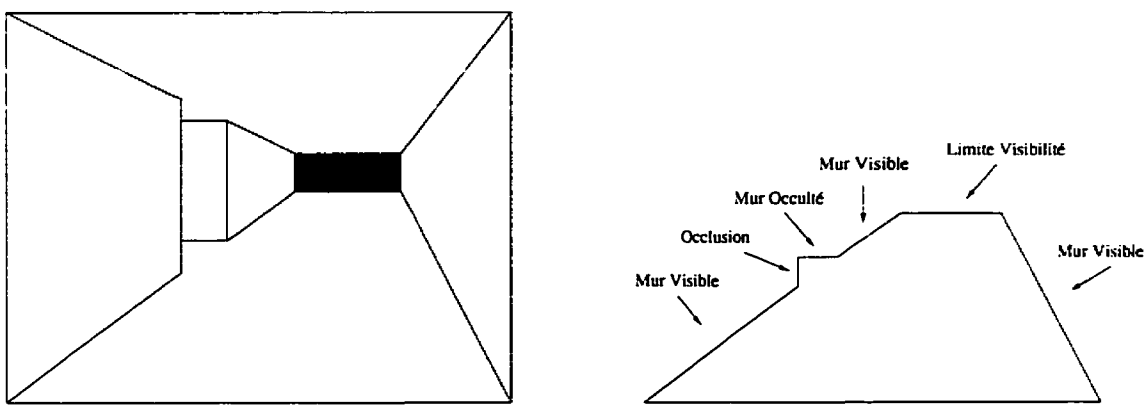


Figure 2.32 : (a) Exemple d'occlusion - (b) Contour du sol étiqueté

- "Mur Occulté" MO,
- "Limite de Visibilité" LV.

La figure 2.32(b) montre un exemple d'étiquetage des segments du contour du sol pour une intersection en T.

Une étiquette "Mur Visible" est assignée à un segment qui borde un mur complètement visible du robot, autrement dit qui ne comporte pas de parties occultées par d'autres surfaces. Une étiquette "Occlusion" est attribuée à un segment qui limite une surface occultant une autre surface. Une étiquette "Mur Occulté" indique que le segment borde une surface qui est occultée.

Les étiquettes "Occlusion" et "Limite de Visibilité" sont directement déduites de l'estimation des segments à partir des points de contour. Le problème consiste à différencier les murs visibles des murs occultés. Il suffit de considérer les deux segments situés de part et d'autre des segments portant l'étiquette "Occlusion". Dans une telle paire de segments, l'un appartient au mur visible, l'autre au mur occulté. Il suffit d'évaluer la profondeur de chacun pour faire la distinction. Le plus proche est le mur visible, l'autre est le mur occulté. Les segments restant appartiennent à des murs visibles.

Dans ce travail, nous n'effectuons pas à proprement parler l'étiquetage des segments. Mais plusieurs avenues ont été explorées ces dernières années pour résoudre ce problème. Citons Hummel et Zucker (1983) qui ont développé une théorie pour caractériser les processus de relaxation d'étiquettes. Les processus de relaxation d'éti-

quettes sont basés sur l'utilisation parallèle de contraintes locales entre des étiquettes. La théorie est basée sur une définition de la consistance entre étiquettes, élargissant la notion de satisfaction de contraintes. Canning et al. (1987) proposent une méthode basée sur l'étiquetage des pixels pour détecter des courbes dans l'image. À chaque pixel est associé un ensemble de masques possibles. Les masques qui sont géométriquement consistants avec les masques voisins sont identifiés et retenus. Les liens de consistance entre les masques permettent d'assembler les composants connectés et d'en extraire les lignes.

2.4.4 Détermination du nombre de corridors

Nous avons déterminé les segments le sol dans l'image et nous leur avons assigné une étiquette. Nous obtenons ainsi une séquence ordonnée **S** des parties de murs apparaissant dans la séquence d'images considérée.

Le robot se déplace dans un couloir dont il connaît l'axe ("droit" ou "courbe") et la largeur pour les avoir déterminés à l'intersection précédente. Il arrive à une nouvelle intersection si :

- un autre corridor s'ouvre sur le corridor dans lequel il se déplace ;
- un tournant apparaît alors qu'il se trouve dans un corridor droit ;
- un corridor droit apparaît alors qu'il se trouve dans un tournant ;
- le corridor dans lequel il se trouve, se rétrécit ou s'élargit ;

Nous définissons un évènement E comme étant :

- une occlusion O,
- une limite de visibilité LV.

Un évènement révèle l'apparition d'une intersection à moins que :

- le robot ne soit dans un couloir droit et que le seul évènement qu'il détecte soit une limite de visibilité ;
- le robot ne soit dans un corridor courbe et que le seul évènement qu'il détecte soit une occlusion.

Le tableau 2.33 donne le nombre de corridors dans la scène à partir des étiquettes des segments.

Figure 2.33 : Détermination du nombre de corridors

Évènement	Corridor connu	
	Droit	Courbe
1 LV	1	2
1 O	2	1
#E>1	#E+1	#E+1

La figure 2.34 décrit les types de situation résumés par le tableau 2.33.

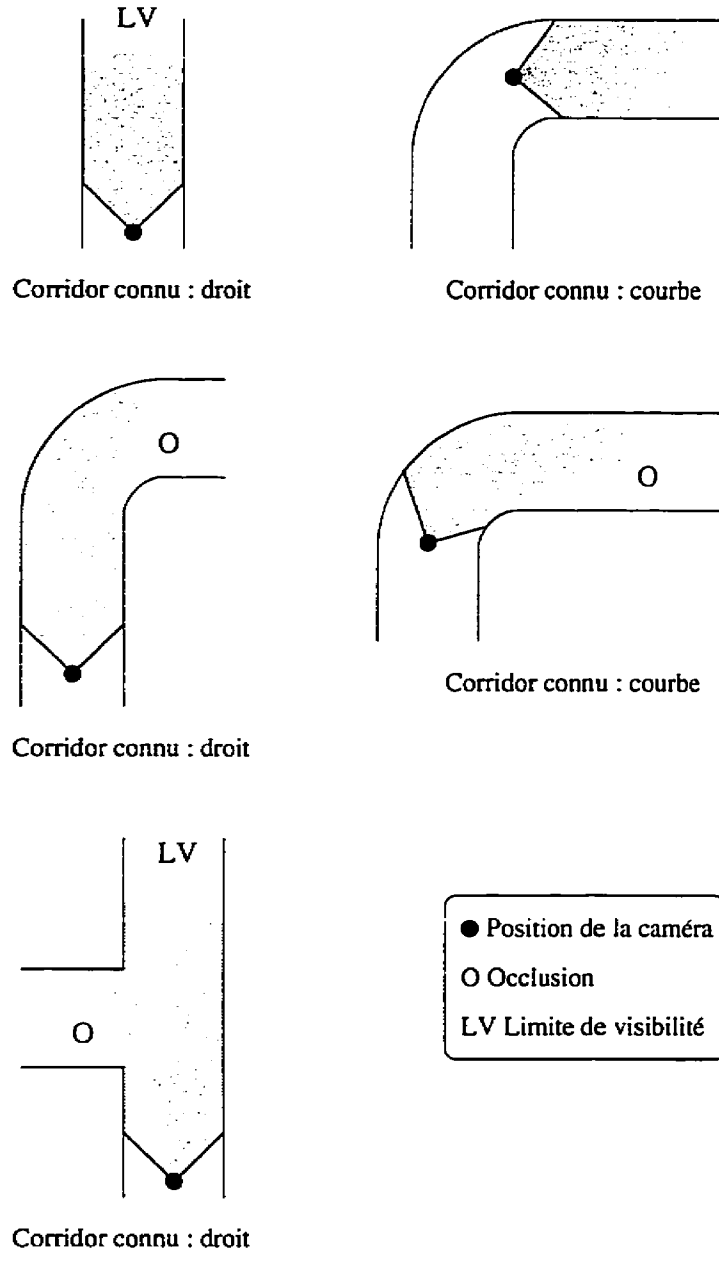
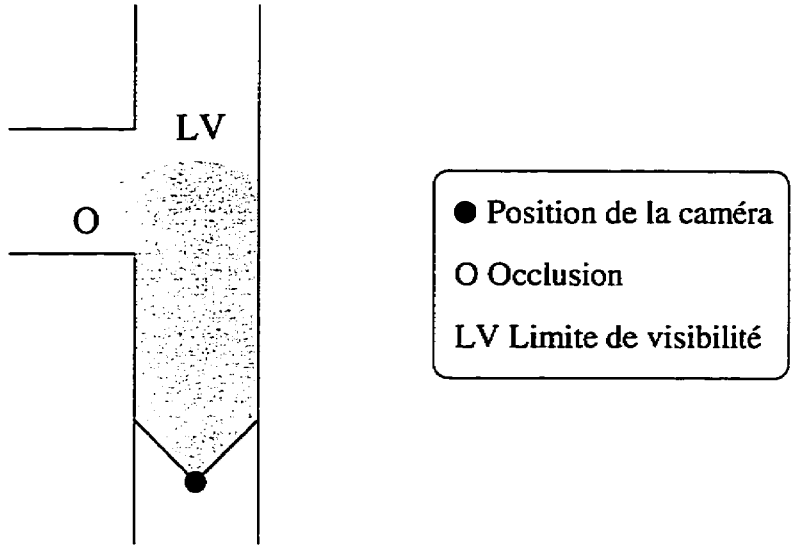


Figure 2.34 : Détermination du nombre de corridors à partir des étiquettes

Il se peut que le robot se trouve dans une configuration temporaire très particulière décrite à la figure 2.35. Cette situation où une étiquette "Occlusion" et une étiquette



Corridor connu : droit

Figure 2.35 : Cas particulier où une étiquette "Occlusion" et une étiquette "Limite de Visibilité" sont consécutives.

"Limite de Visibilité" sont consécutives est temporaire car à l'image suivante, le robot aura suffisamment avancé pour que la limite de visibilité soit repoussée au delà de l'intersection. Dans ce cas de figure, nous conclurons simplement qu'il est prématuré d'analyser la scène et qu'il vaut mieux attendre l'image suivante.

Ce premier résultat permet dès cette étape de décider si le robot est confronté à une intersection ou non. En effet, le nombre de corridors est égal à 1, il n'y a pas d'intersection et le robot poursuit son exploration. Si, au contraire, le nombre de corridors est strictement supérieur à 1, cela indique la présence d'une intersection. Le système procède alors à une analyse plus approfondie de la séquence d'images pour déterminer le modèle de l'intersection et le mémoriser. Cette analyse s'effectue

pendant que le robot poursuit sa route.

Grâce à ce résultat, le système peut dès cette étape classer l'intersection en fonction du nombre de corridors qui la composent. Outre un critère déterminant pour la différentiation des intersections, cette classification joue un rôle fondamental dans la suite du processus de reconnaissance et d'apprentissage de l'intersection rencontrée. En effet, le système mémorise une intersection en se basant sur ce critère. Par ailleurs, les étiquettes fournissent de prime abord une indication sur la localisation d'éventuels corridors s'ouvrant sur le corridor principal. À un évènement **E** est associé un corridor. C'est par ce moyen que nous pouvons déterminer l'ordre de succession des corridors. Ce critère était essentiel pour décomposer une super-intersection en intersections. L'approximation du sol en segments analysé conjointement avec les étiquettes va permettre au système de construire le modèle topologique de l'intersection.

2.5 Construction du modèle topologique

Nous montrons dans cette section comment nous déterminons le modèle topologique d'une intersection à partir des segments approximant le contour du sol et des étiquettes. Autrement dit, nous montrons comment calculer le centre de l'intersection ainsi que les axes, les largeurs et le sens des corridors dans un repère lié à l'intersection.

2.5.1 Modèle de la caméra

Pour modéliser la caméra, nous utilisons le modèle du trou d'épingle qui produit une projection perspective des points de la scène dans l'image (Horn 1986). Un point M de la scène est représenté par son vecteur de coordonnées par rapport au repère de la caméra R_c, $M = (X, Y, Z)'$. Sa projection dans l'image est représentée par son vecteur de coordonnées par rapport à R_c, $m = (x, y, f)$. La projection perspective est décrite par la relation suivante :

$$x = \frac{fX}{Z}, \quad y = \frac{fY}{Z},$$

avec f distance focale de la caméra. Ou encore si la distance focale en x est différente de la distance focale en y :

$$x = \frac{f_x X}{Z}, \quad y = \frac{f_y Y}{Z}.$$

Pour travailler avec une image droite et une distance focale positive, le plan de l'image est avancé devant le centre optique (voir la figure 2.36).

Nous supposons que la caméra est placée à une hauteur fixe h par rapport au plan du sol et que son axe optique est parallèle au plan du sol. La figure 2.37 montre le référentiel absolu R_a et le repère de la caméra R_c choisis.

Dans le repère de la caméra et en utilisant la paramétrisation de Hough (voir la figure 2.38), l'équation d'un bord d'un corridor droit s'écrit :

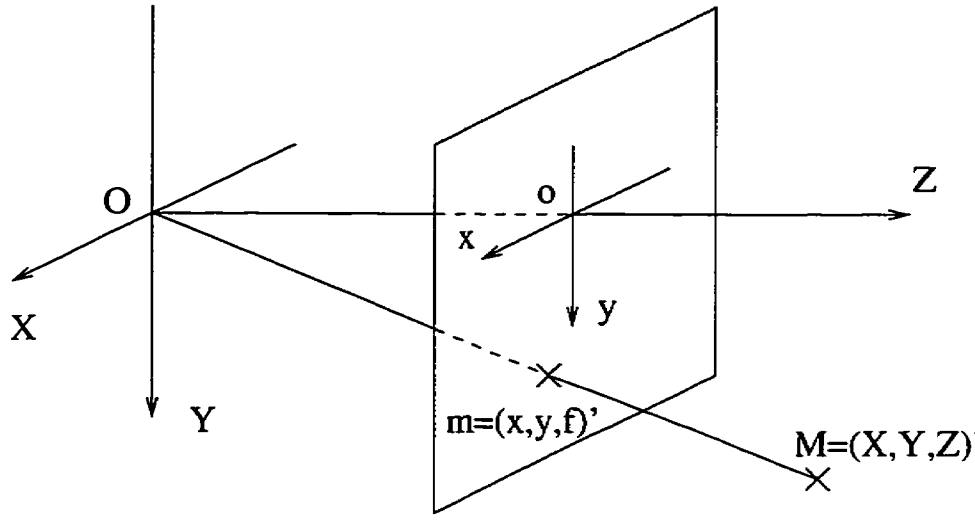


Figure 2.36 : Le modèle du trou d'épingle

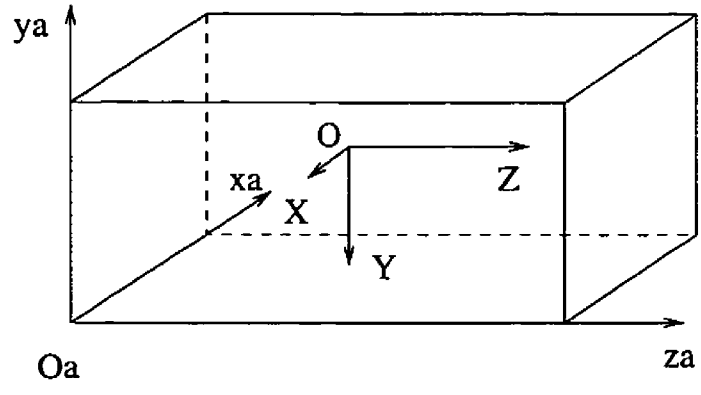


Figure 2.37 : Référentiels choisis dans un corridor idéalisé

$$\begin{cases} R = X\cos\Theta + Z\sin\Theta \\ Y = h \end{cases}$$

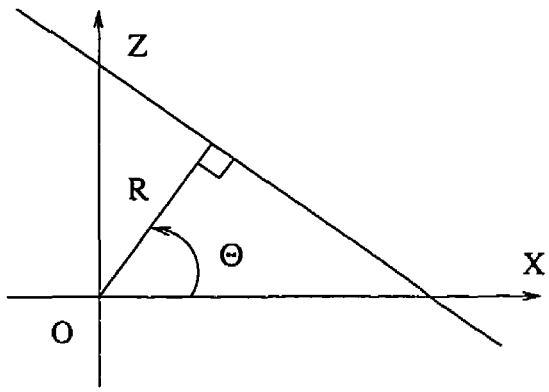


Figure 2.38 : Équation d'une droite dans le repère de la caméra

2.5.2 Corridor connu dans le repère de la caméra

Les bords (B_g) et de (B_d) du corridor connu sont paramétrisés par R_g, R_d et Θ.

En utilisant les équations de la projection perspective, une droite se projette dans l'image en :

$$x + \frac{y}{\frac{-f_y h}{R f_x \cos\Theta}} = -f_x \tan\Theta \tag{2.1}$$

L'équation d'un bord du corridor connu dans l'image s'écrit :

$$x + y/\tan\alpha = a. \tag{2.2}$$

Les paramètres α_g, α_d et a ont été déterminés précédemment pour les bords (B_g) et (B_d) du corridor (voir la figure 2.39).

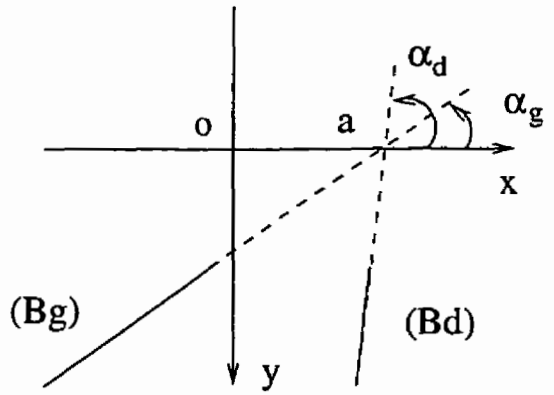


Figure 2.39 : Paramètres des bords du corridor connu dans l'image

En identifiant les équations 2.1 et 2.2, nous obtenons directement les paramètres des équations des bords des corridors dans le repère de la caméra :

$$\begin{aligned} \Theta &= \arctan(-a/f_x) \\ R_g &= \frac{f_y h \sin\Theta}{a \tan\alpha_g} \\ R_d &= \frac{f_y h \sin\Theta}{a \tan\alpha_d} \end{aligned}$$

L'axe du corridor a la même orientation que ses bords, donc même Θ. Son rayon R_a est donné par :

$$R_a = \frac{R_g + R_d}{2}$$

2.5.3 Corridors inconnus dans le repère de la caméra

Nous avons montré qu'un évènement **E** (étiquette "Occlusion" ou "Limite de Visibilité") révélait la présence d'une branche de corridor s'ouvrant sur le corridor principal où se trouve le robot. Par conséquent, nous considérons maintenant les segments du sol étiquetés "Occlusion" et "Limite de Visibilité" pour en déduire les paramètres des corridors correspondants dans le repère de la caméra.

Considérons le cas des occlusions. L'équation du bord du corridor occulté (B_2) dans l'image (voir la figure 2.40) a été déterminée à la section B.3. Nous cherchons l'équation correspondante dans le repère de la caméra :

$$\begin{cases} R_2 = X\cos\Theta + Z\sin\Theta, \\ Y = h. \end{cases}$$

Les équations de la projection perspective nous permettent d'obtenir :

$$\Theta = \arctan(\frac{-r}{f_x\cos\theta_2})$$
$$R_2 = \frac{-hf_y\tan\theta_2\cos\Theta}{f_x}$$

Déterminons la largeur du corridor correspondant à une occlusion. Considérons la figure 2.40.

Soit (B_2) le segment limitant le mur occulté (d'étiquette "Mur Occulté") et (B_1) limitant le mur visible (d'étiquette "Mur Visible"). Soit m l'intersection du segment

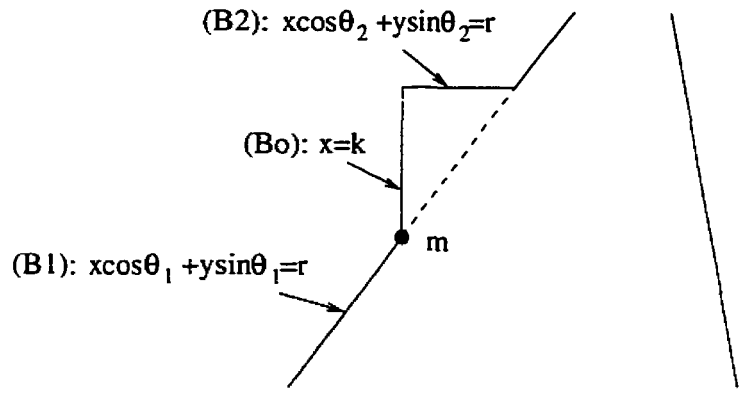


Figure 2.40 : Largeur d'un corridor occlus à partir de l'image

occulté (B_o) avec le bord du corridor (B_1) dans l'image.

$$m = (k, \quad (r_1 - k\cos\theta_1)\sin\theta_1)$$

Les équations de projection perspective permettent de calculer les coordonnées de m dans le repère de la caméra. Soit M le point de la scène correspondant à la projection m dans l'image.

$$M = \left(\frac{khf_y \sin\theta_1}{f_x(r_1 - k\cos\theta_1)}, \quad h, \quad \frac{f_y h \sin\theta_1}{(r_1 - k\cos\theta_1)}\right)$$

La largeur du corridor occulté l est la distance du point M au bord :

$$(B_2) : l = \text{dist}(M, (B_2)).$$

Soit un point N appartenant au bord (B_2) et $\mathbf{n}$ le vecteur normal à (B_2).

$$l = \frac{|\mathbf{MN} \cdot \mathbf{n}|}{|\mathbf{n}|}$$

Soit $N = (0, R_2/\sin\Theta)$ et $\mathbf{n} = (\cos\Theta, \sin\Theta)$ dans le plan (X, Z). Nous obtenons la largeur du corridor occulté l :

$$l = \left| \frac{-hf_y \sin\theta_1 (k\cos\Theta/f_x + \sin\Theta)}{(r1 - k\cos\theta_1)} + R_2 \right|.$$

À partir de la largeur du couloir et de l'équation du bord (B_2), nous pouvons calculer l'équation de l'axe du corridor (B_1) dans le repère de la caméra :

$$R_1 = X\cos\Theta + Z\sin\Theta$$

Il s'agit de déterminer R_1. Nous avons la relation :

$$l = \mid R_2 - R_1 \mid .$$

Nous savons aussi que la profondeur en Z du mur occulté (Mur_2) est plus grande que la profondeur en Z du mur visible (Mur_1). Nous en déduisons que :

$$\begin{aligned} &\text{Si } R_2 > 0, \quad R_1 = R_2 - l/2, \\ &\text{sinon,} \qquad\quad R_1 = R_2 + l/2. \end{aligned}$$

L'axe du corridor en découle directement :

$$R_a = X\cos\Theta + Z\sin\Theta,$$

$$R_a = (R_1 + R_2)/2.$$

Considérons maintenant le cas des limites de visibilité. Si un segment d'étiquette "Limite de Visibilité" est adjacent à un segment d'étiquette "Occlusion", il est préférable de travailler sur l'image suivante dans la séquence, de manière à ce que la partie de la scène occulté devienne visible. Dans le cas où un segment d'étiquette "Limite de Visibilité" est entouré de deux segments d'étiquette "Mur Visible", nous pouvons conclure que le corridor correspondant à cet évènement a le même axe et la même largeur que le corridor dans lequel se trouve la caméra. Si tel n'était pas le cas en effet, nous aurions nécessairement une occlusion.

2.5.4 Centre de l'intersection dans le repère de la caméra

Rappelons que le *domaine de visibilité*, V, d'une intersection est la région formée par l'intersection des rubans au lieu de rencontre de deux corridors ou plus. La détermination de V revient à résoudre un système d'équations représentant les bords des corridors. Nous discrétisons le plan du sol dans l'environnement et le représentons

par une grille d'occupation $O(i,j)$ $n \times n$ dont les valeurs sont :

$$\begin{cases} 0 & \text{si } (i,j) \text{ n'est pas occupée ;} \\ v & \text{sinon.} \end{cases}$$

Chaque bord de corridor divise le plan du sol en deux demi-plans (voir la figure 2.41) dont l'un a pour valeur 0 (non occupé) et l'autre pour valeur 1 (occupé par le corridor). Pour déterminer quel est le demi-plan occupé, nous calculons la normale au bord du corridor. La normale du bord du corridor est toujours orientée vers le centre du corridor (vers l'axe).

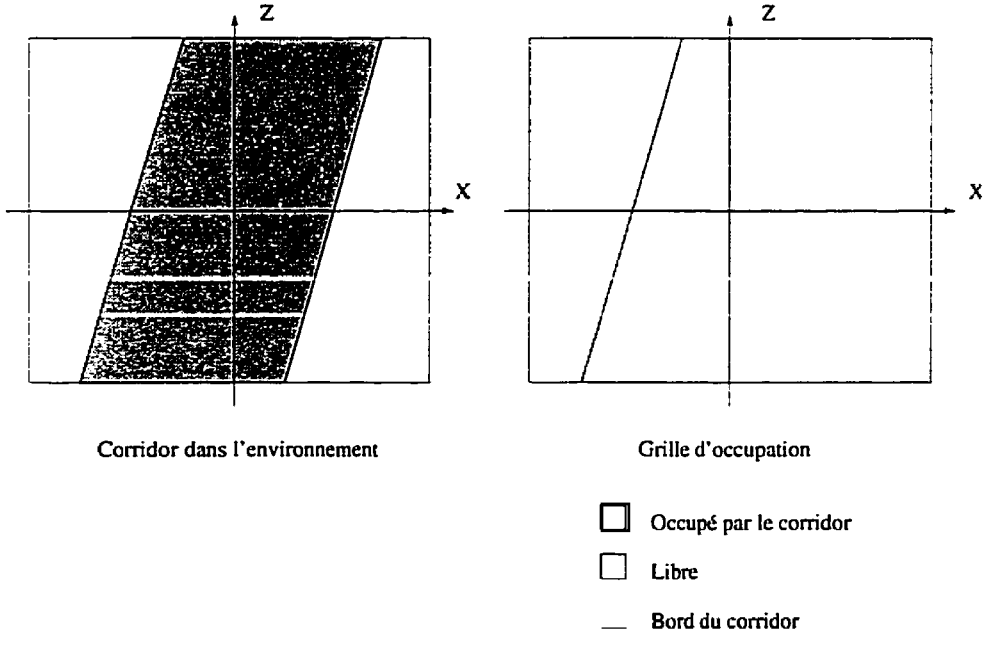


Figure 2.41 : Grille d'occupation d'un corridor

Nous calculons la normale **HA** d'un bord du corridor sachant que :

$$\mathbf{HA} \cdot \mathbf{V} = 0,$$

avec $\mathbf{V} = (-\sin\Theta, \cos\Theta)$ le vecteur directeur de l'axe du corridor, H appartenant au bord du corridor B_i et donc vérifiant son équation et A appartenant à l'axe du corridor (voir la figure 2.42).

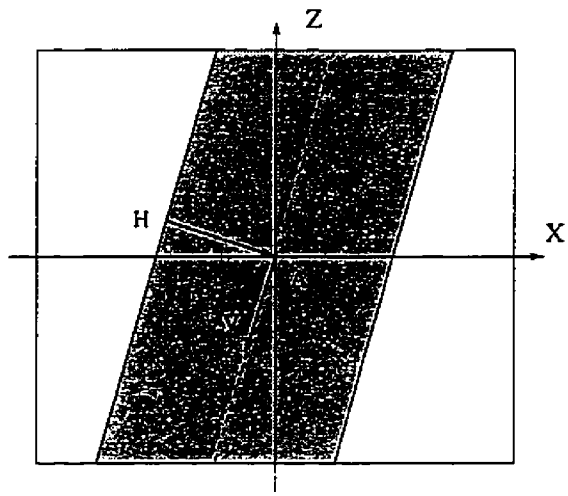


Figure 2.42 : Normale d'un bord de corridor

Nous obtenons :

$$\mathbf{HA} = \begin{pmatrix} (R_a - R_i)\cos\Theta \\ (R_a - R_i)\cos\Theta \end{pmatrix}$$

Nous utilisons la même grille d'occupation pour tous les corridors composant l'intersection. Nous ajoutons une valeur de 1 aux cellules correspondant aux demi-plans engendrés par les bords des corridors. Chaque corridor générant deux demi-plans,

le domaine de visibilité V de l'intersection est la région composée des cellules ayant pour valeur :

$$v = 2 \times \text{nombre de corridors.}$$

Le moment d'ordre jk de la surface définie par le domaine de visibilité est donné par :

$$M_{jk} = \sum_{l=1}^{n} \sum_{m=1}^{n} (x_l)^j (y_m)^k.$$

M_{00} donne l'aire de V c'est-à-dire le nombre de cellules de valeur v. Soit A cette surface. Le moment d'ordre jk s'écrit alors :

$$M_{jk} = \sum_{A} (x)^j (y)^k$$

Pour déterminer le centre de l'intersection P, nous calculons le barycentre de V :

$$\begin{cases} x_P = M_{10}/M_{00}, \\ y_P = M_{01}/M_{00}, \end{cases}$$

ou encore

$$\begin{cases} x_P = \dfrac{1}{A} \displaystyle\sum_{A} x, \\ y_P = \dfrac{1}{A} \displaystyle\sum_{A} y. \end{cases}$$

2.5.5 Corridors dans le repère de l'intersection

Nous avons déterminé les axes des corridors et le centre de l'intersection dans le repère de la caméra. Il nous reste à établir le modèle dans le repère de l'intersection.

Pour passer du repère de la caméra au repère de l'intersection, il faut effectuer les transformations suivantes (voir la figure 2.43) :

- une translation de vecteur **CP**, avec C origine du repère de la caméra et P centre de l'intersection dans ce repère
- une rotation d'angle $(Z, \mathbf{V})$, avec $\mathbf{V}$ vecteur directeur de l'axe du corridor principal où est située la caméra.

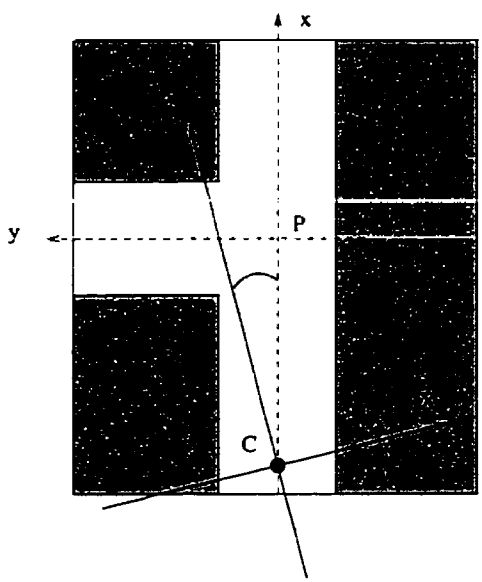


Figure 2.43 : Passage du repère de la caméra au repère de l'intersection

Il nous reste encore à déterminer le sens des axes des corridors dans le repère de l'intersection. Considérons la figure 2.44.

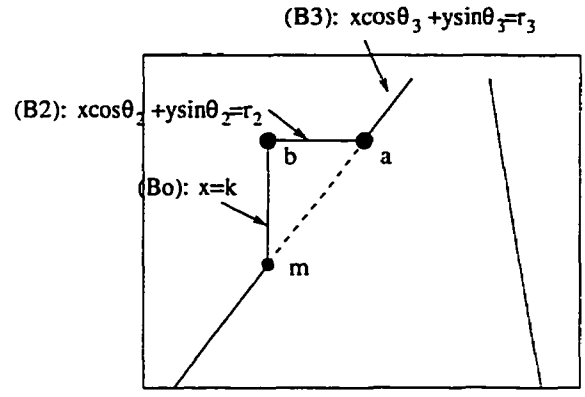


Figure 2.44 : Détermination du sens des axes à partir de l'image

Soit b le point intersection entre le segment d'étiquette "Mur Occulté"et le segment d'étiquette "Occlusion" et a le point intersection entre le segment d'étiquette "Mur Occulté"et le segment d'étiquette "Mur Visible" lui faisant suite dans l'image. L'axe d'un corridor occulté est dans le sens du vecteur **ab**. Nous obtenons les points correspondants dans le repère de la caméra en effectuant la projection perspective inverse, puis dans le repère de l'intersection par les transformations décrites plus haut. Soit $\mathbf{V}$ le vecteur directeur de l'axe du corridor considéré. Le sens de l'axe du corridor est :

$$\begin{cases} +1 & \text{si } \mathbf{V} \cdot \mathbf{AB} > 0 \\ -1 & \text{sinon} \end{cases}$$

Selon les conventions adoptées jusqu'ici, le sens du corridor principal est positif et

le sens du corridor éventuel prolongeant le corridor principal (d'étiquette "Limite de Visibilité" entouré d'étiquettes "Mur Visible") est négatif.

2.6 Expérimentations

Nous avons implanté notre algorithme de construction du modèle topologique à partir du contour du sol segmenté et étiqueté dans l'image. Nous présentons ici les résultats de nos expérimentations.

Nous travaillons sur des images de synthèse représentant des corridors dans un environnement minier. Ces images ont été générées par Philippe Debanné avec Open Inventor (Wernecke 1994). Nous connaissons la topographie de la mine, ce qui nous permet de vérifier nos résultats. Nous avons testé notre algorithme sur différentes images pour vérifier que le système est capable de construire le modèle topologique de l'intersection quelles que soient la position et l'orientation de la caméra, et la configuration de l'intersection. En particulier, les cas suivants ont été testés :

- La caméra est placée au milieu du corridor dans lequel elle se trouve et son axe optique est parallèle à l'axe du corridor.
- L'axe optique de la caméra est parallèle à l'axe du corridor dans lequel elle se trouve mais le robot n'est pas au centre du corridor.
- La caméra n'est pas au centre du corridor et son axe optique n'est pas parallèle à l'axe du corridor.

- Les axes de l'intersection se coupent en un point unique qui est le centre de l'intersection.

- Les axes de l'intersection ne se coupent pas en un point unique. Le problème est de déterminer correctement le centre de l'intersection, c'est-à-dire le centroïde de la région **R** décrite plus haut.

La figure 2.45 représente une image de dimension 400 × 300 avec le repère de l'image et les paramètres que nous utilisons pour les expérimentations.

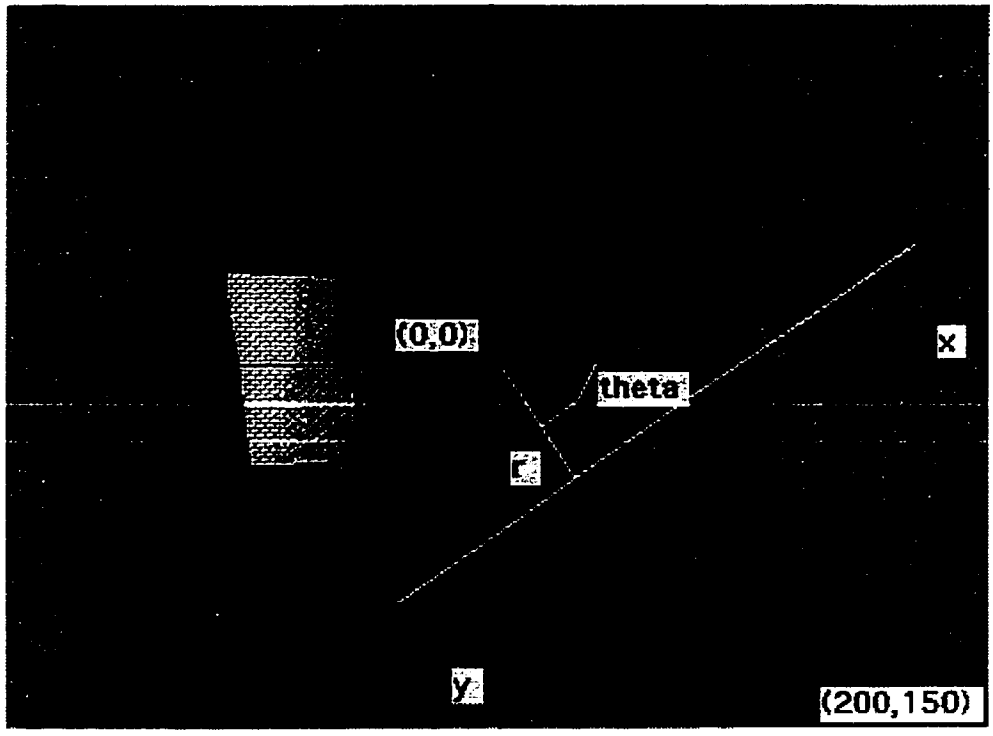


Figure 2.45 : Repère de l'image et paramètres utilisés

Pour tous les cas de figure testés, nous montrons :

- La topographie de la scène avec la position et l'orientation de la caméra dans

la scène. Ces données ne sont pas fournies au système. Elles permettent simplement de vérifier les résultats obtenus.

- Les données d'entrée du système : le contour du sol segmenté (paramètres (r, θ)) et étiqueté dans l'image synthétisée. La liste des segments de droite avec leurs étiquettes est la seule entrée du système.

- Le modèle topologique obtenu : les axes des corridors (r, θ), leur largeur et leur sens dans le repère de l'intersection.

- La représentation tridimensionnelle de l'intersection : cette représentation est obtenue à partir du modèle topologique calculé par le système. Nous verrons dans le chapitre 3 comment nous construisons le modèle tridimensionnel de l'intersection à partir du modèle topologique. Pour le moment, cette représentation facilite la compréhension des résultats.

Nous montrons tout d'abord les résultats obtenus sur une intersection en T. La caméra est placée au centre du corridor et son axe optique est parallèle à l'axe du corridor (voir les figures 2.46 et 2.47).

Les résultats montrent que le système construit correctement la topologie de l'intersection (voir le tableau 2.1). Les angles entre les axes des corridors sont corrects ainsi que leurs largeurs. La représentation tridimensionnelle prouve que le modèle fonctionne (voir la figure 2.48).

Nous testons la même intersection avec des positions et des orientations de la

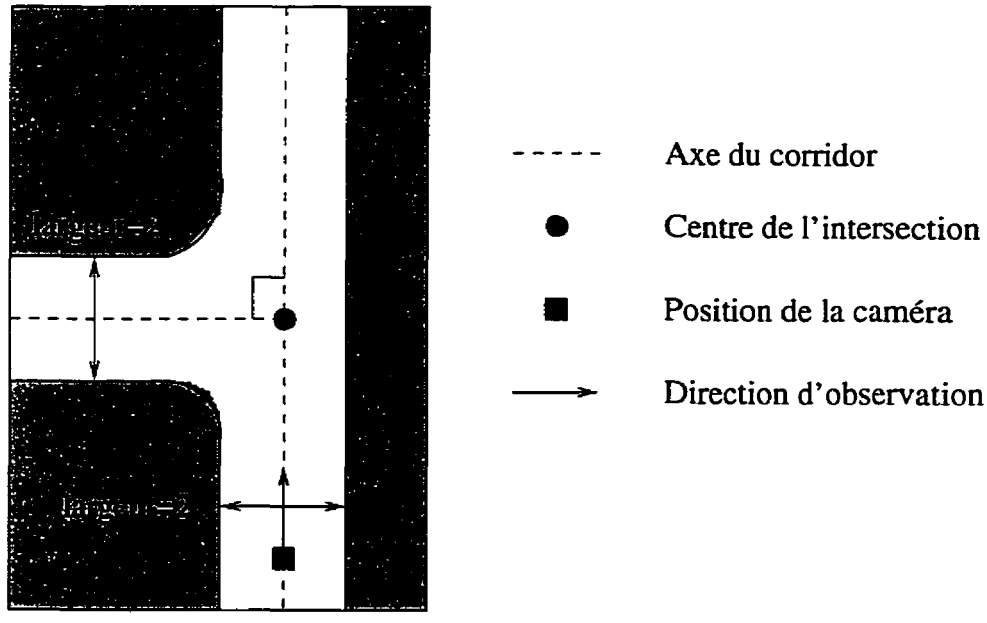


Figure 2.46 : Intersection en T : la caméra est au centre du corridor et son axe optique est parallèle à l'axe du corridor

Tableau 2.1 : Modèle topologique de l'intersection en T calculé par le système

	r	θ (rad)	largeur	sens
corridor 1	0	1,57	2	-1
corridor 2	0,47	0	2	1
corridor 3	0	1,57	2	1

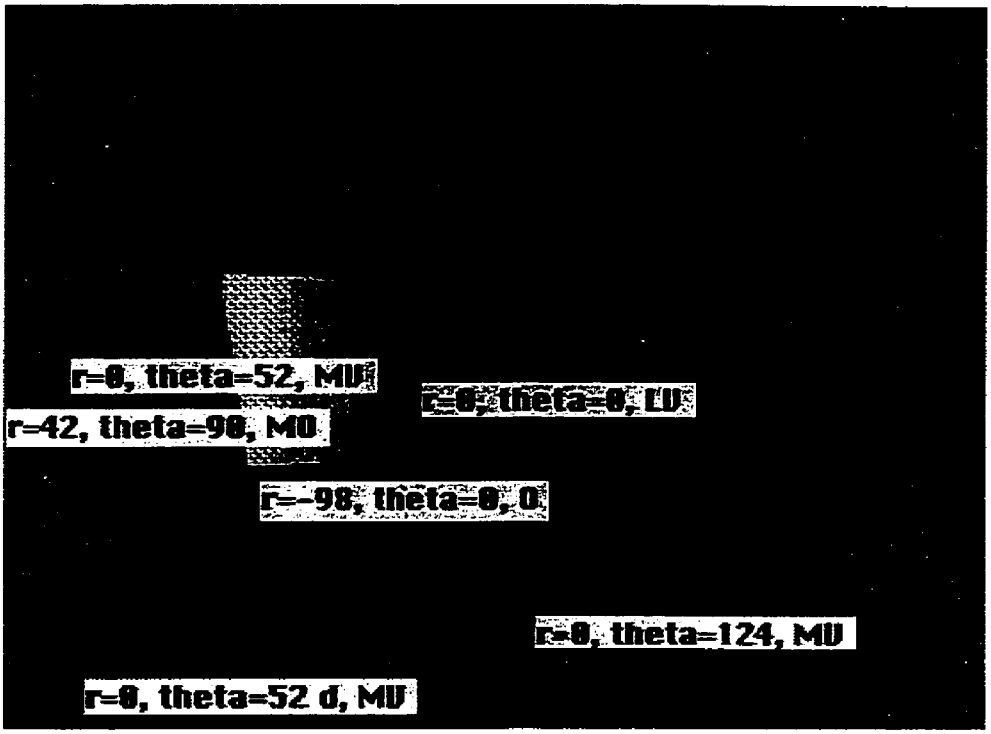


Figure 2.47 : Image de l'intersection en T et contour du sol étiqueté

caméra différentes (voir les figures 2.49, 2.50, 2.52, 2.53, 2.55, 2.56). Nous montrons les modèles topologiques (tableaux 2.2, 2.3, 2.4) et leurs représentions tridimensionnelles obtenues (figures 2.51, 2.54, 2.57). Les paramètres des axes des corridors ont des valeurs différentes d'un modèle à l'autre car le repère de l'intersection est différent dans chaque cas. En effet, l'axe des abscisses est choisi dans la direction d'observation du robot. Mais la représentation finale est la même, ce qui est cohérent avec notre modèle. Le choix du repère de l'intersection n'a aucune influence sur la reconnaissance de l'intersection. D'une visite à l'autre, le robot affine le modèle de l'intersection. Il doit évidemment choisir un repère unique, qui peut par exemple être celui établi lors

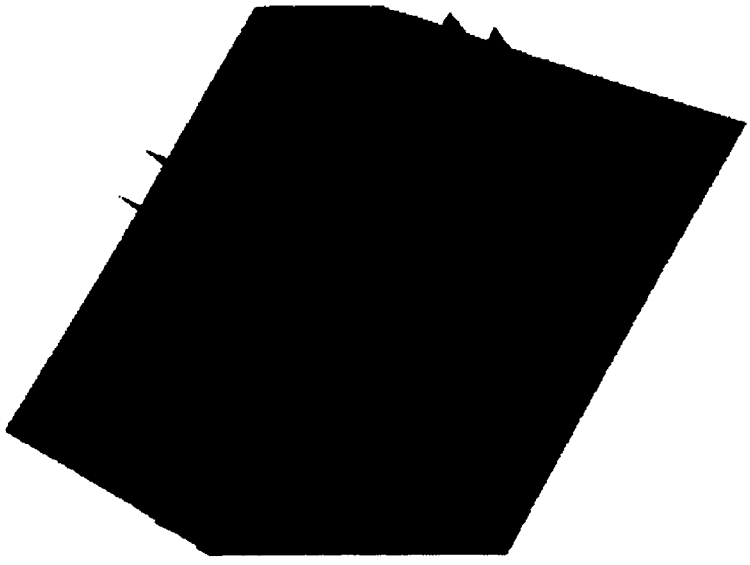


Figure 2.48 : Modèle tridimensionnel de l'intersection en T construit à partir du modèle topologique

de son premier passage par l'intersection. L'important n'est pas tant le repère choisi que la position du centre de l'intersection, les angles entre les axes des corridors et leurs largeurs. Ensuite, par simple rotation du repère de l'intersection, nous pouvons unifier deux représentations.

Tableau 2.2 : Modèle topologique pour l'intersection 2.49

	r	θ (rad)	largeur	sens
corridor 1	0	1,57	2	-1
corridor 2	0,06	3,14	2	-1
corridor 3		1,57	2	1

Nous avons testé d'autres intersections pour valider ce premier ensemble de tests (voir les figures 2.58, 2.59, 2.61, 2.62, 2.64, 2.65, 2.67, 2.68). Les résultats obtenus (voir les figures 2.5, 2.60, 2.6, 2.63, 2.7, 2.66, 2.8, 2.69) montrent le bon fonctionnement de

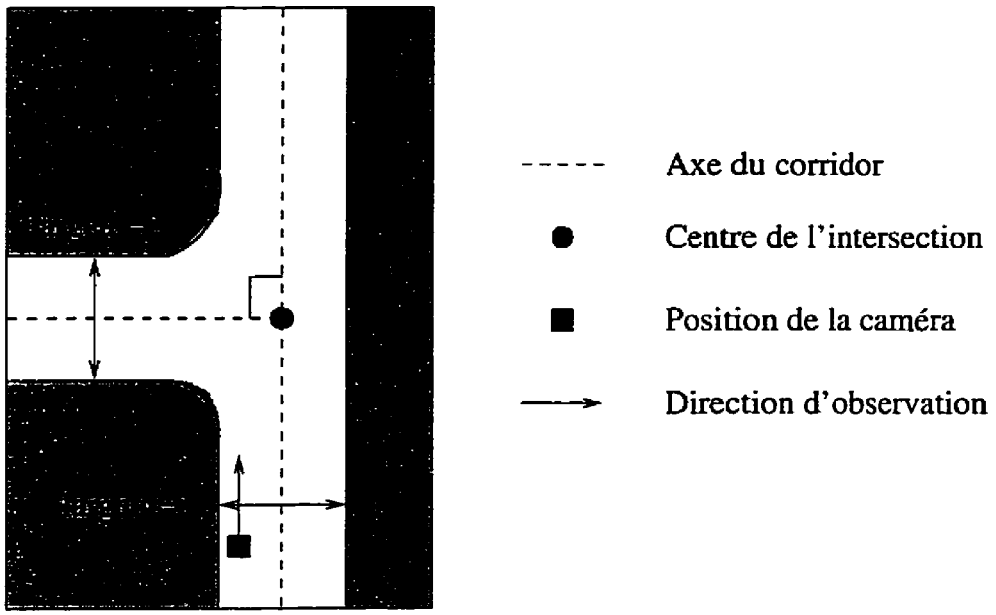


Figure 2.49 : Intersection en T : la caméra est décentrée dans le corridor

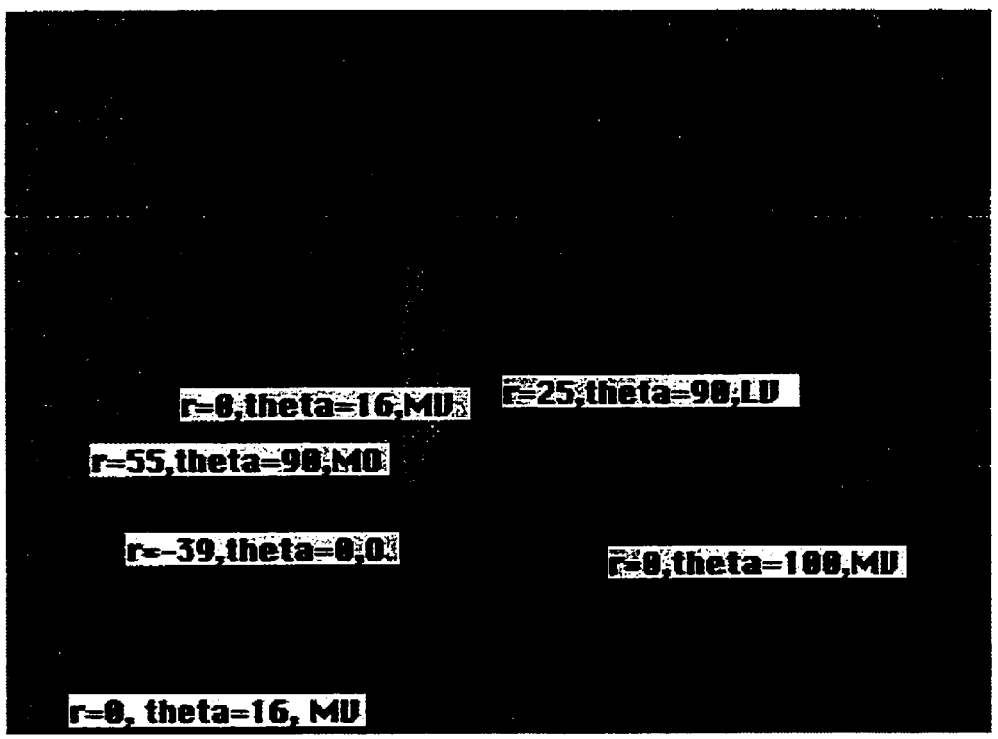


Figure 2.50 : Image de l'intersection en T 2.49 et contour du sol étiqueté

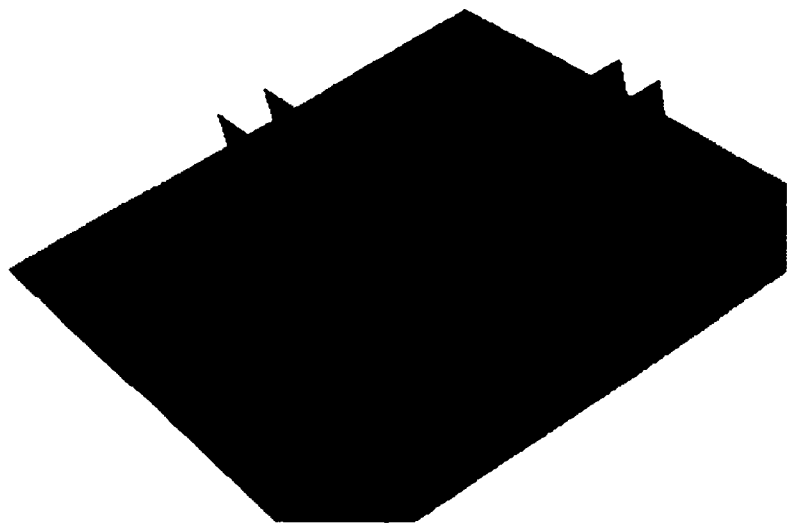


Figure 2.51 : Modèle tridimensionnel de l'intersection 2.49 construit à partir du modèle topologique

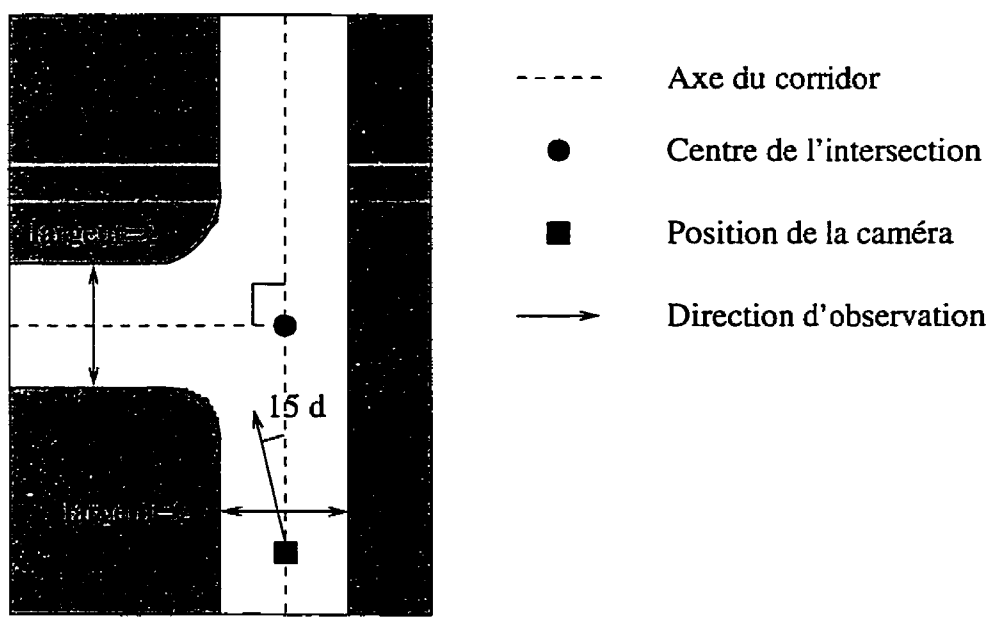


Figure 2.52 : Intersection en T : l'angle entre l'axe optique de la caméra et l'axe du corridor est de 15°

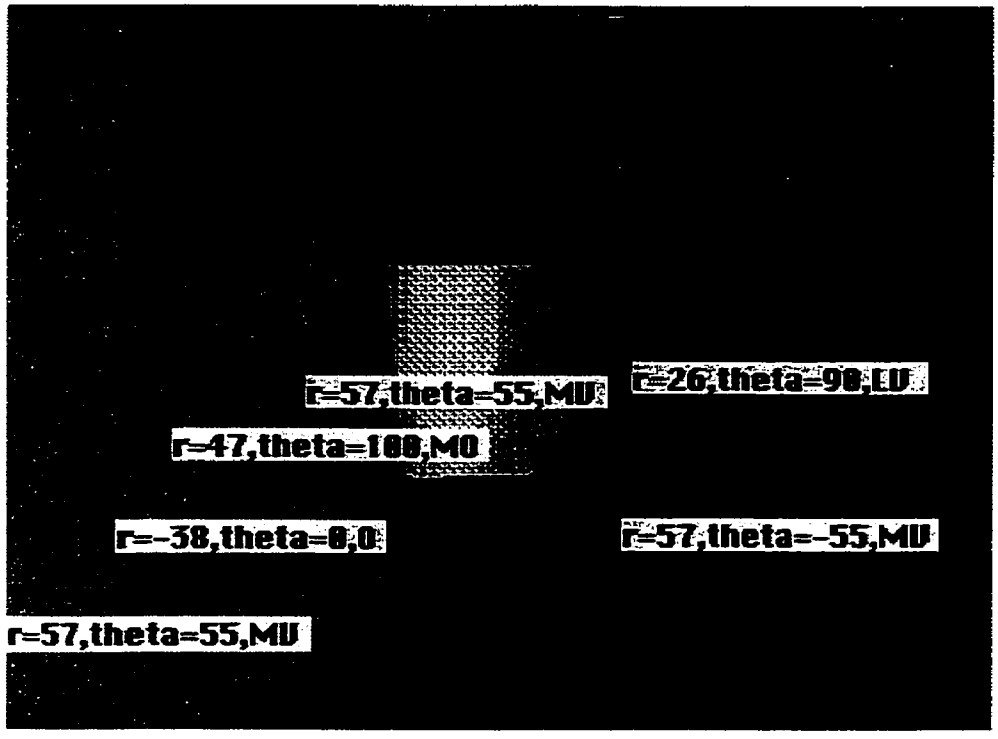


Figure 2.53 : Image de l'intersection en T 2.52 et contour du sol étiqueté

Tableau 2.3 : Modèle topologique de l'intersection 2.52

	r	θ (rad)	largeur	sens
corridor 1	0,24	1,05	2	-1
corridor 2	0,5	-0,57	2	1
corridor 3	0,24	1,05	2	1

Tableau 2.4 : Modèle topologique de l'intersection 2.55

	r	θ (rad)	largeur	sens
corridor 1	0,12	-1,06	2	1
corridor 2	0,29	0,54	2	1
corridor 3	0,12	-1,06	2	-1

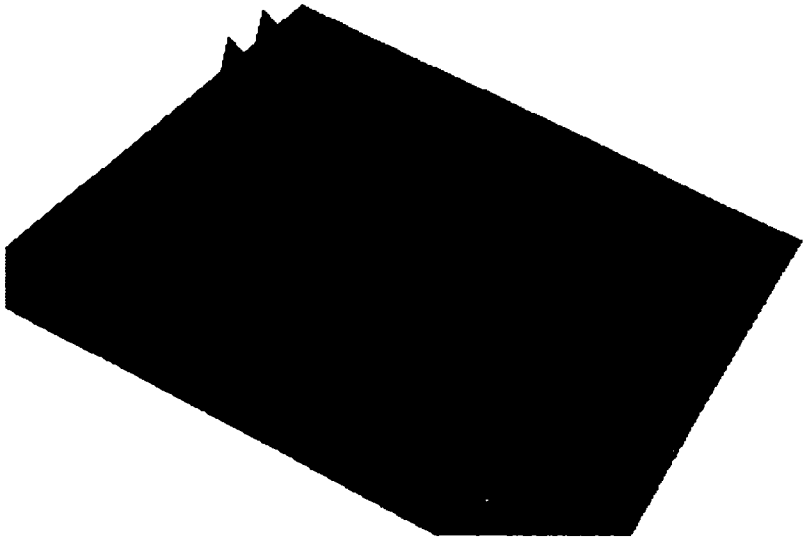


Figure 2.54 : Modèle tridimensionnel de l'intersection 2.52 construit à partir du modèle topologique

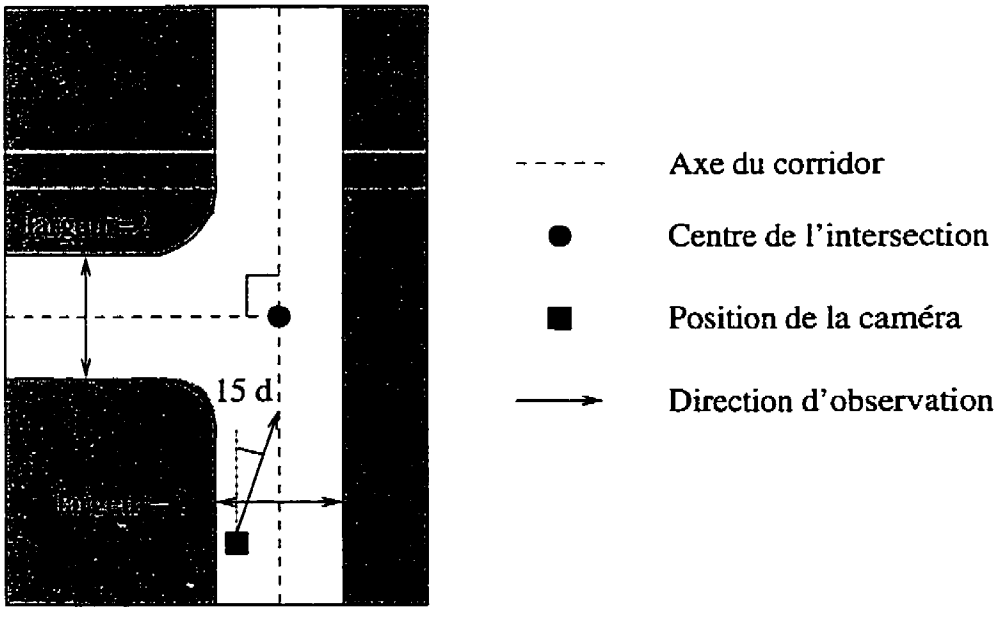


Figure 2.55 : Intersection en T : la caméra est décentrée dans le corridor et l'angle entre l'axe optique de la caméra et l'axe du corridor est de 15°

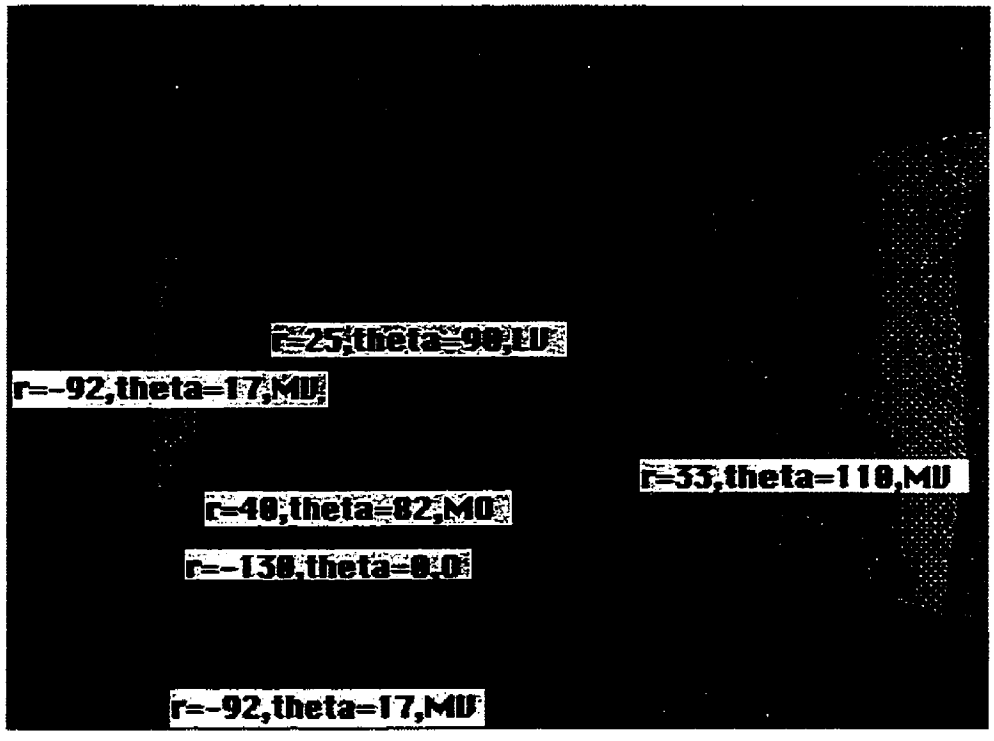


Figure 2.56 : Image de l'intersection 2.55 et contour du sol étiqueté

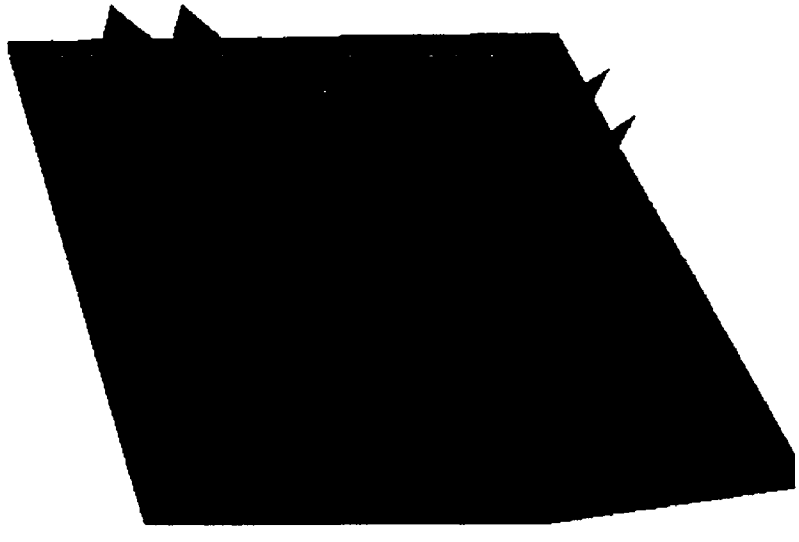


Figure 2.57 : Modèle tridimensionnel de l'intersection 2.55 construit à partir du modèle topologique

l'algorithme.

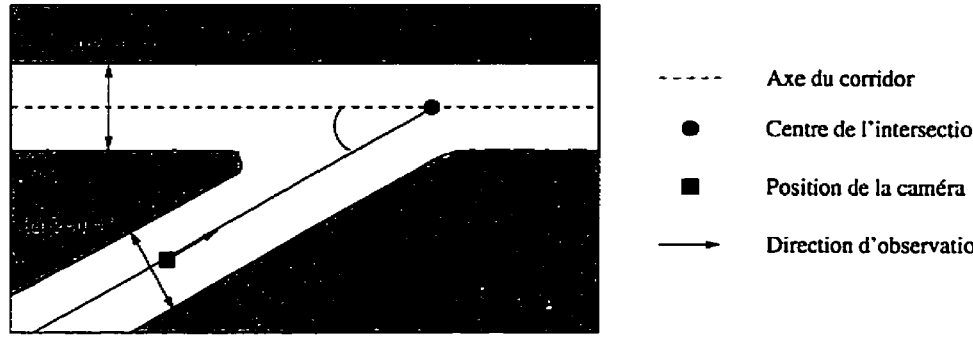


Figure 2.58 : Intersection en λ

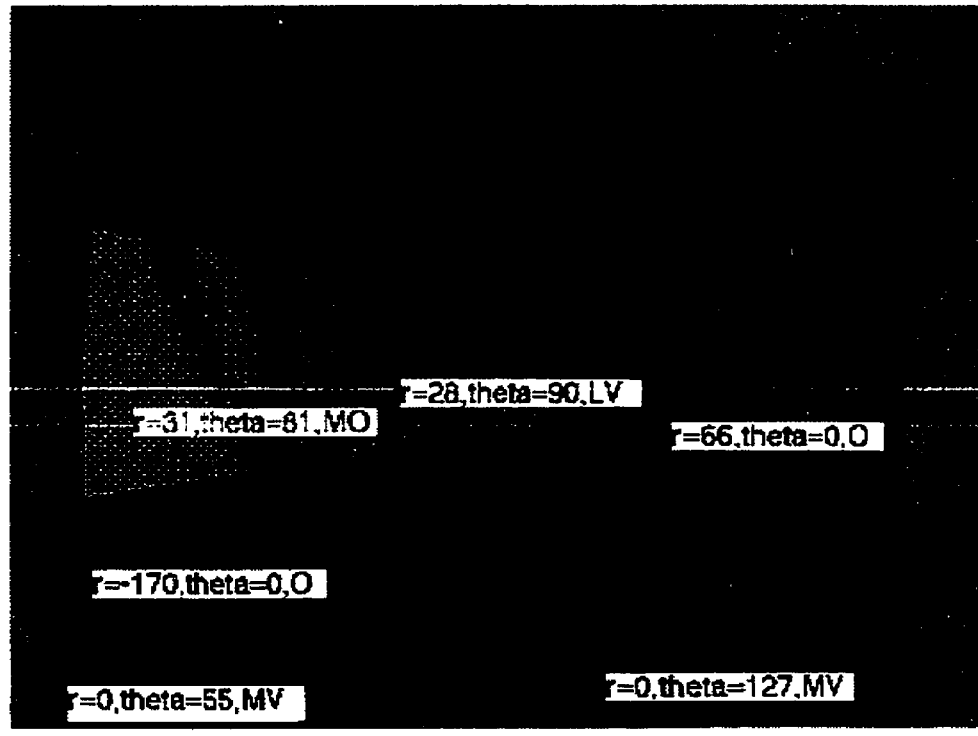


Figure 2.59 : Image de l'intersection 2.58 et contour du sol étiqueté

L'image 2.58, ainsi que les images 2.61 et 2.64, comporte une limite de visibilité consécutivement à une occlusion qui est due à la synthèse de l'image. En effet,

l'intersection est limitée dans l'espace tridimensionnel. Autrement dit, les murs sont "coupés". Dans un cas réel, cette limite de visibilité ne serait probablement pas présente. Les segments appartenant aux murs seraient prolongés. Nous tenons compte de cette complication dans notre système pour modéliser correctement l'intersection. Avec des images réelles, le système serait adapté pour que dans le cas très particulier (et improbable) où une limite de visibilité soit consécutive à une occlusion, l'image ne soit pas traitée. Le système traiterait l'image suivante dans la séquence où la limite de visibilité aurait disparu comme nous l'avons expliqué précédemment.

Tableau 2.5 : Modèle topologique de l'intersection 2.58

	r	θ (rad)	largeur	sens
corridor 1	0,02	3,86	2	1
corridor 2	0,02	3,86	2	-1
corridor 3	0	1,57	2	1

Pour l'intersection 2.58, l'axe du corridor 3 fait un angle de 48° avec l'axe du corridor 1 au lieu des 30° attendus. En revanche, l'angle entre les corridors 1 et 2 est exact.

Tableau 2.6 : Modèle topologique de l'intersection 2.61

	r	θ (rad)	largeur	sens
corridor 1	0,01	-0,68	2	-1
corridor 2	0,01	-0,68	2	1
corridor 3	0	1,57	2	1

Les résultats obtenus avec l'intersection 2.61 confirment ceux obtenus pour l'in-

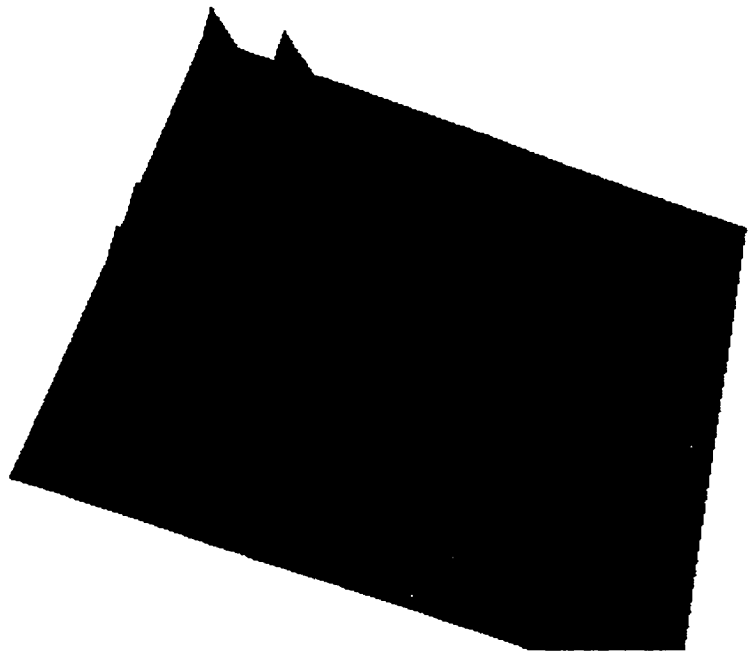


Figure 2.60 : Modèle tridimensionnel de l'intersection 2.58 construit à partir du modèle topologique

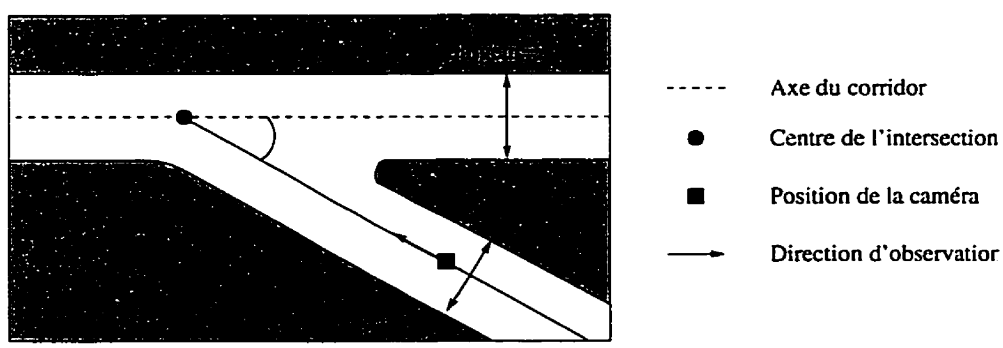


Figure 2.61 : Autre intersection en λ

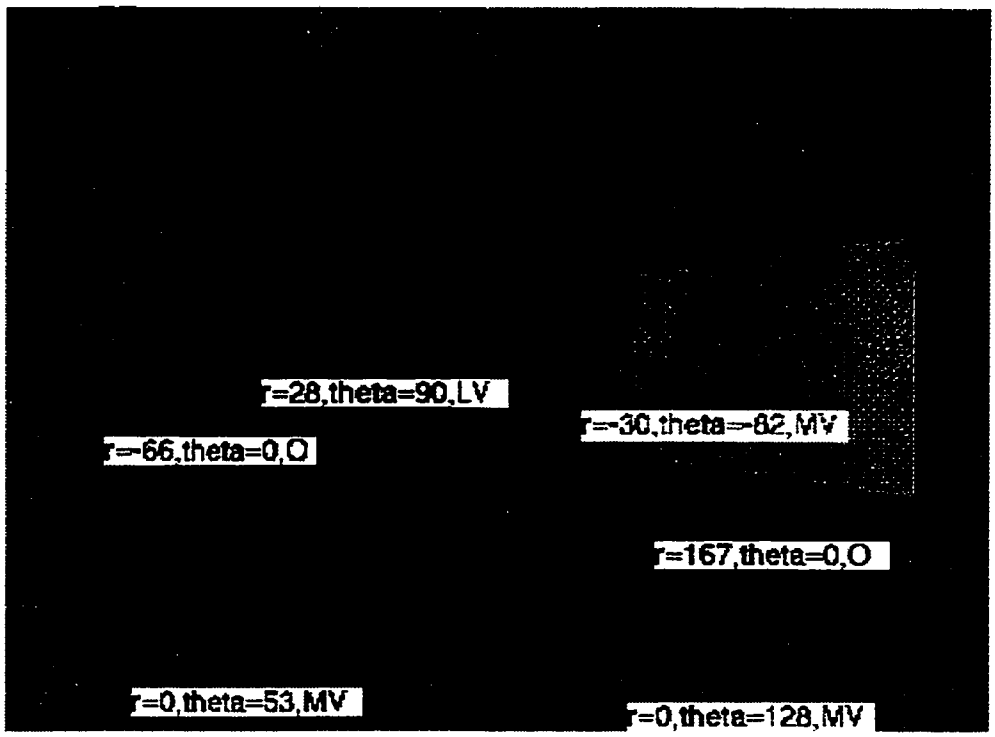


Figure 2.62 : Image de l'intersection 2.61 et contour du sol étiqueté

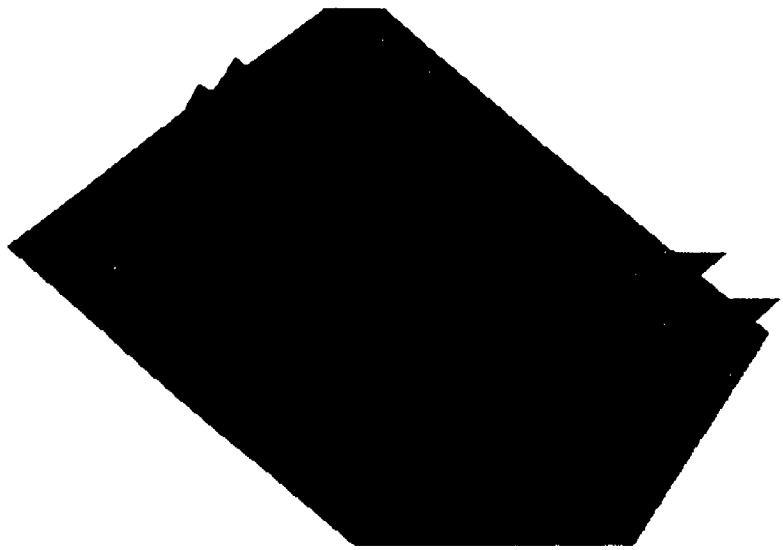


Figure 2.63 : Modèle tridimensionnel de l'intersection 2.61 construit à partir du modèle topologique

tersection 2.58, ce qui est rassurant car ces intersections sont symétriques. L'angle entre les axes des corridors 1 et 2 est juste. Par contre, l'axe du corridor 3 fait un angle de 49° avec l'axe du corridor 1 au lieu des 30° attendus.

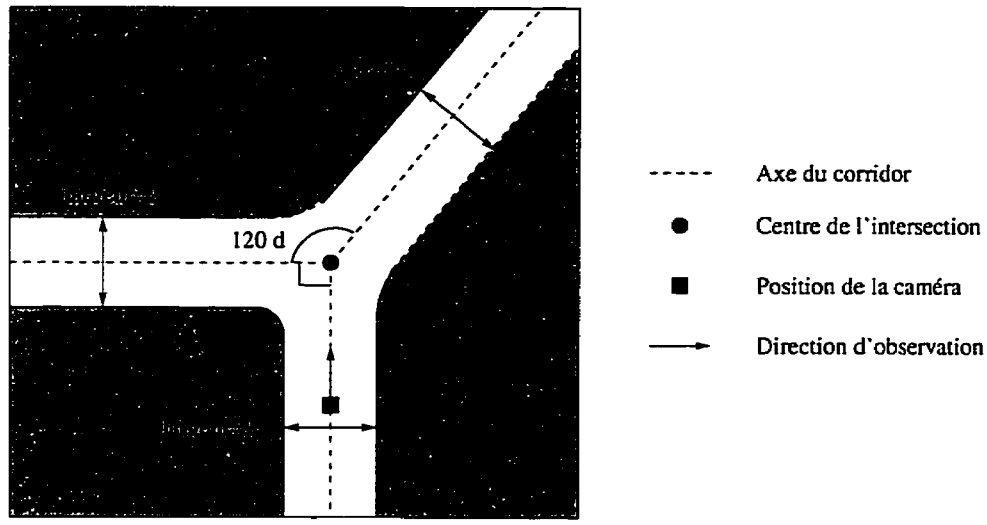


Figure 2.64 : Intersection quelconque

Tableau 2.7 : Modèle topologique de l'intersection 2.64

	r	θ (rad)	largeur	sens
corridor 1	0,40	0,63	2	-1
corridor 2	0,17	0	2	1
corridor 3	0,5	4,71	2	-1

Les résultats obtenus avec l'intersection 2.64 sont très satisfaisants. L'angle entre les corridors 2 et 3 est déterminé sans erreur (90°). Et nous obtenons une petite erreur de 6° sur l'angle entre les axes des corridors 1 et 2.

Pour l'intersection 2.67, nous constatons que nous obtenons effectivement un modèle symétrique. Cependant, l'erreur sur l'angle entre les corridors 1 et 3 d'une part

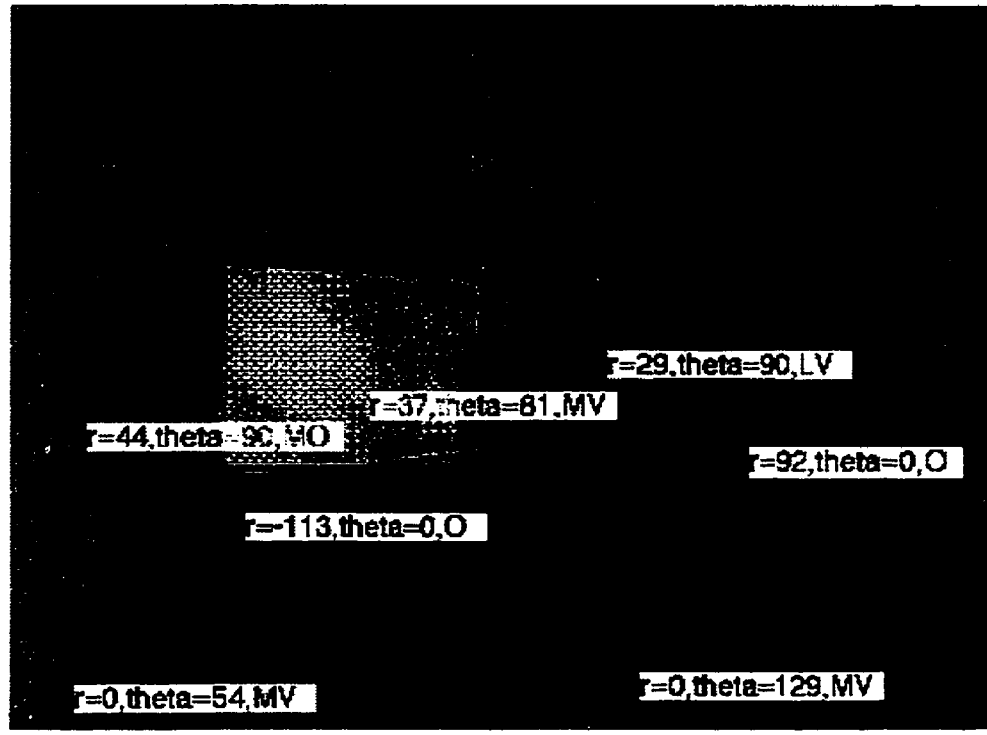


Figure 2.65 : Image de l'intersection 2.64 et contour du sol étiqueté

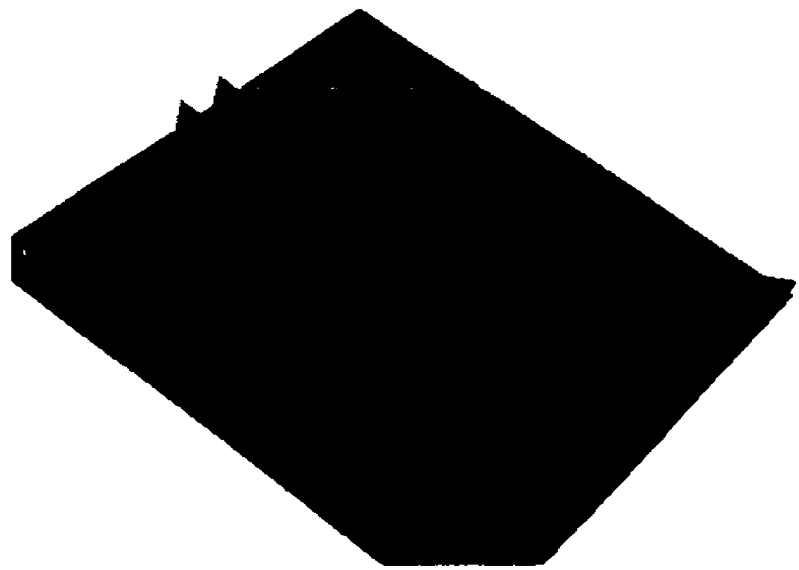


Figure 2.66 : Modèle tridimensionnel de l'intersection 2.64 construit à partir du modèle topologique

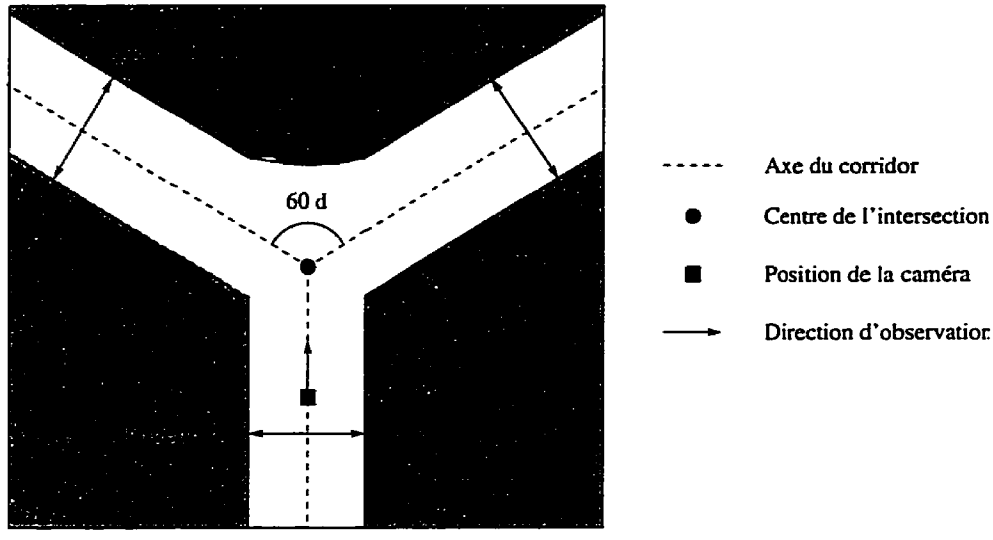


Figure 2.67 : Intersection en Y

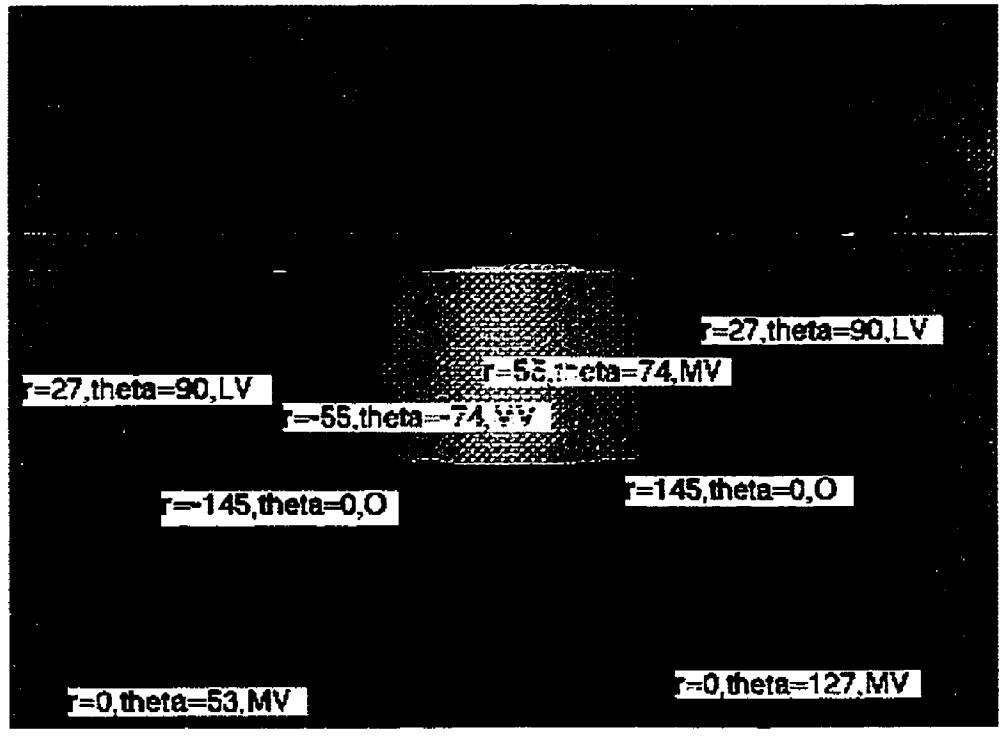


Figure 2.68 : Image de l'intersection 2.67 et contour du sol étiqueté

Tableau 2.8 : Modèle topologique de l'intersection 2.67

	r	θ (rad)	largeur	sens
corridor 1	0,19	3,85	2	1
corridor 2	0,19	2,43	2	-1
corridor 3	0	1,57	2	1

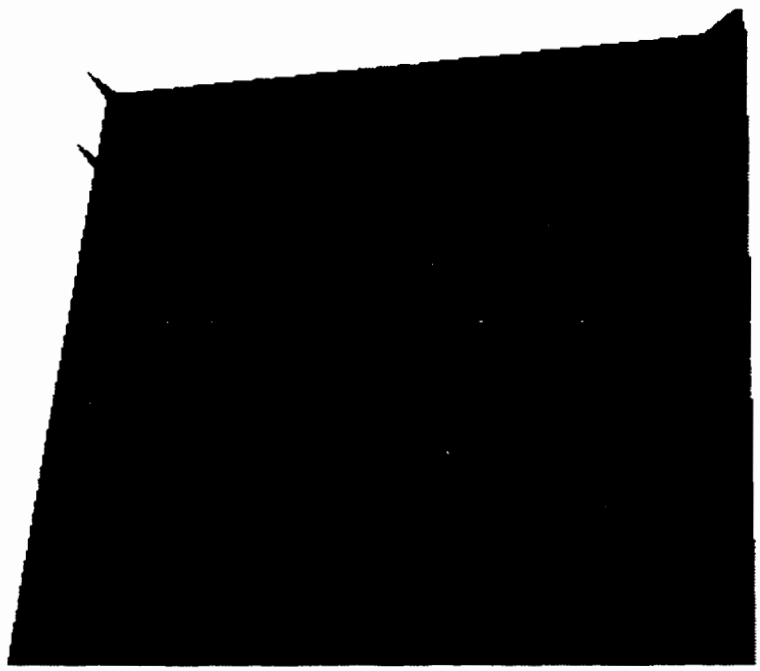


Figure 2.69 : Modèle tridimensionnel de l'intersection 2.67 construit à partir du modèle topologique

et l'angle entre les corridors 2 et 3 d'autre part est de 10°.

Les résultats montrent que le système est capable de modéliser correctement les intersections rencontrées à partir du contour du sol segmenté et étiqueté et quelles que soient la position et l'orientation de la caméra. Les tests ont été effectués sur une seule image de la séquence pour une intersection donnée. L'utilisation de plusieurs images permettrait de corriger les erreurs obtenues. Nous verrons dans le chapitre 3 l'influence de ces erreurs pour la reconnaissance.

Chapitre 3

Reconnaissance d'Intersections

3.1 Introduction

Le modèle topologique que nous venons de décrire est à lui tout seul insuffisant pour identifier une intersection de façon unique. En effet, deux intersections peuvent avoir la même topologie ou une topologie très semblable qui les rende indifférentiables à l'analyse, surtout compte-tenu du bruit et des erreurs de mesure. Nous avons donc besoin d'une représentation qui prenne en compte la spécificité de chaque intersection. Cette information est contenue dans l'apparence de l'environnement c'est-à-dire dans la texture des corridors et est rendue par les images obtenues.

Une solution serait de stocker toutes les images prises par le robot et de mettre en œuvre des techniques de mise en correspondance pour identifier des images. Mais cette solution soulève de gros problèmes. Tout d'abord, l'image d'une scène dépend

beaucoup du point de vue. Des erreurs commises sur l'estimation de la trajectoire du robot influencent beaucoup l'image obtenue. Il faudrait pouvoir s'assurer que, lors de son passage à une intersection, le robot suit la même trajectoire que lors de son passage précédent. De plus, les conditions d'illumination sont un facteur non négligeable. Enfin, la quantité d'information à stocker et à traiter complique sérieusement le processus.

Cela nous amène à l'idée de prendre des images en un point fixe de la scène plutôt que le long d'une trajectoire. Le point le plus intéressant est évidemment le centre de l'intersection. C'est à la fois le point à partir duquel le robot est confronté au minimum d'occlusions et le point de la scène le plus facile à déterminer de façon unique comme nous l'avons vu au chapitre 2. A partir de ce point, le maximum d'informations disponibles, se trouverait dans une séquence d'images représentant l'intersection vue dans toutes les directions (voir la figure 3.1).

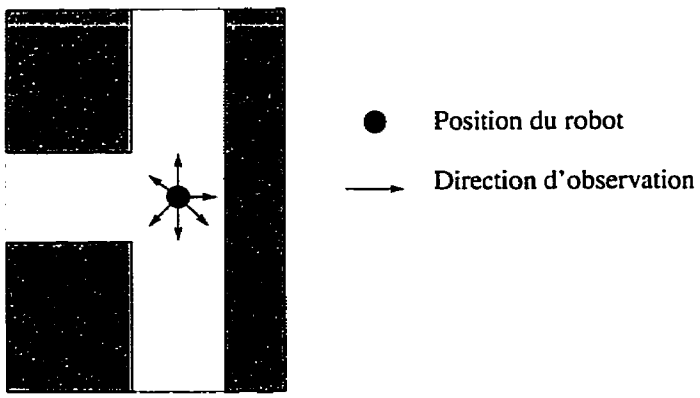


Figure 3.1 : Images panoramiques d'une intersection

Le maximum d'informations non redondantes est obtenu pour des images qui ne

se chevauchent pas. En mettant les images bout à bout, nous obtenons une vue panoramique de l'intersection.

Nous choisissons donc de représenter une intersection par une vue panoramique sur 360 degrés prise à partir du centre de l'intersection.

Le problème consiste donc à mémoriser les intersections que le robot rencontre et à reconnaître celles qu'il a précédemment rencontrées. Les réseaux de neurones sont bien adaptés à ce type de problème qui fait appel aux mécanismes d'apprentissage et de rappel. La machine neuronale doit réaliser une association entre des représentations sur 360° des intersections (en entrée) et des identificateurs correspondant aux intersections rencontrées (en sortie). Le système est décrit sur la figure 3.2.

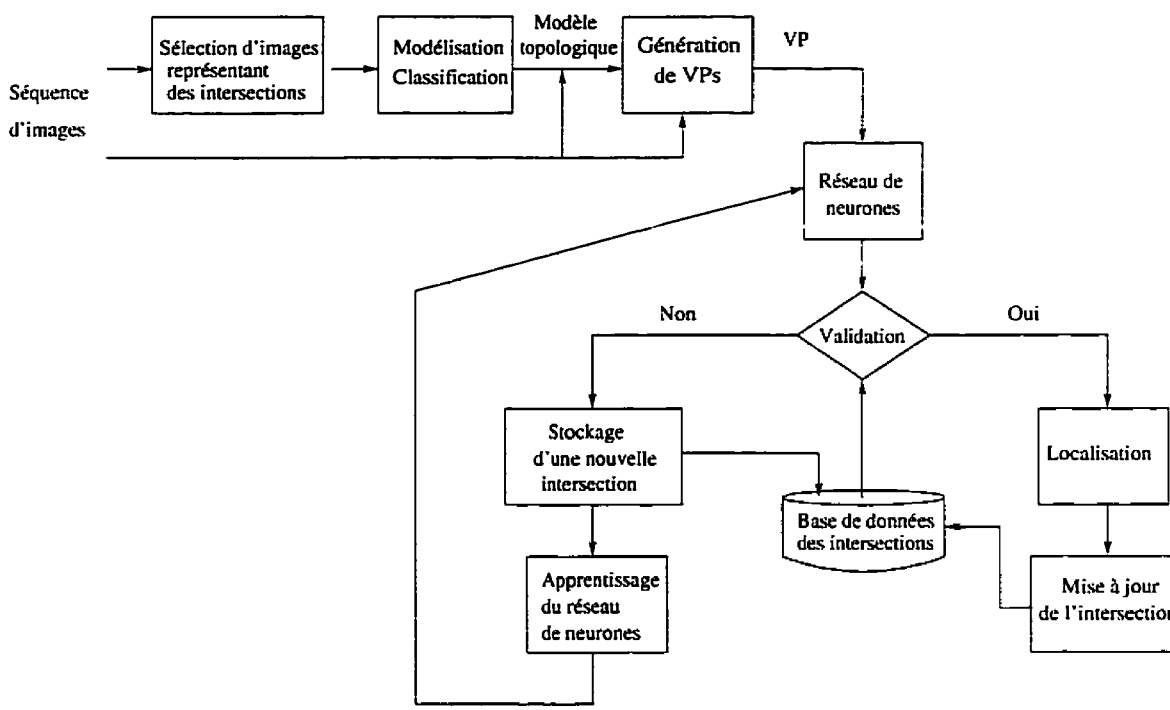


Figure 3.2 : Description du système

Le système analyse la séquence d'images que le robot enregistre au cours de son déplacement pour détecter les éventuelles intersections. Quand le robot rencontre une intersection, le système la modélise et la classe selon son nombre de corridors. Ensuite, le système utilise le modèle topologique et les images pour générer la vue panoramique correspondante (PV). Cette vue panoramique est présentée en entrée au réseau de neurones qui est capable de l'associer à l'identificateur de l'intersection enregistrée dans la base de données (si le robot est déjà passé par cette intersection). Si le réseau de neurones identifie l'intersection, le système est alors capable de se localiser dans l'environnement. Des opérations de mise à jour des modèles de l'intersection stockés dans la base de données sont alors effectuées. Autrement, les modèles de l'intersection inconnue sont stockés dans la base de données et le réseau de neurones apprend sa vue panoramique.

3.2 Modèle panoramique d'une intersection

3.2.1 Difficultés

Les problèmes de déformation de l'image et de génération de vues panoramiques sont récemment devenues un sujet d'intérêt dans les communautés scientifiques de la vision par ordinateur et du graphisme tridimensionnel. Seitz et Dyer (1996), McMillan et Bishop (1995), Mase et Nishira (1996) s'intéressent aux problèmes d'interpolation et de déformation d'images données pour obtenir des images vues d'un point de vue

différent de la scène. Ils exploitent en particulier les contraintes de continuité entre deux images prises de points de vue proches. Southwell et al. (1996) proposent un système stéréoscopique omnidirectionnel pour la commercialisation. La technologie actuelle permet de plus en plus de tendre vers des techniques d'animation et même d'interactivité avec l'utilisateur. Ainsi, Harris (1996) propose un système qui permet de synthétiser des images et de créer des animations. Citons encore Apple qui a développé une technologie qui permet de créer des animations panoramiques (Falco et McBride 1996) et tente d'imposer un format (Cannon 1996).

Erreurs d'alignement Il est peu probable que le robot passe au centre de l'intersection et fasse un tour sur lui-même pour prendre des images dans toutes les directions. En fait, à cause des caractéristiques cinématiques et dynamiques de la plate-forme, il est très possible même que le robot ne passe jamais au centre de l'intersection mais plutôt dans son voisinage. Par conséquent, une des difficultés est de reconstituer les images telles que le robot les verrait s'il était réellement placé au centre de l'intersection (voir la figure 3.3).

Vues manquantes Considérons maintenant la trajectoire du robot dans la figure 3.4. Certaines parties de la scène sont visibles du centre de l'intersection, P, mais n'apparaissent pas dans les images prises au cours du déplacement du robot. Par conséquent, la vue panoramique générée avec une telle séquence d'images contiendra des parties inconnues que nous devons remplir pour que le processus de reconnaissance

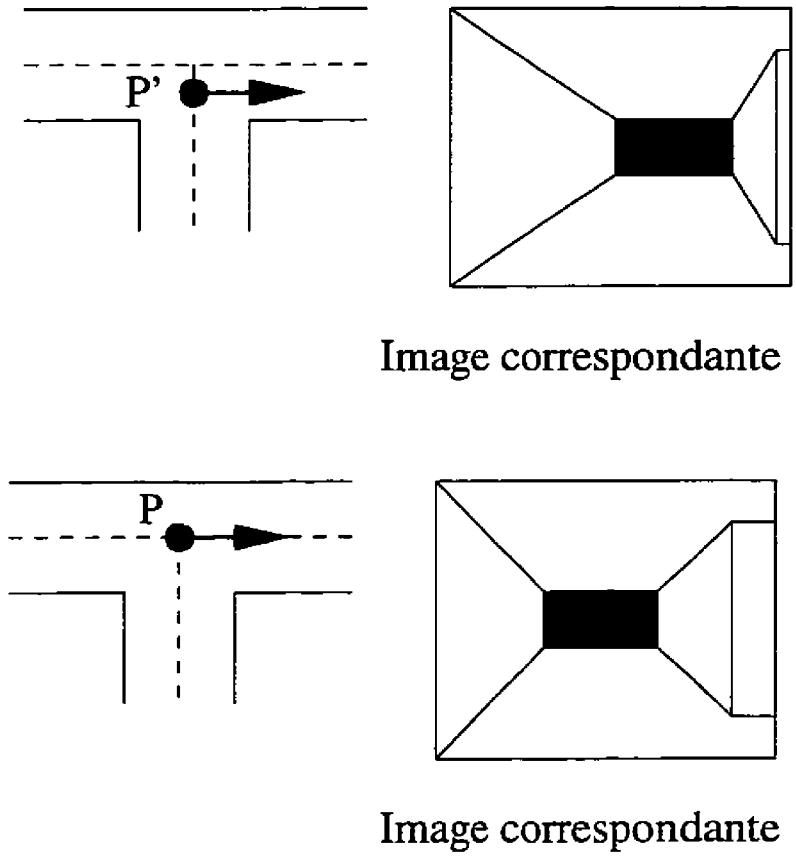


Figure 3.3 : Compensation des erreurs d'alignement

puisse s'amorcer. Un choix sur la méthode pour générer les parties manquantes devra être fait.

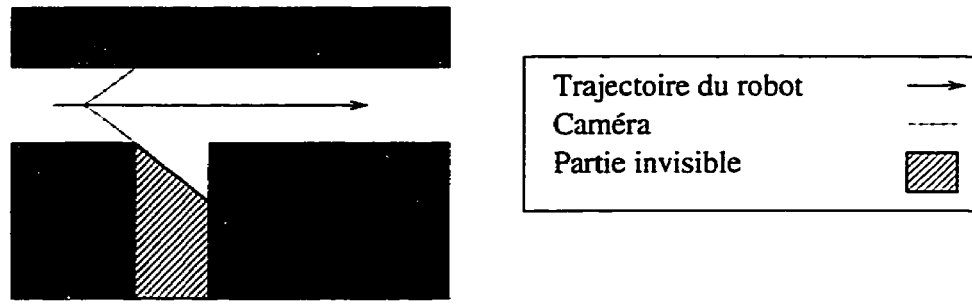


Figure 3.4 : Vues manquantes

Distorsion de la vue panoramique Si nous nous contentons de mettre les images bout à bout pour constituer la vue panoramique, nous obtiendrons des effets de distorsion. Pour éviter ces effets, nous prenons en fait des images qui se chevauchent, autant qu'il le faut pour réduire la distorsion. Le problème consiste à corriger les effets d'écran plat, à superposer les images et à les assembler. Nous verrons comment utiliser la technologie mise au point par Apple dans QuickDraw3D (QD3D) pour résoudre ces problèmes.

3.2.2 Description du processus

Le processus de modélisation panoramique est décrit sur la figure 3.5. Il s'appuie sur la technologie QuickDraw3D (Apple Computer Inc. 1995), librairie graphique qui permet entre autres de définir des modèles tridimensionnels, de leur appliquer des

couleurs, de la texture ou tout autre attribut caractérisant un modèle ou une partie d'un modèle et de générer leur image. QuickDraw3D propose un certain nombre de types d'objets géométriques, dont certains nous seront fort utiles et aussi diverses fonctions permettant de choisir un type d'illumination pour la scène, des méthodes de projection, etc.

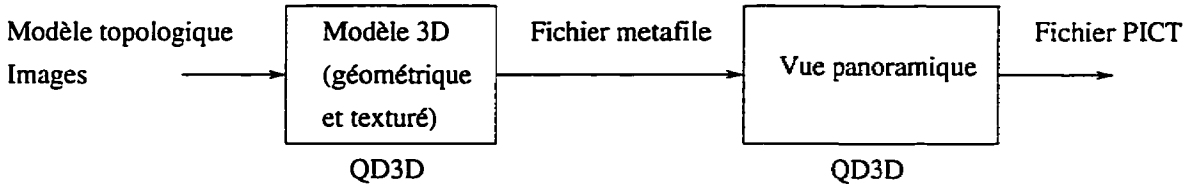


Figure 3.5 : Description du processus de modélisation panoramique d'une intersection

Pour construire une vue panoramique d'une intersection, nous disposons du modèle topologique de l'intersection établi selon la méthode décrite à la section 2.3 et des images. Une première étape consiste à déterminer le modèle tridimensionnel de l'intersection à partir du modèle topologique 2D. Ce modèle sera constitué d'objets géométriques QD3D. La texture des parois de l'intersection est extraite des images et appliquée sur le modèle 3D. Le modèle 3D ainsi obtenu (géométrie et texture) est sauvegardé dans un fichier "metafile", dont le format est défini par QD3D. Le deuxième module a pour objet de lire le modèle dans le fichier metafile, de générer la séquence d'images panoramiques et de les assembler pour former une vue panoramique dans le format d'un fichier image (PICT : format Macintosh).

3.2.3 Modèle 3D

QuickDraw3D est une librairie graphique orientée objet. Toutes les données manipulées sont donc des objets. En particulier, nous utilisons des objets géométriques auxquels nous appliquons des objets "transformation" dans l'espace et des objets "texture". Ces objets sont regroupés dans des objets "groupe".

3.2.3.1 Objets géométriques

Domaine de définition Pour construire le modèle d'une intersection, il nous faut circonscrire l'espace dans lequel nous voulons le définir, autrement dit ses limites. Une intersection sera donc définie dans un domaine $2L \times 2L \times h$ où $2L$ est à la fois la longueur et la largeur de la boîte et h sa hauteur. Le centre de l'intersection est placé au centre de la boîte. Le plafond de l'intersection ne sera pas représenté. En effet, cette partie de la scène est difficilement détectable dans l'image. Par ailleurs, nous n'avons pas vraiment besoin de cette information. La vue panoramique que nous présentons au réseau de neurones est coupée à la hauteur de la ligne d'horizon, c'est-à-dire à mi-hauteur de l'image puisque l'axe optique de la caméra est parallèle au plan du sol. La figure 3.6 donne une illustration pour une intersection en T.

Pour une super-intersection, le problème est un petit peu plus complexe. Rappelons qu'une super-intersection est composée d'intersections. Par conséquent, le domaine de définition d'une intersection est inclus dans l'union des domaines de définition des intersections la composant. Le domaine de définition d'une super-

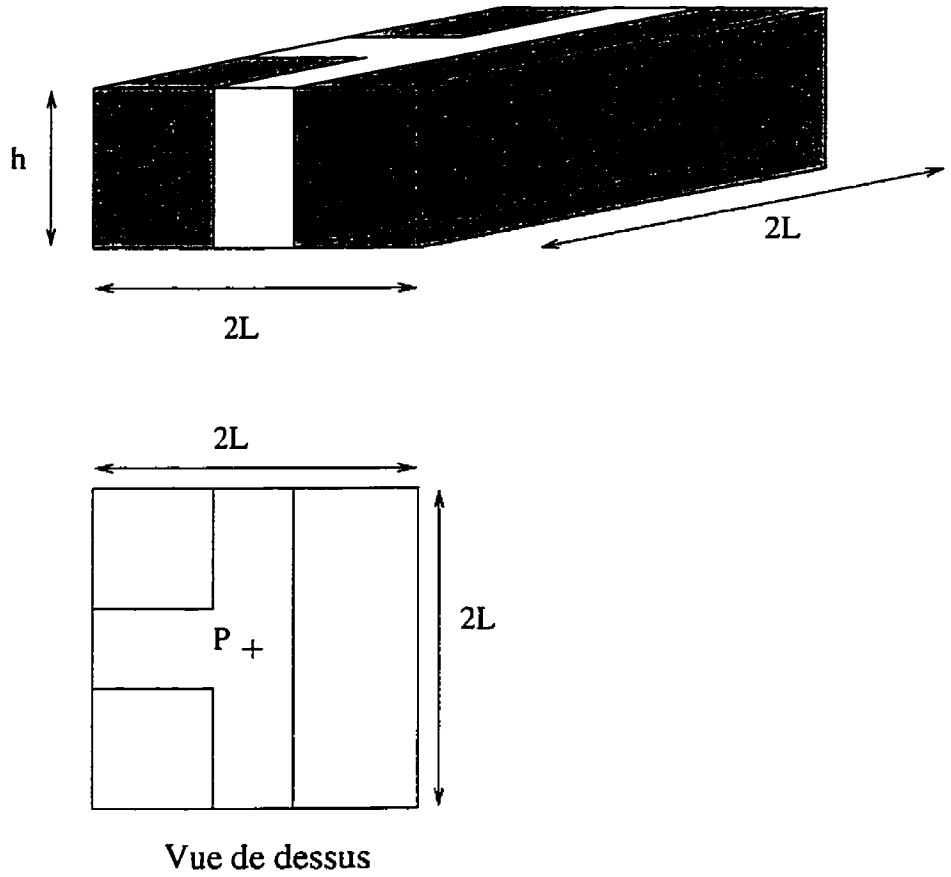


Figure 3.6 : Domaine de définition d'une intersection en T

intersection est la boîte circonvenant l'union des boîtes sur lesquelles sont définies les intersections.

L'objet trigrille Une intersection est constituée des objets géométriques suivants :

- des surfaces planes représentant les murs,
- des raccords entre les murs de deux corridors adjacents,
- une surface plane représentant le sol.

Le plafond n'est pas représenté.

Pour représenter ces surfaces, nous utilisons un objet géométrique QD3D : la trigrille. Une trigrille est une grille rectangulaire composée de facettes triangulaires (voir la figure 3.7). Elle est définie par ses sommets ordonnés de gauche à droite et de bas en haut. Nous pouvons définir des attributs pour les sommets, les facettes et/ou la trigrille tout entière. Ses dimensions sont définies par le nombre de sommets par rangée et le nombre de sommets par colonne.

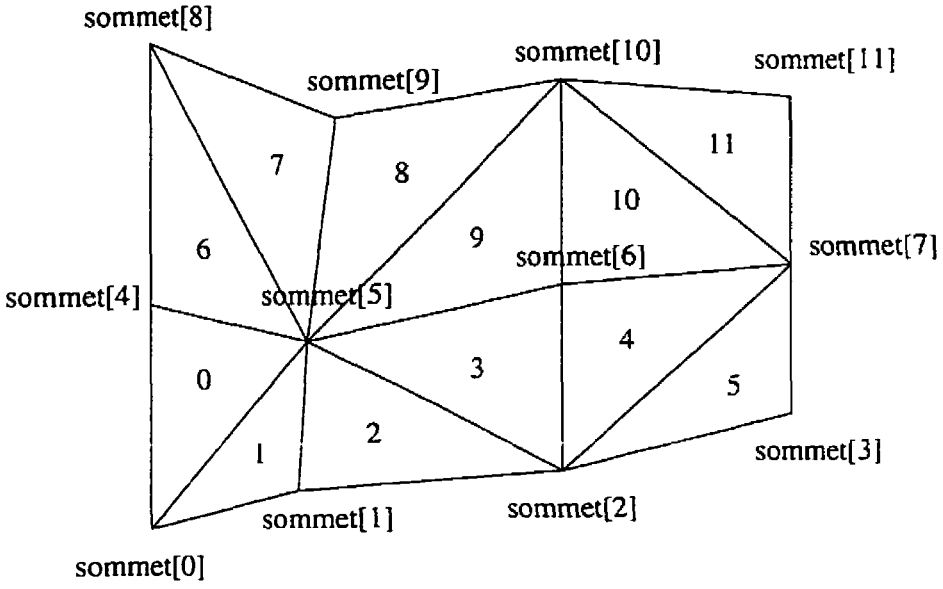


Figure 3.7 : Trigrille

Nous représentons les murs et le sol par une trigrille rectangulaire plane. Pour les raccords, nous utilisons une trigrille suivant la forme d'une surface parabolique (voir la figure 3.8).

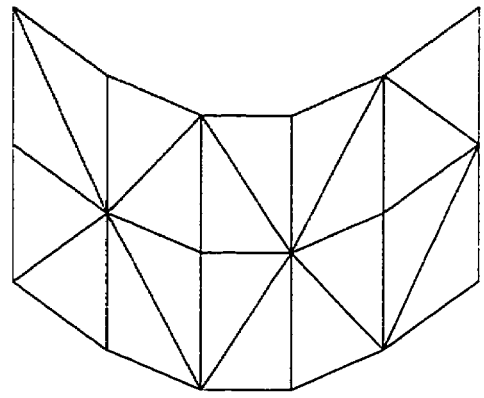


Figure 3.8 : Forme du raccord entre deux corridors

Construction du modèle géométrique 3D Le modèle géométrique 3D doit être compris au sens de QD3D. C'est l'ensemble des objets géométriques et des transformations spatiales qui leur sont appliquées qui permettent de représenter la scène. Pour construire un tel modèle, nous nous basons sur le modèle topologique de l'intersection.

Pour fixer les choses, considérons l'exemple de la figure 3.9. Le modèle topologique donne le centre de l'intersection, les axes orientés des corridors et leur largeur dans le repère de l'intersection. Trois corridors C_1, C_2 et C_3 sont représentés. Les équations de leurs axes sont portés sur la figure ainsi que leur sens. C_1 et C_3 ont le même axe mais une direction opposée. Les largeurs respectives l_1, l_2 et l_3 des corridors sont indiquées sur la figure.

Il s'agit donc, dans un premier temps, de déterminer les équations des murs et des raccords dans ce repère. Le sol est défini sur le domaine de définition de l'intersection complet. Dans un deuxième temps, il faudra passer du repère de l'intersection (2D) dans l'espace 3D.

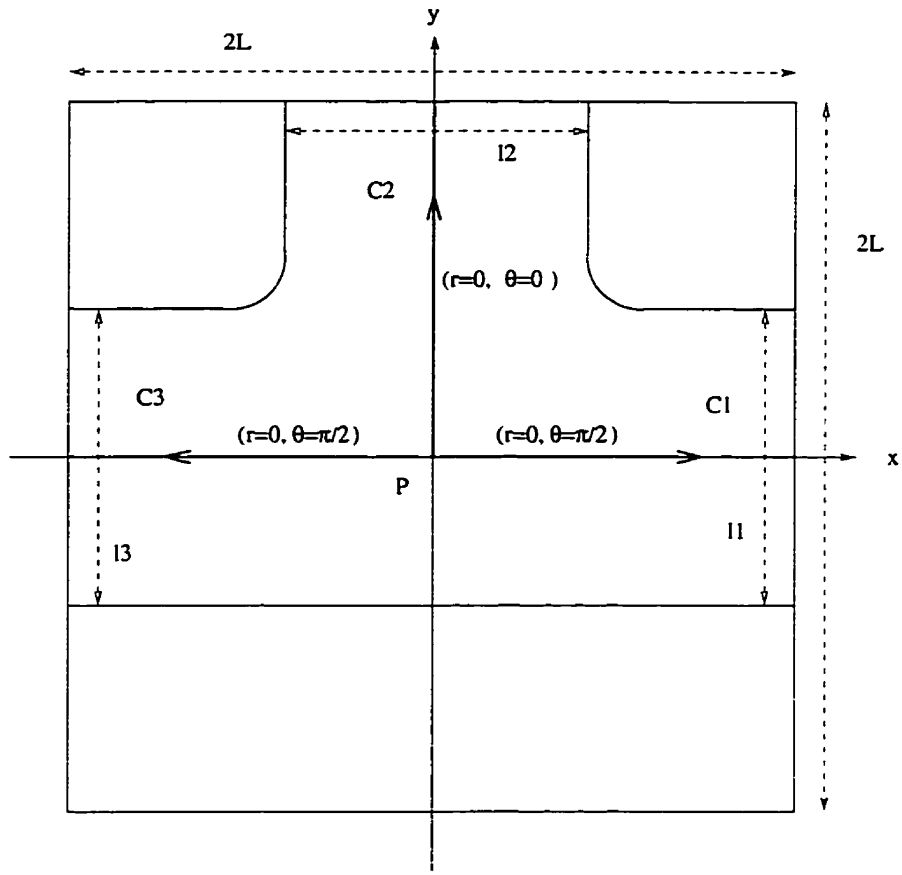


Figure 3.9 : Modèle géométrique pour une intersection en T

Murs À partir de l'équation des axes, nous pouvons déterminer les équations des murs (voir la figure 3.10).

L'équation du mur de gauche est donnée par :

$$\begin{cases} r_g = r - sens \times l/2 \\ \theta_g = \theta \end{cases}$$

Avec $sens = 1$ pour le sens positif et $sens = -1$ pour le sens négatif.

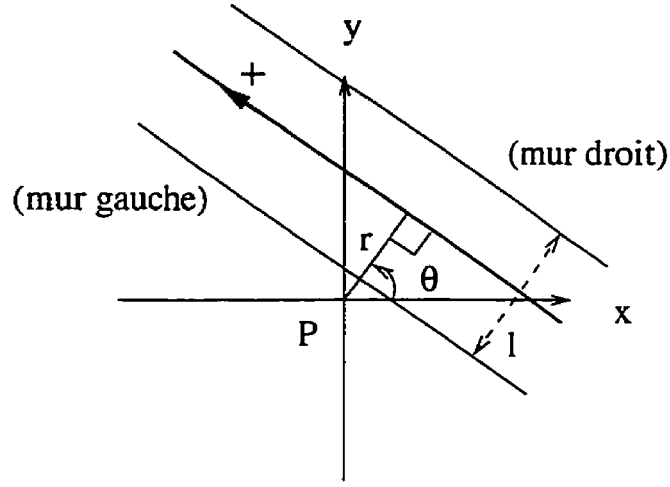


Figure 3.10 : Équations des murs à partir des axes dans le repère intersection

L'équation du mur de droite est donnée par :

$$\begin{cases} r_d = r + sens \times l/2 \\ \theta_d = \theta \end{cases}$$

Nous cherchons maintenant à calculer les points délimitant les murs (voir la figure 3.11.

L'algorithme parcourt les corridors dans le sens trigonométrique en les prenant deux à deux. A chaque itération, nous calculons les points délimitant le mur gauche du premier corridor (F,S_1) et les points délimitant le mur droit du deuxième corridor considéré (D, S_2). Cela comprend les points de raccord entre les deux (D, F). Nous calculons aussi le point M, intersection des murs considérés.

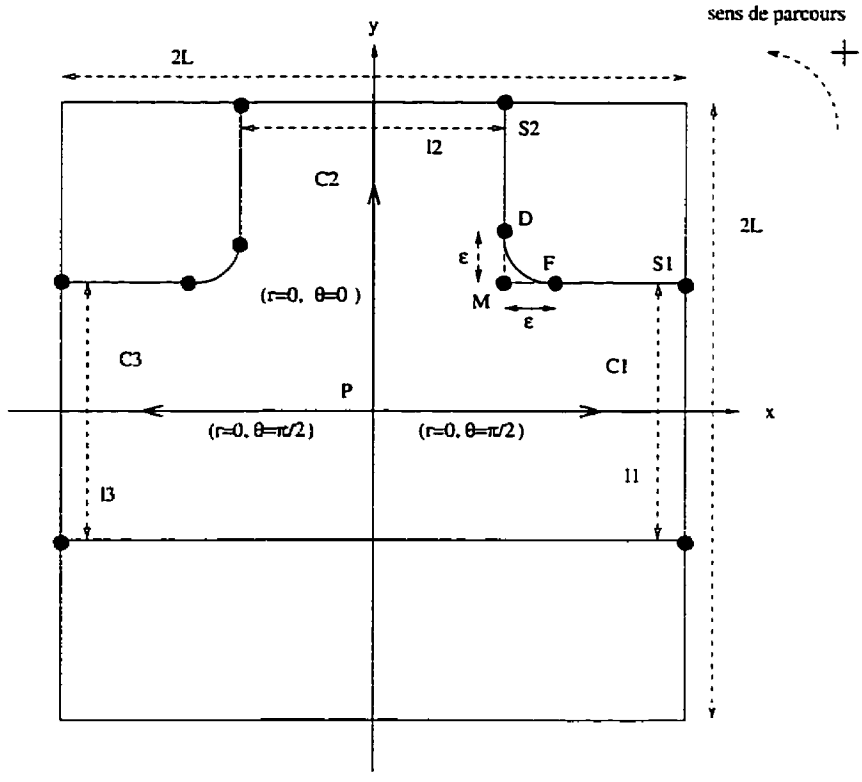


Figure 3.11 : Points à déterminer pour construire les murs

Calculons l'intersection des murs M. Nous connaissons les équations des murs :

$$\begin{cases} r_{g1} = x\cos\theta_{g1} + y\sin\theta_{g1} \\ r_{d2} = x\cos\theta_{d2} + y\sin\theta_{d2} \end{cases}$$

Nous en déduisons les coordonnées du point M dans le repère intersection :

$$\begin{cases} x_M = \dfrac{r_{d2}\sin\theta_{g1} - r_{g1}\sin\theta_{2d}}{\sin(\theta_{g1} - \theta_{d2})} \\ y_M = \dfrac{r_{g1}\cos\theta_{2d} - r_{d2}\cos\theta_1}{\sin(\theta_{g1} - \theta_{d2})} \end{cases}$$

Nous pouvons alors calculer les points de raccord D et F. Nous prenons :

$$sens(D, M) = \epsilon.$$

Soit $\mathbf{v}$ le vecteur directeur de l'axe d'un corridor. Nous avons $\mathbf{v} = (-\sin\theta, \cos\theta)$.

$$\mathbf{MD} = sens \times \epsilon \times \mathbf{v}$$

De même,

$$\mathbf{MF} = sens \times \epsilon \times \mathbf{v}$$

Il reste à calculer les points $S1$ et $S2$. S_i vérifie l'équation du mur :

$$r = x\cos\theta + y\sin\theta.$$

De plus, $\mathbf{MS_i}$ est dans le sens de la droite. D'où :

$$\mathbf{MS_i} = k \times sens \times \mathbf{v}$$

avec $k > 0$. Nous avons de plus une contrainte supplémentaire : $x_{Si} = \pm L$ ou $y_{Si} = \pm L$.

Nous considérons donc les cas suivants :

- Cas particulier : axe du corridor parallèle à l'axe y

$$x_{Si} = x_M$$

$$y_{Si} = sign(\cos\theta) \times sens \times L$$

- Cas particulier : axe du corridor parallèle à l'axe x

$$x_{Si} = -sign(\sin\theta) \times sens \times L$$

$$y_{Si} = y_M$$

- $x_{Si} = L$

$$k = \frac{x_M - x_{Si}}{(sens \times \sin\theta)}$$

$$\text{si } k > 0, \quad y_{Si} = k \times sens \times \cos\theta + y_M$$

Vérifier que $-L \leq y_{Si} \leq L$.

- $x_{Si} = -L \Rightarrow$ même chose que pour $x_{Si} = L$.

- $y_{Si} = L$

$$k = \frac{y_M - y_{Si}}{(sens \times \cos\theta)}$$

$$\text{si } k > 0, \quad x_{Si} = k \times sens \times \sin\theta + x_M$$

Vérifier que $-L \leq x_{Si} \leq L$.

- $y_{Si} = -L \Rightarrow$ même chose que pour $y_{Si} = L$.

Ayant déterminé les points S_1, F, D et S_2 dans le repère intersection, nous pouvons calculer les points décrivant les murs et positionner les objets "mur" dans le repère du monde.

Un mur est un objet trigrille, rectangulaire et plan. Connaissant les dimensions du mur (longueur FS_1 ou DS_2 × hauteur h), nous pouvons calculer ses sommets répartis à intervalles réguliers (voir la figure 3.12) dans le repère du monde. L'espacement entre

les points est fixé. Comme le montre la figure 3.12, il se peut donc que la dernière colonne (celle de droite) et/ou la dernière rangée (celle du haut) ait un espacement plus petit.

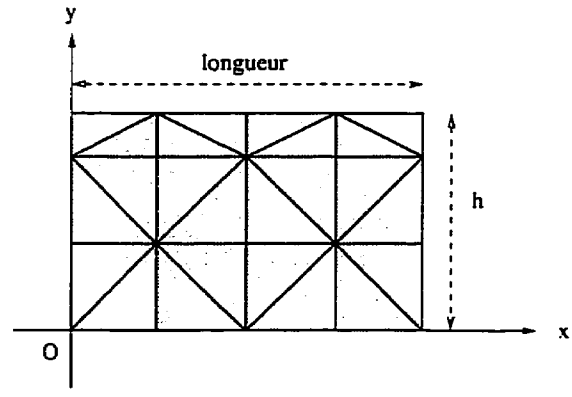


Figure 3.12 : Sommets d'un objet mur

La surface est paramétrisée comme indiqué sur la figure 3.13. Cette paramétrisation est nécessaire pour pouvoir appliquer une texture sur la surface.

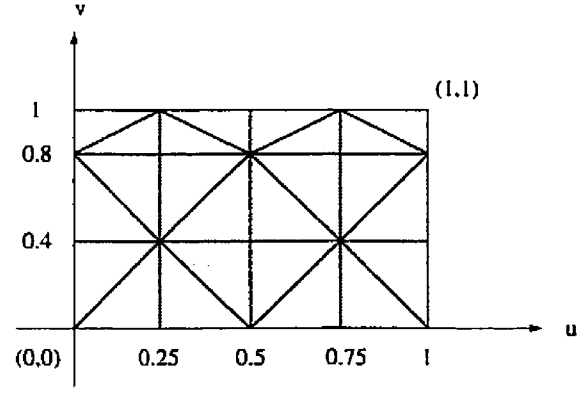


Figure 3.13 : Paramétrisation d'un objet mur

Il reste à placer l'objet mur dans le repère du monde. Considérons le mur droit

du couloir C_2 de la figure 3.14. Nous appliquons successivement une rotation d'angle $(x, \widehat{\mathbf{FS_1}})$ autour de l'axe y puis une translation de vecteur **OF**.

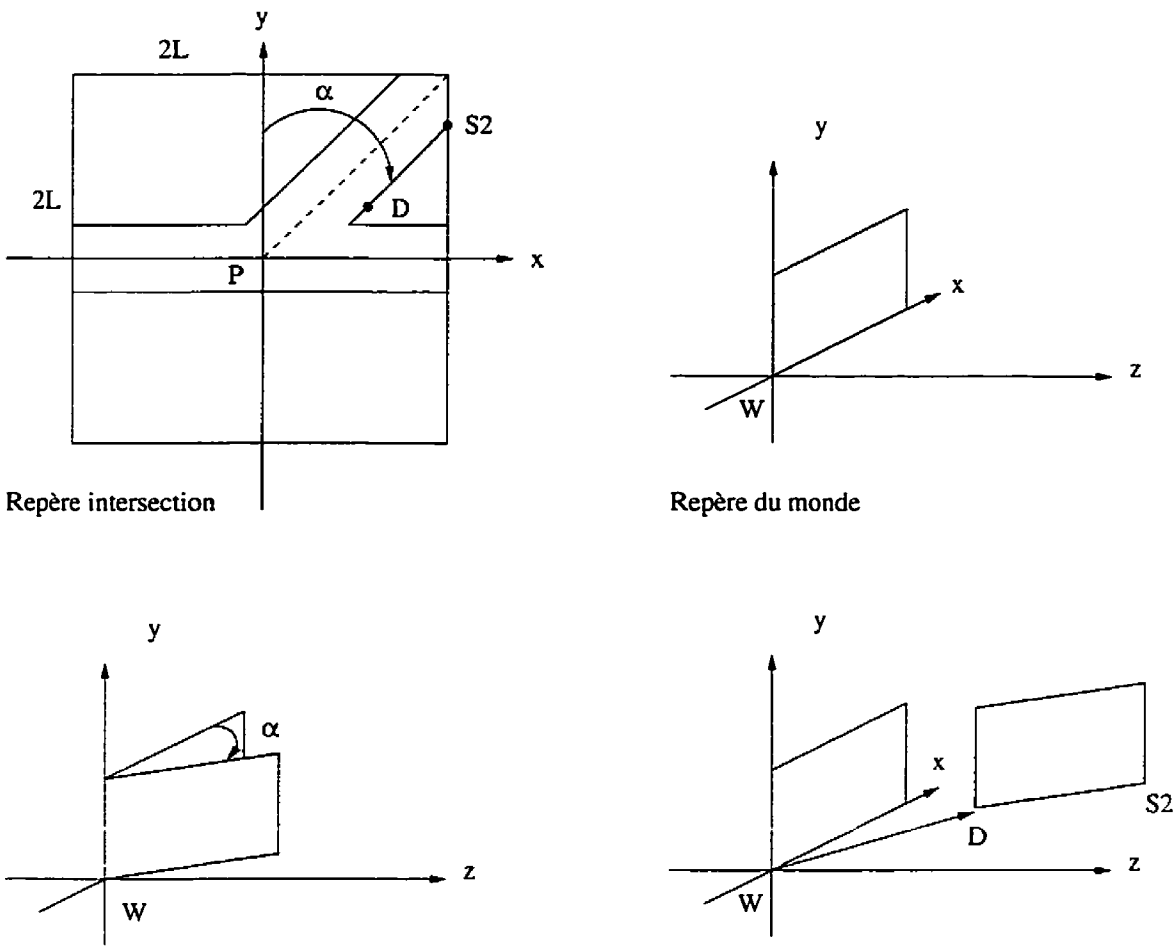


Figure 3.14 : Positionnement d'un objet mur dans le monde

Enfin, il faut donner une orientation à la surface. La normale à une surface mur doit pointer vers le centre de l'intersection.

Sol Le sol est construit sur le même principe que les murs. Ses dimensions sont $2L \times 2L$. Nous appliquons successivement une rotation d'angle $\pi/2$ autour de y pour transformer la trigrille dans le plan (x, z) et une translation de vecteur $(-L, 0, -L)$ pour placer le centre du sol au centre du monde.

Raccords Les raccords sont aussi définis par des trigrilles. Ils répondent aux contraintes suivantes (voir la figure 3.15) dans le repère intersection :

- Ils suivent l'équation d'une parabole.
- Ils passent par les points D et F.
- Les dérivées en D et F sont parallèles aux murs.
- La droite perpendiculaire à la droite (DF) et passant par le milieu de $[D, F]$ est un axe de symétrie.

Il s'agit donc tout d'abord de déterminer l'équation de la parabole décrivant le raccord. Cette parabole est définie dans le repère $O(\mathbf{i}, \mathbf{j})$ déterminé comme suit. L'origine O de ce repère est le milieu de $[D, F]$.

$$\mathbf{PO} = \mathbf{DF}/2$$

L'axe i passe par les points D et F.

$$\mathbf{i} = \mathbf{DF}/\|DF\|$$

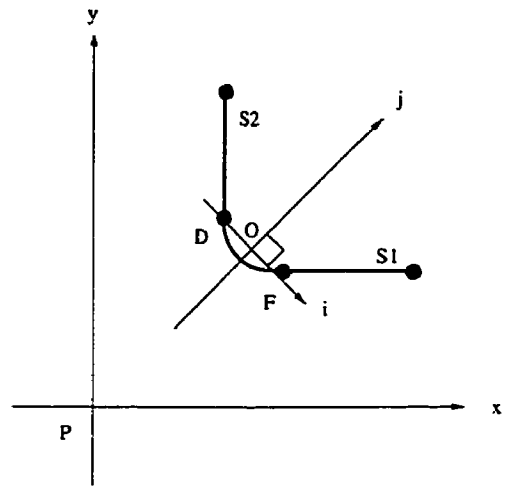


Figure 3.15 : Objet raccord

$$\mathbf{j} = \mathbf{z} \times \mathbf{i}$$

Dans ce repère $O(\mathbf{i}, \mathbf{j})$, la courbe décrivant le raccord a pour équation :

$$Y = aX^2 + Y_0$$

Déterminons les paramètres a et Y_0.

Soit $\mathbf{v_F} = (\mathbf{V_{Fx}}, \mathbf{V_{Fy}})$ la dérivée du mur en F dans le repère $P(x, y)$. Sa transformée $\mathbf{V_F}$ dans le repère $O(\mathbf{i}, \mathbf{j})$ s'obtient par une rotation d'angle $(\widehat{x, \mathbf{i}})$ autour de O.

La dérivée de la courbe s'écrit :

$$\frac{dX}{dY} = 2aX.$$

La continuité au point F s'exprime par la relation suivante :

$$\left(\frac{dX}{dY}\right)_F = \frac{V_{Fy}}{V_{Fx}}$$

F a pour coordonnées : $(0, DF/2)$. Donc

$$\left(\frac{dX}{dY}\right)_F = aDF.$$

D'où

$$a = \frac{1}{DF} \times \frac{V_{Fy}}{V_{Fx}}.$$

Nous savons aussi que la courbe passe par F. Donc F vérifie son équation. Nous en déduisons Y_0 :

$$Y_0 = -a \times DF^2/4$$

A partir de ces données, nous pouvons déterminer les sommets de la trigrille représentant le raccord. La figure 3.16 représente les points de la trigrille dans le repère du monde ainsi que la paramétrisation (u, v) de sa surface.

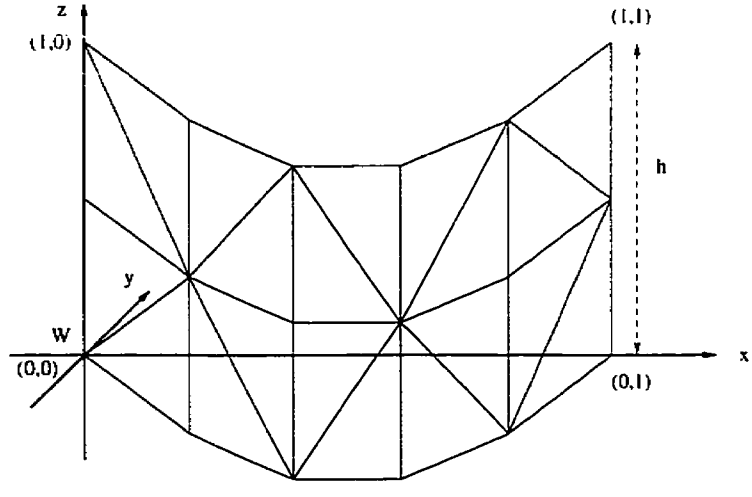


Figure 3.16 : Trigrille représentant le raccord : repère et paramètres

Pour positionner correctement la trigrille, nous lui appliquons les transformations suivantes. La trigrille a été calculée dans le repère $O(\mathbf{i},\mathbf{j})$. Nous lui faisons subir une rotation d'angle $(\widehat{x,\mathbf{i}})$ autour de O et une translation de vecteur $\mathbf{PO}$ pour la placer dans le repère de l'intersection. Ensuite, une rotation de $\pi/2$ autour de z puis une rotation de $\pi/2$ autour de x la transforment dans le repère du monde.

3.2.3.2 Texture

Méthode Nous avons construit le modèle géométrique de l'intersection avec des surfaces planes. Il nous reste à reproduire leur texture en 3D à partir de l'image de la scène. La figure 3.17 reprend les grandes étapes du processus de recherche et d'application de la texture.

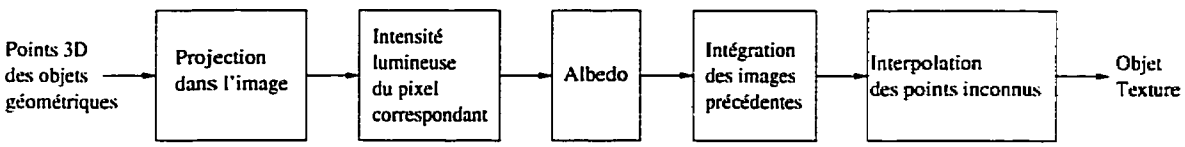


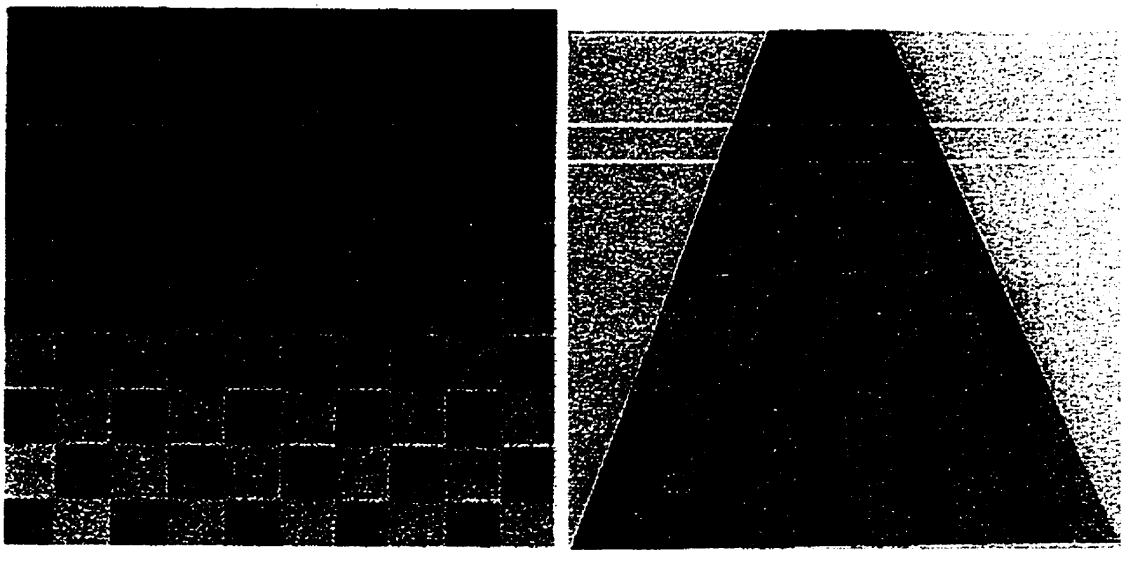
Figure 3.17 : Processus de recherche et d'application de la texture

Nous cherchons à retrouver la texture de la scène à partir de l'image de la scène prise par une caméra dont la position et l'orientation nous sont connues. Les points du modèle géométrique que nous venons de calculer, sont projetés dans l'image. Pour chaque point, nous obtenons l'intensité lumineuse du pixel correspondant. Connaissant l'orientation de la source lumineuse et la normale à la surface (calculée à partir du modèle), nous en déduisons l'albedo du point considéré. En fait, nous ne traitons pas

une image, mais une séquence d'images prises par le robot au cours de son déplacement. Nous avons donc une étape d'intégration des données des images précédentes. Enfin, certaines parties de la scène restent occultées. Il faut donc un procédé qui permette de remplir les parties invisibles. Cela se fait par l'interpolation des points connus par la méthode des fractales.

Notons que les effets de spécularité sont éliminés par le fait que la source de lumière et l'observateur (caméra) sont placés au même point (en fait en des points très proches). Cela nous évite d'effectuer un prétraitement sur l'image.

Autre remarque : quand nous effectuons la projection des points dans l'image, un problème classique apparaît : un point projeté se répartit nécessairement sur un ou plusieurs pixels (Wolberg 1990) (voir la figure 3.18).



(a) (b)

Figure 3.18 : (a) Objet 3D – (b) Projection dans l'image (Wolberg 1990)

Pour éviter l'accumulation d'erreurs, nous ne gardons que les points suffisamment proches d'un pixel selon le critère de la distance de Manhattan. Les autres points sont interpolés.

Technique QD3D Comme nous l'avons mentionné plus haut, QuickDraw3D définit des objets "texture" qui permettent d'appliquer une image prédéfinie (ou texture) à une surface d'un objet du modèle. QuickDraw3D fournit des outils qui permettent au développeur de créer par exemple une image d'une surface en bois et de l'appliquer sur les objets pour leur donner l'aspect du bois. Les objets texture sont définis par des PixMaps qui contiennent l'image à appliquer. Un PixMap est un objet créé à partir d'une image en format PICT.

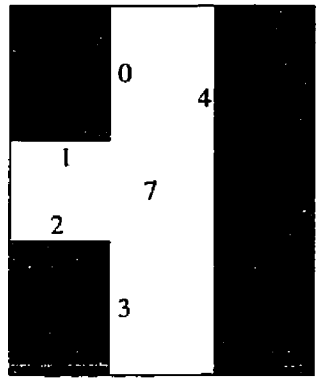
Pour appliquer une texture à une surface, QD3D fait la correspondance entre la texture et la surface. Cette mise en correspondance est spécifiée en utilisant la paramétrisation de la surface que nous avons décrite plus haut.

Segmentation de l'image Pour calculer la texture des objets du modèle géométrique tridimensionnel, nous aurons besoin d'étiqueter les pixels de l'image pour savoir s'ils appartiennent à des murs ou au sol.

Nous voulons attribuer une étiquette au sol et aux murs de l'intersection, autrement dit segmenter l'image en parties de murs et de sol. Les raccords n'ont pas d'étiquette propre. Ils possèdent l'étiquette des murs qui leur sont adjacents. De plus, nous n'attribuons pas d'étiquette au plafond car il n'est pas représenté dans

notre modèle. Nous considérons seulement la moitié inférieure de l'image. En effet, nous supposons que l'axe optique de la caméra est parallèle au plan du sol. Par conséquent la ligne d'horizon est au milieu de l'image.

Pour notre problème de segmentation de l'image, nous considérons qu'un corridor est composé de deux murs. Une étiquette différente est attribuée à chaque mur. De plus, si nous avons deux murs appartenant à deux corridors différents au sens topologique mais qui se prolongent l'un l'autre géométriquement, nous leur attribuons la même étiquette. La figure 3.19 donne un exemple pour une intersection en T.



Modèle topologique étiqueté

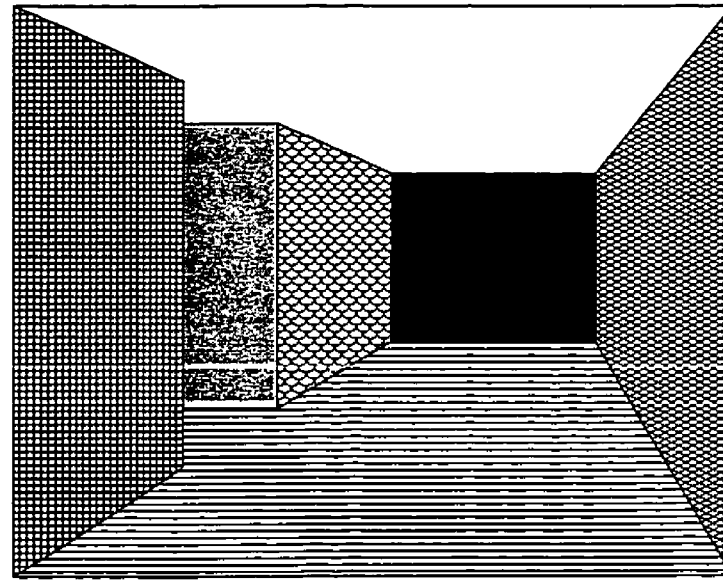


Image de la scène segmentée et étiquetée

Etiquette 0
Etiquette mur 1
Etiquette mur 3
Etiquette mur 4
Etiquette sol 7

Figure 3.19 : Étiquettes des pixels de l'image

Les étiquettes sont attribuées en commençant par les murs et en les prenant dans le sens de parcours du modèle géométrique de l'intersection c'est-à-dire dans le sens trigonométrique. Nous terminons par le sol.

Pour segmenter notre moitié d'image, nous nous basons sur le contour du sol approximé par des segments étiquetés obtenu dans le chapitre 2 à la section 2.4. Le contour du sol borde la région de l'image qui portera l'étiquette du sol. Chaque segment du sol étiqueté "Mur Visible" ou "Mur Occulté" borde une région de l'image qui portera l'étiquette d'un mur. Les segments portant une étiquette "Occlusion" ou "Limite Visibilité" ne sont pas pris en compte. Considérons l'exemple de la figure 3.20.

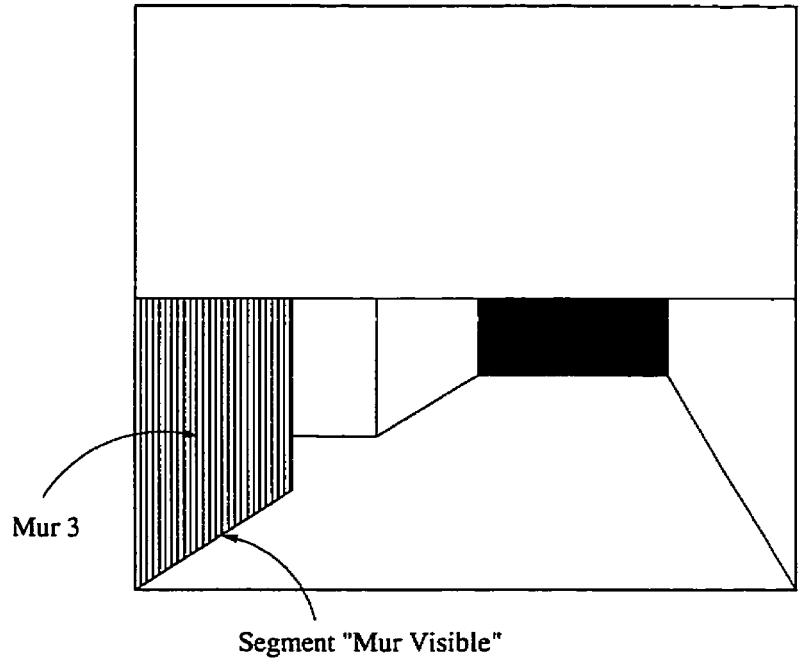


Figure 3.20 : Segmentation de la demi-image

Nous partons de l'hypothèse selon laquelle les murs sont verticaux de façon "globa-

le". Autrement dit, les aspérités des murs sont négligées devant leur verticalité. Donc les arêtes des murs sont considérées comme verticales dans l'espace tridimensionnel. Leur projection dans l'image donne des segments de droite verticaux. Par conséquent, un mur sera délimité par quatre segments :

- Un segment de sol portant une étiquette "Mur Visible" ou "Mur Occulté",
- Deux segments verticaux partant des extrémités du segment de sol,
- Un segment horizontal situé au milieu de l'image et joignant les extrémités des segments verticaux.

Il suffit donc de partir de chaque segment de sol étiqueté "Mur Visible" ou "Mur Occulté" et de balayer la région située verticalement au dessus de lui en s'arrêtant au milieu de l'image.

Algorithme Nous travaillons sur une image en niveaux de gris. La position et l'orientation de la caméra ayant pris cette image de la scène sont connus. Nous traitons chaque objet géométrique c'est-à-dire chaque mur, chaque raccord et le sol séparément. Nous calculons une texture pour chaque objet à partir de l'image et en tenant compte des textures calculées à partir des images précédentes de la séquence pour chaque objet.

Calcul de la texture pour un mur La figure 3.21 rappelle les notations utilisées. Un mur est défini par deux points F et S_1 (ou S_2 et D) dans le repère de

l'intersection. Connaissant ces points, nous calculons facilement la normale au mur dans le repère du monde. La surface est orientée vers le centre de l'intersection.

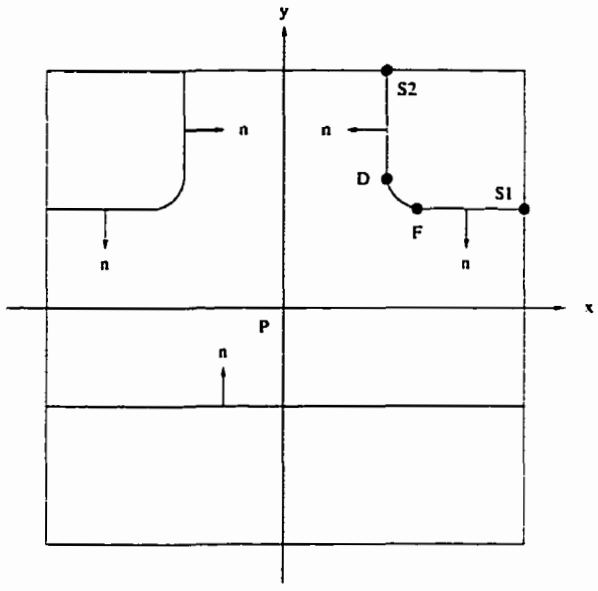


Figure 3.21 : Orientation de la normale à un mur

Nous considérons ensuite chacun des pixels appartenant à la texture que nous voulons appliquer sur le mur. Il s'agit de trouver leur valeur de niveau de gris à partir de l'intensité lumineuse des pixels de l'image.

Considérons le couloir droit de la figure 3.22. Nous montrons à partir de cet exemple les transformations successives qui doivent être accomplies pour déterminer la texture du mur (A, B, C, D).

La figure 3.23 décrit la suite de transformations qui permettent de passer d'un pixel du PixMap qui doit être appliqué sur le mur (A, B, C, D) au pixel correspondant dans l'image. Nous supposons que la caméra est placée au centre de l'intersection (centre

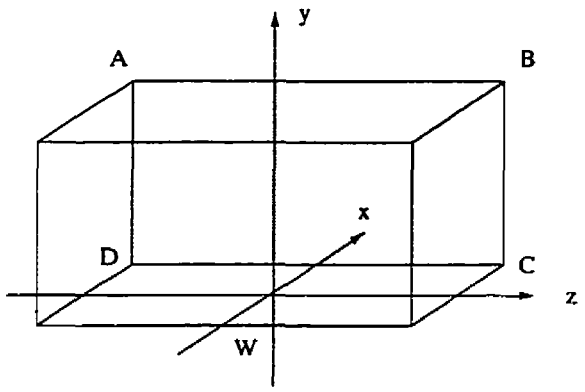


Figure 3.22 : Couloir droit considéré

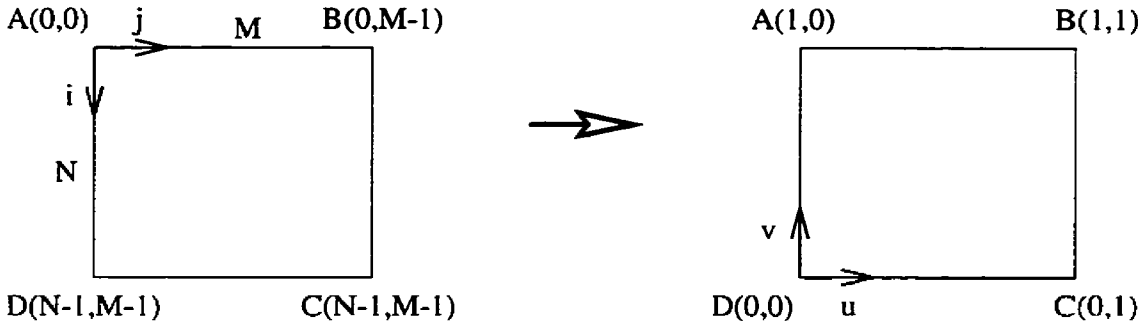
du monde) et est orientée dans la direction de l'axe z du monde.

Ayant trouvé la projection d'un point du mur dans l'image, nous vérifions que ce point se projette bien dans les limites de l'écran autrement dit que ce point est visible dans l'image de la scène. Par exemple, les points A et D ne se projettent pas dans l'image.

Cela fait, pour passer du point projeté (x, y) dans l'image au pixel (i, j), nous arrondissons au point le plus proche (x_i, y_j).Nous vérifions que la distance du point (x, y) au point (x_i, y_j) est petite. Nous utilisons la distance de Manhattan comme critère :

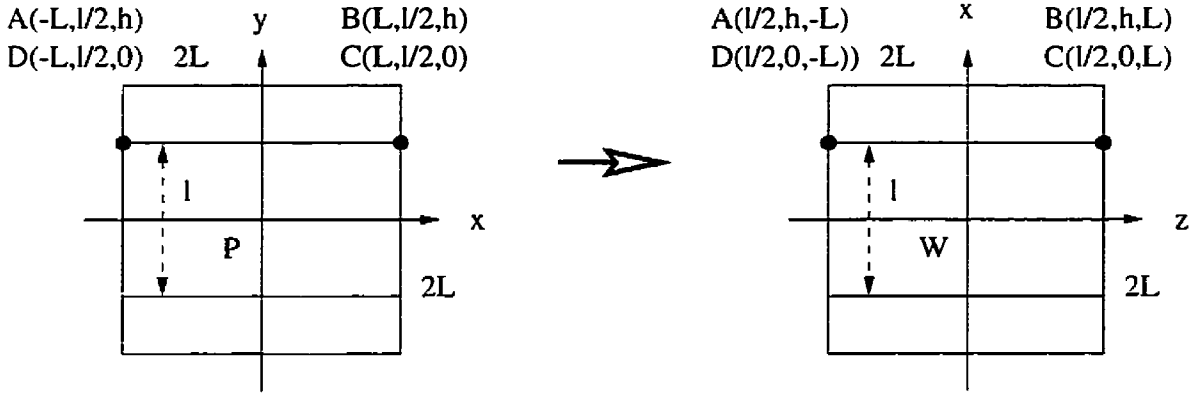
$$|x - x_i| + |y - y_j| < \varepsilon \text{ avec } \varepsilon > 0$$

La figure 3.24 illustre cette notion. Pour les points M et N, le point (x_i, y_j) est le plus proche mais la distance de N au point (x_i, y_j) est inférieure à ε alors que la distance de N au point (x_i, y_j) ne l'est pas.



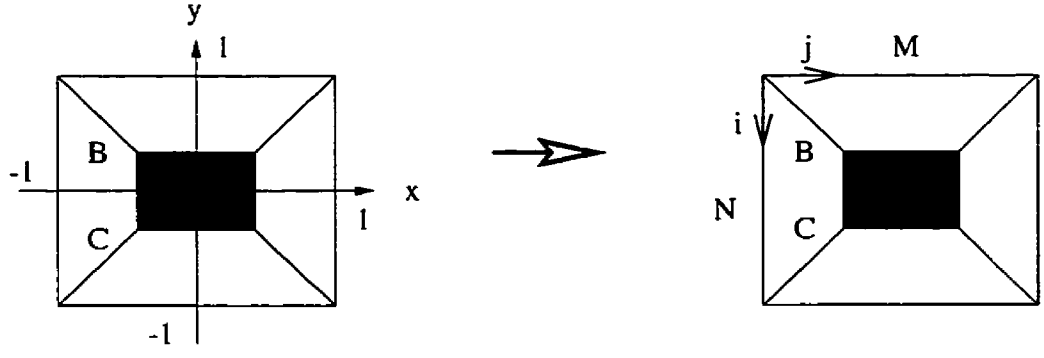
PixMap MxN
Même taille que l'image

Objet paramétrisé



Mur dans le repère de l'intersection

Mur dans le repère du monde



Projection perspective
dans l'image

Transformation dans
l'espace du PixMap

Figure 3.23 : Transformations pour passer d'un PixMap à un point image

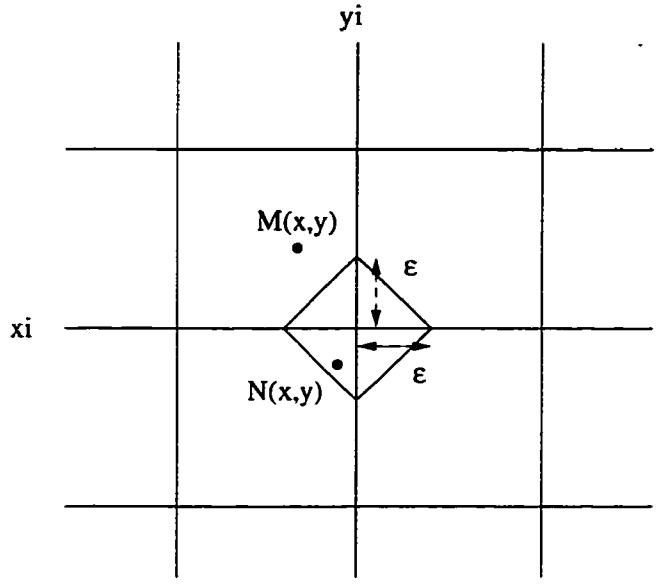
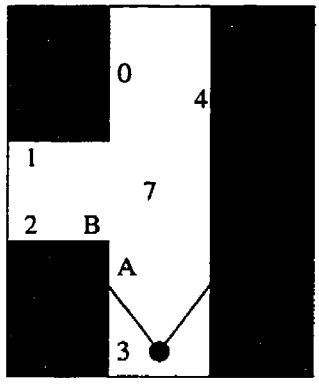


Figure 3.24 : Distance de Manhattan

Enfin, il faut vérifier que le pixel obtenu n'appartient pas à une partie cachée (occlusion) de la scène. Cette question est résolue grâce à l'étiquette attribuée aux pixels de l'image. L'étiquette d'un pixel indique en effet à quel mur il appartient. Par exemple, sur la figure 3.25, le pixel $(i,j)_1$ a pour étiquette 1. Il appartient au mur 1. Le mur 2 par exemple est totalement occulté. Aucun pixel ne porte donc l'étiquette 2. Considérons les points A et B appartenant respectivement aux murs 2 et 3. A et B se projettent dans l'image sur le même pixel $(i,j)_2$. Mais A est occulté par B. Donc le pixel aura pour étiquette 3. Donc si un point M appartenant à un mur se projette sur l'image en un pixel ayant une étiquette différente du mur auquel il appartient, alors M est occulté.

Dans le cas lambertien, l'intensité lumineuse normalisée vérifie l'équation suivante



Modèle topologique étiqueté

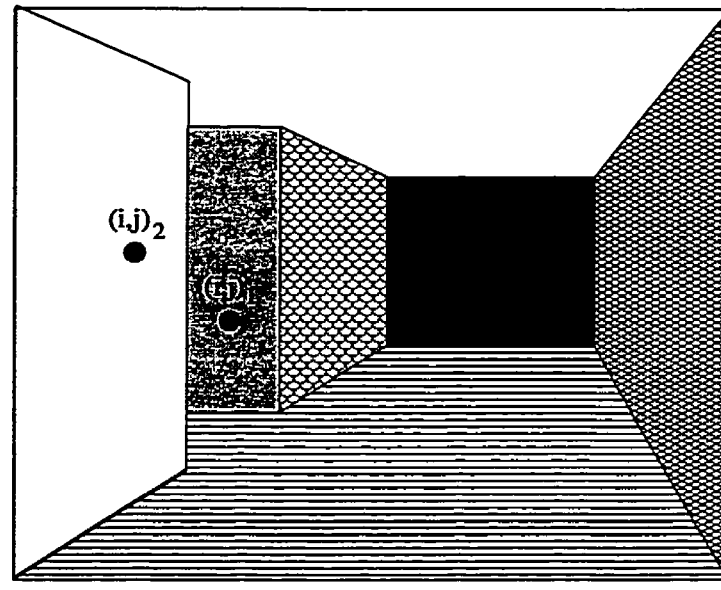


Image de la scène segmentée et étiquetée

Etiquette 0
Etiquette mur 1
Etiquette mur 3
Etiquette mur 4
Etiquette sol 7
Caméra

Figure 3.25 : Problème des occlusions

(tirée de (Horn 1986)) :

$$I(x,y) = \rho(x,y)\mathbf{s}\cdot\mathbf{n}\ g(d,a,e),$$

avec $\mathbf{s}$ la direction de la source de lumière, $\mathbf{n}$ la normale à la surface en (x,y), ρ l'albedo de la surface, g une fonction qui dépend de l'éloignement du point considéré, des propriétés d'éclairage, du coefficient d'absorption de l'air. Comme nous utilisons un modèle de source lumineuse dont l'intensité ne dépend pas de la distance dans

nos expérimentations sur des images de synthèse, nous prenons $g = 1$. La figure 3.26 décrit les paramètres de l'équation de l'intensité.

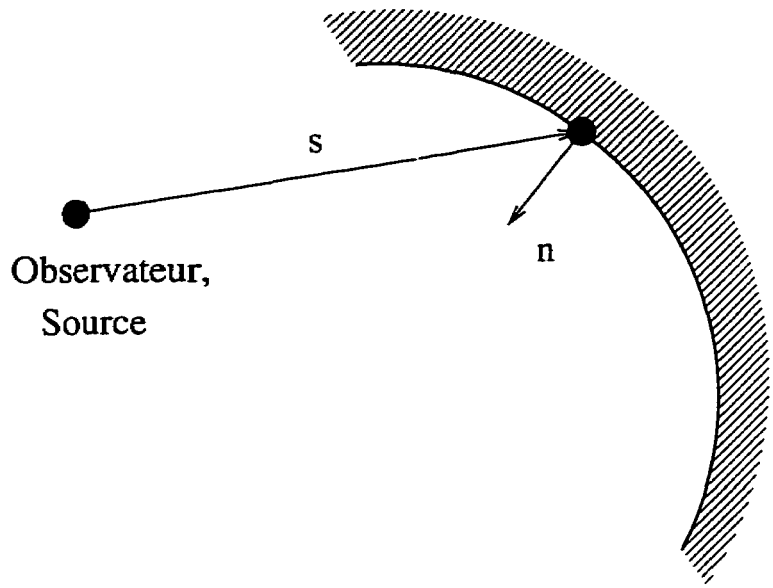


Figure 3.26 : Intensité lumineuse

Connaissant la position de la caméra, nous calculons le vecteur s. L'albedo d'un point (x, y) s'écrit :

$$\rho(x, y) = \frac{I(x, y)}{\mathbf{s} \cdot \mathbf{n}\ g(d, a, e)}$$

Nous convertissons la valeur obtenue en intensité de gris et la conservons dans le PixMap.

Nous pouvons ensuite intégrer les valeurs des données calculées précédemment pour ce mur à partir des images précédentes. Une image obtenue à l'instant t nous permet de calculer un estimé instantané du PixMap, $\hat{P}(x, y)_t$, que nous intégrons dans le PixMap $P(x, y)_{t-1}$ (voir la figure 3.27). La nouvelle texture à l'instant t, $P(x, y)_t$

est calculée en utilisant la formule :

$$P(x,y)_t = (1-\alpha)P(x,y)_{t-1} + \alpha\hat{P}(x,y)_t,$$

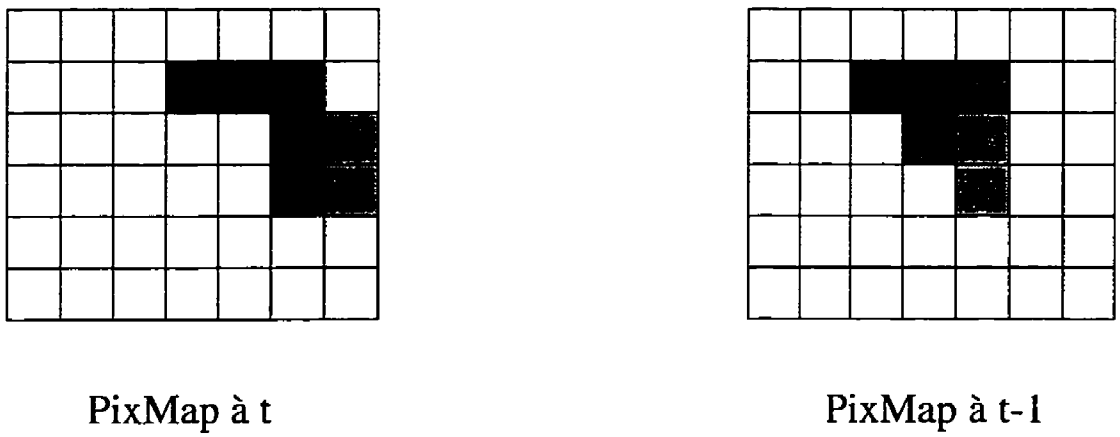


Figure 3.27 : Combinaison des PixMaps aux instants $t-1$ et t

où α est un facteur de pondération qui traduit l'importance accordée à l'image considérée. Nous prenons $\alpha = 0.2$.

Les valeurs inconnues (i,j) de $\hat{P}(x,y)_t$ sont purement remplacées par les valeurs correspondantes (i,j) de $P(x,y)_{t-1}$. Les valeurs inconnues de $P(x,y)_{t-1}$ ne sont évidemment pas prises en compte.

Une fois que tout le PixMap a été parcouru, il reste encore des pixels qui n'ont pas de valeur pour l'une des deux raisons :

- ils appartiennent à des parties de la scène non visibles dans l'image ;

- leur projection se répartit entre plusieurs pixels.

Pour ces points, nous effectuons une interpolation à partir des points connus par la méthode des fractales. Nous adaptons l'algorithme décrit par Peitgen et al. (1988). Nous utilisons une grille de dimension carrée. La méthode consiste à calculer un point inconnu à partir de la valeur de points connus proches à laquelle une valeur aléatoire gaussienne est ajoutée (voir l'exemple en 1D de la figure 3.28). Sur l'exemple, le point M est calculé comme le milieu des points A et B et en ajoutant une valeur aléatoire gaussienne. Les valeurs aléatoires suivent une loi de Gauss de moyenne nulle. La déviation standard est déterminée expérimentalement. Pour utiliser cet algorithme, nous attribuons une valeur aux quatre coins de la grille (sauf s'ils sont déjà connus) et nous nous servons des points calculés sans les modifier. A la fin de l'interpolation, tous les pixels du PixMap ont une valeur.

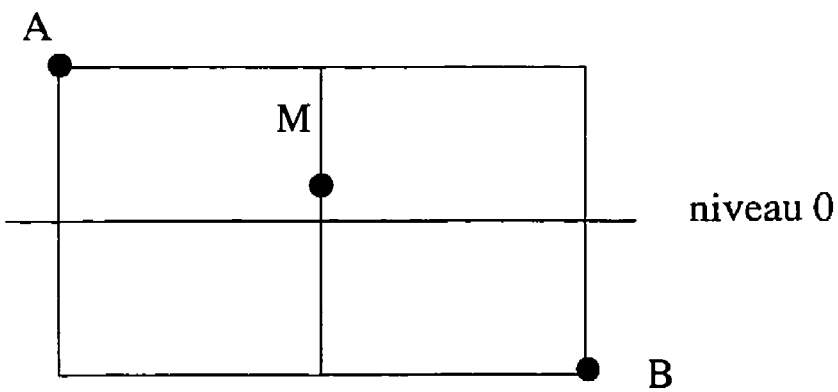


Figure 3.28 : Interpolation par les fractales

Calcul de la texture du sol Pour le sol, la procédure est très semblable. Seules les transformations concernant le positionnement du sol dans le repère du monde changent.

Calcul de la texture des raccords Pour les raccords, il faut calculer la normale en chaque point. L'équation de la courbe décrivant le raccord est donnée par :

$$Y = aX^2 + Y_0.$$

La dérivée en un point de la courbe s'écrit :

$$\frac{dY}{dX} = 2aX.$$

Nous obtenons la normale en un point du raccord dirigée vers le centre de l'intersection : $\mathbf{n} = (2aX, -1)$.

D'autre part, il se pose un problème supplémentaire par rapport aux murs et au sol qui est dû aux occlusions. En effet, la méthode qui consiste à examiner l'étiquette d'un point n'est pas suffisante pour déterminer si un point appartenant à un raccord est occulté ou non. En effet, nous n'attribuons pas d'étiquette spécifique à un raccord. Un point appartenant à un raccord a pour étiquette celle de l'un des deux murs qu'il raccorde. Par exemple, dans la figure 3.29, un point du raccord 6 aura pour étiquette 2 ou 3.

Mais deux points du raccord peuvent se projeter sur le pixel image, l'un occultant

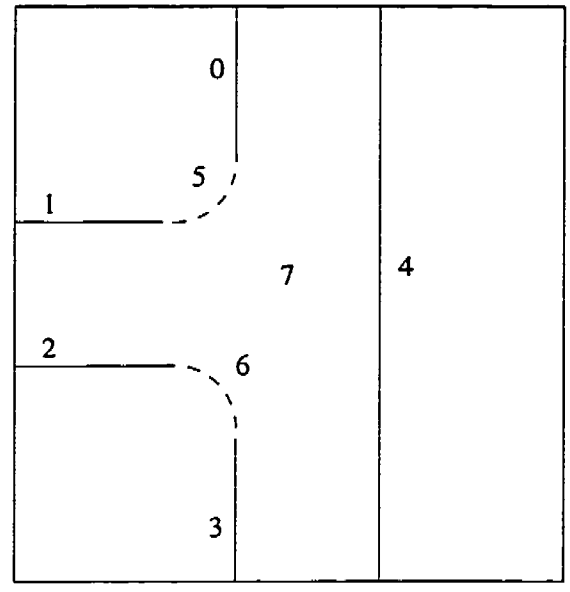


Figure 3.29 : Occlusion d'un point appartenant à un raccord

l'autre. Si nous nous fions uniquement à l'étiquette de la projection d'un point, nous nous ne pouvons pas nous assurer qu'il est effectivement visible. Pour résoudre ce problème, nous nous ramenons au modèle topologique de l'intersection. Nous cherchons à déterminer si :

1. Le rayon lumineux provenant de la caméra et illuminant le point M considéré intersecte une paroi (un mur ou un raccord). Appelons I ce point d'intersection s'il existe.

2. Le point M est occultant ou occulté par I.

Considérons le cas de l'intersection du rayon lumineux avec un mur (voir la figure 3.30).

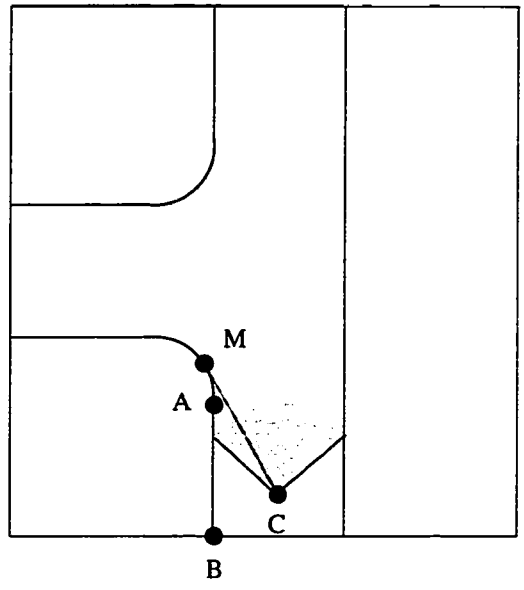


Figure 3.30 : Intersection du rayon lumineux avec un mur

Nous déterminons les équations du mur et du rayon lumineux dans le repère de l'intersection à partir des points A et B délimitant le mur et des points C (position de la caméra) et M définissant le rayon lumineux.

La droite (AB) a pour équation : $(AB) : y = ax + b$. La droite (CM) a pour équation : $(CM) : y = cx + d$. Connaissant les coordonnées des points A, B, C et M, nous calculons a, b, c et d. Nous en déduisons les coordonnées du point intersection I de (AB) avec (CM) :

$$x_I = (b - d)/(c - a),$$

$$y_I = ax_I + b.$$

Le point I obtenu n'est susceptible d'occulter M que si I appartient effectivement au segment $[A; B]$. Il faut donc que la relation suivante soit vérifiée :

$$\mathbf{AI} = k\mathbf{AB}, \quad 0 \leq k \leq 1$$

Enfin, il faut déterminer lequel des deux points occulte l'autre. M est occulté par I si :

$$\text{dist}(C, I) < \text{dist}(C, M).$$

Pour déterminer l'intersection d'un rayon lumineux avec un raccord, nous nous plaçons dans le repère du raccord (voir la figure 3.31).

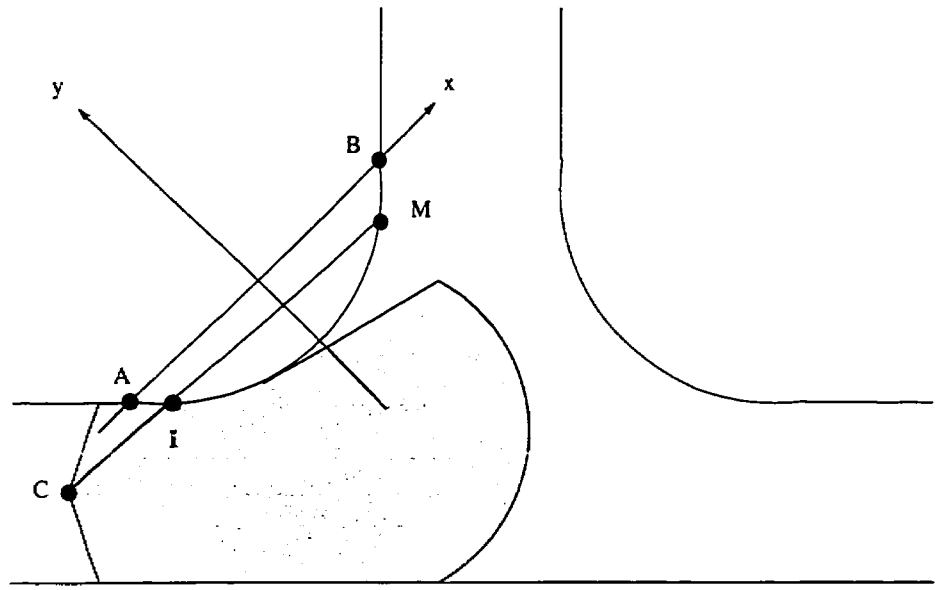


Figure 3.31 : Intersection du rayon lumineux avec un raccord

Nous déterminons l'équation du rayon lumineux dans le repère du raccord. L'équation du raccord (parabolique) est connue. L'intersection de l'équation du rayon lumineux avec l'équation du raccord revient à résoudre une équation du second degré. Comme pour le mur, nous vérifions que la ou les solutions obtenues appartiennent

effectivement au raccord :

$$x_A \leq x_I \leq x_B.$$

Enfin, nous regardons quel point, I ou M, occulte l'autre de la même manière que pour un mur.

Insertion de la texture dans le modèle Un objet texture est ainsi construit pour chaque objet géométrique et lui est associé dans un sous-groupe. Le modèle géométrique est le groupe formé par l'ensemble des sous-groupes contenant les objets de l'intersection (voir la figure 3.32).

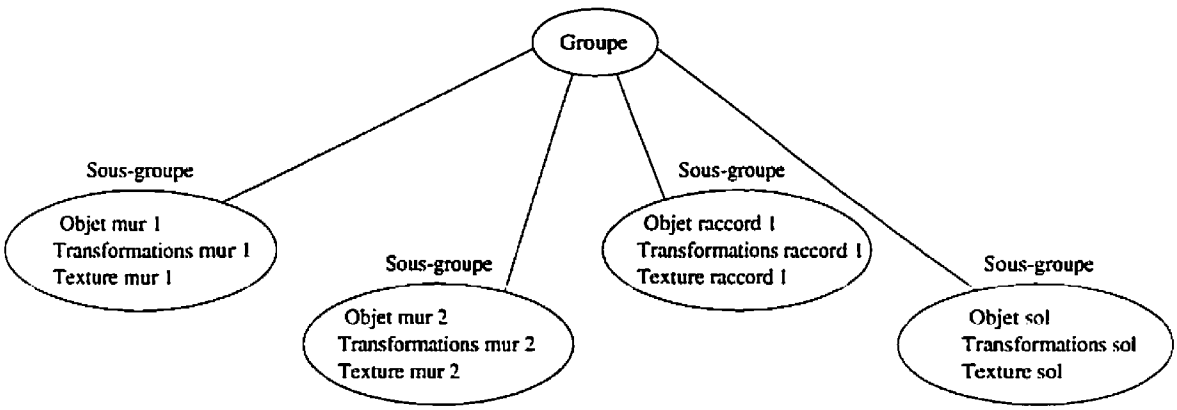


Figure 3.32 : Modèle 3D d'une intersection

Nous sauvegardons ce modèle dans un fichier metafile, fichier QD3D qui permet de sauvegarder des objets QD3D.

3.2.4 Vue panoramique de l'intersection

Apple (Cannon 1996) propose deux méthodes pour créer des vues panoramiques. Une première approche consiste à simuler une caméra réelle, à la tourner pour générer une série d'images et ensuite à assembler ces images en utilisant des techniques de déformation pour créer une vue panoramique 360° c'est-à-dire un fichier PICT. Mais cette méthode nécessite l'utilisation d'un outil d'assemblage QuickTime VR. C'est pourquoi nous avons opté pour la seconde solution qui consiste à rendre la vue panoramique directement.

Nous commençons par lire le modèle stocké dans le fichier metafile. Ce modèle ne comporte pas d'objet éclairage. Nous rajoutons donc un spot lumineux que nous plaçons un peu au dessus de la caméra et que nous faisons tourner en même temps qu'elle. Une procédure permet ensuite de calculer le centre du modèle à partir de ses dimensions. Ce centre correspond en fait au centre de l'intersection où nous plaçons la caméra.

L'idée est de prendre des tranches très minces dans les images et de les "coller" bout à bout, au lieu de prendre des images entières. Nous tournons donc à chaque fois la caméra d'un angle très petit et nous stockons seulement une tranche mince au centre de l'image, approximant ainsi un cylindre (voir la figure 3.33). Plus la tranche est petite et moins il y a d'erreurs de déformation.

Cette représentation se construit au fur et à mesure que le robot repasse par l'intersection. Son premier passage lui fournit une vue partielle de l'intersection. Il

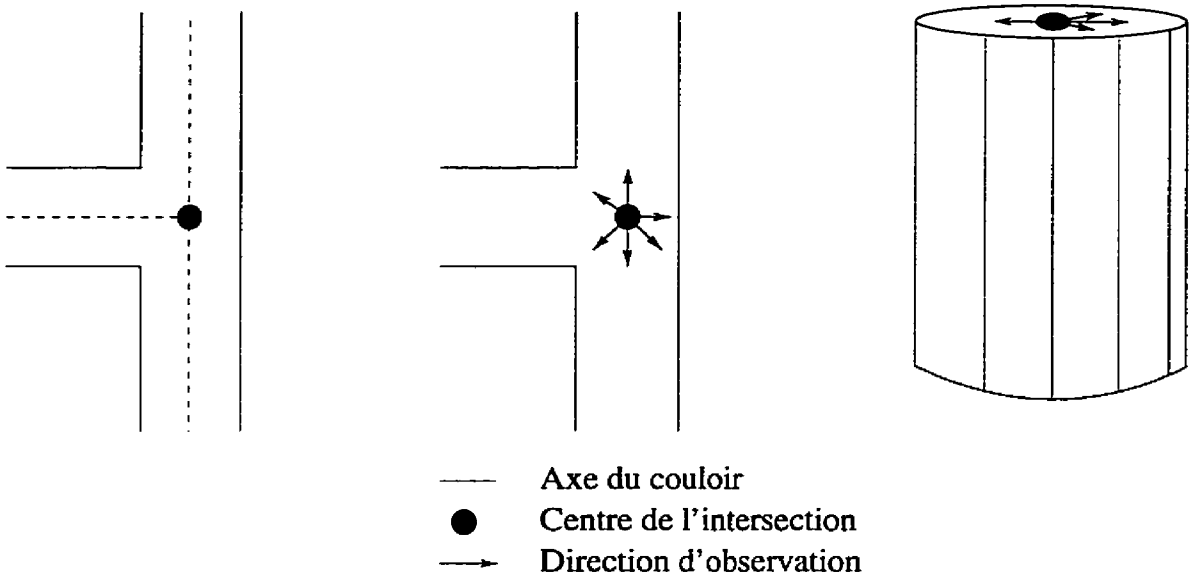


Figure 3.33 : Vue panoramique d'une intersection

complètera cette vue quand il retournera à l'intersection par un autre chemin.

3.3 Reconnaissance neuronale

Comparées aux méthodes basées sur la mise en correspondance de caractéristiques, les méthodes basées sur les réseaux de neurones sont plus susceptibles de donner de meilleurs résultats dans le cas d'une mine. En effet, dans ce type d'environnement, il y a peu de caractéristiques aisément détectables permettant la mise en correspondance. De plus, une fois que le réseau de neurones a effectué son apprentissage, son temps de réponse est très rapide pour la reconnaissance. L'apprentissage peut être long, mais nous n'avons pas d'exigence particulière sur le temps de calcul car le modèle peut être établi après que les scènes ont été enregistrées.

Lippman (1987) propose une taxonomie des réseaux de neurones les plus classiques utilisés pour la reconnaissance de formes. Il classe dans les mémoires associatives à apprentissage supervisé le réseau de Hopfield et le perceptron multi-couches. Nous choisissons le perceptron multi-couches comme étant l'architecture de réseaux de neurones la plus utilisée dans ce type de problème. Ce type de réseau accepte des valeurs non binaires en entrée, ce qui est notre cas puisque nous traitons des images en niveaux de gris.

Le problème peut être traité de deux façons. Une première méthode consiste à extraire des caractéristiques des exemples avec certaines propriétés d'invariance et de les présenter au réseau. Le rôle du réseau est alors de partionner l'espace des caractéristiques en régions de décision correspondant à chaque classe. La deuxième approche consiste à laisser le soin au réseau de trouver les invariances dans l'ensemble de données échantillons sur lequel il fait son apprentissage. Dans cette approche, les invariances sont incorporées dans la structure du réseau. Le réseau est ensuite capable de discerner les invariances dans des données inconnues et de les reconnaître. Ces deux approches sont décrites plus en détail dans l'annexe A.3.

La deuxième approche semble la plus appropriée étant donnée la difficulté de trouver les caractéristiques pertinentes à extraire dans les images sur lesquelles nous travaillons. En effet, des critères géométriques ne nous apporteraient rien de plus que ce que nous avons déjà avec notre modèle topologique. Nous devons chercher une information supplémentaire dans la texture même des objets de la scène. Dans

l'approche proposée par Greespan et al. (1994), le système fonctionne bien pour des textures qui sont très différentes à l'œil humain. Or dans le cas d'une mine, d'une intersection à l'autre, la texture des parois est très semblable. En fait, ce que le réseau apprend dans (Greespan et al. 1994), c'est à classer des textures. En revanche, Jain et Karu (1996) montrent que les réseaux de neurones ont cette capacité d'apprendre à différencier les régions texturées d'une image directement à partir d'exemples qui leur sont présentés. De plus, cette approche nous permet d'exploiter les propriétés d'invariance en rotation de notre représentation. En effet nous apprenons au réseau à associer à une même intersection, un ensemble de vues panoramiques ayant subi une rotation autour du centre de l'intersection.

Finalement, cette approche connexionniste nous permet d'exploiter à la fois les propriétés sur la géométrie et la texture de la scène pour apprendre et reconnaître les intersections.

3.3.1 Entrée du réseau

Nous définissons *les directions initiales d'observation* d'une vue panoramique comme les directions des axes composant l'intersection. Le système génère tout d'abord la vue panoramique de l'intersection rencontrée (dans le sens horaire), puis, par simple rotation de cette vue panoramique, l'ensemble des vues panoramiques commençant à partir de toutes les directions initiales d'observations. Cet ensemble de vues panoramiques permet de proposer au réseau de neurones une représentation invariante

en rotation. Par exemple, pour une intersection en T, il y a trois directions initiales d'observation. Cela donne donc trois vues panoramiques. Nous supposons que le système peut faire une erreur d'au plus 10° sur la direction d'observation initiale dans le sens horaire ou antihoraire. Pour prendre en compte ces erreurs, le système génère des vues panoramiques supplémentaires en tournant une vue panoramique de $+10°$ ou $-10°$ (voir la figure 3.34). Le réseau de neurones apprend ces vues panoramiques supplémentaires avec les autres vues panoramiques pour être indépendant des erreurs d'orientation.

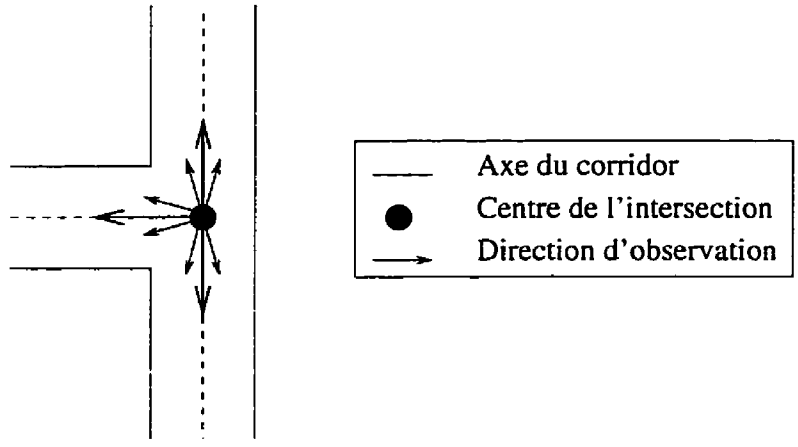


Figure 3.34 : Invariance en rotation de la vue panoramique

Avant d'être présentée au réseau, une vue panoramique est traitée de la manière suivante (voir la figure 3.35).

- Elle est dépliée à partir de la direction initiale d'observation.
- Sa résolution est réduite à 24×128.

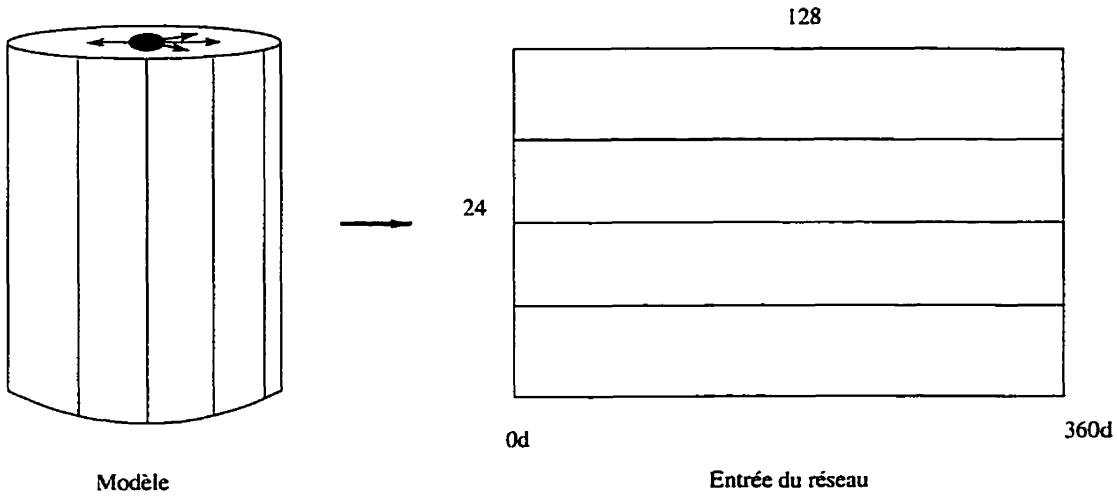


Figure 3.35 : Vue cylindrique dépliée et partionnée en bandes

- Elle est partionnée en 24 bandes horizontales de 128 pixels qui vont constituer les 24 vecteurs d'entrée du réseau.

Les vecteurs d'entrée ont pour valeur des intensités de gris qui sont normalisées.

3.3.2 Modèle d'un neurone

Un neurone est une unité de traitement composée des éléments suivants (Haykin 1994) (voir la figure 3.36) :

- Un ensemble de connexions synaptiques caractérisées par leur poids. Un signal xj, à l'entrée de la synapse j connectée au neurone k, est multiplié par le poids synaptique w_{kj}.

- Un additionneur effectuant la somme des signaux d'entrée pondérés par leurs synapses respectives.

- Une fonction d'activation qui permet de limiter l'amplitude de sortie du neurone.
- Un seuil θ_k qui permet de diminuer l'entrée fournie à la fonction d'activation. L'entrée fournie à la fonction d'activation peut être au contraire augmentée par un biais (de valeur négative).

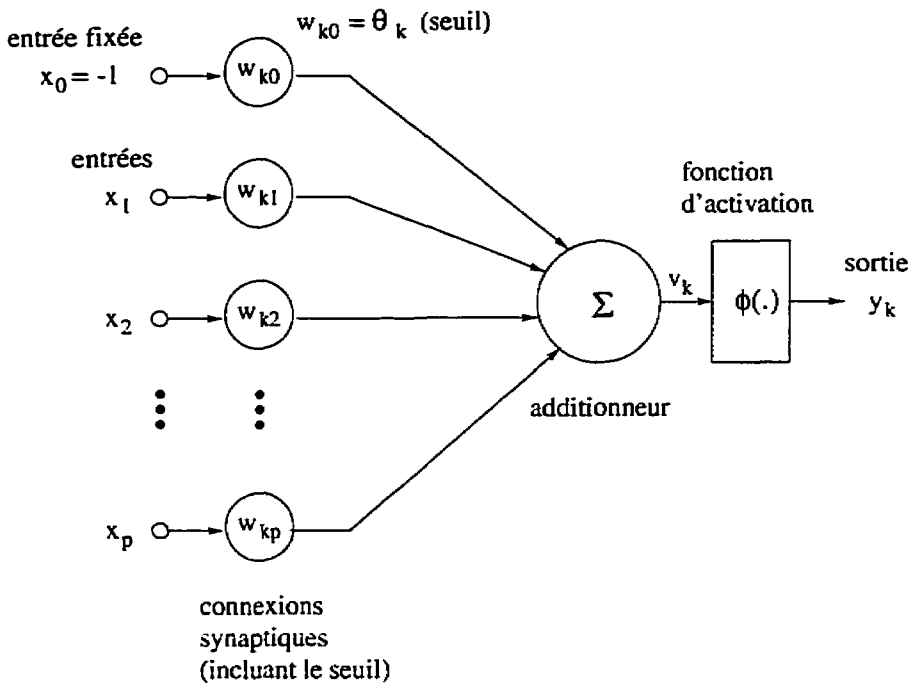


Figure 3.36 : Modèle d'un neurone

Mathématiquement, le neurone est décrit par les équations suivantes :

$$v_k = \sum_{j=1}^{p} w_{kj} x_j,$$

$$y_k = \phi(v_k).$$

La fonction d'activation $\phi(.)$ définit le niveau d'activité du neurone à son entrée. Nous utilisons une fonction d'activation de type sigmoïde : la fonction log hyper-

bolique qui a pour équation :

$$\phi(v) = \frac{1}{1 + \exp(-v)}.$$

Cette fonction limite la sortie du neurone à des valeurs comprises entre 0 et 1.

3.3.3 Architecture du réseau de neurones

Le réseau est de type perceptron multi-couches comportant 24 neurones sur la première couche, 24 neurones sur la deuxième couche et 64 neurones sur la couche de sortie correspondant à 64 intersections différentes. Chaque neurone de la première couche reçoit un vecteur de 128 entrées. Il n'y a pas de chevauchement entre les bandes. Les deux autres couches sont complètement connectées (voir la figure 3.37). Toutes les fonctions d'activation sont des sigmoïdes.

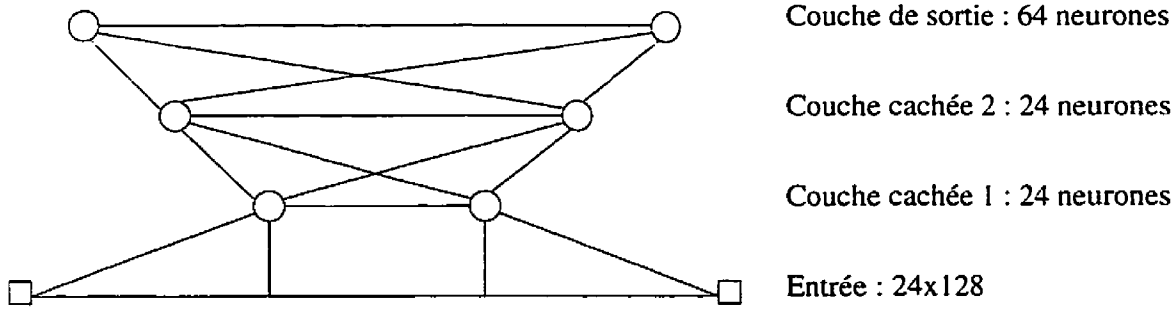


Figure 3.37 : Architecture du réseau

La première couche cachée extrait les caractéristiques locales de chaque bande de la vue panoramique. La deuxième couche cachée permet d'extraire des caractéristiques globales. Le neurone gagnant de la couche de sortie ("Winner Takes All") permet

de classer l'intersection en indiquant son identificateur dans la base de données des intersections.

3.3.4 Apprentissage

Nous utilisons la rétropropagation (RP) comme règle d'apprentissage pour ajuster les poids synaptiques du réseau. Le processus de rétropropagation de l'erreur se fait en deux étapes : une étape "vers l'avant" où des exemples sont présentés en entrée au réseau et activent les neurones des différentes couches du réseau. Dans cette étape, les poids sont fixés. La deuxième étape est une étape "vers l'arrière" où les poids sont ajustés selon la règle de correction d'erreur. La réponse réelle du réseau est comparée à la réponse désirée et les poids synaptiques sont ajustés pour réduire l'erreur sur la sortie avec un certain taux d'apprentissage (voir la figure 3.38).

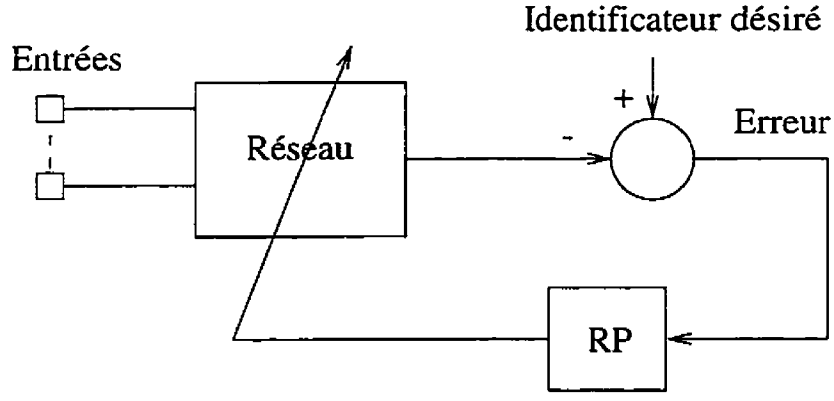


Figure 3.38 : Fonctionnement du réseau pendant l'apprentissage

Le mode d'entraînement est le mode "pattern" (selon le terme employé par Haykin (1994)), dans lequel l'ajustement des poids est effectué à chaque exemple de l'ensem-

ble d'apprentissage présenté en entrée. Ce mode d'entraînement s'oppose au mode "batch", dans lequel l'ajustement des poids se fait après avoir présenté tous les exemples. Dans ce mode, seule l'erreur globale sur tous les exemples est considérée. Par ailleurs, pour un apprentissage efficace, les données sont présentées dans un ordre aléatoire.

Plus le taux d'apprentissage est petit, plus l'apprentissage est lent. Plus nous augmentons le taux d'apprentissage, plus le réseau apprend vite mais plus il est instable (les poids oscillent). Pour augmenter le taux d'apprentissage et éviter l'instabilité, nous modifions la règle d'apprentissage en rajoutant un terme d'inertie ou "momentum" (Rumehart et al. 1986).

Les couches du réseau sont numérotées de 0 (couche d'entrée) à 3 (couche de sortie). L'algorithme d'apprentissage est le suivant :

1. *Initialisation* : Les poids du réseau sont initialisés aléatoirement selon une distribution uniforme et dans une plage de valeurs restreinte de façon à ne pas saturer le réseau. Cette plage de valeurs est fonction des entrées du réseau.

2. *Présentation des données* : À chaque itération, toutes les vues panoramiques contenues dans la base de données sont présentées au réseau. Les étapes 3 et 4 sont réalisées à chaque itération.

3. *Phase "avant"* : Soit une vue panoramique représentée par un vecteur d'entrée $x(n)$ et le vecteur de sortie $d(n)$ correspondant à l'identificateur de l'intersection correspondante désiré. Nous calculons couche par couche l'activité v_j^l du

neurone j dans la couche l :

$$v_j^l = \sum_{i=0}^{p} w_{ji}^{(l)}(n) y_i^{(l-1)}(n)$$

où $y_i^{(l-1)}(n)$ est la sortie du neurone i dans la couche précédente $l-1$ à l'itération n et $w_{ji}^{(l)}(n)$ est le poids de la connexion synaptique entre le neurone j de la couche l et le neurone i. p est la dimension du vecteur d'entrée du neurone j. Pour la première couche cachée partiellement connectée, chaque neurone reçoit en entrée un vecteur de dimension 128. Pour la deuxième couche cachée complètement connectée, chaque neurone reçoit toutes les sorties de la première couche cachée, soit 128 entrées. La couche de sortie, elle aussi complètement connectée, reçoit 128 entrées. Pour $i = 0$, nous avons $y_0^{(l-1)}(n) = \theta_j(l)(n)$ où $\theta_j(l)(n)$ est le seuil appliqué au neurone j de la couche l. La sortie du neurone j de la couche l s'écrit :

$$y_j^{(l)}(n) = \frac{1}{1 + \exp(-v_j^{(l)}(n))}.$$

Si le neurone j appartient à la première couche cachée, c'est-à-dire si $l = 1$, alors nous avons :

$$y_j^{(0)}(n) = x_j(n)$$

où $x_j(n)$ est le je élément du vecteur d'entrée $x(n)$. Si le neurone j appartient à la couche de sortie, c'est-à-dire si $l = 3$, alors nous avons :

$$y_j^{(2)}(n) = o_j(n)$$

avec $o_j(n)$ la sortie du neurone j à l'itération n. Nous calculons le signal d'erreur $e_j(n)$ entre la sortie désirée $d_j(n)$ et la sortie obtenue $o_j(n)$:

$$e_j(n) = d_j(n) - o_j(n)$$

4. *Phase de rétropropagation de l'erreur* : Nous calculons les gradients locaux δ du réseau en procédant couche par couche, de la couche de sortie vers la couche d'entrée :

   $\delta_j^{(3)} = e_j^{(3)}(n) o_j(n)[1 - o_j(n)]$, pour le neurone j de la couche de sortie 3

   $\delta_j^{(l)}(n) = y_j^{(l)}(n)[1 - y_j^{(l)}(n)] \sum_k \delta_k^{(l+1)}(n) w_{kj}^{(l+1)}(n)$, pour une couche cachée l

   Les poids synaptiques de la couche l du réseau sont ajustés selon la règle delta généralisée :

   $$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha[w_{ji}^{(l)}(n) - w_{ji}^{(l)}(n-1)] + \eta \delta_j^{(l)}(n) y_i(l-1)(n),$$

   où η est le taux d'apprentissage et α l'inertie.

5. *Itération* : Nous considérons que l'algorithme de rétropropagation a convergé quand l'erreur quadratique moyenne sur une itération est inférieure à un certain seuil.

Classiquement, la phase d'apprentissage est séparée de la phase de reconnaissance.

L'apprentissage sert à ajuster les poids du réseau. Ensuite les poids restent fixes et le réseau cherche simplement à reconnaître ce qui est présenté en entrée. Dans notre système, le réseau apprend au fur et à mesure de la progression du robot. Comme l'algorithme de rétropropagation minimise l'erreur globale, après quelques intersections, le réseau "oublie" ce qu'il a appris. Pour pallier à ce défaut, nous nous proposons d'adapter la méthode de rafraîchissement de la mémoire présentée par Pomerleau (1991) pour le système ALVINN. Les nouvelles intersections ainsi que les anciennes sont stockées dans une base de données d'entraînement. Pour chaque nouvelle intersection, par exemple neuf vues panoramiques dans le cas d'une intersection en T, neuf vues panoramiques sont retirées de la base de données d'entraînement et remplacées par les vues panoramiques de la nouvelle intersection. Les vues panoramiques à retirer sont choisies pour une partie parmi celles qui produisent la plus petite erreur en sortie et pour le reste, aléatoirement. La rétropropagation de l'erreur est alors effectuée sur la base de données d'entraînement. Les vues panoramiques retirées pourront éventuellement être remises dans la base de données d'entraînement à l'intersection suivante.

3.3.5 Reconnaissance

La figure 3.39 décrit le fonctionnement du réseau pendant la phase de reconnaissance.

Nous mesurons l'erreur entre la sortie du réseau et le neurone gagnant pour valider

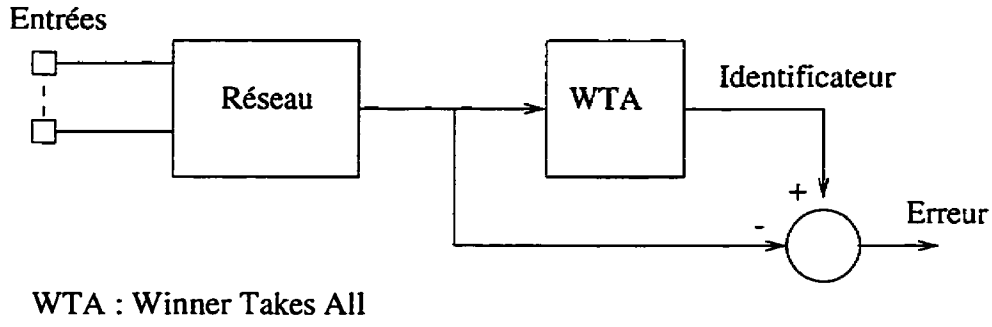


Figure 3.39 : Fonctionnement du réseau pendant la phase de reconnaissance

le résultat obtenu. Le vecteur gagnant, de dimension 64, est formé de 0 et d'un 1 correspondant au neurone gagnant. Le système calcule la somme des moindres carrés entre le vecteur gagnant et le vecteur de sortie du réseau. Si l'erreur est supérieure à un certain seuil, le système considère que l'intersection est inconnue.

3.4 Expérimentations

3.4.1 Performances du réseau de neurones

Nous présentons les résultats obtenus sur des images de synthèse d'intersections à trois branches. La figure 3.40 montre l'exemple d'une intersection en T avec la trajectoire suivie par le robot. La figure 3.41 montre les images prises aux positions A et B.

La figure 3.42 montre la vue panoramique que le système a généré à partir de la séquence d'images et du modèle topologique de l'intersection et la figure 3.43, la vue panoramique correspondante présentée au réseau.

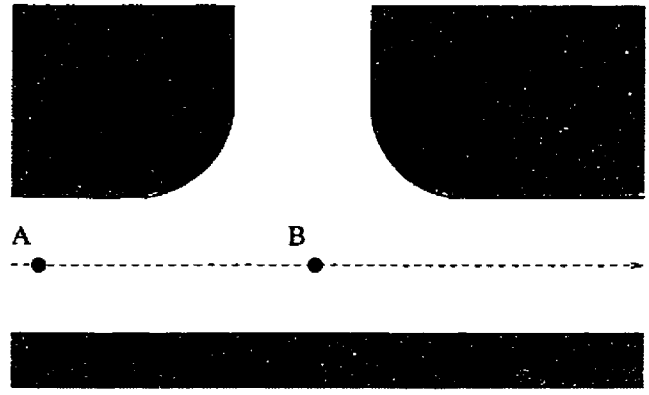


Figure 3.40 : Exemple d'intersection utilisée pour les tests

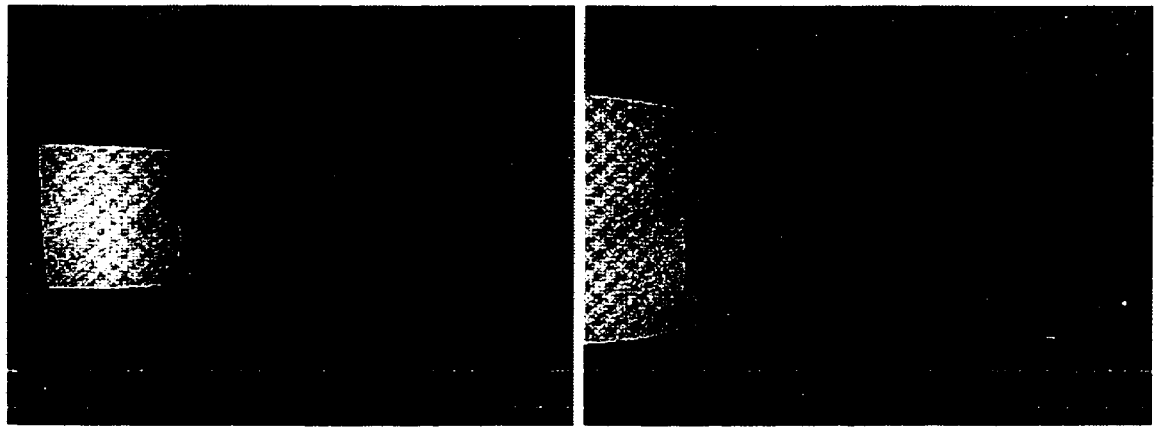


Figure 3.41 : Exemples d'images de synthèse d'intersections

Figure 3.42 : Exemple de vue panoramique générée

Figure 3.43 : Entrée du réseau (128 × 24)

Nous avons travaillé sur onze intersections différentes. Pour chaque intersection, nous avons considéré différentes trajectoires (entre une et trois) du robot pour générer les vues panoramiques (voir la figure 3.44). Nous avons obtenu en tout vingt-sept vues panoramiques.

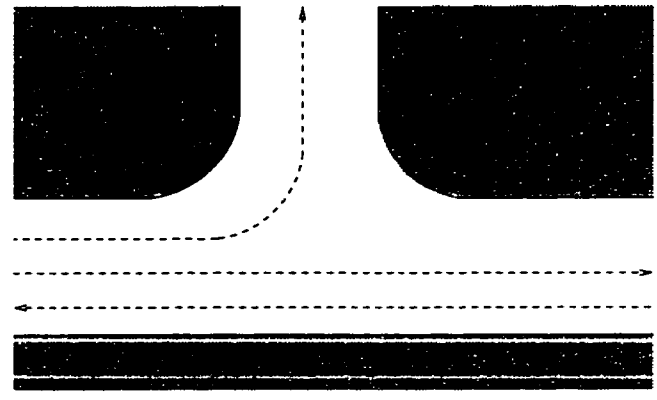


Figure 3.44 : Selon la trajectoire du robot, le système ne peut pas générer exactement la même vue panoramique car d'une trajectoire à l'autre, des parties différentes de l'intersection sont occultées.

Nous avons utilisé ces vues panoramiques pour vérifier que :

- Le réseau de neurones peut reconnaître une intersection pour laquelle la direction d'observation initiale est erronée jusqu'à 10°.

- Le réseau de neurones peut reconnaître une intersection que le robot a déjà

rencontré mais en arrivant par un autre chemin. Cela signifie en particulier que d'un passage à l'autre, des parties différentes de la scène ont été cachées.

- Le réseau de neurones peut distinguer des intersections que le robot n'a jamais vues, des intersections déjà apprises.

Nous avons entraîné notre réseau de neurones sur huit vues panoramiques correspondant à huit intersections différentes. Quinze autres vues panoramiques générées à partir d'une trajectoire différente ont été utilisées pour tester des intersections déjà rencontrées. Nous avons aussi utilisé trois intersections inconnues. Le tableau 3.1 montre les résultats obtenus.

Le taux d'apprentissage et l'inertie ont une grande influence sur les résultats. Si le taux d'apprentissage est trop grand (0.9) et l'inertie trop petite (0.1), le système est instable. La convergence peut être très rapide ou très lente. Dans ce dernier cas, cela signifie que le réseau est piégé dans des minimums locaux et qu'il n'arrive pas à apprendre. Avec de tels paramètres d'apprentissage, il est logique d'obtenir des taux de reconnaissance variables. En moyenne, les taux de mauvaise classification ne sont pas très bons (10% pour des intersections déjà rencontrées et 40% pour des intersections inconnues).

Avec un taux d'apprentissage plus petit (0.1), la convergence est plus lente (environ 160 itérations par intersection) mais les résultats sont plus fiables : seulement 3% ou 4% de mauvaise classification pour des intersections déjà rencontrées. Les meilleurs résultats sont obtenus pour un taux d'apprentissage de 0.1 et une inertie

Tableau 3.1 : Performances du réseau de neurones

l_r=0.1 m_c=0.9		PVs avec des erreurs de rotation	Intersections déjà rencontrées	Intersections jamais vues
	Bien reconnu	93%	50%	
	Mal reconnu	0	4%	25%
	Non identifié	7%	46%	75%
	Nb. moyen d'itérations	160		
l_r=0.9 m_c=0.1		PVs avec des erreurs de rotation	Intersections déjà rencontrées	Intersections jamais vues
	Bien reconnu	95%	56%	
	Mal reconnu	0	10%	40%
	Non identifié	5%	34%	60%
	Nb. moyen d'itérations	variable		
l_r=0.1 m_c=0.5		PVs avec des erreurs de rotation	Intersections déjà rencontrées	Intersections jamais vues
	Bien reconnu	94%	62%	
	Mal reconnu	0	3%	25%
	Non identifié	6%	35%	75%
	Nb. moyen d'itérations	162		

de 0.5. Le réseau peut identifier des intersections déjà visitées en arrivant par un corridor différent avec des taux de réussite de 62% et un bon facteur de confiance (3% de mauvaise classification). Il peut aussi différencier une intersection jamais vue des intersections qu'il a déjà apprises avec 75% de réussite.

Les résultats montrent aussi que le réseau gère très bien des erreurs sur la direction initiale d'observation du robot, avec des taux de réussite de 95% et sans mauvaise classification.

3.4.2 Simulation

La figure 3.45 montre l'environnement que nous avons simulé ainsi que le chemin suivi au cours de l'exploration du robot.

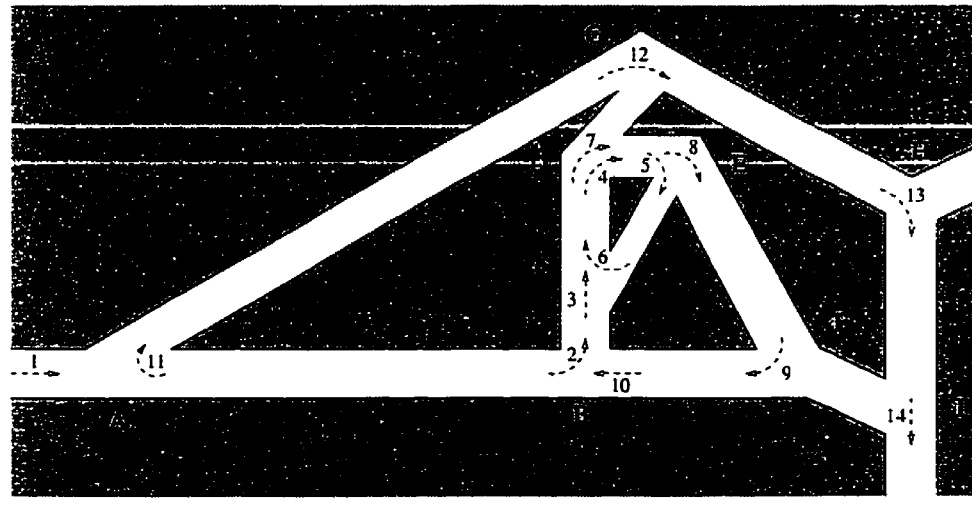


Figure 3.45 : Environnement simulé

La méthode est la suivante. Quand le robot arrive à une intersection, il essaie de

l'identifier. Cela signifie que le système construit la vue panoramique de l'intersection et la présente au réseau de neurones. Plus précisément, la vue panoramique est tournée neuf fois (trois par direction initiale d'observation) et chaque vue panoramique résultante est présentée au réseau de neurones donnant ainsi neuf réponses. Chaque "bonne" réponse du réseau (avec une erreur inférieure à un certain seuil) donne un vote pour une intersection. L'intersection identifiée est celle qui gagne le plus de votes. Nous calculons le facteur de confiance f de la reconnaissance :

$$f = \frac{\text{Nombre de votes de l'intersection gagnante}}{\text{Nombre de vues panoramiques pour une intersection}}$$

Si f est supérieur à 0.5, l'intersection est considérée comme reconnue. Dans ce cas, sa représentation est mise à jour dans la base de données. Autrement, il s'agit d'une nouvelle intersection qui est ajoutée à la base de données et apprise par le réseau de neurones.

Le tableau 3.2 montre les résultats de la simulation.

Les résultats sont cohérents avec les performances du système évaluées dans la section 3.4.1. Nous constatons que le système a une bonne capacité à ne pas confondre des intersections qu'il n'a jamais rencontrées avec celles qu'il a déjà apprises. Seule l'intersection C est confondue avec l'intersection B, mais avec un facteur de confiance peu élevé. Par ailleurs, lors du deuxième passage du robot, le système reconnaît les intersections D, E et B. Pour l'intersection D, il a suivi le même chemin, mais pour les intersections E et B, sa trajectoire est différente. En revanche, le système n'est

Tableau 3.2 : Résultats de la simulation

Étape	Intersection rencontrée	Intersection identifiée	f	Commentaires
1	A	?		
2	B	?		
3	C	B	0,6	Jamais rencontrée auparavant
4	D	?		
5	E	?		
6	C	C	0,1	Déjà rencontrée
7	D	D	1	Déjà rencontrée
8	E	E	0,9	Déjà rencontrée
9	F	?		
10	B	B	0,6	Déjà rencontrée
11	A	?		N'identifie pas
12	G	?		
13	H	?		
14	I	?		

pas capable d'identifier les intersections C et A lors du deuxième passage du robot, probablement parce que les points de vue sont trop différents.

3.4.3 Facteurs d'influence

3.4.3.1 Influence relative de la texture et de la géométrie

Nous avons montré que notre système est capable d'apprendre et de reconnaître des intersections. Nous montrons dans cette section quelle est la part de la géométrie et quelle est la part de la texture dans le processus d'identification. Pour cela, nous faisons apprendre au réseau des intersections qui ont la même géométrie, auxquelles nous appliquons des textures différentes. Et par ailleurs, nous lui apprenons aussi des

intersections qui ont des géométries différentes, auxquelles nous appliquons la même texture. Nous avons sélectionné pour cela différents types de texture : "brique", "bois", "marbre". Le tableau 3.3 montre les résultats obtenus.

Tableau 3.3 : Influence relative de la texture et de la géométrie

l_r=0.1 m_c=0.5		PVs avec des erreurs de rotation	Intersections déjà rencontrées	Intersections jamais vues
	Bien reconnu	83%	78%	
	Mal reconnu	0	0%	22%
	Non identifié	17%	22%	78%

Ces résultats confirment les résultats précédents et, en particulier, montrent que le réseau différencie les intersections en se basant sur les deux critères de la géométrie et de la texture. En effet, deux intersections de même géométrie ne sont pas confondues, pas plus que deux intersections ayant la même texture. Les taux de bonne classification sont similaires à ceux obtenus lors de l'évaluation de la performance du réseau, ce qui ne serait pas le cas si le réseau ne se basait que sur un seul des deux critères.Le résultat obtenu serait entaché de beaucoup plus d'erreurs.

3.4.3.2 Influence des erreurs de modélisation

Nous évaluons dans cette section l'influence des erreurs de modélisation sur la reconnaissance. Nous utilisons le réseau entraîné sur le parcours considéré dans la section 3.4.2. Nous refaisons passer le robot aux intersections B et F. Nous introduisons des erreurs sur leur modèle topologique comme le montre la figure 3.46.

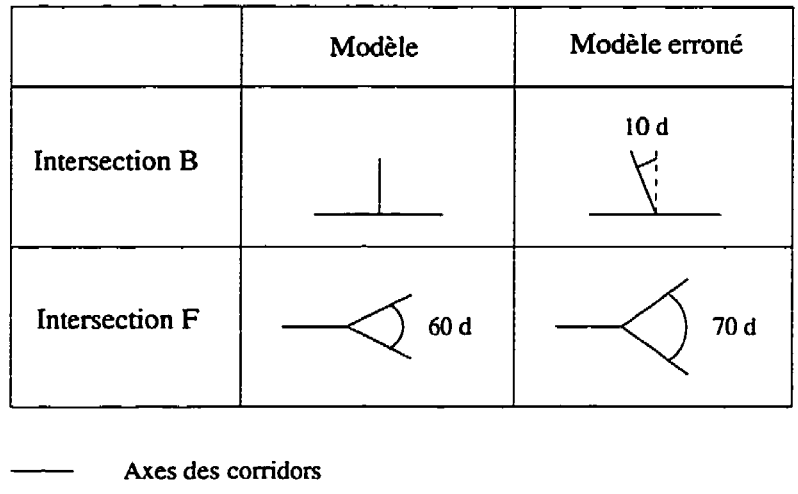


Figure 3.46 : Erreurs de modélisation sur les intersections B et F

Sur ces modèles erronés, nous construisons les vues panoramiques à partir des séquences d'images prises par le robot. Nous proposons ces vues panoramiques au réseau de neurones que nous avons entraîné lors de sa précédente exploration. Nous obtenons les résultats suivants. L'intersection B est reconnue par le réseau avec un facteur de confiance de 0.1 s'il suit le chemin 2 et avec un facteur de confiance de 0.6 s'il suit le chemin 10. L'intersection F est reconnue avec un facteur de confiance de 0.6. Ces résultats montrent que le réseau est capable d'identifier l'intersection malgré les erreurs de modélisation. La confiance est très faible pour l'intersection B selon le chemin 2 mais nous constatons que le réseau classe correctement l'intersection. Dans les deux autres cas, l'intersection est reconnue avec un bon facteur de confiance.

Par ailleurs, nous avons testé quelle serait l'influence d'erreurs sur les mesures odométriques, autrement dit sur la trajectoire du robot. Nous reprenons toujours le même réseau ayant appris l'environnement simulé sur la figure 3.45. Sur les modèles

topologiques des intersections B et F, nous construisons les vues panoramiques à partir des séquences d'images et en introduisant une erreur latérale sur la trajectoire de 0.1 unités (sachant que la largeur du corridor est de 2 unités). Cette erreur induit nécessairement des erreurs sur la texture appliquée sur les murs et le sol. Avec ces données, le réseau reconnaît l'intersection B avec un facteur de confiance de 0.2 pour le chemin 2 et un facteur de confiance de 0.6 pour le chemin 10. L'intersection F est reconnue avec un facteur de confiance de 0.7. Donc, malgré des erreurs de mesure odométriques, le système est capable de reconnaître des intersections précédemment rencontrées.

Ces résultats montrent la capacité de généralisation de notre réseau, qui peut identifier des intersections précédemment rencontrées malgré des erreurs sur le modèle topologique ou des erreurs sur la trajectoire.

Conclusion

Nous avons conçu et développé un système de vision artificielle qui permet à un robot mobile de se localiser dans un environnement inconnu. La méthode que nous proposons, basée sur la construction et la reconnaissance de vues panoramiques, en fait son intérêt et son originalité.

Ce projet apporte sa contribution à la recherche dans les domaines de la modélisation de l'environnement, du graphisme tridimensionnel, de la réalité virtuelle et de l'intelligence artificielle. En effet, nous avons proposé un modèle des intersections général notamment grâce à l'introduction du concept de super-intersection. Ce modèle permet donc de représenter aussi bien des intersections simples que complexes (ce qui n'était pas le cas avec le modèle de Jochem et al. (1995)). Un des avantages de notre méthode est que le modèle est déterminé directement à partir de l'image. Une autre innovation que nous apportons dans les techniques de détection de routes et d'intersections, est une méthode simple et efficace pour calculer le nombre de branches composant l'intersection directement à partir de l'image. À titre d'application immédiate, ce résultat améliorerait considérablement la méthode de Jochem et al. (1995).

En effet, comme le nombre de branches n'était pas prédéterminé, les auteurs devaient ajouter incrémentalement des branches de routes au modèle dans le but de l'améliorer mais avec le risque d'introduire des erreurs. Dans beaucoup d'approches, la détection des caractéristiques est basée sur la couleur. Ces méthodes sont inapplicables dans le cas d'une mine étant données les mauvaises conditions d'illumination et l'uniformité des couleurs. Nous avons donc proposé une méthode basée sur la construction de vues panoramiques pour la reconnaissance des intersections. Notre modèle possède des propriétés d'invariance qui le rendent indépendant de la trajectoire suivie par le robot, ce qui était la grande faiblesse de l'approche de Zheng et Tsuji (1992). Nous avons proposé des techniques pour construire de telles vues panoramiques et pour les mettre à jour au fur et à mesure de la progression du robot pour une amélioration constante du modèle. Nous avons aussi proposé une architecture de réseaux de neurones qui permet d'identifier les intersections rencontrées. Ce qui fait la force de cette approche neuronale, c'est la capacité de généralisation du réseau : il est capable de reconnaître des intersections dont le robot n'a perçu qu'une vue partielle. De plus, notre approche est adaptée à un environnement dynamique, ce qui est une des grandes difficultés en matière de modélisation de l'environnement.

Les résultats montrent que le système est capable de modéliser correctement les intersections. Par ailleurs, notre système est capable de reconnaître une intersection quel que soit le chemin d'arrivée du robot avec 62% de réussite. Le robot est aussi capable de distinguer une intersection inconnue avec 75% de réussite. Nous avons

montré que des erreurs de modélisation ou des erreurs de mesure sur la trajectoire n'empêchent pas le réseau d'identifier des intersections précédemment rencontrées. Un des avantages de la méthode est que le processus de reconnaissance est très rapide. Le réseau de neurones pourrait être facilement parallélisé pour améliorer la performance. L'apprentissage est lent, mais il peut se faire plus tard, une fois que les scènes ont été enregistrées. De plus, l'ajout d'une nouvelle intersection n'accroit pas la taille de l'ensemble d'apprentissage. Par conséquent, le temps de convergence ne devrait pas augmenter au cours de l'exploration du robot.

Les travaux réalisés ont fait l'objet de publications dans des conférences (Béranger et Hervé 1996*c*), (Béranger et Hervé 1996*b*), (Béranger et Hervé 1996*a*) et de rapports techniques (Béranger et Hervé 1996*d*), (Béranger et Hervé 1995).

Pour compléter ce travail, il faudrait tester le système sur des images réelles. Nous pensons que les résultats seraient meilleurs sur des images réelles que sur des images de synthèse car une scène réelle contiendrait nécessairement plus de caractéristiques propres à la distinguer d'une autre scène. C'est d'ailleurs ce qu'avait expérimenté Pomerleau (1991) pour le système ALVINN. Des possibilités d'amélioration du système seraient à explorer en rajoutant le plafond au modèle ou en modélisant finement les raccords. Un atout pour ce système et aussi une possibilité d'ouverture découlent de sa modularité. Nous pourrions en effet sélectionner des repères autres que des intersections (pour des scènes extérieures par exemple) et conserver notre méthode de reconnaissance basée sur les vues panoramiques de l'environnement. La

difficulté serait de déterminer le centre des vues panoramiques. Mais tout le processus de construction de la vue panoramique serait le même.

Bibliographie

Apple Computer Inc. (1995), *3D Graphics Programming With QuickDraw 3D*, Addison Wesley, Reading, MA.

BALLARD, D. H. et BROWN, C. M. (1982), *Computer Vision*, Prentice Hall, Englewood Cliffs, New Jersey.

BASRI, R. et RIVLIN, E. (1993), Localization using combinations of model views, dans *Proc. of the IEEE Fourth International Conference on Computer Vision*, Berlin, Allemagne, pp. 226–230.

BEHRINGER, R. et HÖTZL, S. (1994), Simultaneous estimation of pitch angle and lane width from the video image of a marked road, dans *Proc. of the International Conference on Intelligent Robots and Systems*, Munich, Allemagne, pp. 966–973.

BÉRANGER, V. et HERVÉ, J.-Y. (1995), Reconnaissance d'intersections dans un environnement de corridors, rapport technique GRPR-RT-9511, Groupe de Recherche en Perception et Robotique, École Polytechnique, Montréal, QC.

BÉRANGER, V. et HERVÉ, J.-Y. (1996*a*), Détection et reconnaissance d'intersections dans un environnement de corridors, dans *Actes du 64e congrès de l'ACFAS*, Montréal, QC.

BÉRANGER, V. et HERVÉ, J.-Y. (1996*b*), Neural-network based recognition of mine environments, dans *Proc. Canadian Conference on Electrical and Computer Engineering*, Calgary, Canada.

BÉRANGER, V. et HERVÉ, J.-Y. (1996*c*), Recognition of intersections in corridor environments, dans *Proc. of the IEEE International Conference on Pattern Recognition*, Vienne, Autriche, pp. 133–137.

BÉRANGER, V. et HERVÉ, J.-Y. (1996*d*), Recognition of intersections in corridors environments, rapport technique GRPR-RT-9606, École Polytechnique, Montréal, QC.

BRON, C. et KERBOSCH, J. (1973), Finding all cliques of an undirected graph, *Communications of the ACM* **16**(9).

CANNING, J., KIM, J. J. et ROSENFELD, A. (1987), Symbolic pixel labeling for curvilinear feature detection, dans *Proc. 1987 DARPA Image Understanding Workshop*, Los Angeles, CA, pp. 242–256.

CANNON, J. (1996), Quicktime VR 1.0 panorama movie file format, rapport technique Technote 1035 - Release 1.0, Apple Computer, Cupertino, CA.

CHANDRAN, S., DAVIS, L. S. et DEMENTHON, D. (1987), An overview of vision-based navigation for autonomous land vehicles 1986, rapport technique CAR-TR-285, Computer Vision Laboratory, University of Maryland, College Park, MD.

CHATILA, R. et LAUMOND, J.-P. (1985), Position referencing and consistent world modeling for mobile robots, dans *Proc. of the IEEE International Conference on Robotics and Automation*, St. Louis, MO, pp. 138–145.

CHIOIU, A. et HERVÉ, J.-Y. (1996), Detection of the ground space for visual navigation, dans *Actes du 64e congrès de l'ACFAS*, Montréal, QC.

CRISMAN, J. et THORPE, C. E. (1993), SCARF: a color vision system that tracks roads and intersections, *IEEE Trans. on Robotics and Automation* **9**(1).

DICKMANNS, E.-D. (1989), Vehicule guidance by computer vision, dans *High Precision Navigation*, K. Linkwitz and V. Hangleiter editors, Springer-Verlag, New York, pp. 86–96.

DUDEK, G., JENKIN, M., MILIOS, E. et WILKES, D. (1991), Robotic exploration as graph construction, *IEEE Trans. on Robotics and Automation* **7**(6).

DUDEK, G., JENKIN, M., MILIOS, E. et WILKES, D. (1993), Map validation and self-location in a graph-like world, dans *Proc. of the 13rd International Conference on Artificial Intelligence*, pp. 1648–1653.

EDLINGER, T. et PUTTKAMER, E. (1994), Exploration of an indoor-environment by an autonomous mobile robot, dans *Proc. of the International Conference on Intelligent Robots and Systems*, Munich, Allemagne, pp. 1278–11284.

ELFES, A. (1987), Sonar-based real-world mapping and navigation, *IEEE Journal of Robotics and Automation* **RA-3**(3).

ENGELSON, S. et MCDERMOTT, D. (1992), Error correction in mobile robot map learning, dans *Proc. of the 1992 IEEE International Conference on Robotics and Automation*, Nice, France, pp. 2555–2560.

EVANS, J. R. et MINIEKA, E. (1992), *Optimization Algorithms for Networks and Graphs*, Dekker, New York.

EVEN, S. (1979), *Graph Algorithms*, Computer Science Press, Potomac, Maryland.

FALCO, P. et MCBRIDE, P. (1996), Generating QuickTime VR movies from QuickDraw 3D, *Develop* **25**.

FUKUMI, M., OMATU, S., TAKEDA, F. et KOSAKA, T. (1992), Rotation-invariant neural pattern recognition system with application to coin recognition, *IEEE Trans. on Neural Networks* **3**(2).

GREESPAN, H., GOODMAN, R., CHELLAPA, R. et ANDERSON, C. H. (1994), Learning texture discrimination rules in a multiresolution system, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **16**(9).

GREINER, R. et ISUKAPALLI, R. (1996), Learning to select useful landmarks, *IEEE Trans. on Systems, Man, and Cybernetic* **26**(3).

HARRIS, C. (1996), Re-synthetizing reality: from image to graphic to interactive imaging, *Advanced Imaging* **11**(7).

HAYKIN, S. (1994), *Neural Networks: A Comprehensive Foundation*, Macmillan, Hamilton, Canada.

HORN, B. K. P. (1986), *Robot Vision*, MIT Press, Cambridge, MA.

HOUGH, P. V. C. (1962), Method and means of recovering complex patterns, rapport technique US Patent 3 069 654.

HUMMEL, R. et ZUCKER, S. (1983), On the foundations of relaxation labeling processes, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **5**(3).

ILLINGWORTH, J. et KITTLER, J. (1988), A survey of the Hough transform, *Computer Vision, Graphics, and Image Processing* **44**(1).

JAIN, A. K. et KARU, K. (1996), Learning texture discrimination masks, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **18**(2).

JANÉT, J. A., GUTIERREZ-OSUNA, R., CHASE, T. A., WHITE, M. et LUO, R. C. (1995), Global self-localization for autonomous mobile robots using self-organizing Kohonen neural networks, dans *Proc. of the International Conference on Intelligent Robots and Systems*, Japon, pp. 504–509.

JOCHEM, T. M., POMERLEAU, D. A. et THORPE, C. E. (1995), Vision-based neural network road and intersection detection and traversal, dans *Proc. of the International Conference on Intelligent Robots and Systems*, Japon, pp. 344–349.

KIM, H. et NAM, K. (1995), Object recognition of one-DOF tools by a back-propagation neural net, *IEEE Trans. on Neural Networks* **6**(2).

KOSAKA, A. et KAK, A. (1992), Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties, *Computer Vision, Graphics, and Image Processing* **56**(3).

KRIEGMAN, D. J. et BINFORD, T. O. (1988), Generic models for robot navigation, dans *Proc. of the IEEE International Conference on Robotics and Automation*, Philadelphia, PA, pp. 746–751.

KUIPERS, B. et BYUN, Y. (1987), A qualitative approach to robot exploration and map-learning, dans *Proc. Workshop on Spatial Reasoning and Multi-sensor Fusion*, pp. 390–404.

LABROSSE, F., HERVÉ, J.-Y. et COHEN, P. (1996*a*), Modélisation de mines avec un modèle générique d'objets, dans *Proc. Canadian Conference on Electrical and Computer Engineering*, Calgary, Canada.

LABROSSE, F., HERVÉ, J.-Y. et COHEN, P. (1996*b*), GNOMine: An application of the generic GNOME model to the representation of mines, rapport technique GRPR-RT-9611, École Polytechnique, Montréal, QC.

LACROIX, S., CHATILA, R., FLEURY, S., HERRB, M. et SIMÉON, T. (1994), Autonomous navigation in outdoor environment: Adaptive approach and experiment, dans *Proc. of the International Conference on Robotics and Automation*, Vol. 1, San Diego, CA, pp. 426–432.

LECUN, Y., BOSER, B., DENKER, J., HENDERSON, D., HOWARD, R. E., HUBBARD, W. et JACKEL, L. D. (1990), Handwritten digit recognition with a back-propagation network, dans *Advances in Neural Information Processing Systems*, San Mateo, CA, pp. 396–404.

LEVITT, T. S. et LAWTON, D. T. (1990), Qualitative navigation for mobile robots, *Artificial Intelligence* **44**(1-2).

LI, S., NAGATA, S. et TSUJI, S. (1995), A navigation system based upon panoramic representation, dans *Proc. of the International Conference on Intelligent Robots and Systems*, Japon, pp. 142–147.

LI, S., TSUJI, S. et HAYASHI, A. (1996), Qualitative representation of outdoor environment along route, dans *Proc. of the IEEE International Conference on Pattern Recognition*, Vienne, Autriche, pp. 176–180.

LIOU, S.-P. et JAIN, R. C. (1987), Road following using vanishing points, *Computer Vision, Graphics, and Image Processing* **39**(1).

LIPPMAN, R. (1987), An introduction to computing with neural nets, *IEEE Acoustics Speech and Signal Processing Magazine* **4**.

LOWE, D. G. (1987), The viewpoint consistency constraint, *International Journal of Computer Vision* **1**(1).

MARR, D. (1982), *Vision*, Freeman, San Francisco, CA.

MASE, K. et NISHIRA, H. (1996), Computing the field-of-view of a stitched panorama to create FoV sensitive virtual environments, dans *Proc. of the IEEE International Conference on Pattern Recognition*, Vienne, Autriche, pp. 151–155.

MATTHIES, L. et ELFES, A. (1988), Integration of sonar and stereo range data using a grid-based representation, dans *Proc. IEEE International Conference on Robotics and Automation*, Philadelphia, PA, pp. 727–733.

MCINTOSH, J. H. et MUTCH, K. M. (1988), Matching straight lines, *Computer Vision, Graphics, and Image Processing* **43**(3).

MCMILLAN, L. et BISHOP, G. (1995), Plenoptic modeling: an image-based rendering system, dans *Proc. of SIGGRAPH 95*, Los Angeles, CA.

PEITGEN, H., SAUPE, D. et BARNSLEY, M. F. (1988), *The Science of Fractal Images*, Springer-Verlag, Berlin, Allemagne.

PILON, M. (1996), Détection et analyse des occlusions basées sur les combinaisons d'images, Master's thesis, École Polytechnique, Montréal, QC.

POMERLEAU, D. (1991), Efficient training of artificial neural networks for automous navigation, *Neural Computation* **3**(1).

RENCKEN, W. (1994), Autonomous sonar navigation in indoor, unknown and unstructured environments, dans *Proc. of the International Conference on Intelligent Robots and Systems*, Munich, Allemagne, pp. 431–438.

ROHR, K. (1994), Towards model-based recognition of human movements in image sequences, *Computer Vision, Graphics, and Image Processing: Image Understanding* **59**(1).

ROSENFELD, A. (1986), Axial representations of shape, *Computer Vision, Graphics, and Image Processing* **33**(2).

RUMEHART, D. E., HINTON, G. E. et WILLIAMS, R. J. (1986), Learning representations by back-propagating errors, *Nature* **323**.

SÄCKINGER, E., BOSER, B., BROMLEY, J., LECUN, Y. et JACKEL, L. (1992), Application of the ANNA neural network chip to high-speed character recognition, *IEEE Trans. on Neural Networks* **3**(3).

SEITZ, S. et DYER, C. (1996), Toward image-based scene representation using view morphing, dans *Proc. of the IEEE International Conference on Pattern Recognition*, Vienne, Autriche, pp. 84–89.

SOUTHWELL, D., BASU, A., FIALA, M. et REYDA, J. (1996), Panoramic stereo, dans *Proc. of the IEEE International Conference on Pattern Recognition*, Vienne, Autriche, pp. 378–382.

STEIN, F. et MEDIONI, G. (1995), Map-based localization using the panoramic horizon, *IEEE Trans. on Robotics and Automation* **11**(6).

SUGIHARA, K. (1988), Some location problems for robot navigation using a single camera, *Computer Vision, Graphics, and Image Processing* **42**(1).

THORPE, C., HEBERT, M., KANADE, T. et SHAFER, S. (1989), Vision and navigation for the Carnegie Mellon Navlab, dans *High Precision Navigation*, K. Linkwitz and V. Hangleiter editors, Springer-Verlag, New York, pp. 97–122.

WAGNER MINING EQUIPMENT CO. (1989), *Sales Manual.*

WERNECKE, J. (1994), *The Inventor Mentor*, Addison Wesley, Reading, MA.

WOLBERG, G. (1990), *Digital Image Warping*, IEEE Computer Society Press, Los Alamitos, CA.

ZHENG, J. et TSUJI, S. (1992), Panoramic representation for route recognition by a mobile robot, *International Journal of Computer Vision* **9**(1).

Annexe A

Revue bibliographique

A.1 Modélisation de l'environnement

Une des principales difficultés rencontrées en robotique mobile est liée à la complexité de l'environnement. Un robot mobile doit faire face à une grande variété d'objets tant sur le plan qualitatif que quantitatif, tels que des obstacles à éviter ou des objets à manipuler. Considérons par exemple le cas d'un véhicule minier chargé d'opérer dans des zones à risques d'une mine. Comment le rendre capable de se situer dans cet environnement qui lui est inconnu ? Il est essentiel pour cela de doter le robot mobile des moyens de construire et d'utiliser des modèles de son environnement. Ce problème de modélisation de l'environnement est sous-jacent aux problèmes de planification de chemin, de navigation, de planification et d'exécution de tâches spécifiques, etc.

Une difficulté majeure du problème vient de la complexité de l'environnement. Il y a une grande diversité d'éléments à traiter. Comment modéliser des éléments aussi différents que des obstacles à éviter, des objets à manipuler ou encore des évènements (par exemple une porte qui se ferme) ? Une réponse à cette question est fournie dans (Labrosse et al. 1996*a*, Labrosse et al. 1996*b*). Ces travaux décrivent la modélisation d'une mine par un modèle générique d'objets. Par ailleurs, et c'est ce que nous évoquerons plus en détail, dans un environnement non-structuré, il est très difficile de définir des points d'intérêt sur des objets. En général, les objets ne sont pas polyédriques mais ont des formes courbes. Le problème consiste à construire et à mettre à jour un modèle (des objets et de l'espace), qui reste consistant tout au long de l'exploration du robot, aussi bien lorsqu'il explore de nouvelles régions que lorsqu'il retourne dans des régions déjà visitées. Un dernier aspect du problème est la localisation du robot. La principale difficulté réside dans l'incertitude sur les mesures fournies par les capteurs qui peut engendrer une accumulation d'erreurs importante sur de grandes distances, tant sur la position des objets que sur la position du robot lui-même.

Nous présentons dans cette section trois approches classiques du problème. Tout d'abord, nous verrons une approche "métrique" qui consiste à représenter l'environnement sous forme de carte de type géographique. Puis, nous présentons une approche "probabiliste" qui prend en compte les incertitudes sur les mesures. L'espace est décomposé en régions probablement vides ou probablement occupées. Enfin,

une troisième approche, qualifiée de "qualitative", se base sur l'idée que les humains reconnaissent des repères caractéristiques.

A.1.1 Modélisation "métrique"

Cette modélisation, proposée par Chatila et Laumond (1985), permet de représenter l'environnement selon différents niveaux d'abstraction. Les auteurs décrivent principalement le niveau le plus bas qui est une carte 2D très précise de l'environnement.

Modèle Le monde est représenté selon trois modèles, sur la supposition que les objets sont polyédriques. Tout d'abord, au plus bas niveau, un modèle géométrique représente la position des sommets des objets projetés sur le sol dans un repère absolu. Ensuite, au niveau suivant se trouve un modèle qui permet de structurer l'espace et pour lequel deux notions sont introduites : la place et le connecteur. Une place est un espace qui a une unité fonctionnelle ou topologique. Une station de travail, une pièce sont des exemples. Un connecteur permet de relier des places (exemple : un corridor). Le modèle topologique est constitué de graphes de connectivité entre les places. Le plus bas niveau est représenté par un graphe d'adjacence construit de la manière suivante. L'espace est partionné en polygones convexes qui seront les places. Un bord commun entre deux polygones est un connecteur. Le graphe d'adjacence représente les places reliées par les connecteurs. Au deuxième niveau, certaines places du graphe sont regroupées pour introduire un contexte sémantique. Il peut s'agir par exemple d'une pièce, d'une porte, d'un corridor, etc. Enfin, un troisième modèle, dit

sémantique, contient les informations sur les propriétés et les relations des objets.

Trois moyens permettent de décrire la position du robot :

- Par la position dans un repère absolu. Cela nécessite des balises de repère fixes et connues.
- Par l'intégration de la trajectoire.
- Par la position relative par rapport à des objets ou à des places.

Méthode Les auteurs adoptent une approche multi-sensorielle pour répondre à différents besoins : connaître à la fois la forme et la localisation des objets, ainsi que la localisation du robot lui-même. De plus, les senseurs n'ont pas tous les mêmes capacités selon la proximité ou l'éloignement des objets. Les auteurs ont utilisé deux types de senseurs : un laser qui fournit une carte de profondeur de l'environnement et un système odométrique pour l'intégration de la trajectoire.

Le modèle prend en compte les erreurs. Les erreurs peuvent être dues aux mesures ou aux algorithmes d'interprétation. Les erreurs de mesure systématiques sont éliminées par calibration. Les erreurs aléatoires sont modélisées par une variable aléatoire gaussienne.

A tout moment, le robot connaît :

- le modèle de l'environnement ;
- sa position ;

- un modèle géométrique de l'environnement perçu à l'instant présent.

Le problème consiste donc à mettre à jour le modèle grâce à sa position et au modèle de perception et à corriger la position du robot si cela est possible.
La méthode se base sur les principes suivants :

- Associer un repère relatif à chaque objet nouvellement découvert et des incertitudes sur le repère et sur les points de l'objet.

- Introduire un modèle de prédiction sur ce qui devrait être vu lors du prochain déplacement.

- Mettre en correspondance le modèle de prédiction avec le modèle de perception pour mettre à jour les positions précédentes du robot.

La construction de la carte se fait de la façon suivante. Tout d'abord, un modèle de prédiction représentant les zones qui devraient être vues, les zones invisibles et les zones inconnues lors du prochain déplacement est construit à partir du modèle de l'environnement. Il faut ensuite identifier les zones vues précédemment : connaissant la position du robot, ce qui doit être vu est mis en correspondance avec ce qui est effectivement vu. Les perceptions nouvelles sont ajoutées au modèle dans un repère relatif à un objet précédemment identifié ou dans un nouveau repère. La position du robot est mise à jour à partir du modèle courant et du modèle de perception. Cela permet de corriger les positions du robot précédentes et de mettre à jour le modèle en conséquence. La figure A.1 montre le modèle final obtenu dans un environnement

composé de deux pièces.

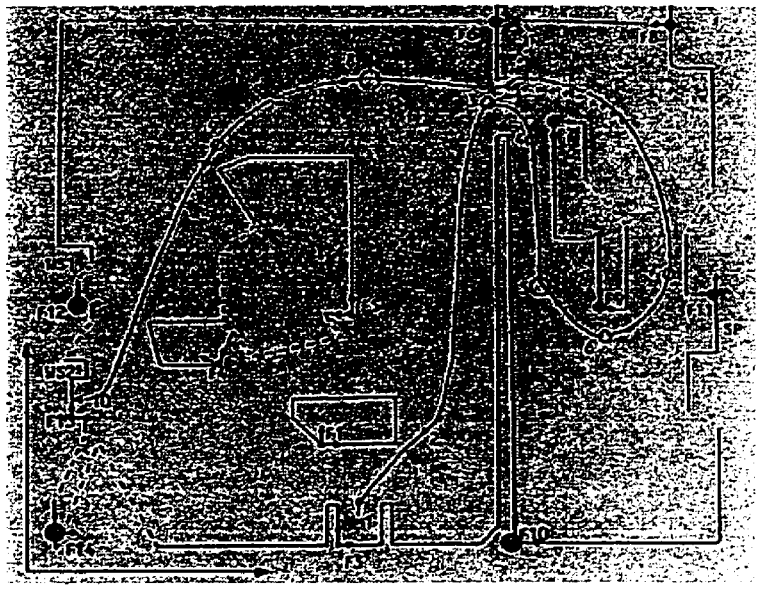


Figure A.1 : Modèle métrique final pour la trajectoire représentée (Chatila et Laumond 1985)

Lacroix et al. (1994) proposent une approche adaptative pour la navigation autonome dans un environnement extérieur. Le comportement du robot est adapté au terrain rencontré et présente trois modes de navigation. Le mode de navigation "planifié 2D" s'applique quand le terrain est quasiment plat et repose sur l'exécution d'une trajectoire 2D planifiée en utilisant une description binaire de l'environnement, qui s'exprime en termes d'espace traversable ou non. Le mode de navigation "planifié 3D" s'applique en terrain accidenté et exige un modèle précis du terrain pour lequel une trajectoire 3D est planifiée. Le mode "réflexe" est gouverné par un but donné au robot ou encore simplement par la nécessité d'éviter les obstacles et n'exige donc au-

cun modèle de l'environnement. Les auteurs définissent différentes représentations de l'environnement pour répondre aux besoins de planification de trajectoire, de localisation et de prise de décision. Pour la planification de trajectoire 2D, une représentation binaire est utilisée. Pour la planification de trajectoire 3D, une carte d'élévation est construite sur une grille cartésienne. Pour la localisation, le robot peut utiliser un ensemble de points 3D ou une carte globale de repères détectés. Pour décider des stratégies de navigation, les auteurs utilisent une représentation topologique sous forme de graphe de connexion. Les modes de navigation planifiés sont testés sur un terrain de 20×50 mètres carrés constitué d'espaces plats et accidentés et comportant des obstacles. L'environnement est totalement inconnu au robot. Ces modes de navigation fonctionnent séparément mais l'intégration des différents modes de navigation n'est pas encore réalisée.

Edlinger et Puttkamer (1994) proposent une stratégie d'exploration d'un environnement intérieur et en construisent une carte géométrique et topologique. Un capteur optique est utilisé pour déterminer la position des obstacles ainsi que la position du robot dans le système de coordonnées global. L'algorithme proposé se compose d'une stratégie d'exploration locale qui opère au niveau d'une pièce et d'une stratégie d'exploration globale. L'idée est que le robot se tient dans une "bulle" virtuelle initialement limitée par la portée maximale de ses senseurs. Cette bulle se déforme au fur et à mesure de la progression du robot. Quand le senseur détecte un objet, la surface de l'objet devient une frontière réelle de la bulle. L'exploration est terminée quand

la bulle n'a plus que des frontières réelles ou des frontières virtuelles appartenant à des portes. Le robot cherche toujours à se déplacer vers les frontières virtuelles de la bulle. En fait, la bulle est représentée par deux sous-structures. La bulle des senseurs gère les frontières réelles et virtuelles sous forme de lignes. La bulle de l'horizon contient les informations sur l'environnement et est faite de l'intersection de cercles. Une méthode est proposée pour fusionner la bulle des senseurs courante avec la bulle des senseurs globale. Cette méthode est basée sur des contraintes de voisinage entre les lignes et sur la minimisation de la longueur de toutes les lignes virtuelles de la bulle résultante. Les auteurs proposent aussi une méthode pour déterminer les points d'intérêt de l'environnement à explorer. La figure A.2 montre la carte topologique générée par l'exploration (locale) d'une pièce simple.

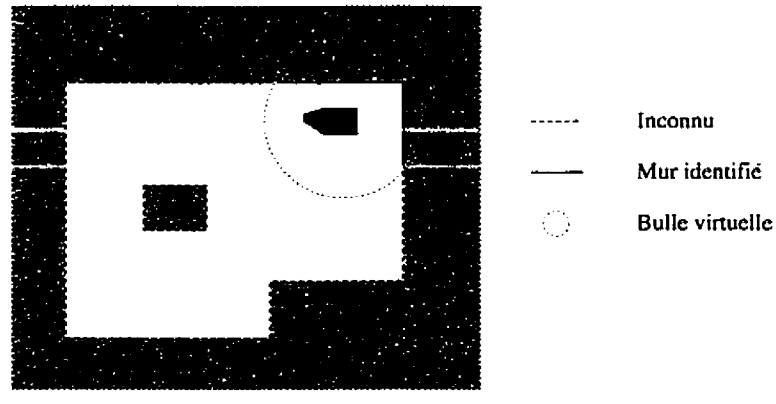


Figure A.2 : Carte topologique générée par l'exploration d'une pièce simple selon l'approche de Edlinger et Puttkamer (1994)

A.1.2 Modélisation "probabiliste"

Modèle Cette approche, proposée par Elfes (1987), permet de modéliser les incertitudes sur les erreurs. Le modèle est composé de plusieurs représentations hiérarchisées. La représentation de plus bas niveau est une représentation des données sous forme de carte bidimensionnelle composée de régions probablement occupées, probablement vides et inconnues. La représentation de plus niveau est une représentation symbolique.

Méthode Le robot est équipé d'un sonar dont les unités sont disposées sur un anneau. Les mesures sont interprétées comme des volumes de l'espace délimité par le cône de chacun des senseurs (d'angle solide 30°). Cette information est modélisée par des profils de probabilité qui sont projetés sur une carte 2D de régions probablement occupées, probablement vides et inconnues. Le résultat obtenu est une grille de cellules avec un facteur associé : 0 pour inconnu, [-1,0) pour vide, (0,1] pour occupé. Les mesures prises par les senseurs et en des positions différentes du robot sont intégrées dans la carte. Cela permet de réduire les incertitudes au fur et à mesure du déplacement du robot. Les espaces vides (ou pleins) qui se recouvrent augmentent la certitude. De plus, les frontières entre espaces vides et espaces occupés sont affinées. Pour la mise à jour de la carte ou pour la reconnaissance d'espaces connus, une méthode de mise en correspondance par convolution est proposée. La carte finale montre les régions probablement occupées, probablement vides et inconnues (voir la

figure A.3).

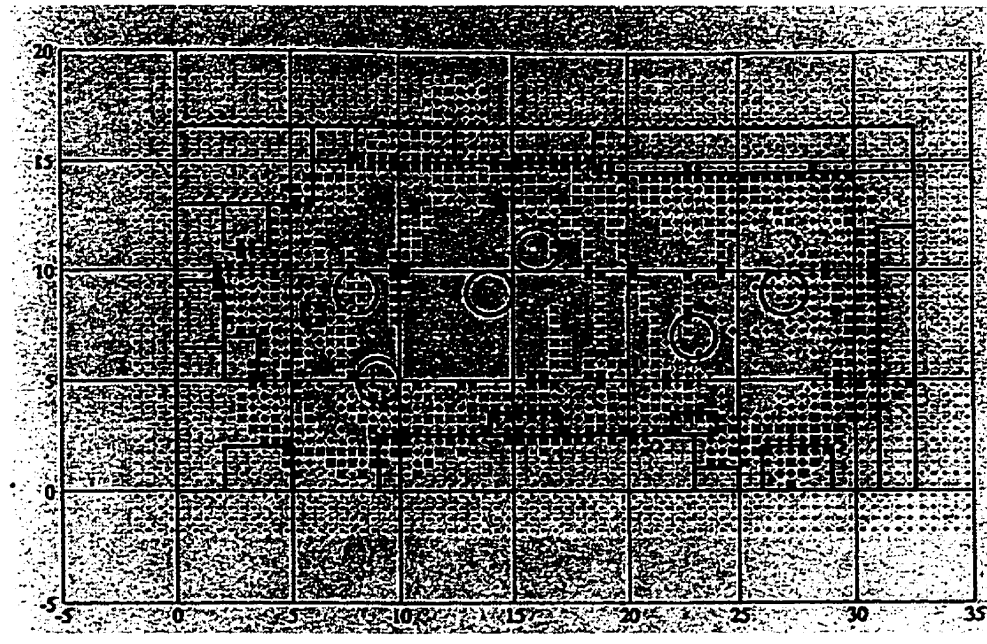


Figure A.3 : Carte bidimensionnelle représentant les régions d'occupation. Les régions probablement vides sont représentées par des espaces blancs (grande fiabilité) ou par des + (faible fiabilité). Les régions probablement occupées sont représentées par des x, les régions inconnues par des points. Les cercles montrent les positions du robot d'où les mesures ont été prises. Les traits pleins représentent l'environnement (Elfes 1987).

Axes de représentation L'auteur propose une hiérarchie de représentations selon trois axes. L'axe d'abstraction permet de représenter les données sensorielles jusqu'à des niveaux d'interprétation et d'abstraction plus élevés. L'axe géographique permet de représenter des espaces géographiques de plus en plus larges. L'axe de résolution représente différentes échelles de la carte.

Dans (Matthies et Elfes 1988), ce modèle probabiliste est exploité pour intégrer

des données provenant de deux types de senseurs différents : des sonars et un système de vision stéréoscopique dans une description commune de l'environnement.

Les cartes des régions occupées ou vides sont construites indépendamment pour chaque senseur et les résultats sont combinés dans une carte unique. Le processus d'interprétation des données et d'intégration est le suivant. Un modèle d'interprétation de l'espace est développé pour chaque type de senseur (sonar ou vision). Ce modèle transforme les données sensorielles en estimations des régions ou des volumes de l'espace probablement occupés ou probablement vides. Chaque senseur fournit ainsi une vue. Les vues des senseurs de même type sont combinées pour obtenir un modèle unifié du monde perçu par le robot à un instant t en utilisant un modèle d'incertitude sur la position des senseurs. Un modèle bayésien de mise à jour permet ensuite de combiner les vues prises à l'instant t avec la carte établie à l'instant $t-1$. Puis les données provenant de types de senseurs différents sont intégrées dans une carte unique par un mécanisme de Bayes. Enfin, pour décider explicitement de l'étiquette des cellules de la grille d'occupation (occupée, vide ou inconnue), les auteurs utilisent une fonction de vraisemblance maximale ou une règle de décision par seuil.

La construction de la carte pour les mesures fournies par les sonars est décrite dans (Elfes 1987). Pour le système de vision stéréoscopique, Matthies et Elfes (1988) utilisent deux caméras montées l'une au dessus de l'autre juste au dessus de l'anneau de sonars pour détecter approximativement la même partie de la scène. Les axes optiques des caméras sont coplanaires et parallèles au plan du sol. Pour obtenir

les grilles d'occupation à partir des images stéréoscopiques, seules les parties des images autour de la ligne d'horizon sont considérées. Une mise en correspondance des caractéristiques est effectuée dans cette partie de l'image en utilisant un algorithme de programmation dynamique. Les auteurs utilisent un modèle d'incertitude basé sur le même principe que le modèle d'incertitude sur les mesures fournies par le sonar. Ce modèle permet de créer la grille d'occupation à partir du profil de profondeur évalué à partir des images stéréoscopiques.

Les informations fournies par le système stéréoscopique et les sonars sont complémentaires. La vision permet de détecter et et de localiser les frontières des surfaces et les surfaces texturées mais fonctionne mal sur les surfaces non texturées. Au contraire, un sonar avec un grand angle donne de meilleurs résultats pour de larges surfaces non-structurées et révèle des régions vides entre le robot et les objets proches. Cependant, le sonar définit moins bien les arêtes des surfaces et la structure des surfaces. De plus, l'incertitude sur les mesures est différente pour les deux types de senseurs. La vision localise bien les caractéristiques dans la direction perpendiculaire à la ligne de vue alors que le sonar obtient une meilleure certitude le long de la ligne de vue.

Les résultats montrent que ces propriétés permettent de corriger les cartes obtenues à partir de chaque type de senseur. Les informations semblables se confortent tandis que les informations conflictuelles font apparaître des régions inconnues.

Plus récemment, Rencken (1994) propose une approche qui consiste aussi à estimer les erreurs introduites à la fois dans la construction de la carte et dans la localisa-

tion du robot. Une méthode heuristique par hypothèses multiples est proposée pour décider de quelle mesure provient chaque caractéristique détectée.

A.1.3 Modélisation "qualitative"

Cette approche, proposée par Kuipers et Byun (1987), est très différente des deux autres. Elle est basée sur des études des méthodes de repérage propres à l'homme.

Modèle Il s'agit d'un modèle topologique, c'est-à-dire d'un graphe de nœuds reliés par des arcs. Le modèle accepte éventuellement des informations métriques. Un nœud est une place distinctive, qui est définie par un maximum local d'une certaine caractéristique. La signature d'une place est le sous-ensemble des caractéristiques qui ont une valeur maximale à cet endroit. Un arc est un chemin local reliant deux places. Les chemins sont définis en termes de stratégies de contrôle locales. Une stratégie serait par exemple "suivre le milieu du couloir".

Méthode Quand le robot explore son environnement, il cherche à tenir à jour sa position courante qui est décrite par l'index de la place courante, l'orientation de la place courante et le chemin d'arrivée. Quand le robot arrive dans le voisinage d'une place distinctive, il effectue une recherche par tâtonnement en cherchant toujours à progresser vers la solution. Si la place était déjà connue, il cherche à la reconnaître par sa signature. Il connaît les caractéristiques qu'il doit maximiser. S'il ne connaît pas cette place, il doit trouver les caractéristiques distinctives. Le robot détermine

l'ensemble des directions possibles à partir de la place courante. Cela comprend la direction dont il vient et la direction où il va. Les autres directions (celles qu'il doit explorer) sont portées dans son programme d'exploration.

Le robot utilise ce programme d'exploration pour déterminer les places et les directions qu'il doit explorer. S'il rencontre une place dont la description correspond à une place connue, il la considère comme un candidat possible. Pour lever l'indétermination, il construit alors une route entre la place connue et une autre qui lui est proche à partir du modèle qu'il a de l'environnement. Le robot essaie ensuite de suivre cette route et de revenir à la place courante. S'il est capable de suivre la route prévue, la place courante correspond bien à celle connue. Cette méthode est appelée "processus de répétition" ("rehearsal procedure").

La stratégie globale d'exploration est la suivante :

- Explorer des régions inconnues.
- Minimiser le nombre d'éléments dans le programme.

La figure A.4 montre le modèle topologique d'un environnement simple avec la trajectoire suivie par le robot et les places distinctives retenues.

Levitt et Lawton (1990) s'intéressent principalement à la navigation autonome qualitative. Dans ce cadre, ils proposent une représentation spatiale de l'environnement sur plusieurs niveaux, fondée sur l'observation et la réacquisition d'évènements distinctifs visuels, c'est-à-dire de repères ("landmarks"). Cette représentation se base

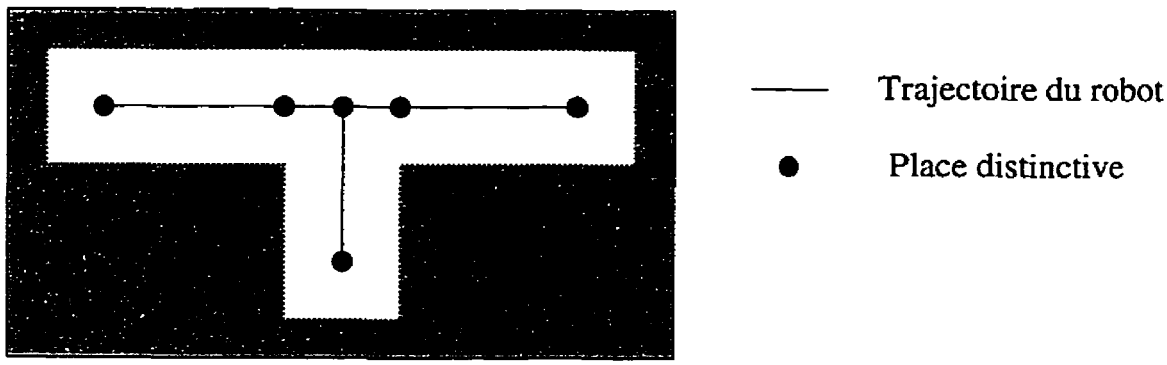


Figure A.4 : Carte topologique de l'environnement selon l'approche de Kuipers et Byun (1987)

sur la mémoire visuelle sous la forme d'une base de données. Celle-ci inclut une représentation topologique et sans coordonnées de la localisation spatiale relative, mais intègre cependant les connaissances métriques disponibles sur les angles et les distances relatifs ou absolus. Cette représentation permet de développer des algorithmes de planification de chemin et de suivi de chemin qualitatifs. De la fiabilité de la reconnaissance des repères dépend la robustesse du système. Les informations sur la position géographique relative permettent *a posteriori* de construire une carte topologique de l'environnement traversé par le robot.

Les auteurs s'appuient sur les concepts formulés par Kuipers et Byun. Leur théorie est basée sur les principes selon lesquels un robot devrait :

- organiser sa mémoire visuelle autour de systèmes de coordonnées locaux basés sur des repères ("landmarks") ;
- tenir compte de la nature des évènements visuels et en particulier utiliser des

représentations qui se suffisent de mesures angulaires et métriques imprécises tout en exploitant pleinement les indices visuels (par exemple, les occlusions) ;

- maintenir des structures dans la mémoire qui associent des systèmes de repères ("landmarks") locaux le long des trajectoires du robot ;
- admettre des processus d'inférence sur la mémoire visuelle qui lui permettent de naviguer de façon fiable malgré la mauvaise qualité des données quantitatives.

Pour réaliser leurs objectifs, les auteurs définissent la notion de "place" géographique en termes de données sur les repères visibles. Une place est définie par les repères et les relations spatiales entre les repères qui peuvent être observés d'un point fixe. Plus généralement une place est définie comme une région de l'espace dans laquelle un ensemble fixé de repères peut être observé de n'importe quelle position dans la région. La condition est que les relations entre ces repères ne changent pas au sens qualitatif. Les données sur les places sont stockées dans des structures appelées vues, frontières et régions d'orientation. La figure A.5 illustre ces trois niveaux de représentation spatiale.

Les vues définissent une place en termes d'angles relatifs et d'erreur angulaire entre les repères et une estimation très grossière de la distance absolue des repères au point d'observation. Les frontières et les régions d'orientation définissent une place de façon plus qualitative. La ligne joignant deux repères divise virtuellement la surface du sol. L'orientation relative des repères que l'on observe, c'est-à-dire l'ordre de gauche à droite de la paire de repères indique de quel côté de la frontière ("Landmark

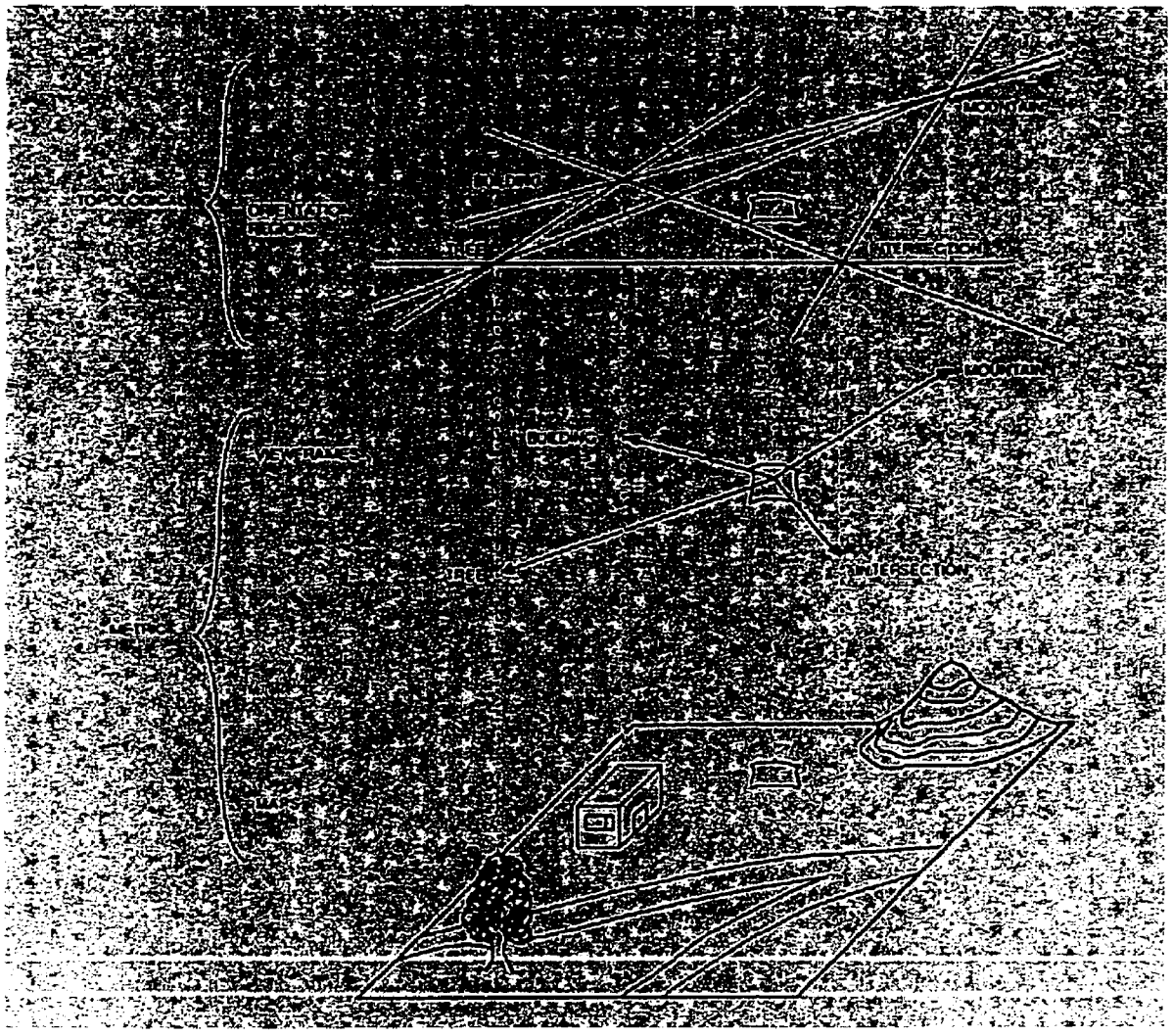


Figure A.5 : Niveaux de représentation spatiale selon Levitt et Lawton (1990)

Pair Boundary" ou LPB) est situé le robot. Un ensemble de frontières orientées détermine une région sur le sol appelée région d'orientation. Les frontières peuvent être obtenues en considérant les paires de repères dans les vues. Les auteurs parlent dans ce cas de l'ensemble des régions d'orientation induites par la vue. Les concepts de vue, de frontière et de région d'orientation permettent de se localiser dans l'espace

par rapport à l'ensemble des repères observés sans utiliser de carte préétablie de l'environnement. Les chemins dans le monde sont représentés comme des séquences d'observation des repères, des vues, des frontières et des autres évènements visuels. Les auteurs calculent les directions approximatives entre les vues qui ont des repères communs. Ils définissent une direction connue comme un état du monde qui peut être créé par une action du robot, typiquement celle de tenir un cap. Tant que le robot tient son cap, le système de vision construit une représentation visuelle de l'environnement rencontré. Quand le robot s'aperçoit visuellement ou de façon calculatoire d'un changement de localisation dans l'espace, le système s'arrête et enclenche des processus d'inférence spatiaux si la destination n'a pas été atteinte.

Dans tout ce que nous avons présenté jusqu'à présent, les auteurs tentent de réduire les erreurs au fur et à mesure de l'établissement de la carte de l'environnement. Engelson et McDermott (1992) proposent une méthode qui permet de détecter et de corriger les erreurs dans la carte.

Le modèle utilisé est basé sur les travaux de Kuipers et Byun (1987). Le robot a un répertoire d'actions qui lui indiquent la marche à suivre pour atteindre la place distinctive la plus proche. Le déplacement du robot est mesuré par un odomètre. Les auteurs supposent que l'erreur sur les déplacements relatifs du robot est bornée. Un ensemble de vues est stocké pour chaque place. Une vue est un vecteur de mesures sur des propriétés physiques de l'environnement telles que sa forme ou des propriétés visuelles telles que sa couleur. À chaque élément de la vue est associée une marge

d'erreur. Le robot identifie une place mémorisée si la vue qu'il en a correspond à une vue de l'ensemble des vues de la place mémorisée.

Les auteurs élargissent ce modèle en utilisant une représentation, qu'ils qualifient de "diktiométrique", qui inclut la forme des chemins c'est-à-dire les relations géométriques entre les places. Les positions relatives de deux places sont estimées et données par des ensembles de positions possibles. Les positions sont représentées dans des repères locaux pour éviter l'accumulation d'erreurs. Un graphe de référence représente les relations entre les repères locaux.

Pour faire la mise en correspondance, le système garde en mémoire un ensemble de pistes qui sont des estimations de l'état courant du robot. Chaque piste est constituée de l'estimation de la position du robot dans un repère local, de la vue courante, du type de place, de la dernière place repérée dans la carte. La mise en correspondance se fait selon le schéma classique de "prédiction-mise en correspondance-mise à jour". Une piste correspond à une place si leurs positions correspondent et si la vue de la piste correspond à l'ensemble des vues de la place. Un résultat incohérent fait apparaître une erreur dans la carte. Les auteurs répertorient ces erreurs et proposent des mécanismes de correction. Ainsi, une erreur peut être commise sur l'estimation de la position d'une place. Une erreur de ce type peut être corrigée par un "α-match", qui autorise une piste à correspondre approximativement à une place en élargissant l'intervalle toléré sur la position estimée. Un autre type d'erreur peut apparaître quand une piste a été associée à une place incorrecte. Cela est corrigé en associant la

piste à une autre place proche ("β-match"). Le système est aussi capable de fusionner des places de la carte qui représentent une place unique dans la réalité ou encore de séparer plusieurs places représentées par une place unique dans la carte.

La logique de la modélisation qualitative se retrouve aussi dans (Dudek et al. 1991, Dudek et al. 1993). Ils proposent une stratégie pour l'exploration d'un graphe représentant l'environnement. Le robot est capable de parcourir les arêtes du graphe, de reconnaître les sommets et d'énumérer les arêtes incidentes au sommet courant de façon ordonnée par rapport à l'arête par laquelle il est arrivé. Le robot ne peut pas mesurer les distances et n'a pas de compas pour s'orienter. Les auteurs montrent que ce problème d'exploration ne peut pas être résolu en général sans marqueurs. Pour résoudre le problème, le robot est équipé d'un ensemble de marqueurs distincts qu'il peut poser ou prendre à un sommet et qu'il est capable d'identifier. L'information sensorielle que le robot acquiert à un sommet est donc constituée des marqueurs situés à ce sommet et des arêtes ordonnées incidentes de ce sommet. Si le robot visite deux fois le même sommet, par deux arêtes différentes, il doit réordonner les arêtes incidentes de ce sommet pour obtenir la même représentation. Les auteurs proposent un algorithme pour mettre à jour le graphe d'un environnement connu. Quand le robot rencontre de nouveaux sommets, l'algorithme les ajoute au graphe ainsi que leurs arêtes incidentes. Ces nouvelles arêtes conduisent à des places inconnues et doivent donc être explorées.

Kriegman et Binford (1988) proposent un modèle générique pour représenter des bâtiments. Ce modèle utilise des mécanismes s'apparentant au concept orienté objet : notion de classe, d'héritage, d'héritage multiple, de contraintes ou relations entre objets (géométriques, de type, de cardinalité ou symboliques). Les auteurs présentent les résultats de l'instantiation d'un corridor. Pour cela, ils utilisent à la fois des images stéréoscopiques et des images monoculaires. Les images stéréoscopiques permettent de déterminer les positions des lignes verticales croisant le plan horizontal. Les portes sont instantiées en regroupant des paires de lignes verticales. Les bords détectés dans l'image monoculaire confirment les résultats précédents. Les portes trouvées dans une image sont regroupées en ensembles de portes coplanaires. La coplanarité des portes est une contrainte sur les murs. Les murs peuvent ainsi être instantiés. Ainsi, les contraintes et les valeurs instantiées permettent de construire le modèle peu à peu. Ce modèle, intéressant du point de vue conceptuel, ne peut malheureusement pas être implanté efficacement dans le but de se localiser dans l'environnement car il ne s'appuie pas sur des caractéristiques précises de l'environnement.

A.1.4 Comparaison des modèles

Nous avons présenté différentes méthodes de modélisation de l'environnement. En résumé, le modèle métrique donne une représentation de l'environnement de type cartographique. Le modèle probabiliste représente l'environnement par des régions probablement occupées ou probablement vides. Le modèle qualitatif est un modèle

topologique. L'environnement est représenté par des graphes de places distinctives reliées par des chemins décrits comme des stratégies de contrôle.

Ces modèles abordent les difficultés liées à la complexité de l'environnement, aux erreurs de mesure et à la consistance du modèle de façon différente.

Le modèle métrique prend un modèle du monde simplifié : les objets ont une forme polyédrique. Pour ce qui est des erreurs, le robot s'appuie sur des repères connus qui lui permettent d'ajuster sa position. En revanche le modèle ne prend pas en compte des modifications éventuelles de l'environnement, par exemple le cas d'une porte qui peut être ouverte ou fermée. Ce problème est traité par l'utilisation d'un système à ultrasons qui permet d'éviter des obstacles proche

Le modèle probabiliste suppose que l'environnement est non structuré. Par contre, il n'y a aucun moyen d'ajustement de la position du robot et les erreurs s'accumulent. Enfin, un point positif : le modèle est continuellement mis à jour en fonction de la situation présente.

Le modèle qualitatif travaille aussi dans un environnement non structuré. Mais il s'agit d'un environnement simple, tel que Kuipers et Byun (1987) le montrent sur les exemples. En effet, les caractéristiques des places distinctives et des stratégies de contrôle sont peu élaborées. En effet, un surplus de caractéristiques complexifie le modèle et rend la méthode inapplicable. Par contre, s'il n'y en a pas assez, il est impossible de se repérer dans un environnement complexe. D'autre part, la position du robot n'étant pas définie dans un référentiel, elle est peu précise mais la marge

d'erreur est fixe. Enfin, les modifications de l'environnement ne sont pas prises en compte.

Ces approches ont donc chacune leurs forces et leurs faiblesses. Dans chaque modélisation, les auteurs ont été obligés de faire des hypothèses simplificatrices qui diminuent leur capacité de généralisation.

A.2 Relocalisation dans l'environnement

Le problème de la localisation d'un robot mobile s'inscrit dans le cadre de la navigation autonome. En effet, pour naviguer efficacement, un robot autonome doit être capable de déterminer rapidement et précisément sa position courante dans l'environnement. Deux lignes d'approche peuvent être dégagées selon que l'environnement est connu ou inconnu. En général, dans un environnement connu, la méthode consiste à chercher à mettre en correspondance des caractéristiques ou "repères" extraits de l'image avec des caractéristiques extraites d'un modèle de l'environnement. Une difficulté importante réside dans l'incertitude sur le modèle et sur les mesures fournies par les capteurs. Dans le cas d'un environnement inconnu, le problème se ramène à l'apprentissage et à la reconnaissance de scènes de l'environnement, ce qui rejoint le problème de la modélisation de l'environnement. Deux approches ont été abordées : l'une basée sur la mise en correspondance de vues panoramiques et l'autre basée sur les réseaux de neurones.

A.2.1 A partir d'une carte de l'environnement

Sugihara (1988) considère le problème de localisation en utilisant une carte de l'environnement dans lequel le robot se déplace et l'image prise par une caméra monoculaire. Deux types de problème sont envisagés. Dans le premier type de problème, les bords verticaux sont extraits de l'image. Une carte des pôles verticaux de l'environnement est donnée. La position du robot est déterminée en établissant la correspondance entre les bords verticaux dans les images et les pôles donnés par la carte. Dans le deuxième type de problème, la position du robot est estimée à partir de l'ordre dans lequel les bords sont trouvés quand l'image est balayée de gauche à droite. La direction des bords ne peut pas être mesurée avec précision. L'auteur propose des algorithmes pour résoudre ces problèmes géométriquement.

Kosaka et Kak (1992) décrivent un système de vision pour la navigation d'un robot mobile dans un bâtiment. Le système est basé sur l'idée qu'il est possible de borner la recherche de repères dans l'image à des régions d'incertitude. Un modèle des incertitudes sur la position du robot est proposé. Ces modèles d'incertitude sont construits en analysant les différences entre les déplacements réels du robot et les déplacements commandés. Ils impliquent les paramètres usuels d'incertitude, tels que la moyenne, les variances et les corrélations et les transformations spatiales correspondant aux déplacements. La façon dont ces incertitudes sont transformées par le déplacement du robot permet de déterminer les régions où sont situés les repères

dans l'image. La figure A.6 illustre le fonctionnement du système.

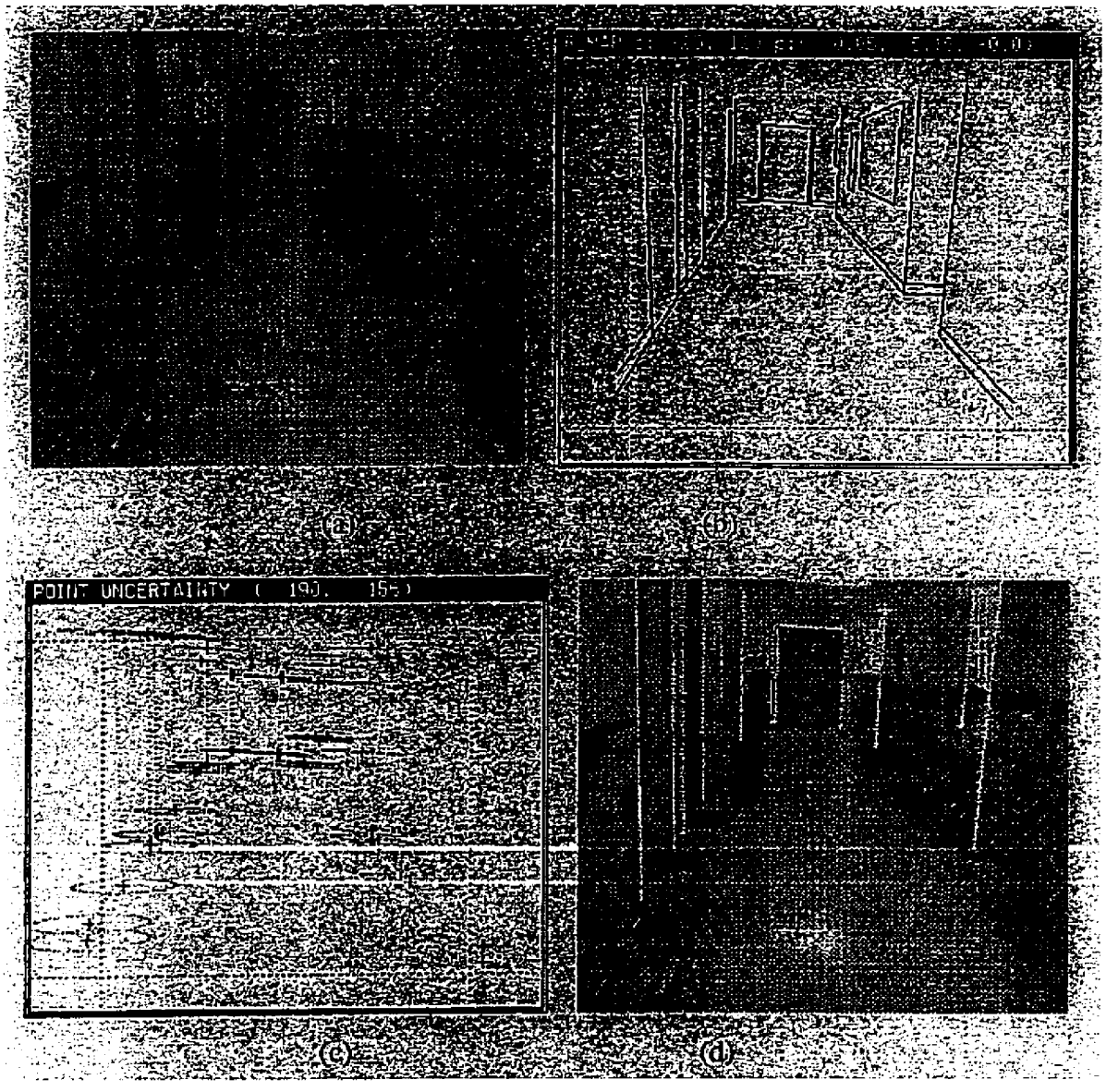


Figure A.6 : (a) Image de la caméra – (b) Carte de prédiction de la scène rendue à partir du modèle tridimensionnel du corridor – (c) Les ellipses représentent les incertitudes associées aux sommets des arêtes dans (b). Le système utilise les données de (a), (b) et (c) pour la localisation du robot. – (d) est une reprojection des arêtes du modèle sur l'image de la caméra une fois que le robot a déterminé sa position dans le corridor (Kosaka et Kak 1992).

La figure A.6(a) montre l'image prise par la caméra d'une certaine position du robot dans le corridor. La figure A.6(b) est la carte prédite rendue à partir du modèle tridimensionnel connu du corridor en supposant que le robot est situé au centre de cette région d'incertitude. Étant donnée l'incertitude à cette position particulière du robot, la figure A.6(c) montre les régions d'incertitude associées aux sommets des différentes arêtes dans la carte de prédiction de la figure A.6. Chaque ellipse représente une unité de distance de Mahalanobis, qui signifie que la probabilité de trouver un sommet correspondant dans l'ellipse est de 39%. Les auteurs montrent comment dériver les régions d'incertitude sur les arêtes à partir des régions d'incertitude sur les sommets. La recherche des arêtes de la scène correspondant aux arêtes de l'image est limitée à ces régions dans l'image et aussi dans l'espace de Hough. La mise en correspondance entre un repère et une caractéristique de l'image s'appuie sur l'utilisation du filtre de Kalman qui permet de réduire à la fois l'incertitude sur la position du robot et la taille des régions d'incertitude associés aux repères. Le système calcule alors la position et l'orientation du robot. La figure A.6(d) montre la reprojection des arêtes du modèle tridimensionnel sur l'image à partir de la position du robot calculée. Une des faiblesses de la méthode est que les incertitudes sur la position du robot s'accumulent. Les auteurs prévoient d'arrêter le robot pour le relocaliser quand un seuil d'incertitude est dépassé.

Greiner et Isukapalli (1996) proposent une méthode pour sélectionner les repères utiles pour la localisation. L'environnement est connu. Étant données une estimation

de la position initiale du robot et une image de l'environnement, la méthode consiste à chercher tout d'abord un ensemble des repères (des objets de l'environnement à des positions connues). Ensuite, une meilleure estimation de la position courante du robot est obtenue à partir de la différence angulaire entre les repères. Le problème est que certains repères ne soient pas visibles ou soient confondus avec d'autres repères. L'idée est d'apprendre au système à sélectionner uniquement les repères fiables. La méthode consiste à apprendre à partir des expériences passées une fonction de sélection qui retourne le sous-ensemble des repères susceptibles d'être visibles dans la scène. La faiblesse de cette méthode est qu'elle est basée sur la mise en correspondance d'arêtes qui reste un problème délicat en vision.

L'approche proposée par Stein et Medioni (1995) pour la localisation d'un observateur sur une carte repose sur une vue panoramique sur 360° de la courbe de l'horizon. La courbe de l'horizon est définie comme étant la frontière entre le ciel et la terre. À partir de la carte du relief digitalisée, les courbes d'horizon sont calculées pour chaque point de la carte et comparées à la vue panoramique perçue par l'observateur. Ces courbes sont approximées par des polygones et groupées en super-segments qui sont codés et stockés dans une table. La courbe panoramique perçue par l'observateur est approximée par un polygone et les super-segments obtenus permettent de trouver les localisations candidates dans la base de données. Le meilleur candidat est sélectionné en appliquant des contraintes géométriques.

L'idée d'utiliser la "courbe d'horizon" est intéressante mais elle suppose un relief

accidenté et suffisamment diversifié pour que les courbes puissent être distinguées les unes des autres. De plus, l'algorithme ne peut pas fonctionner si l'observateur est placé dans une vallée trop profonde pour voir l'horizon. Ces contraintes limitent l'utilisation de cette méthode à un environnement bien particulier. En revanche, l'intérêt est que cette approche ne nécessite pas de connaître la position initiale de l'observateur, ni du chemin suivi pour effectuer la reconnaissance. L'observateur est en quelque sorte "parachuté" dans l'environnement.

A.2.2 Sans carte de l'environnement

Basri et Rivlin (1993) proposent une méthode de localisation basée sur la combinaison linéaire de vues de la scène. L'environnement est modélisé par un ensemble d'images qui ont des éléments en commun (par exemple, un même objet). La localisation est effectuée en retrouvant la combinaison linéaire de vues du modèle qui peut se superposer sur l'image observée. Les coefficients sont déterminés en utilisant quatre points du modèle et leurs points correspondants dans les images et en résolvant un système d'équations linéaires.

Le problème dans cette méthode est essentiellement la mise en correspondance des points de l'image. En particulier, la difficulté consiste à gérer les effets de distorsion dans les images. Un obstacle est aussi la quantité d'images qu'il faudra stocker pour la localisation dans un large environnement.

Zheng et Tsuji (1992) proposent une représentation panoramique pour la recon-

naissance de chemins suivis par un robot mobile en milieu urbain. La reconnaissance est basée sur une vue panoramique généralisée ("GPV") et une vue panoramique ("PV"). Une vue panoramique est une projection des scènes visualisées sur une rétine cylindrique en un point stationnaire. Elle permet de déterminer la position et l'orientation du robot en un point donné. Une vue panoramique généralisée est une projection des scènes visualisées au cours du chemin parcouru par le robot sur la surface d'un cylindre généralisé. Cette GPV est une mémoire visuelle du chemin suivi par le robot. Pour effectuer la reconnaissance, les auteurs emploient des méthodes de programmation dynamique permettant d'optimiser la mise en correspondance des caractéristiques des représentations mesurées et mémorisées. Les caractéristiques retenues sont essentiellement basées sur la détection des arêtes et sur la couleur. Pour mettre en correspondance deux GPVs, les auteurs ont besoin de connaître la position initiale du robot. À partir de ce point de départ, tous les chemins possibles sont explorés. La difficulté majeure réside dans le fait que la représentation dépend de la trajectoire suivie, de l'orientation et de la vitesse du robot. Ces problèmes peuvent être partiellement résolus en transformant les GPVs pour obtenir des GPVs prises à vitesse constante. Enfin, pour mettre en correspondance deux PVs, les auteurs doivent considérer toutes les rotations possibles car il n'y a pas de point de référence fixé. Ils utilisent malgré tout un algorithme qui permet d'optimiser le chemin de recherche et d'éviter d'explorer toutes les combinaisons possibles.

Dans (Li et al. 1996), ces représentations panoramiques, en tant que points de

repère, sont intégrées dans une carte qui fournit des informations sur la structure de l'environnement. La mise en correspondance des caractéristiques est basée sur la contrainte selon laquelle les arrangements spatiaux relatifs sont invariants avec la vitesse du véhicule. Autrement dit, les objets apparaissent toujours dans le même ordre dans la scène. Ils supposent malgré tout que la vitesse du véhicule est constante pour pouvoir mettre en correspondance les caractéristiques des objets par une méthode de programmation dynamique. La justesse de la mise en correspondance est ensuite vérifiée en comparant les modèles de disparité et les distributions de couleur.

Li et al. (1995) présentent un système de navigation basé sur la représentation panoramique de l'environnement décrite dans (Zheng et Tsuji 1992). Le système est monté sur un robot mobile avec des caméras et des projecteurs qui projettent la lumière par des fentes. Une caméra et un projecteur sont placés à l'avant du robot et dirigés vers le sol pour détecter les obstacles. Deux autres caméras et projecteurs sont placés sur les côtés perpendiculairement à la direction du déplacement du robot pour acquérir une vue panoramique de chaque côté de la scène. Le robot se déplace librement dans un environnement intérieur. Les objets 3D de la scène sont extraits des vues panoramiques en utilisant la disparité des vues panoramiques et leur couleur, et sont organisés en une carte structurée qui représente leurs propriétés "iconiques" et leurs relations spatiales. En examinant la structure de la carte, les auteurs trouvent les objets 3D qui peuvent servir de repère au robot pour naviguer.

Ils présentent des résultats sur l'acquisition de cartes de l'environnement à partir

d'un chemin fermé du robot. La trajectoire du robot est estimée en utilisant l'information donnée par son encodeur. L'information visuelle permet de corriger l'erreur sur l'estimation de la trajectoire. Quand le robot se déplace, il fait la correspondance entre les objets qu'il perçoit et la vue panoramique mémorisée. S'il reconnaît un objet dans la séquence, il considère qu'il est revenu à son point de départ et l'erreur sur la trajectoire est compensée. La reconnaissance est basée sur la taille de l'objet, sur sa distance par rapport au robot et sur sa localisation approximative. Les auteurs estiment alors la position de l'objet par rapport au sol à partir de la trajectoire et peuvent ainsi construire la carte de l'environnement.

Janét et al. (1995) présentent une approche par réseau de neurones de Kohonen auto-organisateur pour la localisation globale d'un robot mobile autonome dans un environnement intérieur. L'idée est que le réseau est capable de dégager les caractéristiques d'une région discrète de l'espace (typiquement une pièce) à partir des données sensorielles recueillies par le robot au cours de son exploration. Le robot peut déterminer dans quelle pièce il se trouve en classant l'ensemble des données sensorielles selon leur degré de similarité par rapport aux pièces mémorisées. Le réseau de Kohonen est non-supervisé, c'est-à-dire qu'il n'a pas besoin d'un "professeur" pour lui apprendre à associer les pièces et les données.

Le caractère unique d'une pièce est créé par des nuages de données recueillies par un sonar au cours de l'exploration de la pièce par le robot. Dans la simulation, des fonctions de corruption gaussiennes sont appliquées aux mesures du sonar calculées

pour simuler le bruit dû au sonar et à la trajectoire du robot. Il est important que le robot explore complètement les extrémités d'une pièce pour que le réseau de neurones puisse l'identifier. Au cours de son apprentissage, le réseau de Kohonen s'organise selon une grille de 21×21 centres de regroupement pour représenter les 441 caractéristiques potentielles de chaque pièce. Pendant la phase d'apprentissage, le réseau enregistre la fréquence à laquelle chaque neurone réagit pour pouvoir par la suite éliminer des centres de regroupement redondants ou inactifs. Les auteurs utilisent les règles d'apprentissage et de rappel classiques, basées sur la minimisation de la distance euclidienne d'un point à un centre et sur la règle du "Winner-Take-All". Trois opérations de prétraitement sont appliquées au réseau pour réduire la probabilité de fausse classification. Tout d'abord, une étape de translation grossière est employée pour approximer un problème invariant en translation. Cette étape consiste à ajuster un ensemble de données placé arbitrairement dans un espace 2D à une grille de nœuds de Kohonen placée aussi arbitrairement dans un espace 2D. Ensuite, une étape de translation plus fine consiste à translater le réseau et les données jusqu'à obtenir le meilleur ajustement. Enfin, les neurones dont la fréquence de réaction aux données d'apprentissage est nulle sont rendus inactifs pour la phase de rappel. Les auteurs montrent les résultats de la simulation pour dix pièces différentes. Le taux de fausse classification est de 1%. Ils ne font pas de tests sur des environnements inconnus.

Le problème essentiel avec cette approche est qu'elle ne peut s'appliquer qu'à un environnement statique. La moindre modification de l'environnement modifie sa

signature et oblige le système à tout réapprendre. Dans la même optique, le réseau ne peut pas apprendre de nouvelles pièces incrémentalement.

A.2.3 Comparaison des approches avec carte et sans carte

Les approches basées sur une carte de l'environnement s'appliquent dans le cas d'environnements connus alors que les méthodes n'utilisant pas de carte s'appliquent dans le cas d'environnements inconnus. L'avantage des approches avec carte est que la scène peut être prédite à partir d'un modèle de l'environnement, ce qui facilite le problème de la mise en correspondance de caractéristiques. Certaines approches supportent de légères modifications de l'environnement. Ainsi dans (Kosaka et Kak 1992), tout objet non modélisé sera considéré comme un obstacle. La difficulté dans cette approche consiste à relever les repères pertinents dans l'image et à les mettre en correspondance avec les caractéristiques du modèle. Or dans ce processus, il se peut des repères soient confondus les uns avec les autres ou que des repères imprévisibles apparaissent (notamment dans des environnements dynamiques) ou encore tout simplement que selon le point de vue, des repères ne soient pas visibles. Généralement, les repères proposés sont basés sur les arêtes qui sont difficiles à détecter avec précision en vision. Les erreurs sur les mesures induisent des incertitudes sur la localisation du robot. Et la mise en correspondance des repères souffre de l'accumulation d'erreurs sur la localisation du robot. Par ailleurs, dans ces approches, l'environnement est assez simple. En effet, il faut pouvoir déterminer les repères dans le modèle.

Dans les approches sans carte, l'avantage est que le robot construit son propre modèle de l'environnement, directement à partir de ce qu'il perçoit. L'idée est que le robot se repère par rapport à des caractéristiques de l'environnement, donc sur un plan plus qualitatif que quantitatif, ce qui permet d'éviter les accumulations d'erreur sur la localisation du robot. En revanche, la faiblesse de cette approche est qu'elle suppose que l'environnement est statique. En effet, dans ces méthodes, le robot apprend au fur et à mesure de son exploration. Si l'environnement est modifié, il ne retrouvera plus ses repères.

A.3 Reconnaissance de scènes

A.3.1 Approche par mise en correspondance

Zheng et Tsuji (1992) proposent une approche par mise en correspondance de représentations panoramiques pour la reconnaissance de routes. Ils présentent une vue panoramique généralisée qui est une mémoire visuelle de la route suivie par le robot et une vue panoramique centrée en un point stationnaire. Ils utilisent des techniques de programmation dynamique pour mettre en correspondance les représentations panoramiques. Un des problèmes avec cette méthode est que dans une vue panoramique, il n'y a pas de point de référence. Pour comparer deux vues panoramiques, les auteurs doivent donc essayer toutes les combinaisons possibles, avec la difficulté supplémentaire que les vues panoramiques dépendent de la vitesse et de

la trajectoire suivie par le robot. Malgré tout, les auteurs proposent des techniques d'optimisation qui permettent d'accélérer la recherche.

Comparées aux méthodes basées sur la mise en correspondance de caractéristiques, les méthodes basées sur les réseaux de neurones sont plus susceptibles de donner de meilleurs résultats dans le cas d'une mine. En effet, dans ce type d'environnement, il y a peu de caractéristiques aisément détectables permettant la mise en correspondance. De plus, une fois que le réseau de neurones a effectué son apprentissage, son temps de réponse est très rapide pour la reconnaissance. L'apprentissage peut être long, mais nous n'avons pas d'exigence particulière sur le temps de calcul car le modèle peut être établi après que les scènes ont été enregistrées.

En fait, nous nous trouvons face à une difficulté directement liée au problème plus général de l'analyse de la texture. Les différentes textures dans une image sont en général perçues par un observateur humain mais il n'existe pas de bonne définition mathématique de la texture. C'est cette absence de définition qui rend la description automatique ou la reconnaissance de formes un problème très complexe et non encore résolu. En général, les chercheurs classent les textures en deux catégories principales : les textures structurées et les textures non structurées ou encore stochastiques. Pour les textures structurées, les modèles proposés supposent que les textures sont composées de primitives. Mais cette approche ne peut pas être généralisée à des textures non structurées. Des modèles stochastiques, tels que les modèles de champs aléatoires de Markov, sont plutôt utilisés. Dans ces modèles, l'image est vue comme une

instance d'un processus aléatoire, défini par les paramètres du modèle. Mais cette technique, basée sur un modèle, ne convient pas entre autres pour des textures qui ne sont pas homogènes. D'autres méthodes calculent les caractéristiques de la texture à partir des images filtrées et utilisent ces caractéristiques pour des tâches de segmentation ou de classification. Parmi ces méthodes, une approche connexionniste a vu le jour et s'est imposée comme une alternative pour des tâches d'analyse de la texture (Greespan et al. 1994, Jain et Karu 1996).

A.3.2 Approches connexionnistes

Lippman (1987) propose une taxonomie des réseaux de neurones les plus classiques utilisés pour la reconnaissance de formes. Il classe dans les mémoires associatives à apprentissage supervisé le réseau de Hopfield et le perceptron multi-couches. Nous choisissons le perceptron multi-couches comme étant l'architecture de réseaux de neurones la plus utilisée dans ce type de problème. Ce type de réseau accepte des valeurs non binaires en entrée, ce qui est notre cas puisque nous traitons des images en niveaux de gris.

Le problème peut être traité de deux façons. Une première méthode consiste à extraire des caractéristiques des exemples avec certaines propriétés d'invariance et de les présenter au réseau. Le rôle du réseau est alors de partionner l'espace des caractéristiques en régions de décision correspondant à chaque classe. Ainsi Kim et Nam (1995) utilisent les descripteurs de Fourier invariants en rotation, en translation

et en taille comme caractéristiques d'entrée pour la reconnaissance d'outils dans des images en niveaux de gris. Cependant, ces descripteurs sont basés sur la forme des outils et non pas sur la texture.

Greespan et al. (1994) décrivent un système d'analyse de la texture dans lequel des règles de discrimination de la texture sont apprises à partir d'une représentation à plusieurs échelles de l'image d'entrée. Le système consiste en trois étapes. La première étape a pour objectif d'extraire les caractéristiques de l'image. Les auteurs utilisent pour cela des filtres passe-bas sélectifs quant à l'orientation. Les réponses en orientation et en fréquence sont extraites de régions locales de l'image d'entrée. Les statistiques des coefficients caractérisant une région locale forment le vecteur de caractéristiques représentatif. Les auteurs utilisent la pyramide de Gabor pour définir un ensemble de filtres. Cette étape transforme donc l'espace image en un tableau de vecteurs de caractéristiques qui représentent des fenêtres locales dans l'image originale. L'étape suivante est considérée comme une étape de prétraitement qui permet de réaliser une représentation plus compacte du vecteur de caractéristiques. Il s'agit d'une étape d'apprentissage non-supervisé réalisé par des algorithmes de regroupement ("clustering") et qui a pour but de quantifier les valeurs continues du vecteur de caractéristiques. La dernière étape consiste à étiqueter le domaine d'entrée en utilisant un réseau de neurones à trois couches et par une méthode d'apprentissage supervisé. Finalement, le système apprend une librairie de textures pendant la phase d'entraînement et est capable de classifier et de segmenter une image inconnue com-

portant différentes textures.

La deuxième approche consiste à laisser le soin au réseau de trouver les invariances dans l'ensemble de données échantillons sur lequel il fait son apprentissage. Dans cette approche, les invariances sont incorporées dans la structure du réseau. Le réseau est ensuite capable de discerner les invariances dans des données inconnues et de les reconnaître.

LeCun et al. (1990) utilisent un réseau de neurones de type perceptron multi-couches pour résoudre le problème de la reconnaissance de caractères optiques. L'entrée consiste en des pixels noirs ou blancs représentant des chiffres manuscrits. Le réseau classe les images en dix catégories (dix chiffres). Les images sont directement appliquées au réseau, évitant ainsi le problème d'extraction de caractéristiques. Le réseau est composé de cinq couches. La première et la troisième couche ont pour rôle d'extraire les caractéristiques des images. La seconde et la quatrième couche permettent de réduire la résolution spatiale. Säckinger et al. (1992) implantent ce réseau sur une puce reconfigurable, appelée "ANNA" (analog neural network arithmetic and logic unit) et proposent une méthode pour améliorer les performances du réseau qui consiste à réentraîner la dernière couche. L'apprentissage se fait en deux phases : une passe avant où tous les neurones participent au calcul de la sortie à partir de l'exemple présenté en entrée et une phase de rétropropagation de l'erreur qui ne s'applique que sur la couche de sortie. Les autres poids sont gelés.

Pomerleau (1991) propose un système appelé ALVINN (Autonomous Land Vehicle In a Neural Network) basé sur un réseau de neurones à rétropropagation pour conduire le camion Navlab. Le système apprend à contrôler le camion en observant les réactions d'un conducteur humain. L'architecture du réseau consiste en un perceptron à deux couches complètement connecté (voir la figure A.7).

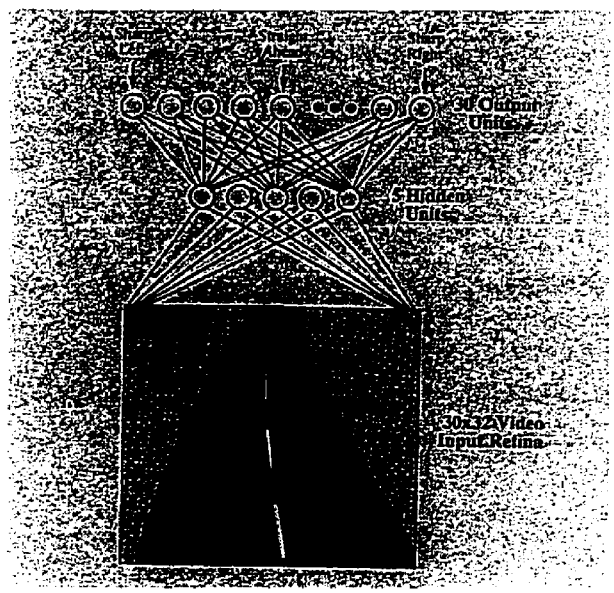


Figure A.7 : Architecture du réseau de neurones dans le système ALVINN (Pomerleau 1991)

Le réseau reçoit en entrée l'image vidéo prise par une caméra montée sur le camion. La couche de sortie comporte trente unités et est une représentation linéaire de la direction que le véhicule devrait suivre pour rester sur la route. L'unité la plus au centre représente la direction en ligne droite ; les unités à gauche et à droite du centre représentent des tournants à gauche et à droite. Pour entraîner le réseau, l'auteur lui présente des images de la route en entrée et la direction du volant correcte comme la

sortie désirée. La règle d'apprentissage utilisée est la rétropropagation classique. Au lieu d'entraîner le réseau à activer seulement une unité de sortie, l'auteur l'entraîne à produire une distribution d'activation gaussienne centrée sur la direction du volant qui gardera le véhicule centré sur la route. L'avantage de cette méthode, d'après l'auteur, est de faciliter la tâche d'apprentissage car des images de la route légèrement différentes devront produire des vecteurs de sortie légèrement différents et non pas radicalement différents. L'auteur propose une méthode d'entraînement qui consiste à apprendre au réseau à imiter la conduite d'un humain dans des conditions réelles. L'apprentissage est effectué alors que la personne conduit le véhicule. L'entrée du réseau est fournie par une caméra montée sur le véhicule et la sortie désirée est la direction dans laquelle le conducteur tient le volant. Au lieu de présenter au réseau seulement l'image vidéo courante avec la sortie désirée, l'auteur translate l'image originale latéralement pour créer quatorze images supplémentaires dans lesquelles le véhicule apparaît décalé par rapport au centre de la route. Ces images sont apprises avec l'image originale. Cela permet d'apprendre au réseau des situations où la direction du véhicule aurait besoin d'être redressée. En effet, ce genre de situation risque peu d'apparaître avec un conducteur normal. ALVINN a été entraîné à conduire dans des situations variées, notamment sur des routes goudronnées ou non, à voie unique ou à plusieurs voies et à des vitesses allant jusqu'à vingt miles à l'heure.

Jain et Karu (1996) décrivent une méthode de classification de la texture par un réseau de neurones de type perceptron multi-couches. Ils introduisent cette approche

comme une généralisation de la méthode par filtrage multicanal. Les auteurs partent de l'observation que l'opération effectuée par un neurone est très similaire à celle effectuée par un canal. Ils remplacent donc chaque neurone par un canal, si bien que les opérations d'extraction de caractéristiques et de classification peuvent être effectuées au sein du même réseau. L'autre avantage de cette représentation est que les techniques d'apprentissage classiques peuvent être utilisées pour améliorer les performances du réseau. Le problème est finalement formulé comme un problème d'optimisation qui consiste à trouver l'ensemble minimal de filtres (à la fois leur nombre et leurs coefficients ou autrement dit les poids du réseau) qui résulte en la meilleure classification. Pour réduire le nombre de neurones (ou de filtres), les auteurs utilisent une méthode d'élimination des neurones basée sur l'évaluation de leur contribution à l'activité du réseau. Ils choisissent de n'éliminer que des neurones appartenant à la première couche du réseau. Les poids sont ajustés par la règle classique de rétropropagation de l'erreur. La méthode est expérimentée sur différentes applications : localisation de codes barres dans des images, segmentation d'une page imprimée en régions contenant du texte, des graphiques et le fond.

(Fukumi et al. 1992) proposent un réseau de type perceptron multi-couches insensible aux rotations des formes présentées en entrée et l'appliquent à la reconnaissance de pièces de monnaie. Les entrées sont des images en niveaux de gris. L'architecture du réseau dépend du nombre de degrés désiré pour l'invariance en rotation. Plus ce nombre est petit, plus il faudra de neurones dans le réseau. L'invariance en rotation

est obtenue par sa structure. Les connexions synaptiques sont créées de telle sorte que les versions transformées de la même pièce de monnaie produisent la même sortie.

La deuxième approche semble la plus appropriée étant donnée la difficulté de trouver les caractéristiques pertinentes à extraire dans les images sur lesquelles nous travaillons. En effet, des critères géométriques ne nous apporteraient rien de plus que ce que nous avons déjà avec notre modèle topologique. Nous devons chercher une information supplémentaire dans la texture même des objets de la scène. Dans l'approche proposée par Greespan et al. (1994), le système fonctionne bien pour des textures qui sont très différentes à l'œil humain. Or dans le cas d'une mine, d'une intersection pour l'autre, la texture des parois est très semblable. En fait, ce que le réseau apprend dans (Greespan et al. 1994), c'est à classer des textures. En revanche, Jain et Karu (1996) montrent que les réseaux de neurones ont cette capacité d'apprendre à différencier les régions texturées d'une image directement à partir d'exemples qui leur sont présentés. De plus, cette approche nous permet d'exploiter les propriétés d'invariance en rotation de notre représentation. En effet nous apprenons au réseau à associer à une même intersection, un ensemble de vues panoramiques ayant subi une rotation autour du centre de l'intersection.

Finalement, cette approche connexionniste nous permet d'exploiter à la fois les propriétés sur la géométrie et la texture de la scène pour apprendre et reconnaître les intersections.

A.4 Détection d'intersections

Nous présentons dans cette section deux approches proposées pour la détection d'intersections dans des séquences d'images. Dans la première approche, les points de la route sont détectés à partir de leur couleur et une méthode est proposée pour modéliser les intersections dans l'image. La deuxième approche utilise un réseau de neurones pour apprendre des intersections.

A.4.1 Approche basée sur un modèle de l'intersection dans l'image

SCARF (Crisman et Thorpe 1993) est un système basé sur la vision qui détecte des routes et des intersections dans des images en couleur. SCARF permet à un robot mobile intelligent de naviguer sur des routes "non-structurées" (pas de lignes peintes sur la surface de la route) sans l'aide d'une carte et dans des conditions extérieures variées. Le système est composé de deux modules principaux : (1) la détection de la surface de la route et (2) son interprétation.

La détection de la route se fait en trois étapes. Tout d'abord, l'image d'entrée est filtrée pour éliminer le bruit et réduire sa résolution. Pour cela, les auteurs utilisent des techniques classiques de moyennage et de sous-échantillonnage. Ensuite, la détection étant basée sur la couleur des pixels, un modèle des couleurs est établi à partir de l'image prédite de la route (image précédente dans la séquence). Les couleurs correspondant à des points "route" et "non-route" sont représentées par différents modèles

de couleur. Cela permet de représenter la couleur de la surface de la route par différentes couleurs. Ces modèles sont formés en déterminant des régions échantillons dans l'image à partir de la prédiction. En utilisant les pixels des régions "route" et "non-route" échantillons, SCARF regroupe les échantillons en ensembles de pixels de couleur semblable. Un modèle gaussien est établi pour chaque ensemble. À partir du modèle des couleurs, SCARF peut classer les pixels de l'image réduite. SCARF calcule la probabilité qu'un pixel x dans l'image soit un pixel "route" selon la justesse avec laquelle la couleur de x correspond au modèle.

Le module d'interprétation met en correspondance un ensemble de modèles pour des routes et des intersections avec l'image de la route probable pour déterminer la meilleure interprétation possible. Une intersection est modélisée dans l'image par les paramètres suivants : le nombre de branches qui se rencontrent en un point commun, appelé "noyau", les angles des axes des branches avec la verticale dans l'image, la largeur de la route au bas de l'image, la position de la ligne horizon. Les auteurs supposent que les routes sont localement droites, de largeur constante et dans un sol plan. Une route droite est une route à une seule branche. Si le nombre de branches est égal à 2, la route fait un tournant. SCARF génère un ensemble d'intersections candidates qui sont des images binaires idéalisées contenant des 1 pour les pixels appartenant à la route, des 0 sinon. Les auteurs calculent la corrélation entre l'image de la route probable (déterminée au module précédent) et chacune des images binaires possibles et en déduit une mesure de la confiance qui peut être accordée à chacun

des modèles. Pour limiter l'espace de recherche, SCARF commence par déterminer le modèle de la route principale dans l'image. Puis le système essaie de trouver une branche partant de la route principale. Si l'ajout de cette branche augmente la confiance du modèle, elle est conservée dans le modèle. Elle permet alors de déterminer la position du noyau à partir duquel le système essaie de rajouter d'autres branches toujours avec l'objectif d'augmenter la confiance du système. À chaque fois qu'une branche est ajoutée au modèle, il faut vérifier qu'elle ne recouvre pas excessivement les autres branches déjà incluses dans l'intersection, sinon l'algorithme l'écarte du modèle. Le système repose sur la détermination de la position du noyau qui n'est malheureusement pas toujours possible. Par exemple, le cas d'une intersection en T pose problème car SCARF ne peut pas déterminer le point de fuite de la branche secondaire. Un autre problème peut survenir quand il n'y a pas assez de pixels "non-route" dans l'image pour déterminer de façon fiable le modèle des couleurs. Cela peut être le cas par exemple si la route tourne. Les limitations du système découlent en partie des hypothèses formulées : les couleurs des points de la route diffèrent des couleurs des points en dehors de la route ; les routes sont localement droites, de largeur constante, et planes.

Le système a été testé sur le robot Navlab dans des conditions extérieures variées. Le système semble fonctionner pour des intersections en Y (la route principale se sépare en deux branches) ou en λ (une branche secondaire vient rejoindre la branche principale de la route), mais éprouve des difficultés pour une intersection ayant trois

branches de direction différente. Les auteurs ne montrent pas de résultats pour des intersections plus complexes.

A.4.2 Approche par réseau de neurones

Jochem et al. (1995) présentent le système ALVINN VC qui permet de détecter des routes et des intersections par réseau de neurones. ALVINN VC utilise le système ALVINN Pomerleau (1991) avec un senseur d'images artificiel, appelé caméra virtuelle. Une caméra virtuelle peut être placée à n'importe quelle position et n'importe quelle orientation dans le référentiel du monde. Cette caméra, qui est donc simulée, crée des images artificielles (ou virtuelles) en utilisant les pixels d'une image prise par une caméra réelle projetés sur un modèle du monde (voir la figure A.8).

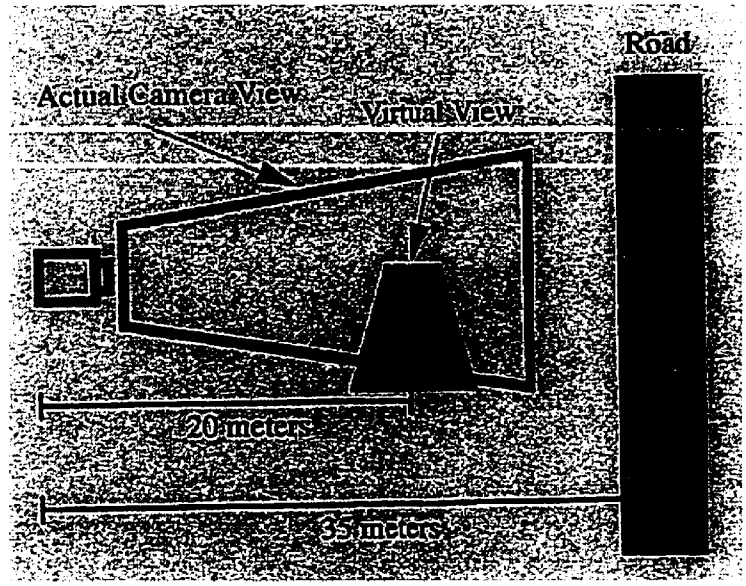


Figure A.8 : Système de caméra utilisé par ALVINN VC (Jochem et al. 1995)

La méthode s'appuie sur un modèle du monde plat. Les images virtuelles sont reconstruites à partir des images prises par la caméra réelle dont la position est connue. ALVINN VC cherche à identifier les images virtuelles aux images apprises par le réseau de neurones pour localiser la route. ALVINN VC utilise des connaissances *a priori* qui lui permettent de spécifier où et quand des caméras virtuelles devraient être créées. Les caméras sont créées par rapport à la position du véhicule et avant que l'intersection ne survienne. La position de la caméra virtuelle (à déterminer) et le réseau de neurones associé dépendent du type de route auquel le robot s'attend (par exemple, une route à une voie). Quand la route ou l'intersection à détecter est présente, la caméra virtuelle fournit au réseau une image qu'il connaît et peut donc identifier avec un certain facteur de confiance.

Les résultats montrent que le robot est capable de détecter une route à voie unique mais éprouve des difficultés à s'aligner avec l'axe de la route et par conséquent à suivre la route. Pour une intersection en Y, le problème est semblable.

Cette approche neuronale présente l'avantage d'être indépendante du type de route. En revanche, le système montre des faiblesses quant à sa capacité à naviguer sur la route détectée. De plus, le système est basé sur des connaissances *a priori* des positions de la caméra virtuelle par rapport au véhicule ce qui impose une contrainte très forte. Le robot doit avoir à l'avance une très bonne idée de ce qu'il va rencontrer.

A.4.3 Comparaison des approches

Le modèle proposé par Crisman et Thorpe (1993) pour les intersections présente l'avantage d'être basé directement sur l'image. Dans cette méthode, il n'y a pas besoin de repasser dans l'espace tridimensionnel. En revanche, le modèle montre des faiblesses quant à ses capacités de généralisation. D'une part, il ne permet pas de modéliser certains types d'intersections qui pourtant risquent classiquement d'être rencontrées (par exemple, le cas d'une intersection en T). Et d'autre part, ce modèle n'est pas adapté à des intersections complexes c'est-à-dire à des intersections dont le nombre de branches est supérieur à trois. En effet, ce modèle suppose que les axes des routes se coupent en un point unique, ce qui n'est pas nécessairement vrai dès qu'une intersection comporte plus de deux branches. Enfin, la méthode proposée risque d'échouer si le nombre de branches composant l'intersection augmente. En effet, comme le nombre de branches n'est pas connu, Crisman et Thorpe (1993) utilisent une méthode incrémentale dont la fiabilité risque de diminuer au fur et à mesure que l'intersection se complexifie. L'approche de Jochem et al. (1995) suppose que le monde est connu et modélisé, contrairement à la méthode précédente. Cette approche pose donc plusieurs problèmes : les problèmes de la modélisation de l'environnement et de la localisation du robot dans l'environnement. Le principal avantage de cette méthode réside dans le mécanisme de reconnaissance de l'intersection. En effet, les réseaux de neurones possèdent cette capacité de généralisation qui leur permettent de reconnaître des formes même légèrement déformées.

A.5 Détection du contour du sol

A.5.1 Approches basées sur la couleur

Thorpe et al. (1989) proposent un système de navigation visuelle intelligent Navlab qui permet de trouver et de suivre des routes ainsi que d'éviter les obstacles. La détection de la route se fait directement dans l'image. Les points de l'image sont classés comme appartenant ou n'appartenant pas à la route selon leur couleur. Comme les conditions extérieures peuvent être très variables (selon la luminosité, l'environnement,...), les auteurs proposent un modèle de couleur adaptatif. Ils supposent que, localement, la largeur de la route est constante, le sol est plan et la route est droite. La route est paramétrisée par son orientation θ et la position du point de fuite P dans l'image (voir la figure A.9).

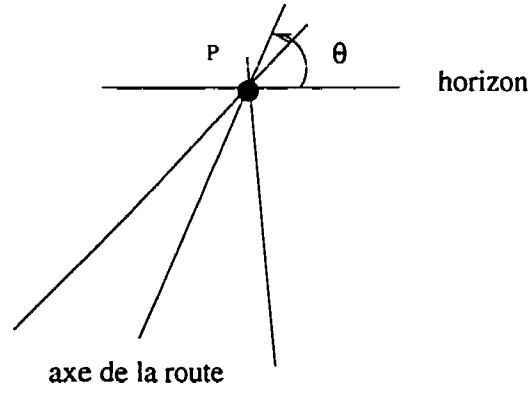


Figure A.9 : Paramétrisation de la route pour Navlab

L'algorithme est le suivant :

1. Classer les pixels de l'image selon le modèle de couleur et de texture déterminé.

2. Utiliser les résultats de la classification pour voter pour la meilleure position de la route (P, θ).

3. Établir de nouvelles statistiques de couleur basées sur les régions de l'image "route" et "non-route" pour mettre à jour le modèle.

A.5.2 Approches basées sur la détection de lignes dans l'image

Chandran et al. (1987) proposent une solution un peu différente pour traiter le problème de la navigation autonome de véhicules de terrain. Leur méthode se base sur l'extraction des bords de la route dans l'image sur le principe suivant. Les auteurs fixent la direction de la caméra à partir de l'image courante et effectuent une prédiction de la scène suivante. Chandran et al. (1987) cherchent à estimer la position des points de la route et tout particulièrement des points qui vont se trouver à la limite d'entrée de l'image (points les plus proches de la caméra). Ils cherchent ensuite à extraire les bords linéaires dans des fenêtres centrées sur le bord estimé. Ils commencent par traiter les parties de l'image correspondant à des parties de la route les plus proches de la caméra (voir la figure A.10).

Deux représentations pour la route sont proposées : l'une basée sur les bords de la route, l'autre sur la région occupée par la route. Pour la première représentation, l'algorithme de détection de la route commence par évaluer les bords les plus proches de la caméra. Les auteurs construisent ensuite l'histogramme des niveaux de gris de

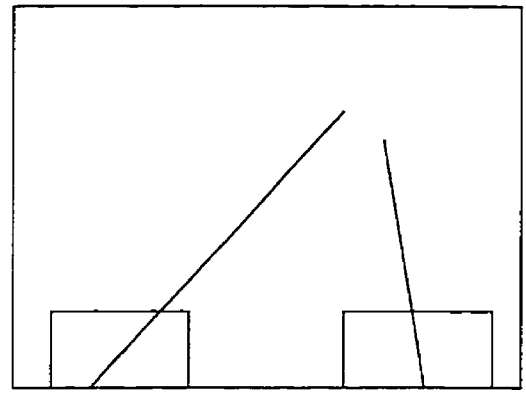


Figure A.10 : Estimation des bords de la route proches de la caméra

cette partie de l'image connue en supposant que les points entre les bords de la route appartiennent à la route. Cela permet de choisir un seuil d'erreur pour déterminer si un point quelconque de l'image appartient ou non à la route. Les auteurs relient les points du bord de la route obtenus et évaluent les droites constituant les bords de la route par la technique de vote dite de Hough. Un deuxième algorithme moins coûteux se base sur les régions de l'image. L'idée est de minimiser dans chaque fenêtre le total des "points de route" dans la région "non-route" et des "points de non-route" dans la région "route".

Liou et Jain (1987) proposent un algorithme qui s'appuie sur le point de fuite dans l'image pour détecter les bords de la route. Les bords de la route sont approximés par des segments de droite. Les auteurs supposent que la courbure de la route varie de façon douce, si bien que d'une image à l'autre le point de fuite dans l'image a un petit déplacement. Cela permet de chercher le point de fuite dans une fenêtre placée

autour du point de fuite détecté dans l'image précédente. Les bords de la route sont localisés en considérant toutes les lignes qui sont susceptibles de converger en ce point (voir la figure A.11).

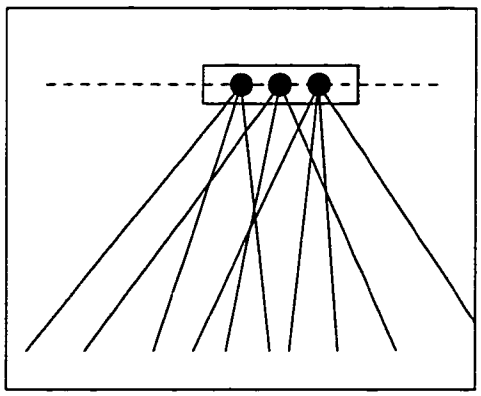


Figure A.11 : Estimation des bords de la route à partir du point de fuite

Les auteurs utilisent l'information sur la largeur de la route et le modèle de la caméra pour sélectionner les paires de lignes appropriées pour les bords de la route. Les deux meilleures lignes sont sélectionnées en cherchant le maximum des "mesures de la performance" calculées pour chaque ligne détectée en se basant sur les critères suivants : longueur de la ligne, localisation, amplitude moyenne du gradient des points, consistance des segments de ligne. Les coordonnées des points situés sur les différentes lignes convergentes possibles sont calculées à partir des bords de la route détectés à l'image précédente.

Dans (Dickmanns 1989), un système intelligent basé sur la vision permet à un véhicule de suivre une route. L'auteur propose un modèle dynamique qui permet de

prédire le vecteur d'état du véhicule à l'instant $t + 1$ à partir du vecteur d'état à l'instant t (par des méthodes de type filtre de Kalman). Ce modèle est basé sur la représentation de la route par une clothoïde. Ce processus permet de prédire l'image à l'instant $t + 1$. L'image réellement mesurée est analysée pour détecter les bords de la route. La recherche s'effectue dans un ensemble de fenêtres rectangulaires placées aux endroits de l'image supposés contenir les bords de la route. Dans chaque fenêtre, l'auteur cherche des éléments de contour linéaires dans une direction préférentielle estimée. L'interprétation des séquences d'images prédites et mesurées permet d'ajuster le modèle de la route.

Dans le cadre du système de vision développé pour la navigation d'un véhicule autonome sur une route, Behringer et Hötzl (1994) proposent une méthode permettant d'estimer l'angle de tangage de la caméra et la largeur de la route. En effet, pour réaliser le contrôle latéral du véhicule de façon autonome, le système doit connaître la position et le cap courants du véhicule par rapport au centre de la route. L'estimation de cet état est réalisée récursivement (filtre de Kalman) en détectant la route dans l'image. La cinématique du véhicule est donnée par un ensemble d'équations différentielles. Le modèle est décrit par les variables d'état Y_v (déplacement latéral du véhicule par rapport au centre de la route), Ψ_v (cap) et β_v (glissement). La route est modélisée par une clothoïde. Un ensemble de vecteurs d'état décrit les modèles de la route et de la dynamique du véhicule dans l'espace et dans le temps (4D). La détection des bords de la route se fait dans des portions de l'image sélectionnées et en

appliquant des techniques de corrélation. La prédiction de la localisation des bords de la route dans l'image dépend de l'angle de tangage de la caméra, qui peut varier au cours du temps, particulièrement dans les situations où le véhicule accélère ou freine brusquement. C'est pourquoi l'angle de tangage est estimé comme une variable d'état et mis à jour à chaque cycle. Les auteurs montrent que l'indice le plus fiable pour estimer l'angle de tangage est la largeur de la route dans l'image à une certaine hauteur de l'image à condition que la largeur réelle soit connue. Un effet de changement de largeur de la route peut être différencié d'un effet de changement d'angle de tangage en effectuant plusieurs mesures de la largeur à des hauteurs différentes dans l'image. Dans le cas où la largeur de la route varie linéairement, l'ambiguïté peut être levée en considérant la séquence d'images et non plus une image unique. Les auteurs introduisent des modèles dynamiques pour l'estimation de ces effets. Ils modélisent le mouvement de l'angle de tangage par un oscillateur harmonique 1D excité par des perturbations. La dynamique de la largeur de la route est représentée par les variables d'état suivantes : la largeur de la route approximée à la position courante du véhicule, la variation de la largeur de la route estimée à une distance L du véhicule et la variation de largeur après le changement de largeur de la route. Pour la prédiction du filtre de Kalman, les auteurs calculent la matrice de transition d'état qui est la solution du système d'équations différentielles homogènes formées pour le vecteur d'état. Les variables d'état sont obtenues à partir des mesures effectuées dans l'image et en utilisant les équations de la projection perspective et des transformations de

coordonnées.

A.5.3 Approches basées sur la prédiction du déplacement des points dans la séquence d'images

Chioiu et Hervé (1996) proposent une méthode d'évitement d'obstacles à partir de la détection des points du sol. La méthode est basée sur la prédiction du déplacement des points du sol entre deux images consécutives, connaissant le déplacement de la caméra. Elle consiste à génèrer l'image virtuelle (prédiction de l'image suivante de la séquence) dans laquelle tous les pixels appartiennent au sol. Dans cette image virtuelle, les régions qui ne correspondent pas effectivement à des points du sol vont être affectés par des distorsions de parallaxe. L'algorithme évalue le degré de similarité entre les régions équivalentes de l'image réelle et de sa prédiction en calculant pour chaque pixel de l'image un coefficient de corrélation entre des fenêtres rectangulaires centrées sur le pixel considéré. Chaque pixel est alors étiqueté comme appartenant ou non au sol en fonction de la valeur de son coefficient de ressemblance.

Annexe B

Approximation du contour du sol

B.1 Recherche des points de contour

En appliquant l'algorithme de Chioiu et Hervé (1996), nous obtenons les pixels de l'image qui correspondent à des points de sol (voir la figure B.1).

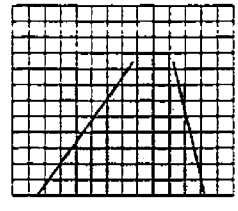


Figure B.1 : Points du sol dans l'image

La première étape consiste à éliminer les pixels isolés correspondant à du bruit. Nous dressons la liste ordonnée des points du contour du sol. La recherche se fait sur la partie inférieure de l'image où sont situés les points de sol. En effet, l'axe optique

de la caméra étant parallèle au plan du sol, la ligne d'horizon est au centre de l'image. Pour trouver les points du contour, il suffit de prendre le premier et le dernier point de chaque ligne de l'image ayant une étiquette "sol".

B.2 Segments correspondant au corridor où est situé le robot

Chaque intersection rencontrée par le robot est modélisée par le système et mémorisée. Quand le robot s'engage dans un des corridors composant l'intersection rencontrée, le système a en mémoire ses caractéristiques : c'est un corridor droit ou courbe dont la largeur est connue. Au cours du déplacement du robot dans le corridor, le système analyse la séquence d'images captée par la caméra montée sur le robot.

Nous proposons une méthode pour déterminer le contour du sol correspondant à la partie du corridor connue. C'est la partie du corridor la plus proche du robot. Notre analyse porte donc sur les points de l'image situés dans une fenêtre de recherche correspondant à des points de sol situés près de la caméra (voir la figure B.2).

B.2.1 Cas d'un corridor droit

Considérons le cas d'un corridor droit. Les bords du corridor se projettent dans l'image par des segments de droite. Nous prolongeons ces segments de droite pour

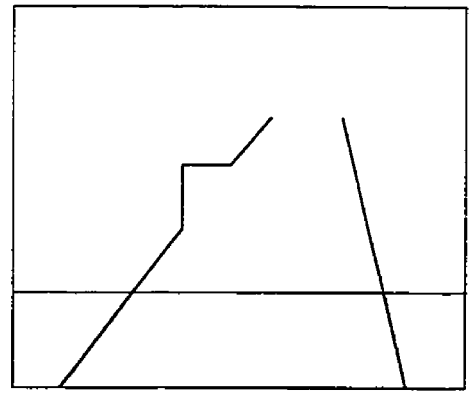


Figure B.2 : Estimation des bords du corridor connu dans une fenêtre de recherche

obtenir des droites que nous paramétrisons par leur point de fuite $P = (a, 0)$ et l'angle par rapport à l'horizon α (voir la figure B.3). Avec cette paramétrisation, l'équation d'une droite dans l'image s'écrit :

$$x + y/\tan\alpha = a.$$

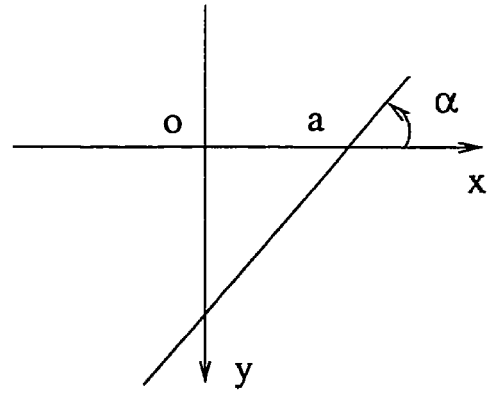


Figure B.3 : Paramétrisation d'une droite dans l'image

Les bords gauche (B_g) et droit (B_d) du corridor dans lequel se trouve le robot

se coupent sur la ligne d'horizon comme le montre la figure B.4. Leurs équations s'écrivent :

$$(B_g) \ : x + y/\tan\alpha_g = a \tag{B.1}$$

$$(B_d) \ : x + y/\tan\alpha_d = a \tag{B.2}$$

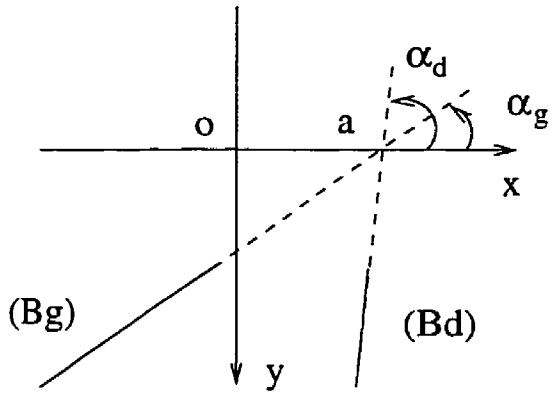


Figure B.4 : Paramètres des bords du corridor connu dans l'image

Les paramètres a, α_g et α_d peuvent être déterminés par la technique de vote de Hough. Le principe de la méthode est décrit par Ballard et Brown (1982). Les points (x, y) appartenant au bord gauche (B_g) du corridor vérifient l'équation B.1 et les points (x, y) appartenant au bord droit (B_d) du corridor, l'équation B.2. Cette technique peut être utilisée pour approximer les segments (B_g) et (B_d). Mais nous avons une contrainte supplémentaire que nous pouvons mettre à profit : le système connaît la largeur du corridor (dans l'espace tridimensionnel). Cela permet à la

fois d'obtenir une mesure plus précise des paramètres des segments et d'éliminer un paramètre. En effet, connaissant la largeur du corridor, un point de (B_d) peut directement voter pour une position de (B_g).

Pour déterminer la relation entre les paramètres de (B_g) et de (B_d), nous sommes obligés de nous ramener à l'espace tridimensionnel dans lequel est connue la largeur du corridor. Dans un premier temps, nous définissons le modèle de la caméra et les repères choisis.

Pour modéliser la caméra, nous utilisons le modèle du trou d'épingle qui produit une projection perspective des points de la scène dans l'image (Horn 1986). Un point M de la scène est représenté par son vecteur de coordonnées par rapport au repère de la caméra R_c, $M = (X, Y, Z)'$. Sa projection dans l'image est représentée par son vecteur de coordonnées par rapport à R_c, $m = (x, y, f)$. La projection perspective est décrite par la relation suivante :

$$x = \frac{fX}{Z}, \quad y = \frac{fY}{Z},$$

avec f distance focale de la caméra. Ou encore si la distance focale en x est différente de la distance focale en y :

$$x = \frac{f_x X}{Z}, \quad y = \frac{f_y Y}{Z}.$$

Pour travailler avec une image droite et une distance focale positive, le plan de l'image est avancé devant le centre optique (voir la figure B.5).

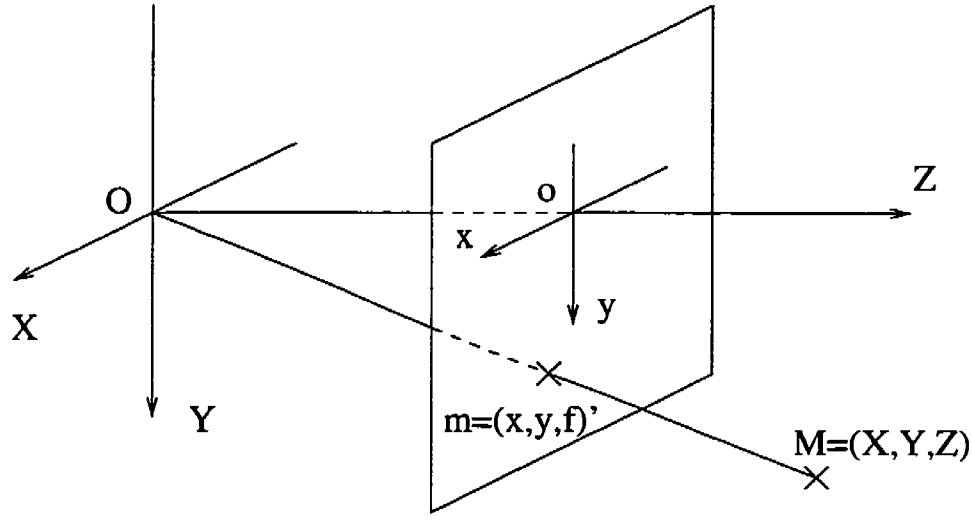


Figure B.5 : Le modèle du trou d'épingle

Nous supposons que la caméra est placée à une hauteur fixe h par rapport au plan du sol et que son axe optique est parallèle au plan du sol. La figure B.6 montre le référentiel absolu R_a et le repère de la caméra R_c choisis.

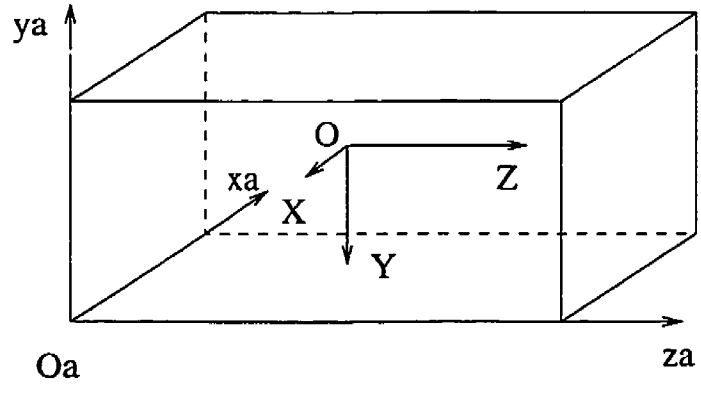


Figure B.6 : Référentiels choisis dans un corridor idéalisé

Dans le repère de la caméra et en utilisant la paramétrisation de Hough (voir la figure B.7), l'équation d'un bord d'un corridor droit s'écrit :

$$\begin{cases} R = X\cos\Theta + Z\sin\Theta \\ Y = h \end{cases}$$

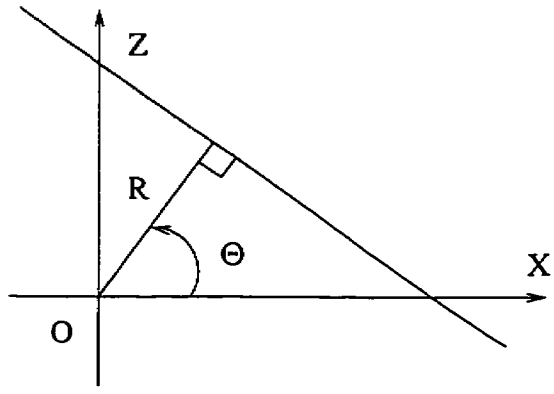


Figure B.7 : Équation d'une droite dans le repère de la caméra

En utilisant les équations de la projection perspective, une droite se projette dans l'image en :

$$x + \frac{y}{\frac{-f_y h}{R f_x \cos\Theta}} = -f_x \tan\Theta \qquad \text{(B.3)}$$

Les deux bords (B_g) et (B_g) du corridor, parallèles, sont représentés avec le même paramètre Θ et des rayons R_g et R_d (voir la figure B.8). Si un corridor a pour largeur l, nous avons la relation :

$$l = \mid R_g - R_d \mid .$$

Nous choisissons $R_g > 0$ dans le cas du corridor connu et nous avons :

$$l = R_d - R_g.$$

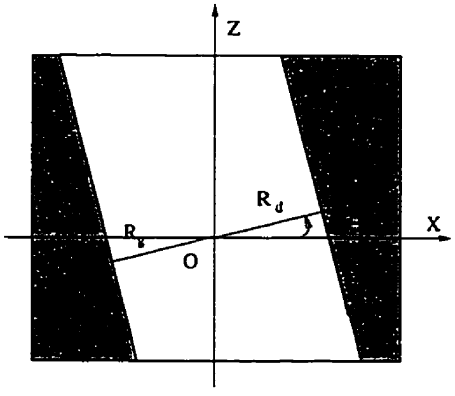


Figure B.8 : Équation des bords d'un corridor dans le repère de la caméra

Nous identifions les équations B.1, B.2 et B.3 pour les bords (B_g) et (B_g) du corridor et nous obtenons les relations suivantes :

$$\begin{aligned} \tan\alpha_g &= \frac{-f_y h}{R_g f_x \cos\Theta} \\ \tan\alpha_d &= \frac{-f_y h}{R_d f_x \cos\Theta} \\ a &= -f_x \tan\Theta \end{aligned}$$

d'où nous déduisons la relation :

$$\frac{1}{\tan\alpha_g} - \frac{1}{\tan\alpha_d} = \frac{l f_x \cos(\arctan(-a/f_x))}{f_y h}$$

de laquelle nous tirons α_d en fonction de α_g :

$$\frac{1}{\tan \alpha_d} = \frac{1}{\tan \alpha_g} - \frac{l f_x \cos(\arctan(-a/f_x))}{f_y h}$$

Les points (x, y) appartenant au bord gauche du corridor (B_g) votent pour les paramètres a et α_g de la droite d'équation B.1 et les points (x, y) appartenant au bord droit du corridor (B_d) votent pour les paramètres a et α_g de la droite d'équation :

$$x + y\left(\frac{1}{\tan \alpha_g} - \frac{l f_x \cos\left(\arctan(-a/f_x)\right)}{f_y h}\right) = a.$$

Obtenant ainsi a et α_g, nous en déduisons α_d.

Nous obtenons ainsi les équations des deux segments de droite (pour un corridor droit) correspondant aux bords du corridor proches du robot. Ces deux segments sont prolongés pour obtenir une approximation de ce que devrait voir le robot s'il n'y avait pas d'intersection (voir la figure B.9).

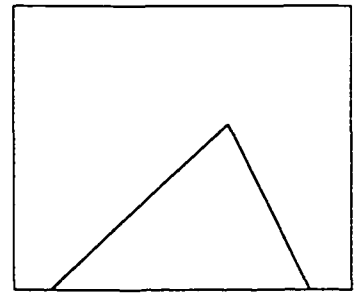


Figure B.9 : Modèle du contour du sol

B.2.2 Cas d'un corridor courbe

L'image d'un cercle C en projection perspective est donnée par l'intersection du plan image avec le demi-cône de sommet O, le centre optique de la caméra, généré par le cercle C. Dans le cas qui nous intéresse ici, l'axe optique de la caméra est parallèle au sol, à une hauteur h au dessus de celui-ci. L'image du cercle C sera donc :

- une ellipse si le centre optique de la caméra se tient en dehors de C (voir la figure B.2.2(a)) ;
- une parabole si le centre optique de la camera se tient à la verticale de C (voir la figure B.2.2(b)) ;
- une branche d'hyperbole si le centre optique de la camera se tient à l'intérieur de C (voir la figure B.2.2(c)).

Bien entendu, on n'observera jamais dans l'image la conique complète, mais un arc de cette conique. Des considérations pratiques nous amènent à simplifier quelque peu le problème. Tout d'abord, nous pouvons éliminer le cas de la parabole qui ne sera en pratique jamais observée. En effet, il correspond à une configuration instable : un déplacement arbitrairement petit de la caméra fera rentrer la caméra dans le cercle ou l'en fera sortir. De même, dans le cas qui nous intéresse, nous pouvons distinguer un arc d'ellipse d'un arc d'hyperbole à l'aide du signe de la courbure de cet arc de courbe. En effet, comme les arcs de cercle observés correspondent à la trace au sol d'une paroi opaque, au plus une moitié du cercle est visible par la caméra. De plus, la courbure

de ces cercles tracés sur le sol est elle-même limitée, puisque les couloirs circulaires doivent être empruntés par des véhicules. Par conséquent, comme le montre la figure suivante, si la caméra se trouve à l'extérieur du cercle observé, elle ne peut en voir que la partie la plus proche, qui donnera un arc convexe (selon la verticale) dans la figure B.2.2(a). De même, si la caméra se trouve à l'intérieur du cercle, l'arc de cercle visible se projettera comme un arc concave dans la figure B.2.2(b).

Le cercle observé a pour équation, dans le plan (X, Z) :

$$(X - X_0)^2 + (Z - Z_0)^2 = R_0^2, \tag{B.4}$$

où (X_0, Y_0) et R_0 sont respectivement les coordonnées du centre et le rayon du cercle, c'est-à-dire les inconnues de notre problème. La hauteur h de la caméra par rapport au sol nous permet d'exprimer la profondeur, Z, en fonction de la coordonnée y dans l'image :

$$Y = h \text{ et } y = \frac{fh}{Z} \Rightarrow \frac{f}{Z} = \frac{y}{h} \text{ ou } Z = \frac{fh}{y}. \tag{B.5}$$

Si nous multiplions les deux côtés de l'équation B.4 par le terme $(f/Z)^2$, nous obtenons :

$$\left(\frac{fX}{Z} - X_0\frac{f}{Z}\right)^2 + \left(f - Z_0\frac{f}{Z}\right)^2 = \left(\frac{f}{Z}\right)^2 R_0^2.$$

$$\left(x - \frac{X_0}{h}y\right)^2 + \left(f - \frac{Z_0}{h}y\right)^2 = \frac{R_0^2}{h^2}y^2.$$

Nous pouvons maintenant obtenir l'expression suivante pour X_0 :

$$X_0 = \frac{h}{y} \cdot \left[x \pm \left(\frac{R_0^2}{h^2} y^2 - \left(f - \frac{Z_0}{h} y \right)^2 \right)^{1/2} \right]$$

$$= \frac{h}{y} \cdot \left[x \pm \left(\left(\frac{R_0 - Z_0}{h} y + f \right) \cdot \left(\frac{R_0 + Z_0}{h} y - f \right) \right)^{1/2} \right].$$

Si nous adoptons une méthode basée sur la transformation de Hough pour détecter et reconstruire les projections d'arcs circulaires dans l'image, chaque pixel d'arête doit voter pour un ensemble de combinaisons de paramètres représentant chacune une conique différente, c'est-à-dire pour un ensemble de valeurs du triplet (X_0, Z_0, R_0). Nous venons de voir que, pour des valeurs de Z_0 et R_0 et un point dans l'image donnés, il est possible de déterminer la valeur de X_0. Il nous reste donc à déterminer l'intervalle sur lequel (Z_0, R_0) peut prendre ses valeurs.

Le rayon de courbure est limité en valeur inférieure par l'encombrement des véhicules. En général, la valeur minimale pour R_0 est de l'ordre de 4m (WAGNER MINING EQUIPMENT CO. 1989). La valeur maximale, R_{max}, dépend par contre de la résolution et du champ de vue de la caméra. Dans le cas où la conique observée est une ellipse (la caméra se trouve en dehors du cercle), nous avons nécessairement $Z_0 > R_0$. En fait, nous pouvons déterminer de manière plus précise la valeur de la borne inférieure pour Z_0, à partir des caractéristiques de la caméra. En pratique, il suffit de considérer que $Z_0 \geq R_0 + 1\text{m}$. La borne supérieure de Z_0 est fixée par la limite de visibilité et par la résolution de la caméra. En pratique, il n'est pas réa-

liste de chercher à déterminer les paramètres de l'ellipse pour $Z_0 - R_0 > 10\text{m}$. Par conséquent, nous obtenons :

$$R_0 \in [4\text{m}, R_{max}] \text{ et } Z_0 \in [R_0 + 1, R_0 + 10].$$

Déterminons R_{max}. Soit M_1 le point de l'espace correspondant au point image d'ordonnée maximum appartenant à l'ellipse et M_2 le point de l'espace correspondant au point image d'ordonnée minimum appartenant à l'ellipse pour un espacement horizontal maximum de la largeur de l'image, $2x_M$. Si ces deux points sont séparés de Δ en ordonnée, les équations de la projection perspective nous donnent :

$$\begin{aligned} X_1 &= -\frac{x_M}{y_M - \Delta}h, & Z_1 &= \frac{fh}{y_M - \Delta}, \\ X_2 &= \frac{x_M}{y_M}h, & Z_2 &= \frac{fh}{y_M}. \end{aligned}$$

Le centre du cercle C est donné par :

$$C = M_1 - R\begin{pmatrix} 0 \\ \\ 1 \end{pmatrix}. \tag{B.6}$$

M_2 vérifie l'équation du cercle :

$$(M_2 - C)^2 = R^2. \tag{B.7}$$

Les équations B.6 et B.7 donnent :

$$\left(M_2 - M_1 + R \begin{pmatrix} 0 \\ \\ 1 \end{pmatrix} \right)^2 = R^2$$

$$(\boldsymbol{M_1 M_2} + R\mathbf{j})^2 = R^2.$$

d'où nous tirons :

$$R_{max} = \frac{M_1 M_2^2}{2\boldsymbol{M_1 M_2} \cdot \mathbf{j}}$$

Dans le cas de l'hyperbole, nous obtenons le même intervalle pour R_0 que dans le cas de l'ellipse. Z_0, par contre, peut théoriquement varier entre $+R_0$ et $1 - R_0$. Si nous prenons en compte la limite de visibilité L_v, nous obtenons :

$$Z_0 \in [1 - R_0, \min(L_v/2, R_0)].$$

B.3 Autres segments

B.3.1 Mesure de similarité

Nous cherchons à comparer le modèle du contour au contour réellement mesuré dans l'image. La conformité du contour mesuré avec le modèle est évaluée en chaque point par une mesure de similarité. Les ensembles de points consécutifs qui ne correspondent pas au modèle prédit devront être étudiés.

Dans la littérature, plusieurs travaux traitent du problème de la mise en corres-

pondance de segments de droite entre deux images ou entre une image et un modèle soit comme un sujet à part entière soit dans le cadre de la reconnaissance de scènes.

McIntosh et Mutch (1988) proposent un algorithme qui permet de mettre en correspondance des segments de droite dans une paire d'images stéréoscopiques à partir d'une fonction de correspondance. Cette fonction combine huit paramètres caractéristiques de chaque segment, pondérés en fonction de leur importance relative dans chaque image. Les lignes qui se correspondent sont celles qui ont la fonction de correspondance la plus élevée. Un segment de droite est défini dans une région de pixels adjacents ayant un gradient de forte intensité et d'orientation similaire. McIntosh et Mutch (1988) considèrent les paramètres suivants : la longueur des segments, leur largeur, leur contraste, leur gradient, l'intensité moyenne des zones claire et foncée, l'angle entre les deux segments comparés et leur disparité. La similarité entre deux segments A et B est définie par :

$$V_i = \frac{min(Param_i(Segment_A), Param_i(Segment_B))}{max(Param_i(Segment_A), Param_i(Segment_B))}$$

avec i = longueur, largeur, contraste, gradient, clair, foncé. Pour l'angle et la disparité, deux seuils maximum sont prédéfinis. Les similarités pour ces deux paramètres sont données par :

$$V_{angle} = \frac{MaxAngleDifference - \mid Angle(Segment_A) - Angle(Segment_B) \mid}{MaxAngleDifference}$$

et

$$V_{disparite} = \frac{MaxDisparite\text{-}Distance(Milieu(Segment_A) - Milieu(Segment_B))}{MaxDisparite}.$$

La fonction de correspondance est alors :

$$\forall i, M_{AB} = \sum V_i W_i,$$

avec W_i =poids du paramètre i.

Lowe (1987) cherche à mettre en correspondance les segments de droite détectés dans l'image avec la projection du modèle de l'objet considéré. Pour cela, Lowe (1987) calcule la probabilité de faire une erreur en mettant en correspondance un segment particulier de l'image avec un segment particulier du modèle. Deux types d'erreur sont considérées : des erreurs provenant de la prédiction et des erreurs dues à la proximité de segments de droite dans l'image. Le premier type d'erreur est modélisé en supposant que les segments de l'image qui sont de mauvais candidats sont uniformément distribués en termes d'orientation, de position et d'échelle. L'expression suivante donne alors une mesure de la probabilité d'obtenir une mauvaise mise en correspondance dans les limites d'orientation et de séparation perpendiculaire spécifiées :

$$N = \frac{4\theta s(p-l)}{\pi l^2},$$

avec les paramètres θ, s, p, l définis sur la figure B.11.

La deuxième source d'erreurs est détectée en examinant toutes les correspondances possibles pour chaque segment prédit pour déterminer l'ambiguïté entre plusieurs segments candidats et la probabilité $P(W)$ de choisir le mauvais candidat parmi les deux meilleurs M et N est donnée par :

$$P(W) = \frac{P(M)}{P(N) + P(M)}.$$

Rohr (1994) cherche à reconnaître dans une séquence d'images les positions d'un humain en train de marcher. Les humains sont modélisés par le modèle 3D de Marr (Marr 1982) constitué de cylindres. Leur projection en 2D donne des segments de droite. Un des problèmes consiste à mettre en correspondance un modèle d'un humain placé dans une certaine position avec les segments de droite détectés dans l'image. Rohr (1994) considère des fenêtres de recherche autour de chaque segment du modèle et utilise une mesure de similarité entre le segment du modèle considéré et chaque segment (ou bout de segment) de l'image se trouvant dans la fenêtre. Cette mesure de similarité prend en compte trois quantités géométriques : la projection d'un segment de l'image sur le segment du modèle l_i, la distance entre le milieu du segment de l'image d_i et le modèle et l'angle entre les deux segments considérés $\Delta\phi_i$ (voir la

figure B.12) et s'exprime par :

$$s_i = l_i \exp\left(-\frac{1}{2}\left(\frac{(l_{Mi} - l_i)^2}{\sigma_{li}^2} + \frac{d_i^2}{\sigma_{di}^2} + \frac{\Delta\phi_i^2}{\sigma_{\Delta\phi}^2}\right)\right).$$

Les paramètres σ_{li} et σ_{di} dépendent de la longueur du segment du modèle l_i. $\sigma_{\Delta\phi}$ est constant.

Dans la méthode proposée par McIntosh et Mutch (1988), l'évaluation des gradients et des intensités lumineuses est pertinente car ils comparent deux images stéréoscopiques. Dans notre cas, ce n'est pas applicable car nous avons seulement un modèle géométrique. De plus, il n'est pas possible non plus de se baser sur les critères de longueur et de largeur des segments car nous cherchons des différences locales avec le modèle et non pas une ressemblance globale de l'ensemble des segments avec le modèle du contour. Enfin, la méthode de détection des segments en régions ne donne pas une bonne précision sur leur localisation, si bien que les critères de position et d'orientation sont peu fiables. La solution de Lowe (1987) est intéressante mais elle n'exploite pas les connaissances à priori sur la position des segments dans l'image. La mesure de similarité de Rohr (1994) est la plus intéressante pour notre problème. Elle utilise uniquement des critères géométriques et les prédictions que l'on peut faire sur l'emplacement des points en travaillant dans des fenêtres de recherche. Nous nous basons sur cette approche pour définir une mesure de similarité entre le modèle du contour et le contour réellement mesuré dans l'image.

Nous travaillons dans des fenêtres de recherche centrées sur le modèle du contour

(voir la figure B.13). Nous utilisons une mesure de similarité entre le modèle du contour et les points en niveaux de gris qui prend en compte deux quantités géométriques : la distance du point au modèle du contour d_i et l'angle entre la direction de l'arête en ce point et le modèle $\Delta\phi_i$. La similarité s_i en un point i s'écrit :

$$s_i = \exp\left(-\frac{1}{2}\left(\frac{d_i^2}{\sigma_d^2} + \frac{\Delta\phi_i^2}{\sigma_{\Delta\phi}^2}\right)\right),$$

avec σ_d et $\sigma_{\Delta\phi}$ constants. Les points ayant une mesure de similarité supérieure à un certain seuil correspondent au modèle. Il reste à étudier les autres points.

B.3.2 Estimation des segments

Nous cherchons tout d'abord à estimer les segments qui correspondent à des ouvertures dans les parois, autrement dit à des branches de corridors s'ouvrant dans le corridor principal. Pour ce faire, nous utilisons un détecteur d'arêtes (par exemple un filtre gaussien) pour trouver toutes les lignes verticales de l'image (voir la figure B.14). Nous éliminons les lignes qui ont une longueur trop petite pour correspondre à des arêtes d'intersection de deux corridors. Nous considérons chaque groupe de points consécutifs extraits du contour du sol qui n'appartiennent pas au modèle du contour et nous les comparons avec les lignes obtenues par la même mesure de similarité que précédemment.

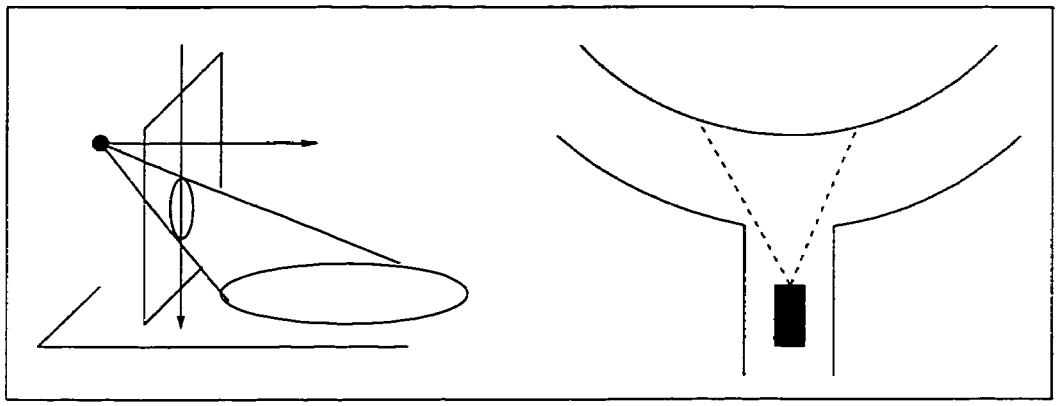
Il s'agit ensuite d'estimer les points suivant (ou précédant) les points d'occlusion qui n'appartiennent pas au modèle. Cela peut se faire par la méthode de Hough,

sachant que nous pouvons facilement avoir au départ une estimation de la position et de l'orientation du segment recherché (voir la figure B.15).

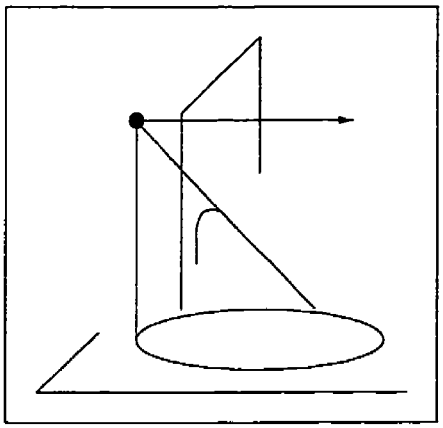
Il nous reste encore à approximer les points qui correspondent à une limite de visibilité du robot. Ces points appartiennent en 3D à un arc de cercle. Ils se projettent donc dans l'image sur un arc de conique. Nous pourrions trouver une approximation par une technique de Hough mais le résultat ne serait sans doute pas très bon étant données la qualité des images et l'imperfection du modèle. Il est en effet peu probable que ces points se projettent effectivement sur un arc de conique. En fait, nous n'avons pas besoin de connaître la courbe décrite par ces points. Il nous suffit de savoir que ces points correspondent à une limite de visibilité du robot. Ces points sont caractérisés par des niveaux de gris très bas. Il suffit donc de faire un histogramme des niveaux de gris de l'image et de regarder si ces points appartiennent bien aux niveaux de gris les plus bas.

B.3.3 Cas particulier

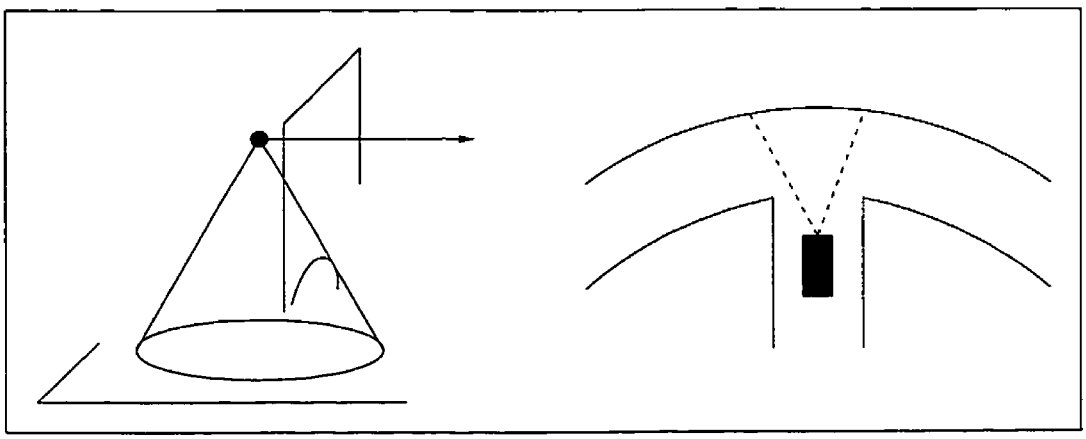
Enfin, une fois tous les cas précédents éliminés, il peut rester un cas très particulier : celui des points qui correspondent à un rétrécissement du corridor (voir l'exemple de la figure B.16). Le point de fuite est connu. Il suffit donc de déterminer les directions des bords droit et gauche du corridor par une technique de Hough. Il reste ensuite les segments joignant les deux corridors.



(a)



(b)



(c)

Figure B.10 : Conique obtenue dans l'image selon la position de la caméra par rapport au cercle $\mathcal{C}$: (a) ellipse ; (b) parabole ; (c) branche d'hyperbole.

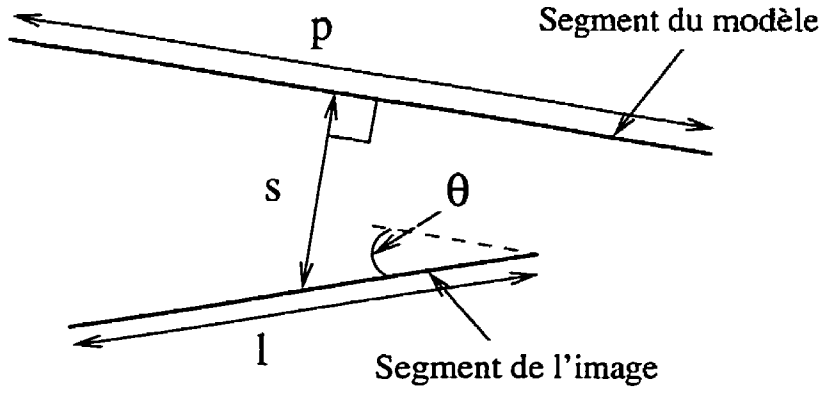


Figure B.11 : Mesures utilisées pour calculer la probabilité de mauvaise mise en correspondance entre un segment de l'image et un segment du modèle

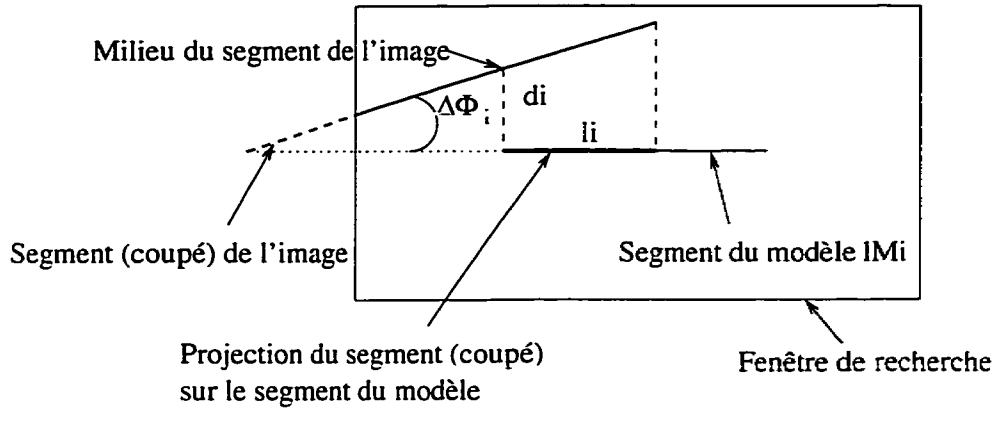


Figure B.12 : Comparaison entre un segment de l'image et le modèle

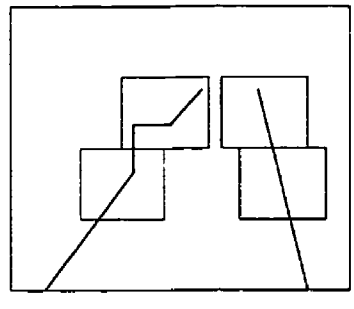


Figure B.13 : Estimation des bords du corridor dans des fenêtres de recherche

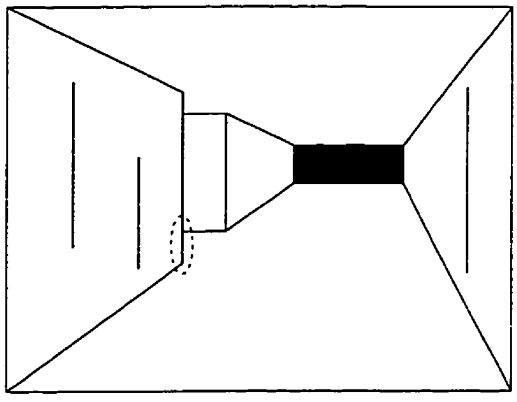


Figure B.14 : Détection des lignes verticales dans l'image

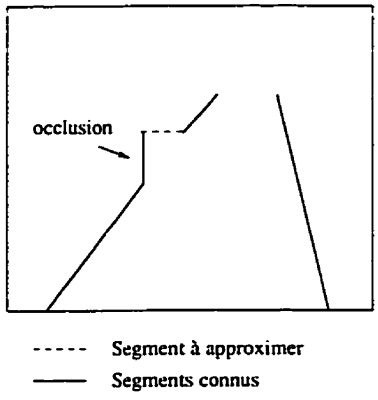


Figure B.15 : Approximation des segments liés à une occlusion dans l'image

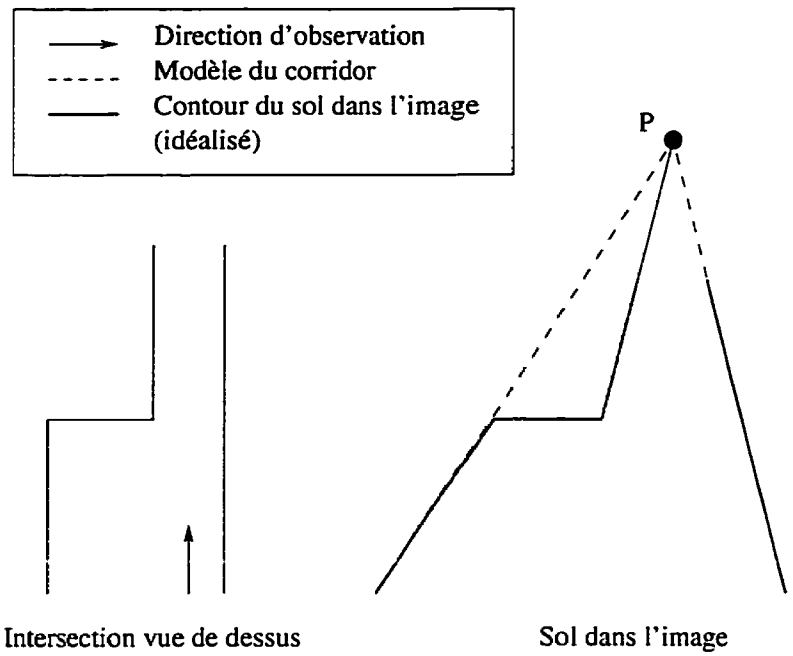
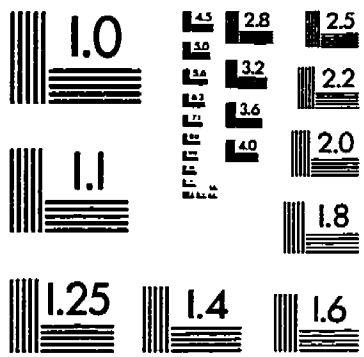
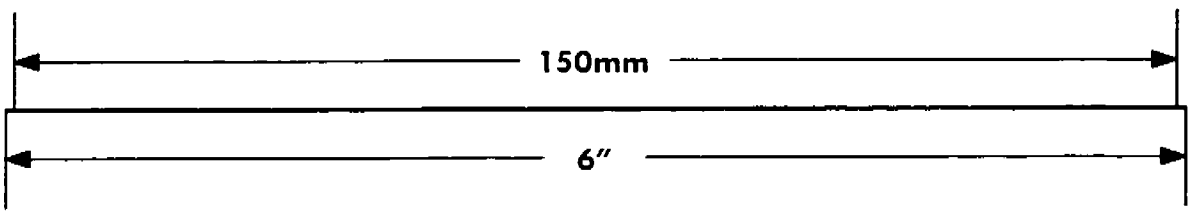


Figure B.16 : Rétrécissement d'un corridor droit

IMAGE EVALUATION
TEST TARGET (QA-3)





APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989