Estimation des segments 2D : un algorithmerobuste

Ming Xie, Patrick Rives

INRIA

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél. (1) 39 63 55 11

Rapports de Recherche

N° 909

ESTIMATION DES SEGMENTS 2D : UN ALGORITHME ROBUSTE

Programme 8

Ming XIE
Patrick RIVES

Octobre 1988

ESTIMATION DES SEGMENTS 2D :

UN ALGORITHME ROBUSTE

Ming XIE
Patrick RIVES

Publication Interne n° 429

Septembre 1988

IRISA INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CÉDEX
FRANCE
Téléphone : 99 36 20 00
Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Estimation Des Segments 2D: Un Algorithme Robuste

===

From Edges To Line Segments: A Robust Algorithm

M. XIE and P. RIVES

INRIA/IRISA de Rennes

Publication Interne n°429 - 34 Pages - Septembre 1988

Campus de Beaulieu

F-35042 Rennes Cedex FRANCE

Tel (33)-99-36-20-00

Résumé

L'extraction de l'information concernant la structure des formes presentées dans une image est un problème très délicat dans le domaine de la vision par ordinateur. Bien qu'il ait été étudié depuis longtemps, il n'y a pas de méthode qui soit en même temps robuste, rapide et générale. Nous présentons dans cet article un algorithme de chaînage des contours qui traite ce problème en deux étapes : (a) L'extraction des chaînes et (b) L'approximation polygonale des chaînes. Dans notre approche, nous introduisons d'abord une technique de réorganisation des données des pixels contours. Cette opération permet d'accélérer considérablement le chaînage des contours. Nous travaillons sur les couples contours au lieu des pixels contour. Un couple contour est défini comme étant un sous-ensemble de pixels contour se situant consécutivement sur une même ligne de l'image. Une image de contours est donc transformée en un tableau des couples contours. Le chaînage des contours est devenu le chaînage

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
(L. A. 227)
UNIVERSITÉ DE RENNES 1 I. N. S. A. DE RENNES

INSTITUT NATIONAL DE RECHERCHE
EN INFORMATIQUE ET EN AUTOMATIQUE
(LABORATOIRE DE RENNES)

des couples contours. Ensuite nous définissons une zone de voisinage causale pour chaque couple contour. Le choix de cette définition provient du fait qu'une zone de voisinage causale possède une propriété intéressante qui assure la robustesse de notre algorithme. De plus, la taille de zone de voisinage n'est pas fixée, ce qui rend l'algorithme de chaînage très souple, adapé à toutes sortes d'images contours. On distingue quatre types de chaînes dans la zone de voisinage d'un couple contour. Selon la nature des chaînes présentes, l'opération de chaînage de notre algorithme consiste à faire : (a) la création des chaînes, (b) le rallongement des chaînes et (c) la fermeture des chaînes. L'autre point important est que les chaînes obtenues par notre approche sont d'une grande simplicité. Tous les pixels contour appartenants à une même chaîne sont alignés séquentiellement en ordre croissant par rapport à X et à Y dans un tableau . Ceci permet d'effectuer rapidement et directement la fraction de chaînes en détectant la courbure maximale de celle-ci. Toutes les chaînes fractionnées sont approximées par des segments droits à l'aide de la methode de "Moindre Carré".

Bien que notre algorithme s'applique originellement au chaînage des contours, il peut s'appliquer aussi au cas où l'on utilise la transformée de HOUGH pour interpoler l'ensemble des points 2D par des courbes.

Mots Clés : Contour, Chaîne, Segment, Zone de voisinage, Chaînage des contours, Fraction de chaîne, Approximation polygonale.

Abstract

The extraction of shap representation from a set of edge points is a essential issue in computer vision. Substantial investigation have been already devoted to the problem. But we can't find really a robuste and general algorithm about this subject. We present in this paper our method that can extract the line segments from a set of edgels succesfully and rapidely. We treat the problem in two steps : (a) the edge linking in order to obtain the chains and (b) the polygonal approximation in order to obtain the line segments. There are three important points in our algorithm. First, we have defined a new structure for representing the edge image (Matrix of Edgels) which allows us to accelerate the process of linking. We know that some edgels may be localized consecutively in a row of image. We can treat them

together. So we define them as a contour element. A edge image is then transformed into a table of contour elements and the edge linking is become the linking of contour elements. Secondly, we have defined a causal region of neighborhood for a contour element, which guarantee that our algorithm is very robust and very simple. Moreover, the size of region of neighborhood is not fixed, this is why that our algorithm is very adaptable and can be applied to all kinds of edge images. Then we define four types of chains that are inside the region of neighborhood for a given contour element. According to the types of chains in a region of neighborhood, the operations of linking consist of : (a) the creation of chain, (b) the lengthening of chain and (c) the closure of chain. Finaly, our algorithm produces the very simple chains. All of edgels belonging to one chain is memorized sequentially in a table with the increasing order in X and in Y direction. That allows each chain be splitted rapidely and directly by measuring the maximal curvature. All of chains splitted are then approximated by the line segments with the least square method.

Although our algorithm is applied originally to the problem of edge linking, it can be used to resolve the problem of interpolating a set of 2D points with 2D lines (HOUGH Transformation).

Key Words : Edge, Chain, Line Segment, Region of Neighborhood, Edge Linking, Splitting of Chain, Polygonal Approximation.

Table des matières

1 Introduction

La vision par ordinateur est une discipline qui tente de développer sur des machines informatiques des algorithmes ayant pour but de percevoir et d'interpréter un environnement 3D afin d'en extraire les caractéristiques nécessaires à la résolution d'un problème donné. L'étape de perception consiste à extraire l'information sensorielle. L'étape d'interprétation consiste à établir la liaison entre l'information sensorielle et une description symbolique du modèle de cet environnement. Le résultat de l'interprétation permet au système décisionnel d'éxécuter les actions nécessaires au bon accomplissement d'une tâche donnée.

Il y a deux types de stratégies utilisées afin d'atteindre les buts de la vision par ordinateur : ascendantes et descendantes. Les stratégies ascendantes tentent de construire à partir de l'information sensorielle (primitives 2D) une représentation de plus haut niveau (ex: primitives 3D) interprétable par le processus décisionnel. Les stratégies descendantes tentent de déduire à partir d'un ensemble d'objets connus par le système une description compatible avec les primitives extraites de l'image, le processus décisionnel étant alors basé sur une mesure de distance entre la représentation extraite de l'image et la description issue du modèle. En fait, la combinaison de ces deux stratégies dans un système de vision constitue une approche plus sophistiquée.

1.1 Contexte

Une des tendances récentes de recherches en vision par ordinateur consiste à prendre en compte une séquence d'images au lieu d'avoir une ou deux images dans le cas de vision statique. Cette approche est connue sous le nom de "vision dynamique" (si le mouvement est contrôlable, on l'appelle aussi "vision active") et possède deux avantages fondamentaux :

1. Certains problèmes rencontrés (par exemple: Shap from shading) en vision par ordinateur peuvent être résolus à partir d'équations linéaires. La solution trouvée est unique et stable.

2. La possibilité d'utiliser la cohérence temporelle au sein de la séquence d'images permet de raffiner les estimation des primitives par des méthodes de filtrage (par exemple : Filtre de KALMAN).

Si nous considérons le cas particulier des robots mobiles, l'utilisation de la vision dynamique procure d'autres avantages :

1. Premièrement, elle donne la possibilité au robot d'acquérir en ligne l'information sur son environnement de façon à pouvoir éxécuter correctement sa tâche dans un univers inconnu ou mal modélisé.

2. Deuxièmement, elle donne la possibilité de réaliser une commande en boucle fermée qui permet d'avoir :

   (a) La capacité de contrôler le mouvement de caméra en vue d'améliorer la qualité d'estimation de primitives 3D (vision active).

   (b) La capacité de compenser l'incertitude sur les primitives (estimées) par une commande en boucle fermée ayant de bonne propriétés de robustesse et de stabilité.

Nous avons focalisé nos recherches sur l'utilisation d'une caméra embarquée sur un robot et d'algorithmes de vision dynamique (ou active). Notre objectif de recherche se définit de la façon suivante :

> *En supposant que le robot se déplace avec une vitesse contrôlabe dans un environnement inconnu composé d'objets polyédriques, essayer de reconnaître cet environnement à partir de la séquence d'images observées par une caméra embarquée sur le robot.*

Nous travaillons dans le cadre de la vision monoculaire. Puisque l'application de notre recherche est restreint à un univers polyédrique, nous portons naturellement attention aux primitives de type : segments 2D et segments 3D qui sont les types de primitives plus distinctifs. Nous avons proposé un système de vision qui répond à ce besoin. Les aspects principaux de ce systèmes sont :

1. Estimation des segments 2D en mouvement (dynamiques).
   Il s'agit d'intégrer des techniques d'extraction de contours statiques, d'extraction de contours spatio-temporels et de chaînage des contours

afin d'estimer les segments 2D dynamiques le long d'une séquence d'images.

2. Commande en boucle fermée.
   En vue d'améliorer la qualité de l'estimation des segments 2D dynamiques, on génère un mouvement local selon un critère de minimisation des erreurs d'estimation. Ce mouvement local est superposé au mouvement global de façon à obtenir un mouvement exécutable.

3. Estimation récursive des segments 3D.
   Connaissant le mouvement de la caméra, il est possible de calculer directement les segments 3D à partir des segment 2D dynamiques selon une méthode générale que nous avons étudié. Cette méthode s'applique aussi à la vision polynoculaire.

4. Identification des polygones 3D.
   On essaye d'extraire les polygones 3D interprétables à partir d'un ensemble des segments 3D estimés. Ceci permet de reconnaître et de localiser les objets polyédriques.

La figure 1 montre schématiquement notre système de vision dans le cadre de la reconstruction d'une scène 3D polyédrique. Nous voulons que chaque niveau de traitement dans notre système de vision donne un resultat robuste. Ceci est le point clef pour assurer la meilleure performance d'un système de vision.

1.2 Description Du Problème

L'extraction de l'information concernant la structure des formes presentées dans une image est un problème très délicat. Bien qu'il ait été étudié depuis longtemps, il n'y a pas de méthode qui soit en même temps robuste, rapide et générale. Etant donnée une image contour, les primitives le plus intéressant sont : (a) les figures et (b) les chaînes. La figure 2 montre la décomposition d'une image contour en de primitives significatifs.

Nous présentons ici un algorithme de chaînage des contours qui traite ce problème en deux étapes :

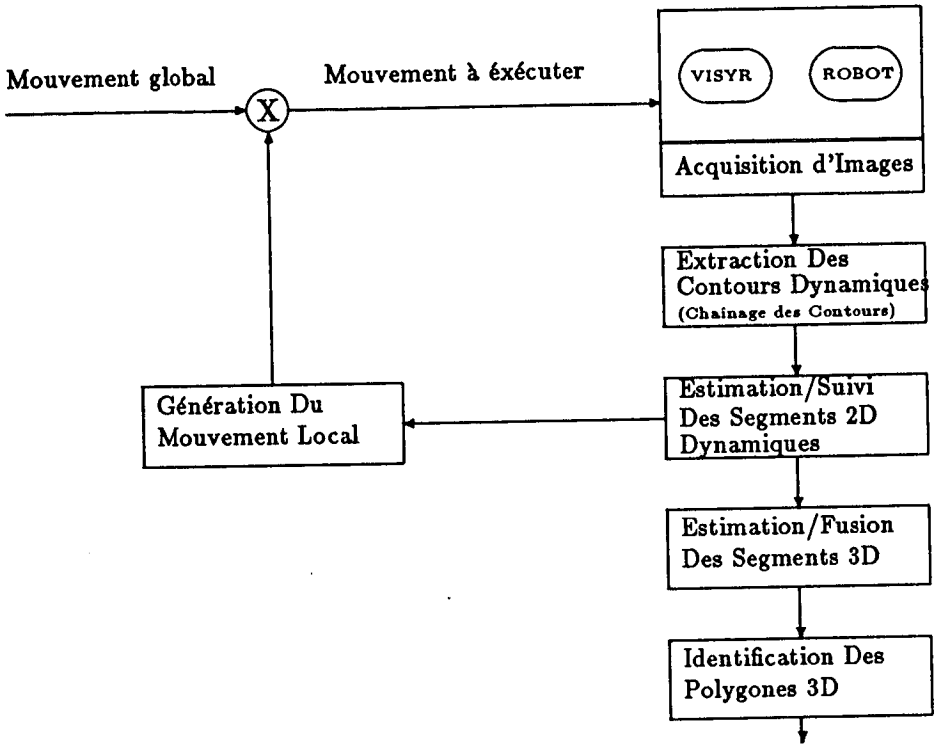


Figure 1 : Un Système pour la Reconstruction d'une Scène 3D Polyédrique

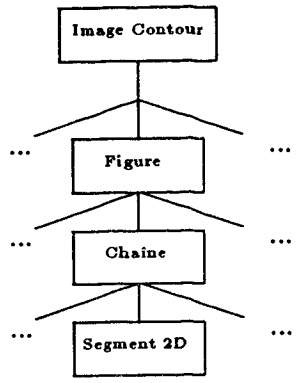


Figure 2 : Arbre De Primitives Multiniveaux Dans Une Image Contour

1. L'extraction des chaînes.

2. L'approximation polygonale des chaînes.

Dans notre approche, nous introduisons d'abord une technique de réorganisation des données des pixels contours. Cette opération permet d'accélérer considérablement le chaînage des contours. Nous travaillons sur les couples contours au lieu des pixels contour. Un couple contour est défini comme étant un sous-ensemble de pixels contour se situant consécutivement sur une même ligne de l'image. Une image de contours est donc transformée en un tableau des couples contours. Le chaînage des contours est devenu le chaînage des couples contours. Ensuite nous définissons une zone de voisinage causale pour chaque couple contour. Le choix de cette définition provient du fait qu'une zone de voisinage causale possède une propriété intéressante qui assure la robustesse de notre algorithme. De plus, la taille de zone de voisinage n'est pas fixée, ce qui rend l'algorithme de chaînage très souple, adapté à toutes sortes d'images contours. On distingue quatre types des chaînes dans la zone de voisinage d'un couple contour. Selon la nature des chaînes présentes, l'opération de chaînage de notre algorithme consiste à faire : (a) la création des chaînes, (b) le rallongement des chaînes et (c) la

fermeture des chaînes. L'autre point important est que les chaînes obtenues par notre approche sont d'une grande simplicité. Tous les pixels contour appartenants à une même chaîne sont alignés séquentiellement en ordre croissant par rapport à X et à Y dans un tableau . Ceci permet d'effectuer rapidement et directement la fraction de chaînes en détectant la courbure maximale de celle-ci. Toutes les chaînes fractionnées sont approximées par des segments droits à l'aide de la methode de "Moindre Carré".

Bien que notre algorithme s'applique originalement au chaînage des contours, il peut s'appliquer aussi au cas où on utilise la transformée de HOUGH pour interpoler l'ensemble des points 2D par des courbes.

En résumé, nous allons traiter dans ce rapport le problème suivant :

Etant donné une image des contours issue d'une scène polyèdrique, identifier tous les segments droits formés par les pixels contours et estimer les paramètres de ces segments droits.

Notre approche possède une grande souplesse et simplicité par rapport à la méthode présentée dans [8]. La derniere impose une contrainte sur l'épaisseur des contours à chaîner et gère un automate assez lourd.

2 Transformation de la Structure de Données de l'Image Contour

La structure de données de l'image contour est en principe organisée sous forme d'une matrice M de dimension $n \times m$. En vue de reduire la taille de mémoire de stockage et plus particulierement le temps de traitement pour le chaînage, on reorganise la matrice $M_{n \times m}$ de façon à obtenir deux tableaux :

1. le tableau PIXCONT de dimension $nbpixcont \times 1$ ("nbpixcont" est le nombre des pixels contour dans l'image).

2. le tableau FDR (Fonction De Répartition des pixels contour sur les lignes d'image) de dimension $n \times 1$ ("n" est le nombre de lignes de l'image).

En balayant une seule fois sur l'image contour, de gauche à droite et ligne par ligne, on mémorise la coordonnée x des pixels contour dans le tableau PIXCONT et le nombre des pixels contour cumulés des lignes déjà balayées dans le tableau FDR. De cette manière, on conserve les trois éléments d'information importants dans ces deux tableaux :

1. Le nombre de pixels contour sur chaque ligne d'image i :

$$nbpixcont(i) = FDR(i) - FDR(i-1).$$

2. La zone de stockage des pixels contour d'une même ligne i :

$$\{ PIXCONT[FDR(i-1)], PIXCONT[FDR(i)] \}$$

3. Les coordonnées de chaque pixel contour j :

$$\begin{cases} pixel\_x_j = PIXCONT[j] \\ pixel\_y_j = i \end{cases} \quad FDR(i-1) \leq j \leq FDR(i).$$

Cette façon de réorganiser les données des pixels contours est efficace dans le cas où les contours sont d'une épaisseur un pixel après raffinage et s'il y a peu de contours horizontaux dans l'image. Mais les contours à chaîner ne sont pas toujours d'une épaisseur un pixel. Dans ce cas, on a interêt à introduire l'idée de "couple contour" qui est défini comme un sous-ensemble de pixels contour alignés consécutivement sur une même ligne. La position d'un couple contour est déterminée par trois paramètres :

- couple_xg : la coordonnée x de l'extrémité à gauche du couple contour.
- couple_xd : la coordonnée x de l'extrémité à droite du couple contour.
- couple_y : la coordonnée y du couple contour.

Au cours du balayage sur l'image contour $M_{n \times m}$, on va générer de la même façon deux tableaux :

1. le tableau MCOUPLE de dimension $nbcouple \times 2$ (nbcouple désigne le nombre des couples contour dans l'image contour).

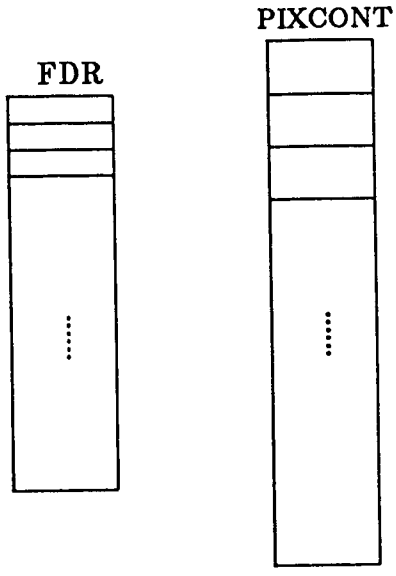


Figure 3 : Tableau de FDR et PIXCONT

2. le tableau FDR (Fonction De Répartition des couples contour sur les lignes d'image) de dimension $n \times 1$.

Ces deux tableaux conservent aussi les trois éléments d'information importants :

1. Le nombre des couples contour sur chaque ligne d'image i :

$$nbcouple(i) = FDR(i) - FDR(i-1).$$

2. La zone de stockage des couples contour d'une même ligne i :

$$\{MCOUPLE[FDR(i-1)], MCOUPLE[FDR(i)]\}$$

3. Les coordonnées de chaque couple contour j :

$$\begin{cases} couple\_xg_j & = & MCOUPLE[j].xg \\ couple\_xd_j & = & MCOUPLE[j].xd \\ couple\_y_j & = & i \end{cases} \quad FDR(i-1) \leq j \leq FDR(i).$$

Il faut souligner que l'opération de transformation de structure des données nécessite un temps d'éxecution négligable. Par contre , il permet le chaînage des contours rapide, effectué en une seule passe sur le tableau FDR.

3 Chaînage des Couples Contours

On désire effectuer le chaînage de tous les couples contour en une seule passe sur le tableau FDR et garder l'information sur les jonctions rencontrées. Pour cela on definit une zone de voisinage ZV pour chaque couple contour (xg_0, xd_0, y_0) qui est :

$$\{ [\, xg_0 - \triangle x,\, xd_0 \,[\parallel y = y_0 \}$$
$$\{ [\, xg_0 - \triangle x,\, xd_0 + \triangle x] \parallel y \in [\, y_0 - \triangle y,\, y_0 \,[\}$$

où $(2 \bullet \triangle x, \triangle y)$ sont les dimensions de ZV.

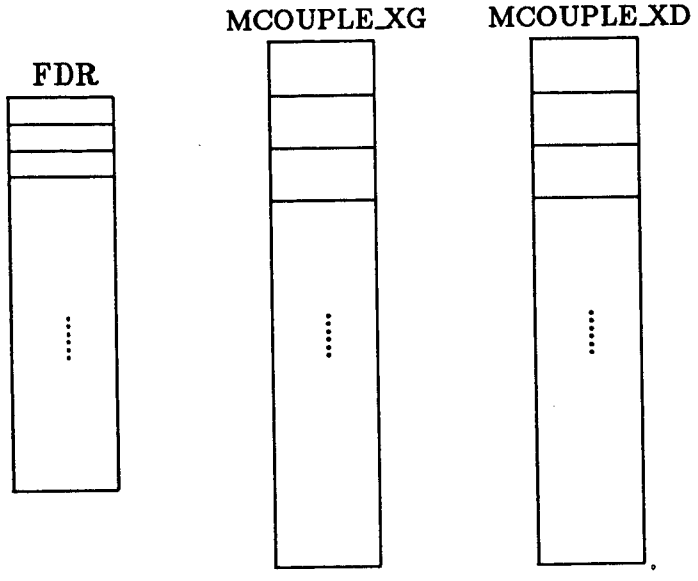


Figure 4 : Tableau de FDR et MCOUPLE

Ensuite on balaye ligne par ligne le tableau FDR, en déplaçant le pavé de ZV et en ne traitant que les couples contour significatifs. Au cours de ce parcours, on doit gérer trois actions: la création des chaînes, le rallongement des chaînes et la fermeture (fusion) des chaînes. La création d'une chaîne consiste à remplir une structure des données de chaîne qui contient trois éléments d'information :

- Les couples contour y appartenants.
- Deux extrémités de la chaîne : le couple tête et le couple queue.
- Une relation d'équivalence qui exprime la relation de jonction avec d'autres chaînes.

L'autre action très importante est la gestion de jonction des chaînes qui permet d'extraire l'information sur la connexité des chaînes. Pour ce fait, nous avons défini une relation d'équivalence pour chaque chaîne créée. Si une chaîne i est connectée à une autre chaîne j, on a :

$$\begin{cases} EQUIV(max\{i,j\}) &= min\{i,j\}. \\ EQUIV(min\{i,j\}) &= min\{i,j\}. \end{cases}$$

Par contre, si une chaîne i est indépendante topologiquement des autres chaînes, on a :

$$EQUIV(i) = i.$$

3.1 Situation des Chaînes dans la ZV d'un Couple Contour

Après avoir défini la zone de voisinage pour un couple contour, on peut identifier un sous-ensemble de chaînes qui intersectent ce pavé de ZV. Parmi ce sous-ensemble de chaînes, on peut distinguer deux catégories de chaînes :

1. les chaînes ouvertes dans lesquelles on s'autorise eventuellement à ajouter de nouveaux couples contour.
2. les chaînes fermées dans lesquelles on s'interdit d'ajouter de nouveaux couples contour.

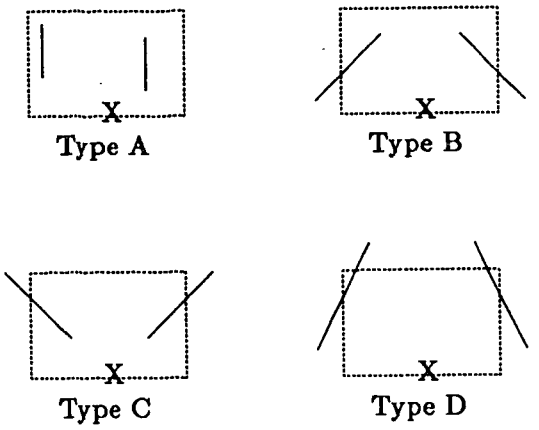


Figure 5 : Quatre Types des Chaînes Ouvertes dans Une ZV

De plus selon la nature de l'intersection avec ZV, les chaînes ouvertes peuvent être classées en quatre types (voir la figure 5) :

1. type A : les chaînes qui sont entièrement à l'intérieur de ZV.
2. type B : les chaînes qui n'ont que leur tête à l'intérieur de ZV.
3. type C : les chaînes qui n'ont que leur queue à l'intérieur de ZV.
4. type D : les chaînes qui n'ont que leur corps à l'intérieur de ZV.

Les chaînes du type D n'impose pas d'influence sur l'action de chaînage pour un couple contour donné, nous les considérons alors comme des chaînes fermées. Les chaînes de type B n'interviennent que pour la gestion de jonction des chaînes; aussi on les considère comme des chaînes demi-fermées. En fonction du type de chaînes presentes dans le pavé de ZV d'un couple contour, on va effectuer des traitements différents.

Nous soulignons ici une propiété important du pavé de ZV :

Si les tailles des couples contours sont localement homogènes, par "causalité" de la zone de voisinage définie, il ne peut pas y

avoir plus de deux chaînes de type A et C dans le pavé de ZV pour un couple contour donné.

La "causalité" d'une ZV représente simplement le fait que toutes les chaînes sont déjà traitées (par exemple: opération de fermeture des chaînes). On peut vérifier très facilement cette propriété :

Supposons qu'il y ait trois chaînes du type A et C dans le pavé de ZV d'un couple contour et que (x_1, y_1), (x_2, y_2) et (x_3, y_3) soient trois points quelconques (à l'intérieur de ZV) de ces trois chaînes, il est évident,en raison de la causalité, qu'on ne peut pas satisfaire en même temps les relations :

$$\begin{cases} max\{|x_1 - x_2|,\ |x_2 - x_3|,\ |x_3 - x_1|\} & \leq & 2 \bullet \triangle x + 2 \bullet taille. \\ min\{|x_1 - x_2|,\ |x_2 - x_3|,\ |x_3 - x_1|\} & > & \triangle x + taille. \end{cases}$$

où *taille* est la dimension d'un couple contour.

Cela signifie qu'on ne peut pas avoir plus de deux chaînes de type A et C dans la ZV d'un couple contour.

C'est grâce à cette propriété que notre algorithme de chaînage possède une très grande simplicité.

3.2 Création des Chaînes

Lorsqu'il n'y a pas de chaîne de type A et C parmi les chaînes presentes dans le pavé de ZV du couple contour courant, on engage l'action de création de chaîne. Dans le cas contraire, ce sont les actions de rallongement de chaîne et de fermeture de chaîne qu'on doit entreprendre.

Quand on crée une chaîne, on prend le couple contour courant $pc(x_g, x_d, y)$ comme couple tête et couple queue de la nouvelle chaîne et on établit une relation d'équivalence, soit avec elle-même, soit avec d'autres chaînes.

D'après l'opération de gestion de relation d'équivalence, on considére trois cas :

1. cas 1 : une nouvelle chaîne est créée lorsqu'il n'y a pas de chaîne ouverte parmi les chaînes presentees dans le pavé de ZV du couple contour courant. Dans ce cas , cette nouvelle chaîne a la relation d'équivalence avec elle-même :

$$EQUIV(new) = new.$$

où *new* désigne la nouvelle chaîne.

2. cas 2 : une nouvelle chaîne est créée lorsqu'il n'y a que des chaînes ouvertes de type B et D parmi les chaînes presentes dans le pavé de ZV du couple contour courant. Dans ce cas, on établit la relation d'équivalence entre la nouvelle chaîne et les chaînes presentes du type B :

$$\begin{cases} EQUIV(i) & = & min\{EQUIV(i) \mid \forall i \in B\}. \\ EQUIV(new) & = & min\{EQUIV(i) \mid \forall i \in B\}. \\ & & \forall i \in B. \end{cases}$$

où *new* désigne la nouvelle chaîne.

3. cas 3 : une nouvelle chaîne est créée lorsqu'il n'y a que des chaînes ouvertes de type D parmi les chaînes presentes dans le pavé de ZV du couple contour courant. Dans ce cas, la nouvelle chaîne a la relation d'équivalence avec elle-même.

$$EQUIV(new) = new.$$

où *new* désigne la nouvelle chaîne. Pour cette raison, on considère les chaînes de type D comme des chaînes fermées.

3.3 Rallongement des Chaînes

Lorsqu'il n'y a qu'une seule chaîne ouverte de type A ou C parmi les chaînes qui sont presentes dans le pavé de ZV du couple contour courant, on prolonge la chaîne vers ce couple contour. S'il y a une autre chaîne ouverte de type B presentee elle-aussi dans le pavé de ZV, on établit simplement la relation d'équivalence entre ces deux chaînes :

$$\begin{cases} EQUIV(i) & = & min\{EQUIV(i), EQUIV(j)\}. \\ EQUIV(j) & = & min\{EQUIV(i), EQUIV(j)\}. \\ & & i \in A \text{ ou } C,\ j \in B. \end{cases}$$

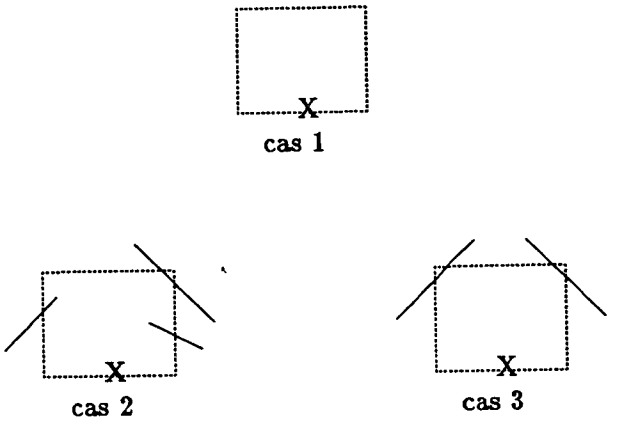


Figure 6 : Création des Chaînes

3.4 Fermeture des Chaînes

Ce traitement intervient lors de trois cas possibles :

1. cas 1 : il n'y a que deux chaînes ouvertes de type A parmi les chaînes presentes dans le pavé de ZV du couple contour courant. Dans ce cas, on établit d'abord la relation d'équivalence entre ces deux chaînes :

$$\begin{cases} EQUIV(i) & = & min\{EQUIV(i), EQUIV(j)\}. \\ EQUIV(j) & = & min\{EQUIV(i), EQUIV(j)\}. \\ & & (i,j) \in A. \end{cases}$$

   Ensuite on prolonge la chaîne $min\{EQUIV(i), EQUIV(j)\}$ vers le couple contour courant et on ferme la chaîne $max\{EQUIV(i), EQUIV(j)\}$.

2. cas 2 : il n'y a que deux chaînes ouvertes de type C parmi les chaînes presentes dans le pavé de ZV du couple contour courant. Dans ce cas, on établit d'abord la relation d'équivalence entre ces deux chaînes :

$$\begin{cases} EQUIV(i) & = & min\{EQUIV(i), EQUIV(j)\}. \\ EQUIV(j) & = & min\{EQUIV(i), EQUIV(j)\}. \\ & & (i,j) \in C. \end{cases}$$

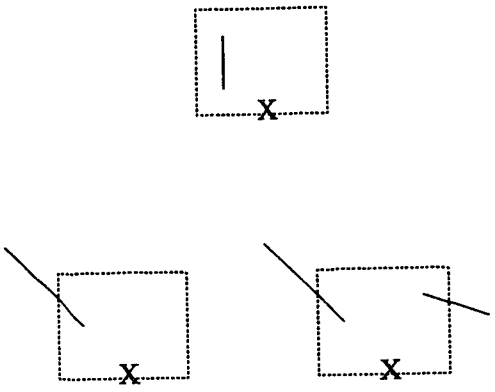


Figure 7 : Rallongement des Chaînes

Ensuite on prolonge une chaîne vers le couple contour courant et on ferme ces deux chaînes.

3. cas 3 : il y a une chaîne ouverte de type A et une autre de type C parmi les chaînes presentes dans le pavé de ZV du couple contour courant. Dans ce cas, on établit d'abord la relation d'équivalence entre ces deux chaînes :

$$\begin{cases} EQUIV(i) = EQUIV(i). \\ EQUIV(j) = EQUIV(i). \\ \qquad\qquad i \in A,\ j \in C \end{cases}$$

Ensuite on prolonge la chaîne i vers le couple contour courant et on ferme la chaîne j.

3.5 Cas Particulier

Nous avons supposé implicitement que les tailles des couples contours étaient localement homogènes . Dans le cas où une chaîne horizontale a été balayée, on peut très bien avoir un couple contour assez grand. Ceci constitue le seul

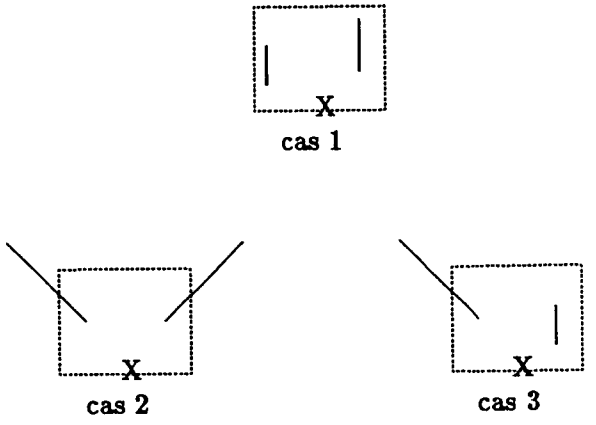


Figure 8 : Fermeture des Chaînes

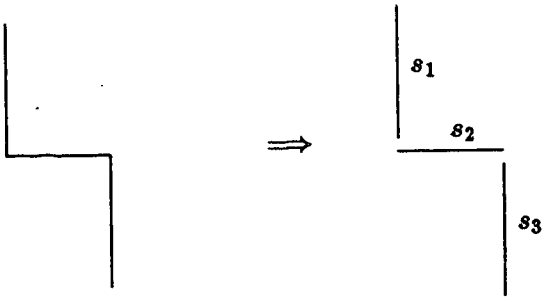
cas particulier de notre algorithme de chaînage.

Dans ce cas particulier, on peut détecter plusieurs chaînes ouvertes de type A et C dans un pavé de ZV. Le traitement de chaînage doit donc être différent. En fait, il est facile de traiter ce cas particulier :

Etant donné un couple contour, s'il a plus de deux chaînes de type A et C dans le pavé de ZV, on ferme toutes les chaînes de type A et C après avoir établi la relation d'équivalence entre elles, et on va créer une nouvelle chaîne à partir de ce couple contour.

4 Approximation Polygonale des Chaînes

Notre algorithme de chaînage assure une simplicité totale des chaînes créées, cela signifie que chaque chaîne n'a aucune branche secondaire (bifurcation). En plus les couples contour d'une même chaîne sont positionnés dans le tableau MCOUPLE de façon sequentielle en ordre croissante par rapport à X et à Y. Cela permet de réaliser directement la fraction des chaînes en morceaux de segments droits.



chaîne segments droits

Figure 9 : Approximation Polygonale d'une Chaîne

Etant donné une chaîne, on prédit d'abord un segment droit en utilisant le couple tête et le couple queue de cette chaîne. Ensuite, on teste la nécessité de l'opération de fraction. S'il y a au moins un couple contour qui s'éloigne de ce segment prédit à un seuil donné, on découpe cette chaîne au couple contour qui s'éloigne le plus et on obtient deux sous-chaînes. Cette opération va continuer jusqu'à ce qu'il n'y ait plus de fraction à faire. L'ensemble des segments droits prédits qui n'ont pas causés la fraction des chaînes constitue donc une approximation polygonale de la chaîne initiale. La figure 9 illustre un exemple de l'approximation d'une chaîne.

Dans le cas de la figure 10, notre approche de l'approximation polygonale des chaînes va échouer. Heureusement notre algorithme de chaînage nous garantit de l'impossibilité produire ces deux types de chaîne.

5 Estimation des Paramètres des Segments 2D

Après l'opération de fraction des chaînes, l'ensemble des couples contour appartenant à une chaîne est partitionnés en des sous-ensembles. Chaque élément de ce sous-ensemble forme un segment droit dont on peut estimer

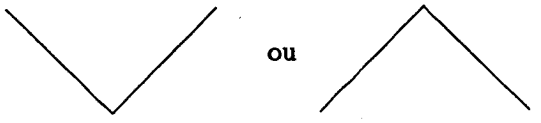


Figure 10 : Chaines non Approximables

les paramètres.

Un segment droit 2D est modélisé en principe dans un espace 2D par le vecteur normal perpendiculaire au segment et la distance à l'origine de ce segment.

Soit $\vec{n} = (a, b)$ le vecteur normal du segment S et c sa distance à l'origine, $f = (a, b, c)$ constitue donc le vecteur paramètre du segment S et son équation s'écrit comme :

$$a \bullet x + b \bullet y - c = 0.$$

où (x, y) est un point sur le segment S.

Etant donné un sous-ensemble de m couples contour $\{(x_i, y_i);\ i = 1, m\}$ où (x_i, y_i) est le point central du couple contour i, le meilleur estimé du vecteur f sera réalisé par optimisation d'un critère tel que :

$$J_f = \min_f \{ J(f) = \sum_{i=1}^{m} (a \bullet x_i + b \bullet y_i - c)^2 \}$$

avec $a^2 + b^2 = 1$.

La minimisation du critère J_f revient à trouver le vecteur pamamètre f satisfaisant les équations suivantes :

$$\frac{\partial J(f)}{\partial f} = 0.$$

C'est-à-dire :

$$\begin{cases} p \bullet a + q \bullet b - s \bullet c &= 0 \\ q \bullet a + w \bullet b - t \bullet c &= 0 \\ s \bullet a + t \bullet b - m \bullet c &= 0 \\ a^2 + b^2 &= 1 \end{cases} \quad (1)$$

où :

$$p = \sum_{i=1}^{m} x_i^2. \quad q = \sum_{i=1}^{m} (x_i\, y_i). \quad w = \sum_{i=1}^{m} y_i^2.$$

$$s = \sum_{i=1}^{m} x_i. \quad t = \sum_{i=1}^{m} y_i.$$

Après quelques développements mathématiques, nous trouvons la solution qui est :

- Cas 1 : segment droit horizontal.

$$f = \begin{pmatrix} 0 \\ 1 \\ t \end{pmatrix}.$$

- Cas 2 : segment droit vertical.

$$f = \begin{pmatrix} 1 \\ 0 \\ s \end{pmatrix}.$$

- Cas 3 : segment droit en orientation quelconque.

$$f = \begin{pmatrix} \frac{w \bullet s - q \bullet t}{L} \\ \frac{p \bullet t - q \bullet s}{L} \\ \frac{p \bullet w - q^2}{L} \end{pmatrix} ; \; avec \; L = \sqrt{(w \bullet s - q \bullet t)^2 + (p \bullet t - q \bullet s)^2}.$$

6 Résultats Expérimentaux

Au cours du développement de notre projet de recherche, nous nous avons mis au point un système expérimental de vision dynamique intitulé VIDYR (Système de Vision Dynamique pour la Robotique). Les Modules principaux de VIDYR sont presentés à la figure 11 où on trouve :

1. Le noyau de VIDYR.
   Sa fonction est de gérer la communication des données entre les modules et éventuellement l'utilisateur. Les données utilisées dans VIDYR sont généralement : (a) les images, (b) les trajectoires, (c) les paramètres et (d) les résultats des traitements. Pour faciliter cette tâche, nous avons défini un jeu de structures de données adapté aux différents niveaux des primitives de la vision. Le noyau de VIDYR gère aussi l'intégration d'une nouvelle application.

2. L'interface graphique.
   Le système VIDYR est développé sur la station SUN sous UNIX. Nous utilisons les utilitaires de SUNVIEW et de SUNCORE pour créer un environnement graphique destiné à l'interfaçage graphique entre les modules de VIDYR et l'utilisateur, et aussi à l'affichage des résultats de traitements (par exemple: l'image des contours, l'image des segments 2D/3D, etc).

3. VISYR.
   Nous avons à notre disposition un simulateur de vision en mouvement intitulé VISYR (Vision Synthétique pour la Robotique) développé à l'IRISA. Ce simulateur a été conçu au sein de l'algorithme de "Lancé de Rayons" qui a pour rôle de synthétiser l'image d'une scène spécifiée par le langage LGRC. L'intégration de VISYR dans VIDYR permet de tester les algorithmes de vision avec des données synthétiques. Ceci est très important pour analyser rapidement la validité de certains algorithmes.

4. Le Robot Mobile.
   Nous avons réalisé un banc expérimental de vision en mouvement. Ce banc est constitué d'un robot manipulateur AID sur le poignet duquel est installée une caméra CCD calibrée de marque MICAM. Le manipulateur est sous contrôle d'un processeur de traitement d'images EDIXIA. Ce dernier a une liaison directement avec la station SUN. Le système VIDYR peut commander facilement le robot en vue d'acquérir une séquence d'images de résolution 256×256 pour une tâche robotique donnée.

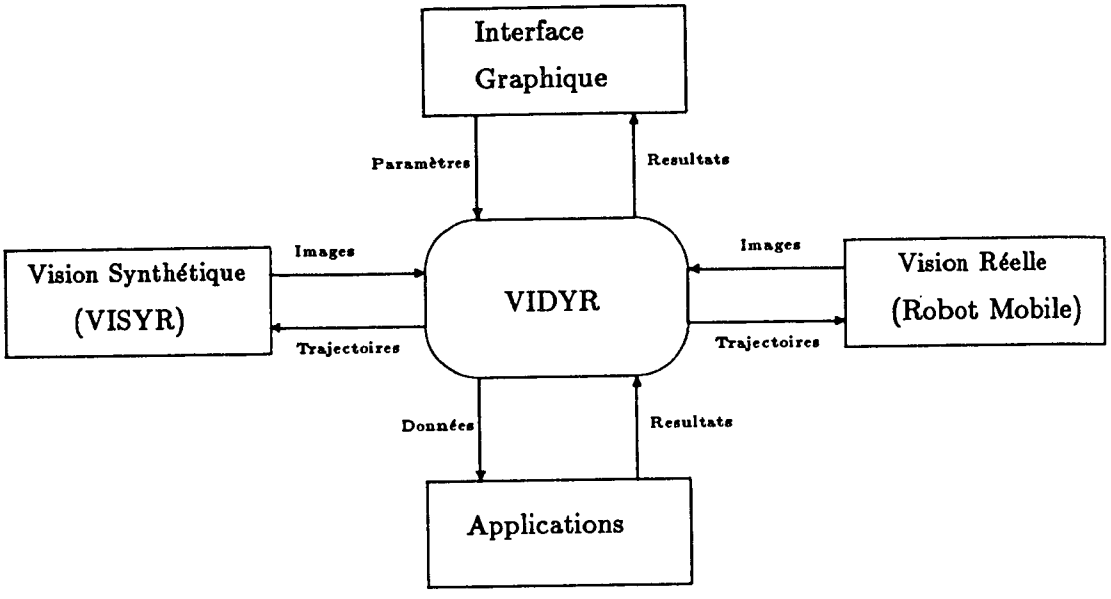


Figure 11 : Description De VIDYR

5. L'application.
   Tous les algorithmes d'applications sont développés au sein de VIDYR. La programmation est basée sur le langage C. En fait, VIDYR représente un outils de recherche pour le développement des algorithmes de vision.

Plus précisément, nous avons utilisé VIDYR pour réaliser le travail concernant la reconstruction d'une scène 3D polyédrique. La figure 12 montre les modules de traitements principaux dans ce contexte.

En tant que module de traitement, l'algorithme de chaînage des contours décrit dans ce rapport a été réalisé au sein de VIDYR. Nous avons testé notre algorithme avec d'images réelles. L'expérimentation nous indique que cet algorithme est efficace. Nous montrons à la fin de ce rapport une série

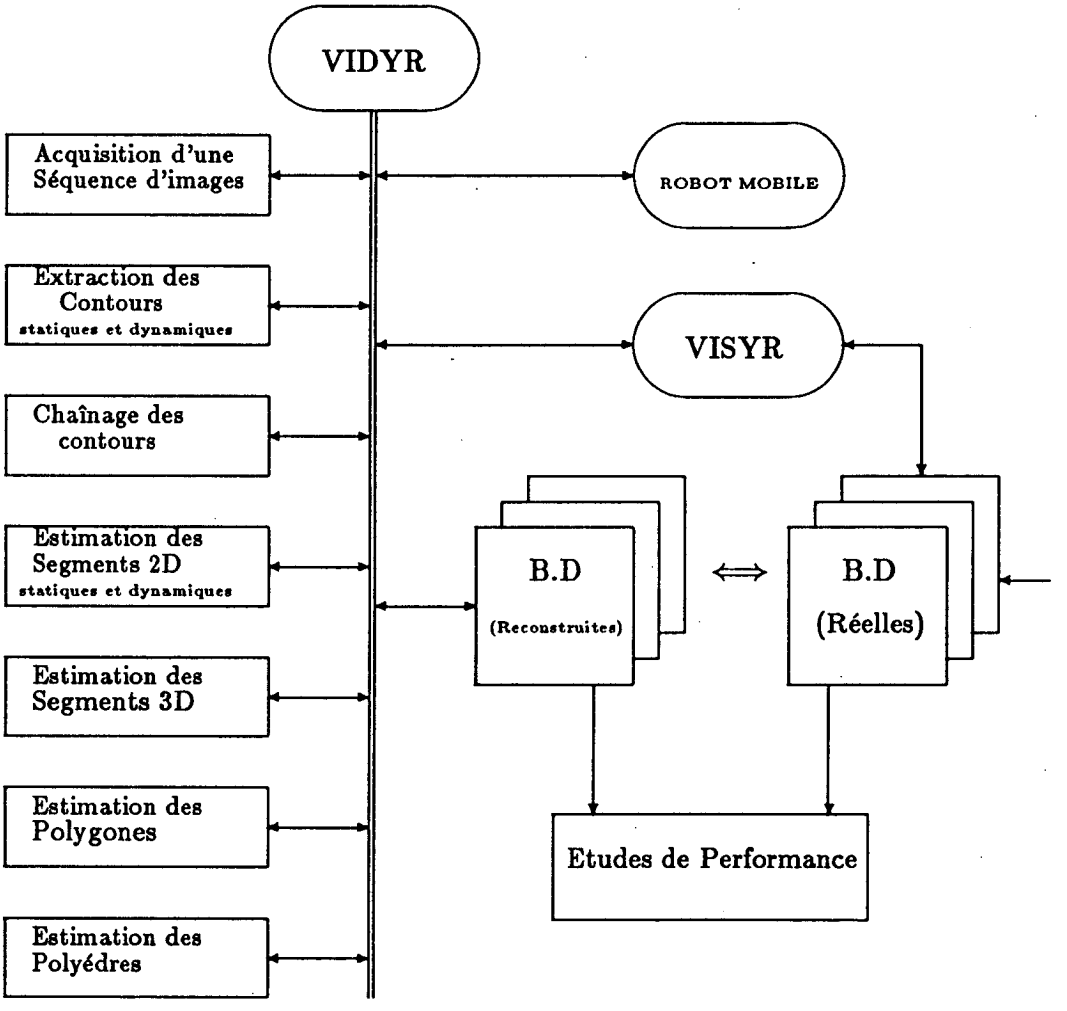


Figure 12 : Application de VIDYR pour la Reconstruction d'une Scène 3D Polyèdrique

de resultats.

7 Conclusion et Discussion

Nous avons résolu le problème de l'estimation des segments 2D dans une image contour d'une façon efficace. Au niveau du traitement de chaînage des contours. Nous avons presenté une technique de réorganisation de la structure des données des contours. Cette opération rend notre algorithme plus rapide et plus efficace. Ensuite nous avons défini une zone de voisinage causal de taille variée. Ceci garantit la robustesse de notre algorithme. Notre algorithme assure en même temps une bonne estimation des segments 2D grâce à la technique de "Moindre Carrée"

Afin de montrer la performance de notre algorithme, nous soulignons simplement trois autres fonctions :

1. Interpolation des points par des courbes sur un plan 2D.
   La résolution de ce problème fait souvent appel à la technique de transformée de Hough qui est une technique peu efficace et assez coûteuse en temps de calcul. Si, dans notre algorithme, on employe une zone de voisinage de taille assez grande, ce problème peut être résolu effectivement en deux étapes: (a) L'identification des points appartenants aux mêmes courbes et (b) L'estimation des paramètres des courbes.

2. Extraction des segments droits horizontaux.
   La dimension d'une zone de voisinage est déterminée par deux paramètres $(\triangle x, \triangle y)$. $\triangle x$ spécifie la largeur dans la direction de X, $\triangle y$ la hauteur dans la direction de Y. On n'a imposé aucune contrainte sur ces deux paramètres. Si pour une application donnée on a :

   $$\begin{cases} \triangle x > 0 \\ \triangle y < 0 \end{cases}$$

   les resultats obtenus seront un ensemble des segments droits horizontaux.

3. Extraction des segments droits verticaux.
   De la même manière, si pour une application donnée on a :

$$\begin{cases} \Delta x < 0 \\ \Delta y > 0 \end{cases}$$

les resultats obtenus seront un ensemble des segments droits verticaux.

Bibliographie

[1] BALLARD.D.H : " Generalizing The HOUGH Transform to Detect Arbitrary Shapes ", *Pattern Recognition Vol.13,No.2* , pp.111-122, 1981.

[2] BOISSONNAT.J.D, O.D.FAUGERAS and E.LEBRAS : "Représentation des donnée stéréo par la triangulation de Delaunay", *Congrès AFCET-RFIA*, Antibes/FRANCE, Novembre, 1987.(in french)

[3] BORGNE.M.L : " QUATERNIONS ET CONTROLE SUR L'ESPACE DES ROTATIONS ", *Rapport Intern de l'IRISA*, No.377, Octobre, 1987.

[4] BOUTHEMY.P : " Estimation Of Edge Motion Based on Local Modelling ", *SPIE Vol.595, Computer Vision For Robotis* pp.162 - 169 , Cannes, December, 1985.

[5] DEURICHE.R : "Using Canny's Criteria to Derive a Recursively Implemented Optimal Edge Detector", *International Journal of Computer Vision*, P167-187,1987.

[6] ESPIAU.B and RIVES.P : " Closed-Loop Recursive Estimation Of 3D Features For a Mobile Vision System ", *IEEE, International Conference on Robotics and Automation* March 30 - Aril 3 , Raleigh, 1987.

[7] FREEMAN.H and DAVIS.L.S : "A corner-finding algorithm for chain-codes curves", *IEEE Trans. Comput. 26(3)*, P297-303, 1977.

[8] GIRAUDON.G : " Chaînage Efficace de Contour ", *Rapport de Recherche No.605*, centre SOPHIA ANTIPOLIS de l'INRIA, 1987.

[9] HEGRON.G, ARNALDI.B and PRIOL.T : "VISYR : A simulation tools of synthetic vision for robotics", *Pro. of MARI87, Vo.II* , Paris, mai 1987.

[10] KUROZUMI.Y and DAVIS.W.A : "Polygonal Approximation by the Minimax Method", *Computer Graphics and Images Processing 19* , 248-264, 1982.

[11] LEU.J.G and CHEN.L : "Polygonal approximation of 2D shapes through boundary merging", *Pattern Recognition letters 7* , 231-238, 1988.

[12] LOWE.D.G : "Three-Dimensional Object Recognition from Single Two-dimensional Images", *Artificial Intelligence 31* , 355 - 395, 1987.

[13] PAVLIDIS.T : "SURVEY : A Review of Algorithms for Shap Analysis", *Computer Graphics and Images Processing 7*, 243-258, 1978.

[14] RIVES.P : "Dynamic Vision: Theoretic Capability and Practical Problems" *Nato Workshop on Kinematic and Dynamic Issues in Sensor Based Control*, IL CIOCCO, TUSCANY Italy, October 25-31, 1987.

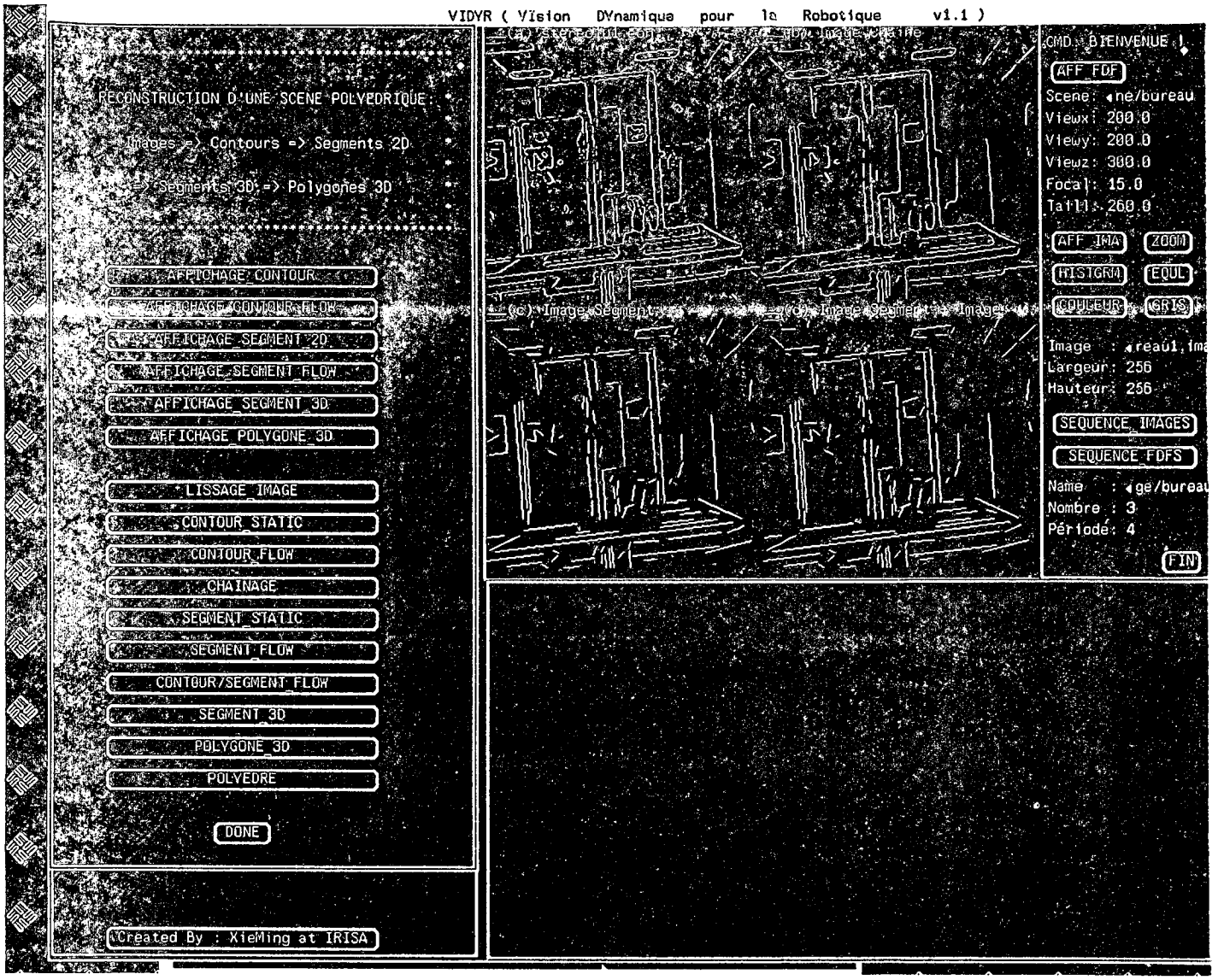


Figure 12 : L'image (a) est l'image contour d'entrée. L'image (b) est l'ensemble des chaînes obtenues. L'images (c) est les segments 2D estimés. L'image (d) est les segments estimés superposés avec l'image originelle.

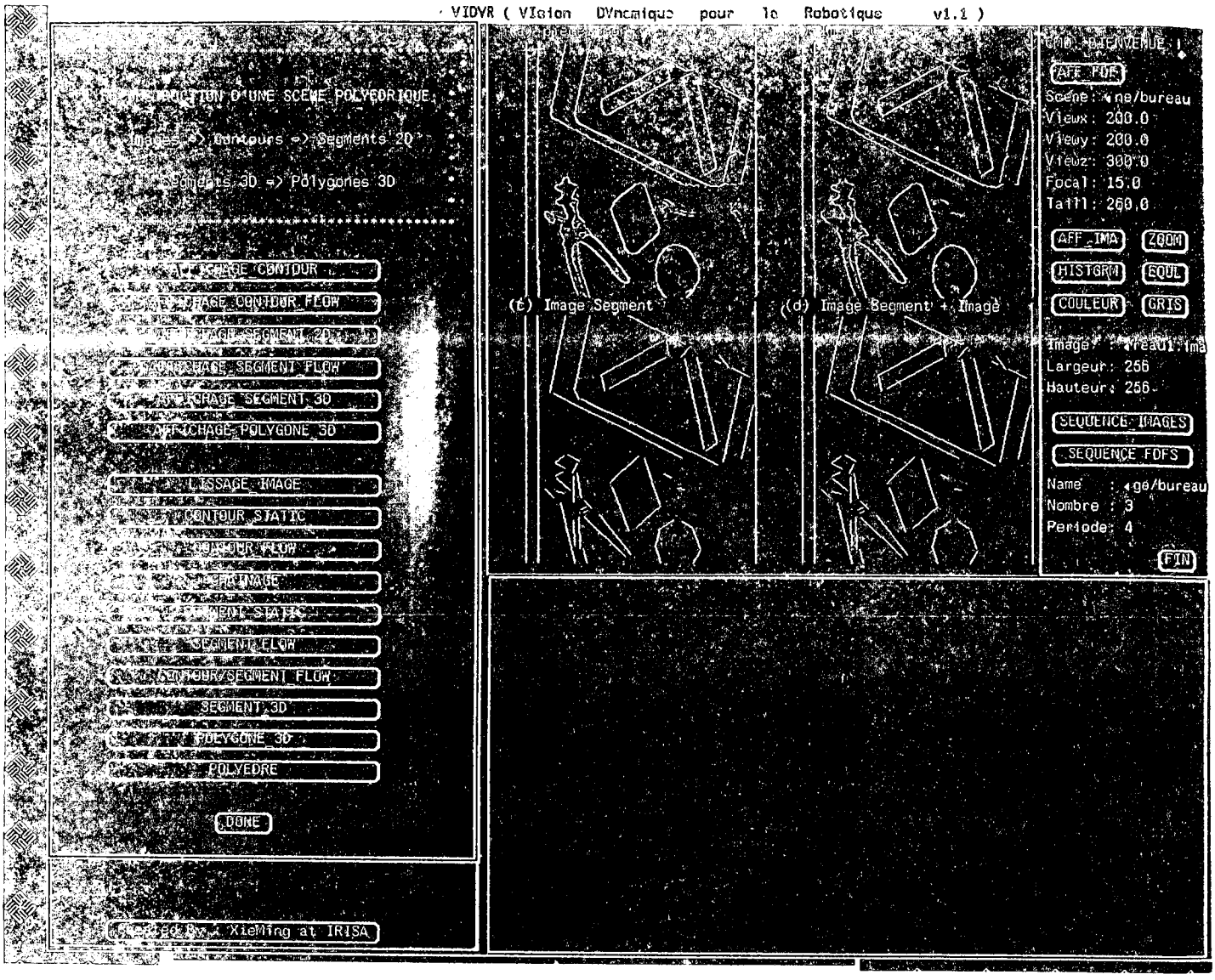


Figure 13 : L'image (a) est l'image contour d'entrée. L'image (b) est l'ensemble des chaînes obtenues. L'images (c) est les segments 2D estimés. L'image (d) est les segments estimés superposés avec l'image originelle.

LISTE DES DERNIERES PUBLICATIONS INTERNES

PI 425 **A MULTISCALE STOCHASTIC APPROACH FOR 2D INVERSE PROBLEM IN CONDUCTIVITY**
Kenneth C. CHOU, Alan S. WILLSKY
40 Pages, Septembre 1988.

PI 426 **CLASSIFICATION EN PRESENCE DE VARIABLES PREORDONNANCES TAXONOMIQUES A CHOIX MULTIPLE. APPLICATION A LA STRUCTURATION DES PHLEBOTOMES DE LA GUYANE FRANCAISE**
Israël-César LERMAN
184 Pages, Septembre 1988.

PI 427 **IDENTIFICATION DES POLYGONES 3D A PARTIR DES SEGMENTS 3D**
Ming XIE, Patrick RIVES
46 Pages, Septembre 1988.

PI 428 **UN MECANISME DE RESOLUTION POUR LE SYSTEME Q**
Christophe BOUCHY
62 Pages, Septembre 1988.

PI 429 **ESTIMATION DES SEGMENTS 2D : UN ALGORITHME ROBUSTE**
Ming XIE, Patrick RIVES
34 Pages, Septembre 1988.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique