Localisation et modélisation tridimensionnelles par approximations successives du modèle perspectif de caméra

Stéphane Christy

THÈSE

présentée par

Stéphane Christy

pour obtenir le grade de **docteur**

de l'**Institut National Polytechnique de Grenoble**

(Arrêté ministériel du 30 Mars 1992)

Spécialité : **Informatique**

Localisation et modélisation tridimensionnelles par approximations successives du modèle perspectif de caméra

---

Date de soutenance : 17 septembre 1998

Composition du jury :	Président :	**Jean-Marie Laborde**
	Rapporteurs :	**Daniel Dementhon**
		Michel Dhome
	Examinateurs :	**Pierre Comon**
		Radu Horaud
		Bruno Mazar
		Brigitte Plateau

Thèse préparée au sein du laboratoire gravir - imag - inria Rhône-Alpes
sous la direction de Radu Horaud

THÈSE

présentée par

Stéphane Christy

pour obtenir le grade de **docteur**

de l'**Institut National Polytechnique de Grenoble**

(Arrêté ministériel du 30 Mars 1992)

Spécialité : **Informatique**

Localisation et modélisation tridimensionnelles par approximations successives du modèle perspectif de caméra

---

Date de soutenance : 17 septembre 1998

Composition du jury :	Président :	**Jean-Marie Laborde**
	Rapporteurs :	**Daniel Dementhon**
		Michel Dhome
	Examinateurs :	**Pierre Comon**
		Radu Horaud
		Bruno Mazar
		Brigitte Plateau

Thèse préparée au sein du laboratoire gravir - imag - inria Rhône-Alpes
sous la direction de Radu Horaud

Résumé

Dans le cadre de cette thèse, nous proposons un algorithme générique permettant de résoudre le problème de calcul de pose et le problème de reconstruction avec un modèle perspectif de caméra.

Étant donnés une image et un modèle 3D de la scène (ou objet) visible dans l'image, le calcul de pose consiste à calculer la position et l'orientation de la caméra par rapport à la scène. Nous étudions successivement le cas de correspondances 2D/3D de points, et le cas des droites. La méthode proposée améliore de manière itérative la pose calculée avec un modèle affine de caméra (orthographique à l'échelle ou paraperspectif) pour converger, à la limite, vers une estimation de la pose calculée avec un modèle perspectif de caméra. Nous étudions les relations mathématiques et géométriques existant entre les modèles orthographique à l'échelle, paraperspectif et perspectif de caméra. Nous introduisons une façon simple de prendre en compte la contrainte d'orthogonalité associée à une matrice de rotation. Nous analysons la sensibilité de la méthode par rapport aux erreurs d'étalonnage de la caméra et nous définissons les conditions expérimentales optimales par rapport à un étalonnage imprécis. Nous étudions la convergence de la méthode sur la base de considérations numériques et expérimentales et nous testons son efficacité avec des données synthétiques et réelles.

Dans un second temps, nous étendons les algorithmes de calcul de pose précédents au problème de la reconstruction euclidienne avec un modèle perspectif de caméra, à partir d'une séquence d'images. La méthode proposée converge en quelques itérations, est efficace du point de vue calculatoire, et ne souffre pas de la nature non linéaire du problème traité. Comparativement à des méthodes telles que la factorisation ou les invariants affines, notre méthode résout le problème de l'ambiguïté de signe d'une façon très simple et fournit des résultats bien plus précis. Nous décrivons la nouvelle méthode en détail, et comparons la complexité de la méthode proposée avec une méthode de minimisation non linéaire.

Nous présentons ensuite une seconde approche du problème de reconstruction euclidienne en considérant un modèle affine de caméra non étalonnée montée sur le bras d'un robot. Nous montrons comment utiliser l'information euclidienne fournie par le déplacement du robot afin d'obtenir une reconstruction euclidienne, et expliquons comment obtenir l'étalonnage du modèle affine de caméra ainsi que l'étalonnage caméra-pince.

Afin de pouvoir utiliser en pratique ces algorithmes de reconstruction, nous présentons une méthode de poursuite de points caractéristiques sur une séquence monoculaire d'images, puis sur une séquence stéréoscopique. Nous proposons également une méthode pour obtenir une précision sous-pixellique des positions des points dans les images pour un faible coût calculatoire.

Mots clés : vision par ordinateur, reconstruction tridimensionnelle (euclidienne et affine), calcul de pose, mise en correspondance, corrélation, modèle perspectif de caméra, modèle affine de caméra, modèle orthographique à l'échelle, modèle paraperspectif, étalonnage d'une caméra.

Abstract

In this report, we propose a generic algorithm to compute objet pose and reconstruction, with a perspective camera model.

Given one image and a 3D model of the scene, object pose consists in recovering the position and the orientation of the camera with respect to the camera. We successively study the case of 2D to 3D point correspondences, and the case of line correspondences. The method consists in iteratively improving the pose computed with an affine camera model (weak perspective or paraperspective) to converge, at the limit, to the pose estimation computed with a perspective camera model. We analyze mathematical and geometric relationships between weak perspective, paraperspective and perspective camera models. We introduce a simple way to take into account the orthogonality constraint associated with the rotation matrix. We analyze the sensitivity to camera calibration errors and we define the optimal experimental setup with respect to imprecise camera calibration. We study its convergence based on numerical and experimental considerations, and we test its efficiency with both synthetic and real data.

In a second time, we extend the previous object pose algorithms for Euclidean reconstruction from a sequences of images, by using a perspective camera model. The proposed method converges in a few iterations, is computationally efficient, and does not suffer from the non linear nature of the problem. With respect to factorization and/or affine-invariant methods, this method solves for the sign (reversal) ambiguity in a very simple way and provides much more accurate reconstruction results. We give a detailed account of the method and compare its complexity with respect to a non linear minimization method.

Then, we present a second approach for recovering Euclidean reconstruction, with an uncalibrated affine camera mounted onto a robot arm. We show how using Euclidean information given by the robot motion. We also explain how obtaining camera calibration and hand-eye calibration.

In order to use these algorithms for reconstruction from a practical point of view, we present a method to do the tracking of characteristic points along a sequence of images. Moreover, we also present a method to obtain a subpixel accuracy of the image point coordinates for a low computation cost.

Keywords : computer vision, 3D reconstruction (Euclidean and affine), object pose, tracking, correlation, perspective camera model, affine camera model, weak perspective model, paraperspective model, camera calibration.

Remerciements

Le travail présenté dans ce rapport a été réalisé au sein de l'Institut National de Recherche en Informatique et Automatique de la région Rhône-Alpes, grâce au soutien financier de la DGA et d'un contrat avec la Société Aérospatiale.

Je souhaite remercier les personnes qui m'ont fait l'honneur de participer à mon jury :

- Jean-Marie Laborde, Directeur de Recherche, pour avoir accepté de présider le jury,
- les rapporteurs de ce mémoire, Daniel Dementhon, Chercheur, et Michel Dhome, Directeur de Recherche,
- les examinateurs, Pierre Comon, Directeur de Recherche, Bruno Mazar, Ingénieur à l'Aérospatiale, Brigitte Plateau, Professeur à l'Institut National Polytechnique de Grenoble,
- mon directeur de thèse, Radu Horaud, Directeur de recherche, qui m'a encadré tout au long de ces trois années.

Je remercie également Roger Mohr, Professeur à l'Institut National Polytechnique de Grenoble, pour m'avoir accueilli au sein de son équipe, et j'exprime toute ma sympathie et ma gratitude aux membres de l'équipe MOVI pour les discussions, conseils, et moments de détente que nous avons eus.

Sommaire

Notations

$\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{e}_i$, $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$, $\boldsymbol{l}$, $\boldsymbol{m}$, $\boldsymbol{m}_0$, $\boldsymbol{m}_j$, $\boldsymbol{n}$, $\boldsymbol{p}$, $\boldsymbol{q}$, $\boldsymbol{r}$, $\boldsymbol{s}$, $\boldsymbol{t}$, $\boldsymbol{u}$, $\boldsymbol{x}$, $\boldsymbol{y}$,

$\boldsymbol{D}_j$, $\boldsymbol{I}$, $\boldsymbol{J}$, $\boldsymbol{K}$, $\boldsymbol{I}_p$, $\boldsymbol{J}_p$, $\boldsymbol{I}_0$, $\boldsymbol{J}_0$,

$\boldsymbol{M}$, $\boldsymbol{M}_0$, $\boldsymbol{M}_j$, $\boldsymbol{S}_j$, $\boldsymbol{\Sigma}_j$, $\boldsymbol{\Omega}_j$ ..... vecteurs (gras italique)

$\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$, $\mathbf{H}$, $\mathbf{K}$, $\mathbf{L}$, $\mathbf{N}$, $\mathbf{O}$, $\mathbf{P}$, $\mathbf{P}_p$, $\mathbf{P}_{pf}$, $\mathbf{P}_{pp}$,

$\mathbf{Q}$, $\mathbf{R}$, $\mathbf{S}$, $\mathbf{T}$, $\mathbf{U}$, $\mathbf{W}$, $\mathbf{Y}_i$, $\mathbf{\Sigma}$ ..... matrices (gras)

a, a_j, b, b_j, c, c_j, d, e, f, i, j, k, n s, t_x, t_y, t_z, u, v, u_c, v_c,

x, y, x_0, y_0, x_j, y_j, x_{ij}, y_{ij}, X, Y, Z, Z_0,

α, α_u, α_v, β, γ, ε, η, λ, μ, ν, φ, ψ, ρ, θ ..... scalaires (italique)

O, A, B, C, D, M, M_0, M_j, N,

m, m_0, m_j, Ω_j ..... points géométriques (italique)

l, l_i, d_j, D_j ..... droites géométriques (italique)

π, π_i ..... plans géométriques

$\mathcal{P}^n$ ..... espace projectif de dimension n

$\boldsymbol{\mathcal{I}}_n$ ..... matrice identité de taille n

${}^t\boldsymbol{I}$, ${}^t\boldsymbol{R}$ ..... la transposée de $\boldsymbol{I}$, $\mathbf{R}$

$\boldsymbol{AB}$ ..... le vecteur allant du point A au point B (gras italique)

$\mathbf{A}^\dagger$ ..... la matrice pseudo-inverse de la matrice $\mathbf{A}$

$[\boldsymbol{u}]_\times$ ..... matrice antisymétrique 3×3 associée au vecteur $\boldsymbol{u}$

$\boldsymbol{u} \cdot \boldsymbol{v}$ ..... produit scalaire des vecteurs $\boldsymbol{u}$ et $\boldsymbol{v}$

$\boldsymbol{u} \wedge \boldsymbol{v}$ ..... produit vectoriel des vecteurs $\boldsymbol{u}$ et $\boldsymbol{v}$

Introduction

> "Il y a diverses sortes de curiosité : l'une d'intérêt, qui nous porte à désirer d'apprendre ce qui nous peut être utile, et l'autre d'orgueil, qui vient du désir de savoir ce que les autres ignorent."
>
> La Rochefoucauld, *Maximes*, 1678.

La vision est un de nos sens les plus puissants. L'œil humain nous permet de percevoir et d'interagir avec l'environnement qui nous entoure, de recueillir des informations tridimensionnelles d'une scène, d'estimer des distances et des vitesses, et de reconnaître des objets ou des personnes. La vision par ordinateur combine l'utilisation de caméras numériques et l'informatique et vise à donner aux robots la capacité de percevoir l'espace qui les entoure. Cependant, la mise en œuvre n'est pas simple car le fonctionnement de la vision humaine reste mal connu.

La vision par ordinateur possède un grand nombre d'applications. Elle nous permet par exemple de pouvoir effectuer des opérations à distance dans des milieux hostiles (sites nucléaire, espace...), en installant des caméras sur des robots. Ceux-ci doivent pouvoir évoluer avec une certaine autonomie, avec l'aide éventuelle d'un opérateur humain (télérobotique, télémanipulation). La création de robots ou de véhicules autonomes conduit à développer des stratégies sur leur comportement, à pouvoir détecter et éviter des obstacles, et éventuellement à l'estimation de la localisation de ces engins (dans le cas de l'évolution dans un site connu). Certaines applications nécessitent également la commande de bras manipulateurs par asservissement visuel (saisie d'objets).

Le problème de la reconnaissance d'objets est également un problème d'actualité, pour des applications robotiques, militaires ou civiles. Différentes solutions peuvent être utilisées pour résoudre ce problème : reprojection du modèle 3D suivant différentes orientations puis identification (reconnaissance 2D-3D), reconstruction de l'objet puis identification 3D-3D, ou recherche dans une base de données constituée d'images (reconnaissance 2D-2D).

La reconstruction à partir d'une séquence d'images ou vidéo est utilisée pour la création automatique ou assistée de modèles tridimensionnels.

L'imagerie médicale est également un domaine en plein essor, les domaines d'application principaux étant l'aide au diagnostic et l'assistance aux opérations chirurgicales.

Enfin, il existe d'autres applications de la vision basées sur la synthèse d'images : synthèse de nouveaux points de vue à partir de vues existantes, création et navigation dans

un monde virtuel, simulation de phénomènes physiques, incrustation d'images (insertion de panneaux publicitaires autour d'un stade pour une retransmission télévisée). L'analyse de séquences vidéo (découpage de films, vidéo cliquable...) connaît également un développement important.

Suivant les applications finales, les traitements sont effectués soit de manière autonome, soit de manière assistée par l'homme.Dans le cadre de cette thèse, nous nous intéresserons plus particulièrement à deux aspects de la vision par ordinateur :

- *au problème du calcul de pose* : il s'agit d'estimer, à partir d'une image et d'un modèle d'un objet, la position de la caméra par rapport à l'objet (ou de manière équivalente la position de l'objet par rapport à la caméra). Les applications courantes du calcul de pose sont la robotique (asservissement visuel afin de saisir un objet avec la pince d'un robot, navigation d'un robot mobile) et les domaines militaires (localisation d'un engin aérien ou terrestre, guidage d'un missile).

- *au problème de la reconstruction à partir d'une séquence d'images* : on s'intéressera plus particulièrement à la reconstruction euclidienne. Les application sont la robotique (robotique mobile, asservissement visuel), reconnaissance d'un objet, construction d'un modèle 3D (modèle numérique de terrain...), retrouver des distances, des angles, et de manière générale des informations 3D à partir d'images 2D.

Nous nous intéresserons également à un problème plus bas niveau, la poursuite de points sur une séquence d'images, afin de mettre des points en correspondance sur une séquence d'images en vue d'effectuer une reconstruction de la scène.

Contributions

Les contributions de cette thèse sont les suivantes.

- Nous avons proposé un algorithme de **poursuite de points** pour une séquence monoculaire d'images – étape indispensable afin d'effectuer une reconstruction 3D. Nous avons montré comment étendre la méthode pour une séquence stéréoscopique, et comment utiliser la géométrie épipolaire pour accélérer et valider les appariements. Nous montrons également comment obtenir une précision sous-pixellique pour un faible coût calculatoire supplémentaire (chapitre 1).

- Nous avons établi le lien existant entre les modèles paraperspectif et perspectif de caméra, et nous avons développé un **algorithme générique** permettant de résoudre les deux problèmes connexes que sont le calcul de pose et la reconstruction 3D. L'algorithme se décline en deux variantes, la première utilisant la relation existant entre le modèle orthographique à l'échelle de caméra et le modèle perspectif, la seconde utilisant la relation entre le modèle paraperspectif et le modèle perspectif.

  Ainsi, nous avons développé une méthode itérative de **calcul de pose à partir de correspondances de points**. L'algorithme proposé effectue un calcul de pose en améliorant de manière incrémentale la pose obtenue avec un modèle de paraperspectif de caméra, pour converger, à la fin, vers la solution obtenue avec un modèle perspectif de caméra (il s'agit d'une extension de la méthode proposée par Dementhon et Davis [Dem 92b, Dem 93]). Nous avons également étudié le cas d'un modèle coplanaire qui

conduit à une résolution spécifique du système d'équations obtenu, et nous avons introduit un moyen simple afin de tenir compte de la contrainte d'orthogonalité de la matrice de rotation 3×3 représentant l'orientation entre l'objet 3D et la caméra (chapitre 3).

- Nous avons ensuite étendu la méthode au problème du **calcul de pose à partir de correspondances de droites**, en faisant le lien entre les modèles orthographique à l'échelle et perspectif, puis paraperspectif et perspectif. Nous avons également étudié le cas d'un modèle coplanaire, ainsi que les configurations valides et dégénérées pour l'application de la méthode proposée (chapitre 4).

- Nous nous sommes intéressés au problème de la reconstruction euclidienne à partir d'une séquence d'images. Nous avons étendu les algorithmes itératifs de calcul de pose au cas de la **reconstruction euclidienne avec un modèle perspectif de caméra étalonnée**. La méthode peut être également vue comme une extension de la méthode de factorisation proposée par Tomasi et Kanade [Tom 91a, Tom 92]. L'avantage de la méthode proposée est d'utiliser les relations mathématiques existant entre les modèles affines de caméra et le modèle perspectif. La méthode résout le problème de l'ambiguïté de signe (solution miroir) inhérent aux modèles affines de caméra, et ne nécessite que des calculs algébriques simples. Nous avons démontré que la reconstruction obtenue est faiblement influencée par l'étalonnage de la caméra, et nous avons comparé le coût de la méthode proposée par rapport à une méthode de minimisation non linéaire (chapitre 5).

- Nous avons également proposé une méthode de **reconstruction euclidienne avec un modèle affine de caméra non étalonnée montée sur le bras d'un robot**, en utilisant l'information euclidienne fournie par le déplacement du robot. Les mouvements effectués par le robot sont connus, mais la position de la caméra par rapport au robot est inconnue. Nous expliquons comment effectuer une reconstruction euclidienne, et nous avons montré comment obtenir l'étalonnage de la caméra ainsi que l'étalonnage caméra-pince (chapitre 7).

- Enfin, cette thèse a fait l'objet d'un **transfert technologique avec la Société Aérospatiale**. Les expérimentations ont été effectuées non seulement sur des données de laboratoire (contexte robotique), mais également sur des images infrarouge fournies par la Société. Les applications qui ont fait l'objet d'une collaboration sont les suivantes : poursuite de points sur des vues aériennes, recalage d'un missile, poursuite de cibles, reconstruction et reconnaissance de bâtiments. Le chapitre 8 présente une partie de ces travaux.

Plan de la thèse

Nous présentons au chapitre 1 un algorithme de poursuite de points à partir d'une séquence monoculaire d'images, puis nous l'étendons pour des images stéréoscopiques. Le problème d'extraction et de mise en correspondances de points entre plusieurs image est une opération de bas niveau indispensable pour pouvoir effectuer une reconstruction 3D d'un objet que nous étudierons au chapitre 5.

Le chapitre 2 effectue le lien entre le modèle perspectif de caméra et les modèles affines, puis introduit un algorithme générique itératif permettant de résoudre à la fois le problème le calcul de pose à partir d'une seule image, et le problème de reconstruction à partir d'une séquence d'images, en utilisant un modèle perspectif de caméra.

Le chapitre 3 traite du problème du calcul de pose à partir d'une image et d'un modèle 3D de la scène (objet), à partir de correspondances de points.

Le chapitre 4 étend l'algorithme de calcul de pose au cas de correspondances de droites. Nous regarderons également comment mixer les points et les droites.

Le chapitre 5 explique les différentes étapes de la méthode itérative afin d'effectuer une reconstruction euclidienne avec un modèle perspectif de caméra étalonnée (extension des algorithmes itératifs de calcul de pose). Nous nous intéresserons également au problème des occultations dans les images, et nous analyserons de manière théorique la complexité de la méthode proposée par rapport à une méthode de minimisation non linéaire.

Le chapitre 6 analyse la convergence des deux variantes de l'algorithme itératif proposé, et l'influence de l'étalonnage de la caméra.

Le chapitre 7 présente une application de la reconstruction euclidienne dans le cas où l'on considère un modèle affine de caméra non calibrée montée sur le bras d'un robot.

Le chapitre 8 présente quelques travaux effectués en collaboration avec la Société AÉROSPATIALE.

Enfin, nous résumerons les différents travaux et présenterons quelques perspectives.

Chapitre 1

Poursuite de points sur une séquence d'images

> "Les principes de la géométrie proviennent de l'apparence générale des objets puisque toutes nos idées proviennent de nos impressions. Ainsi la géométrie ne peut aspirer à l'entière certitude, à cause de l'imprécision qui caractérise le jugement des sens."
>
> David Hume, *Traité de la nature humaine*, XVIII^{e}s.

Les opérations de traitement d'images peuvent être classées en plusieurs catégories. Les opérations de bas niveau extraient des informations simples à partir des images. Il s'agit en général d'extraction de primitives (points, contours courbes ou sous la forme de segments de droites, régions...). Les opérations de plus haut niveau effectuent des opérations plus élaborées comme la reconstruction 3D, la reconnaissance d'objet, l'indexation, l'asservissement visuel, etc. Dans le cadre de ce rapport, nous nous intéressons au problème de la reconstruction 3D à partir d'une séquence d'images. Il nous faut donc dans un premier temps extraire des primitives dans les images, et être capable de les mettre en correspondance d'une image à l'autre. Le problème d'extraction et de mise en correspondance n'est pas le sujet principal de cette thèse, mais c'est cependant une étape nécessaire et trop souvent supposée résolue pour le problème de reconstruction. Dans ce chapitre, nous proposons une méthode pour effectuer la poursuite [1] de points caractéristiques dans une séquence d'images. Nous montrerons également comment obtenir une précision sous-pixellique des positions des points dans les images. Nous nous intéressons d'abord au cas

1. En anglais : tracking.

d'une séquence d'images monoculaires. Nous étendons ensuite la méthode au cas d'une séquence d'images stéréoscopiques.

Plan du chapitre

Le paragraphe 1.1 présente un état de l'art sur les détecteurs de points d'intérêt (sous-paragraphe 1.1.1), sur le problème d'appariement entre deux images (sous-paragraphe 1.1.2), et sur le problème de poursuite de points dans une séquence d'images (sous-paragraphe 1.1.3). Nous nous intéresserons ensuite à la poursuite de points caractéristiques sur d'une séquence monoculaire d'images (paragraphe 1.2.1). Nous proposerons également une méthode permettant d'obtenir une précision sous-pixellique pour les positions des points poursuivis dans les images (paragraphe 1.2.1.2). Enfin, le paragraphe 1.2.2 étend la méthode de poursuite de points sur une séquence d'images stéréoscopiques.

1.1 État de l'art

1.1.1 Détecteurs de points d'intérêts

Les détecteurs de points d'intérêts peuvent être classés en trois catégories :

- Les **approches basées sur les contours** effectuent d'abord une extraction des contours. Ensuite, plusieurs solutions sont possibles pour extraire les points d'intérêt : Asada et Brady [Asa 86] calculent les changements de courbure, ces changements étant classés en plusieurs catégories liés à la nature du point image correspondant (coin...) ; Médioni et Yasumoto [Med 87] effectuent une approximation des contours par des B-splines, puis recherchent les extrema de courbure ; Horaud et al. [Hor 90] effectuent une approximation polygonale, les points d'intérêt étant donnés par les points d'intersection et d'inflexion des segments calculés.

- La seconde catégorie concerne les **approches basées sur le signal**, et conduisent en général à des calculs de dérivées dans l'image. De nombreux travaux ont été effectués suivant cette voie. Beaudet [Bea 78] a proposé un opérateur basé sur les dérivées secondes du signal pour détecter les points d'intérêt. Les approches de Kitchen et Rosenfield [Kit 82] et Dreschler et Nagel [Dre 82] sont basées sur la courbure de courbes planes. Harris et Stephens [Har 88] utilisent une matrice liée à la fonction d'autocorrélation qui prend en compte les valeurs des dérivées premières du signal sur une fenêtre. Cottier [Cot 94] a amélioré la localisation des points détectés par le détecteur de Harris, en appliquant le détecteur de Harris uniquement sur les contours de l'image, et en utilisant successivement deux tailles de support différentes. Schmid [Sch 96a] a également proposé une amélioration du détecteur de Harris, en améliorant la robustesse du calcul des dérivées. Förstner [F87] a utilisé une fonction d'autocorrélation conjointement avec une méthode statistique. Heitger et al. [Hei 92] ont utilisé des filtres de Gabor.

- Enfin, la troisième classe d'approches concerne les **approches basées sur un modèle théorique du signal**, le but étant d'obtenir une précision sous-pixellique de la localisation des points dans l'image. Ces méthodes ne sont utilisables que pour des types bien définis de points d'intérêt : jonctions de plusieurs lignes (Rohr

[Roh 92], Deriche et Blaska [Der 93a]), coins (Deriche et Giraudon [Der 90b], Brand et Mohr [Bra 94]), points elliptiques (Noble [Nob 88]).

Schmid [Sch 96a] a comparé plusieurs de ces approches en mesurant le contenu informatif des points extraits, ainsi que la répétabilité de la localisation des points. Le choix d'utilisation de l'un de ces extracteurs de points d'intérêt dépendra des propriétés recherchées : précision de localisation des points, invariance aux rotations ou aux changements d'échelle, temps de calcul, type d'image (objet polyédrique ou courbe).

1.1.2 Appariement

De manière générale, une seule image ne fournit que peu d'informations sur la scène si l'on ne possède aucune information à priori sur celle-ci. Afin de pouvoir effectuer une reconstruction tridimensionnelle d'un objet, il est nécessaire de considérer plusieurs images de la même scène, sous des points de vue différents. Ceci nous conduit à mettre en correspondance les primitives communes entre les images.

Le problème de la mise en correspondance entre deux images consiste à trouver, pour chaque primitive d'une image, la primitive correspondante dans l'autre image. Nous nous intéresserons ici au cas des correspondances de points. Ces appariements sont effectués après calcul d'une mesure de ressemblance. De nombreuses mesures ont été proposées.

Mesures de corrélation standard. Les mesures de corrélation sont les mesures les plus utilisées pour effectuer des appariements de points entre images. Ces mesures sont basées sur les différences d'intensité lumineuse entre deux sous-fenêtres de taille $(2n+1) \times (2n+1)$ pixels dans chacune des deux images. Quelques mesures de corrélation usuelles sont rappelées dans le tableau 1.1. Aschwanden [Asc 92] a étudié et comparé 19 mesures de corrélation. Ses résultats ont montré que la mesure de corrélation SAD donne de très bons résultats, sauf pour les changements de luminosité. Blanc [Bla 98] a également travaillé sur le problème d'appariement, et il conseille de faire une correction globale sur une des deux images en cas de changement de luminosité importante entre les deux images. Les fenêtres de taille 5×5 et 7×7 pixels sont généralement de bonnes valeurs à adopter ; un masque plus grand aboutit à des résultats similaires, mais respecte moins bien l'hypothèse d'un mouvement de translation (la taille à utiliser dépend cependant du contenu informatif de l'image). Zhang [Zha 94c, Zha 95c] a également travaillé sur le problème d'appariement entre deux images stéréoscopiques à partir de corrélations. Il a proposé une méthode de mise en correspondance robuste basée sur l'estimation de la matrice fondamentale et sur des contraintes géométriques locales autour des points en correspondance.

Mesures de corrélation robustes. D'autres mesures, robustes au problème d'occultation, ont été proposées. Zahib [Zab 94] a défini la transformation *rank*. Celle-ci transforme chaque pixel en une valeur indiquant le nombre de pixels voisins (parmi les 8) qui lui sont inférieurs. Ensuite, la mesure SAD est utilisée pour calculer la ressemblance entre les deux images. Il a également décrit une variante, la transformation *census* qui transforme chaque pixel en une chaîne de 8 bits indiquant quels sont les pixels voisins qui lui sont inférieurs. La mesure de distance utilisée entre les images transformées est une distance de Hamming. Bhat [Bha 96] a proposé une autre transformation consistant à transformer chaque pixel en une suite de 9 chiffres

Mesure	Expression
SAD	$\sum_{i=-n}^{n}\sum_{j=-n}^{n} \lvert I_2(x_2+i,y_2+j) - I_1(x_1+i,y_1+j)\rvert$
SSD	$\sum_{i=-n}^{n}\sum_{j=-n}^{n} \left(I_2(x_2+i,y_2+j) - I_1(x_1+i,y_1+j)\right)^2$
ZSAD	$\sum_{i=-n}^{n}\sum_{j=-n}^{n} \left\lvert \left(I_2(x_2+i,y_2+j) - \overline{I}_2\right) - \left(I_1(x_1+i,y_1+j) - \overline{I}_1\right)\right\rvert$
ZSSD	$\sum_{i=-n}^{n}\sum_{j=-n}^{n} \left(\left(I_2(x_2+i,y_2+j) - \overline{I}_2\right) - \left(I_1(x_1+i,y_1+j) - \overline{I}_1\right)\right)^2$
NCC	$\dfrac{\sum_{i=-n}^{n}\sum_{j=-n}^{n} I_2(x_2+i,y_2+j) I_1(x_1+i,y_1+j)}{\sqrt{\sum_{i=-n}^{n}\sum_{j=-n}^{n} I_2^2(x_2+i,y_2+j) \sum_{i=-n}^{n}\sum_{j=-n}^{n} I_1^2(x_1+i,y_1+j)}}$
ZNCC	$\dfrac{\sum_{i=-n}^{n}\sum_{j=-n}^{n} (I_2(x_2+i,y_2+j) - \overline{I}_2)(I_1(x_1+i,y_1+j) - \overline{I}_1)}{\sqrt{\sum_{i=-n}^{n}\sum_{j=-n}^{n} (I_2(x_2+i,y_2+j) - \overline{I}_2)^2 \sum_{i=-n}^{n}\sum_{j=-n}^{n} (I_1(x_1+i,y_1+j) - \overline{I}_1)^2}}$

TAB. 1.1: *Quelques mesures de corrélation usuelles :* SAD *et* SSD *sont basées sur les différences de niveaux de gris,* ZSAD *et* ZSSD *sont les mesures centrées correspondantes,* NCC *est une mesure normalisée, et* ZNCC *est une mesure normalisée et centrée.*

(allant de 1 à 9) correspondant à la numérotation des 9 pixels du voisinage 3×3 dans l'ordre croissant de leur intensité. Lan [Lan 97] utilise une méthode faisant appel aux statistiques robustes pour détecter les occultations. Le principe consiste à utiliser la mesure de corrélation ZSSD, mais seulement sur les pixels de la fenêtre qui ne sont pas occultés. Pour déterminer les pixels occultés, il suppose que les pixels des fenêtres en correspondance subissent une translation d'intensité qu'il calcule de manière robuste (statistiques), puis estime ensuite la distribution de l'erreur. Il peut alors déterminer les pixels occultés, puis effectuer une corrélation sur la partie non occultée. Enfin, Blanc [Bla 98] a proposé une amélioration de cette méthode en tenant compte de la compacité des pixels occultés : ceux-ci doivent se trouver dans un même voisinage.

Mesures de corrélation sous-pixellique. Ces mesures calculent l'intensité du signal sur des coordonnées non entières dans l'image. Pour ce faire, les méthodes existantes effectuent une interpolation des intensités des pixels entiers les plus proches. Brand [Bra 95] et Gruen [Gru 85] supposent que deux masques en correspondance sont déformés par une transformation affine. Lan [Lan 97] suppose que le mouvement de la caméra est une translation. Enfin, Blanc [Bla 98] sous-échantillonne les pixels pour calculer l'intensité lumineuse sur des pixels non entiers, puis effectue une corrélation.

Mesures basées sur les invariants. Les mesures de corrélation précédentes ne s'appliquent pas en cas de rotation importante ou de changement d'échelle. Schmid [Sch 96a] a proposé 9 invariants aux rotations définis à partir des dérivées du signal-image. La mise en correspondance est ensuite effectuée à l'aide d'une mesure de ressemblance définie par une distance de Mahalanobis.

1.1.3 Poursuite de points

Le problème de la poursuite de points – ou tracking en anglais – suppose que les images sont prises à fréquence élevée, et que le déplacement dans l'image est faible et conserve une certaine cohérence locale.

Tomasi et Kanade [Tom 91b] font l'hypothèse que le déplacement entre deux images peut être modélisé par une translation, plus du bruit. Le problème consiste à minimiser une erreur résiduelle.

Les méthodes basées sur l'estimation du flot optique permettent d'estimer le déplacement d'un ensemble de points entre deux images proches et d'assurer une certaine cohérence locale sur le déplacement des points voisins. Le calcul est généralement effectué en deux étapes : (i) appariement entre les deux images (le plus souvent par corrélation), et (ii) régularisation ou lissage des mouvements des points. De nombreux auteurs s'intéressent à ce problème parmi lesquels Nagel [Nag 83, Nag 87], Barron et al. [Bar 94] (classification et comparaison de différentes approches), Anandan [Ana 89] (approche hiérarchique), Szeliski et Shum [Sze 95b] (modélisation du flot optique par des splines).

D'autres approches sont basées sur la poursuite d'un objet dont le modèle 3D est connu. L'objet est modélisé par un ensemble de points ou de segments de droites [Dau 95, Der 90a, Zha 94b, Har 92a]. L'utilisation d'un filtre de Kalman permet de stabiliser le processus de poursuite et de prédire la position des primitives dans les images suivantes.

Uenohara et Kanade [Uen 96] se sont intéressés à la vérification de la poursuite en utilisant des invariants définis à partir de 5 points coplanaires et des moments affines.

1.2 Poursuite de points sur une séquence d'images

1.2.1 Cas d'une séquence monoculaire d'images

1.2.1.1 Algorithme

Nous nous sommes placés dans un contexte où la fréquence d'acquisition des images est relativement élevé (le déplacement des objets dans l'image est faible entre deux images consécutives), et le calcul des correspondances doit être rapide. De plus, nous voulons un outil général sans faire d'hypothèse à priori sur le type de mouvement de la caméra, qui est supposé inconnu. L'extraction de points caractéristiques étant pénalisante en temps de calcul, nous ne l'effectuons qu'à la première image (ou sur un nombre réduit d'images), et nous utilisons une méthode basée sur la corrélation pour prédire les points d'une image à l'autre. L'algorithme proposé est le suivant.

1. Extraire des points dans la première image, en utilisant par exemple le détecteur de Harris amélioré proposé par Schmid [Sch 96a].

2. Pour chaque nouvelle image, faire une corrélation entre les points de l'image précédente et l'image courante afin de prédire leur position dans la nouvelle image.

Prendre une fenêtre de recherche réduite (généralement comprise entre 20×20 et 60×60 pixels suivant l'échantillonage des images) afin de restreindre le nombre de mauvais appariements et de diminuer le temps de calcul. La mesure de corrélation utilisée est SAD ou SSD avec une taille de masque comprise entre 5×5 et 11×11 pixels.

3. Faire une interpolation sur les scores de corrélation avec les points voisins pour avoir une position sous-pixellique de la localisation des points dans l'image (voir le paragraphe suivant pour le principe de calcul).

4. Pour chaque point prédit dans la nouvelle image, faire une corrélation croisée pour valider les appariements. Il s'agit, à partir des points prédits dans la nouvelle image, de calculer leur position dans l'image précédente par corrélation et de vérifier que l'on retrouve les points initiaux [Fua 91].

5. Si le nombre de points restants dans l'image est inférieur à un certain seuil, refaire une extraction de points. Pour ne pas avoir de points en double, ne conserver les nouveaux points que s'ils sont suffisamment éloignés (typiquement 5 pixels) des points déjà existants, afin de tenir compte d'une dérive éventuelle de la position des points.

1.2.1.2 Précision sous-pixellique

Nous proposons dans ce paragraphe de calculer les positions sous-pixelliques des points trouvés par corrélation dans la nouvelle image, en utilisant les scores de corrélation des points voisins.

Interpolation par deux paraboles

Pour tenir compte des scores de corrélation des points voisins, nous pouvons successivement considérer les scores obtenus avec les deux points voisins situés sur la même ligne ou sur la même colonne de l'image. En supposant que la fonction de corrélation est une fonction continue, nous pouvons obtenir une précision sous-pixellique en interpolant les scores de corrélation calculés sur les positions entières. Si l'on effectue une interpolation sur trois points, il est naturel de choisir une fonction dépendant de trois paramètres.

Nous faisons donc l'hypothèse que chaque point de la nouvelle image correspond à un extremum local pour la fonction de corrélation (minimum pour SAD et SSD), et que le score de corrélation peut être approché par deux paraboles, l'une suivant l'axe des abscisses de l'image, et l'autre suivant l'axe des ordonnées (voir figure 1.1) :

$$\begin{array}{rcccl} s_{x,0} & = & g(x) & = & ax^2 + bx + c \\ s_{0,y} & = & h(y) & = & dy^2 + ey + f \end{array}$$

Nous calculons les coefficients (a, b, c, d, e, f) à partir des cinq mesures de corrélation entre le point de coordonnées $\boldsymbol{p}$ de la première image et le point $\boldsymbol{q}$ de la seconde image et de ses quatre voisins. Nous considérons donc les 5 mesures de corrélation correspondant aux couples suivants : $(\boldsymbol{p}, \boldsymbol{q})$, $(\boldsymbol{p}, \boldsymbol{q} + (-1, 0))$, $(\boldsymbol{p}, \boldsymbol{q} + (1, 0))$, $(\boldsymbol{p}, \boldsymbol{q} + (0, -1))$, $(\boldsymbol{p}, \boldsymbol{q} + (0, 1))$.

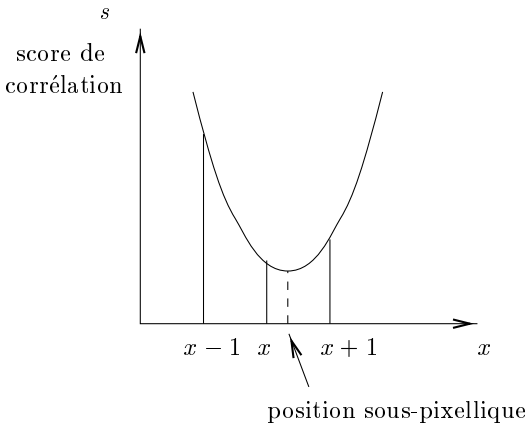


FIG. 1.1: *Interpolation du score de corrélation par une parabole suivant chaque axe (suivant l'axe x sur le schéma).*

Nous obtenons ainsi le système suivant :

$$\begin{cases} a - b + c &= s_{-1,0} \\ c &= s_{0,0} \\ a + b + c &= s_{1,0} \\ d - e + f &= s_{0,-1} \\ f &= s_{0,0} \\ d + e + f &= s_{0,1} \end{cases}$$

d'où l'on déduit les coefficients a, b, d, e :

$$\begin{cases} a &= (s_{-1,0} - 2s_{0,0} + s_{1,0})/2 \\ b &= (s_{1,0} - s_{-1,0})/2 \\ d &= (s_{0,-1} - 2s_{0,0} + s_{0,1})/2 \\ e &= (s_{0,1} - s_{0,-1})/2 \end{cases}$$

et l'extremum correspond au point :

$$\boldsymbol{q}' = \boldsymbol{q} + \left(-\frac{b}{2a}, -\frac{e}{2d} \right)$$

Dans le cas où $a = 0$ ou $d = 0$, on ne fait pas d'ajustement dans la direction considérée, et on prend respectivement $x_{\boldsymbol{q}'} = x_{\boldsymbol{q}}$ ou $y_{\boldsymbol{q}'} = y_{\boldsymbol{q}}$. Comme la corrélation entre les points $\boldsymbol{p}$ et $\boldsymbol{q}$ donne un meilleur score que les deux voisins respectivement sur l'axe des abscisses ou sur l'axe des ordonnées, cette méthode assure que l'ajustement du point est inférieur à 1 pixel dans chaque direction.

Interpolation par un paraboloïde

L'interpolation précédente ne tient compte que du voisinage en 4-connexité de la seconde image, sans utiliser l'information donnée par les voisins diagonaux. Nous supposons maintenant que le score de corrélation décrit un paraboloïde qui est une généralisation naturelle d'une parabole. On estime les coefficients à partir des valeurs de corrélation entre

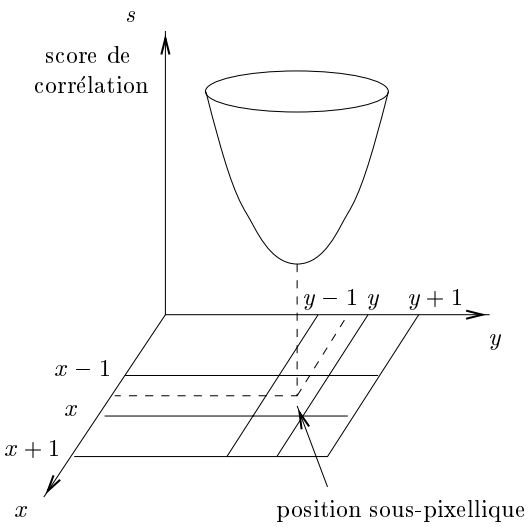


FIG. 1.2: *Interpolation du score de corrélation par un paraboloïde.*

$\boldsymbol{p}$ et $\boldsymbol{q}$ et ses 8 voisins (voir figure 1.2) :

$$s_{x,y} = ax^2 + by^2 + cxy + dx + ey + f$$

Nous avons donc 9 équations fournies par les valeurs de corrélation, et 6 inconnues (a, b, c, d, e, f) que l'on peut estimer au sens des moindres carrés. Le système matriciel s'écrit sous la forme :

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$$

où

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 0 & 0 & -1 & 0 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 \\ 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & -1 & 1 & -1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \qquad \boldsymbol{x} = \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} \qquad \boldsymbol{b} = \begin{pmatrix} s_{-1,-1} \\ s_{-1,0} \\ s_{-1,1} \\ s_{0,-1} \\ s_{0,0} \\ s_{0,1} \\ s_{1,-1} \\ s_{1,0} \\ s_{1,1} \end{pmatrix}$$

La solution de ce système est donnée par :

$$\boldsymbol{x} = \mathbf{A}^{\dagger}\boldsymbol{b}$$

où l'exposant † indique qu'il s'agit de la matrice pseudo-inverse, et :

$$\mathbf{A}^{\dagger} = \frac{1}{36} \begin{pmatrix} 6 & 6 & 6 & -12 & -12 & -12 & 6 & 6 & 6 \\ 6 & -12 & 6 & 6 & -12 & 6 & 6 & -12 & 6 \\ 9 & 0 & -9 & 0 & 0 & 0 & -9 & 0 & 9 \\ -6 & -6 & -6 & 0 & 0 & 0 & 6 & 6 & 6 \\ -6 & 0 & 6 & -6 & 0 & 6 & -6 & 0 & 6 \\ -4 & 8 & -4 & 8 & 20 & 8 & -4 & 8 & -4 \end{pmatrix}$$

L'extremum local est atteint en un point où les dérivées partielles par rapport à x et y s'annulent, on obtient ainsi :

$$\boldsymbol{q}' = \boldsymbol{q} + \frac{1}{c^2 - 4ab}\Big(2bd - ce, 2ae - cd\Big)$$

Le calcul est très rapide car la matrice $\mathbf{A}^\dagger$ peut être précalculée, et les coefficients a, b, c, d, e et f sont obtenus par une simple multiplication matricielle. Contrairement à la méthode d'interpolation par deux paraboles, nous ne sommes plus assurés d'avoir un ajustement inférieur à 1 pixel car la résolution est effectuée au sens des moindres carrés (voir figure 1.3). Si l'ajustement est supérieur à 1 pixel, ou si le dénominateur s'annule, nous rejetons la valeur trouvée, et nous pouvons soit prendre $\boldsymbol{q}' = \boldsymbol{q}$, soit faire une interpolation par deux paraboles (voir le paragraphe précédent). Ce problème arrive dans environ 5% des cas.

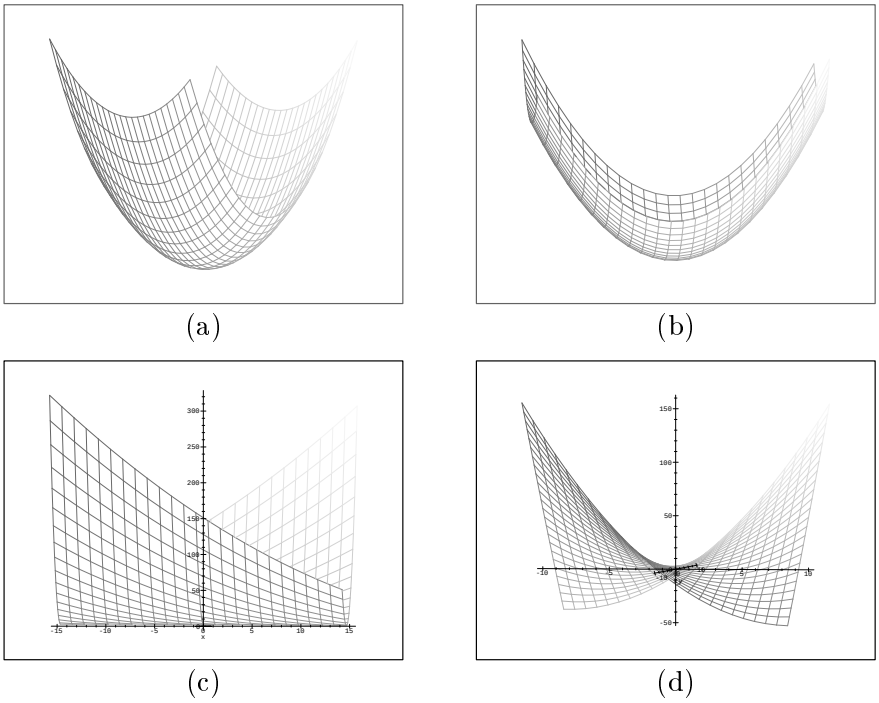


FIG. 1.3: *Illustration de la méthode d'interpolation du score de corrélation par un paraboloïde : les figures (a) et (b) représentent le comportement dans le cas général, et les figures (c) et (d) représentent deux cas dégénérés où la courbe interpolée n'est pas un paraboloïde. Ces deux derniers cas conduisent en général à un ajustement supérieur à un pixel, et celui-ci est par conséquent rejeté.*

1.2.1.3 Résultats expérimentaux

Dans ce paragraphe, nous nous intéressons à deux types d'expérimentations.

– Dans un premier temps, nous étudions de manière qualitative les méthodes de corrélation sous-pixellique par rapport à une corrélation pixellique, ainsi que par rapport

à deux autres méthodes de corrélation sous-pixellique proposée respectivement par Blanc [Bla 98] et Lan [Lan 97].

– Ensuite, nous montrons des résultats obtenus en effectuant une poursuite de points sur plusieurs séquences d'images par la méthode proposée.

La figure 1.4 représente le couple d'images de synthèse qui a été utilisé afin d'analyser les performances des différents algorithmes de corrélation. Ces images représentent trois surfaces tridimensionnelles texturées. Le processus expérimental utilisé est le suivant.

– Nous avons d'abord effectué une extraction des points caractéristiques dans les deux images en utilisant le détecteur de Harris amélioré proposé par Schmid [Sch 96a].

– Nous avons ensuite effectué une mise en correspondance automatique entre les deux images à l'aide de corrélations (il peut donc y avoir quelques mauvais appariements).

– Sur les correspondances trouvées, nous n'avons conservé que celles correspondant à un minimum local pour le score de corrélation, soit 606 points. Cette hypothèse doit en effet être vérifiée pour utiliser les méthodes d'interpolation décrites précédemment. Notons par ailleurs que cette hypothèse est toujours vérifiée pour la méthode de poursuite de points que nous avons proposée.

– Nous avons ensuite utilisé successivement chaque méthode de corrélation sous-pixellique pour réestimer de manière plus précise les points de l'image de droite, les points de l'image de gauche étant fixés.

– Enfin, pour chaque méthode de corrélation sous-pixellique, nous avons calculé l'erreur sur la localisation des points dans l'image de droite, entre la position théorique et la position estimée par corrélation. Pour connaître la position théorique du point, nous reconstruisons le point 3D à partir de l'image de gauche en effectuant du ray-tracing inverse, puis nous reprojetons le point 3D trouvé dans l'image de droite. Nous pouvons ainsi calculer l'erreur en x et y par rapport à la position trouvée par corrélation.

Les figures 1.5, 1.6, 1.7, 1.8, et 1.9 représentent les résultats obtenus pour chacune des méthodes de corrélation. Plus le pic est resserré et haut autour de zéro, meilleure est la méthode. Nous pouvons en déduire que les méthodes de corrélation sous-pixellique donnent des résultats beaucoup plus précis qu'une corrélation pixellique simple. La méthode itérative proposée par Blanc [Bla 98] est celle qui donne les meilleurs résultats (précision de l'ordre du dixième de pixel), mais c'est de loin la plus lente (temps de calcul supérieur à une minute sur une UltraSparc 1/170). La méthode d'interpolation par un paraboloïde et la méthode de Lan [Lan 97] donnent des résultats légèrement moins bons (précision de l'ordre de 0,15 pixel), suivi par la méthode d'interpolation par deux paraboles (précision de l'ordre de 0,25 pixel) (voir tableau 1.2).

Dans un second temps, nous avons testé l'algorithme de poursuite de points sur plusieurs séquences d'images réelles. Les figures 1.10, 1.11, 1.12, 1.13 et 1.14 représentent, pour chaque séquence, la première image dans laquelle les points caractéristiques ont été extraits par le détecteur de Harris, ainsi que plusieurs images intermédiaires dans lesquelles figurent les trajectoires des points. Remarquons que c'est dans les premières images que l'on perd le plus de points, car le déplacement des points sur le devant ou le derrière de

Méthode	Précision (en pixel)
pixellique	0,5
2 paraboles	0,25
paraboloïde	0,15
Blanc	0,10
Lan	0,15

TAB. 1.2: *Ordre de grandeur des précisions des méthodes de corrélation sous-pixellique.*

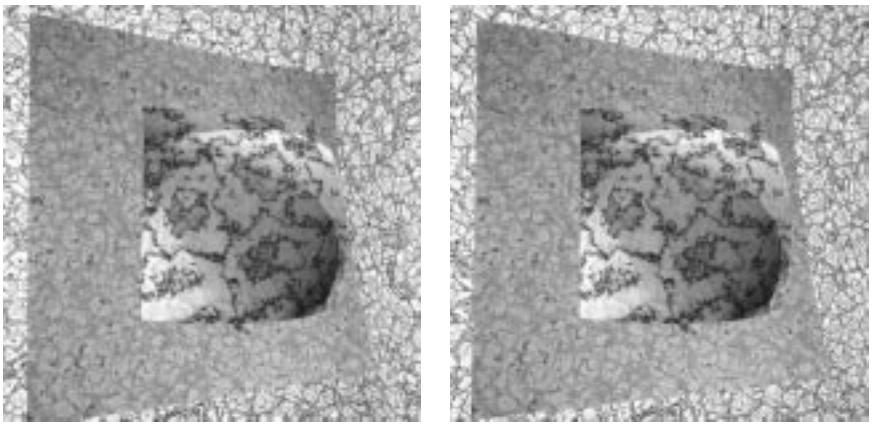


FIG. 1.4: *Couple d'images de synthèse utilisé pour comparer les mesures de corrélation.*

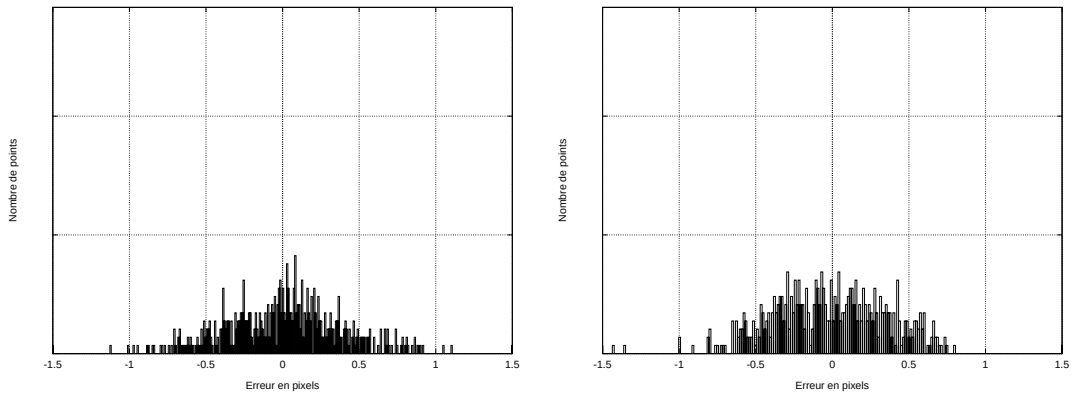


FIG. 1.5: *Erreur d'appariement dans l'image en pixels (en x à gauche et en y à droite) en utilisant une corrélation pixellique.*

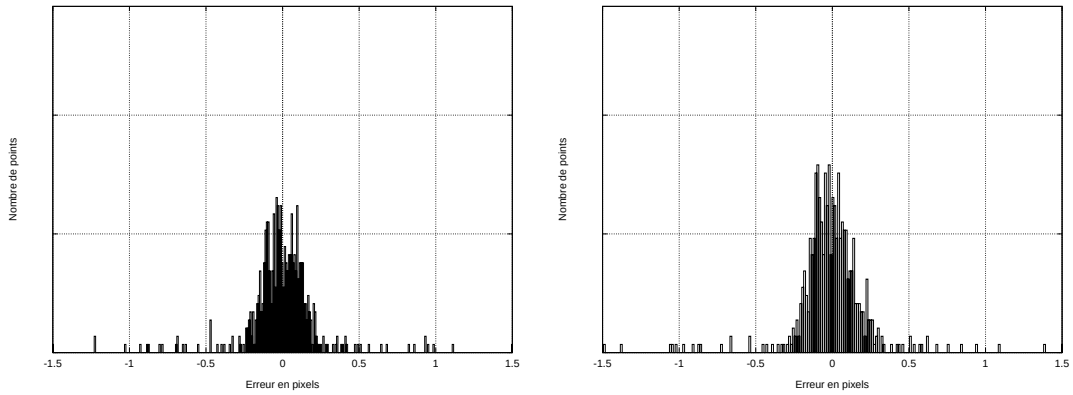


FIG. 1.6: *Erreur d'appariement dans l'image en pixels (en x à gauche et en y à droite) en utilisant une corrélation sous-pixellique (interpolation par deux paraboles).*

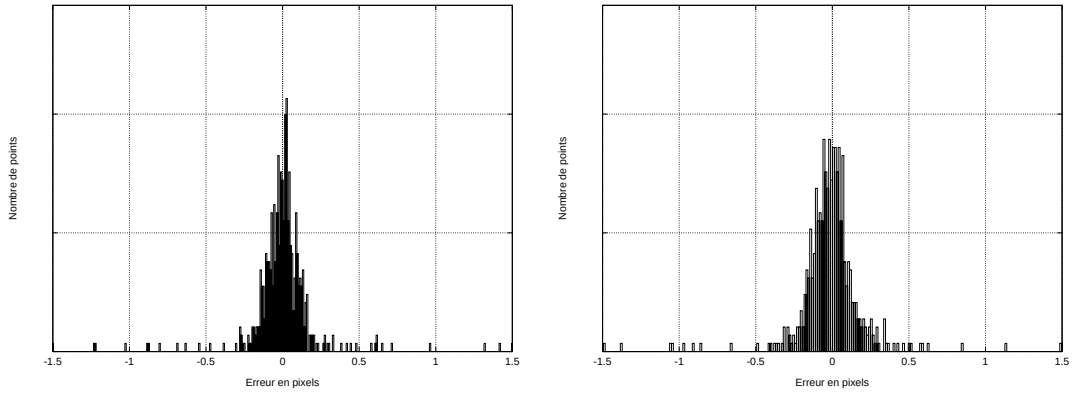


FIG. 1.7: *Erreur d'appariement dans l'image en pixels (en x à gauche et en y à droite) en utilisant une corrélation sous-pixellique (interpolation par un paraboloïde).*

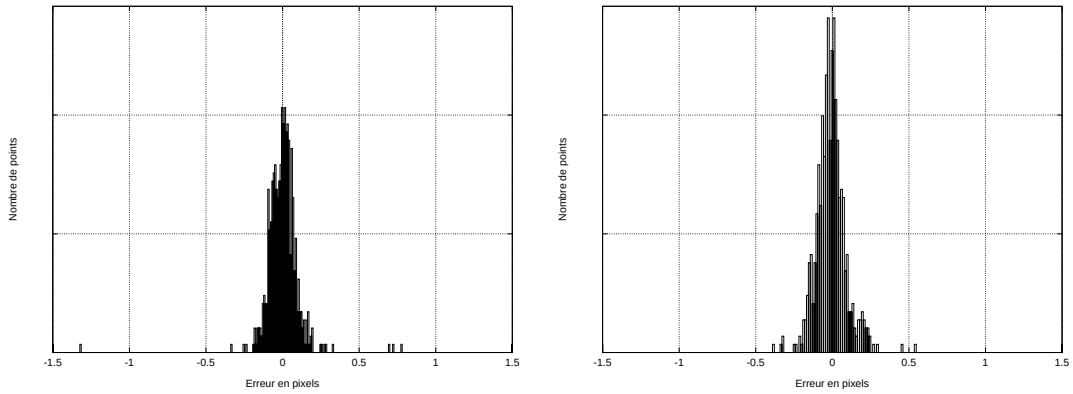


FIG. 1.8: *Erreur d'appariement dans l'image en pixels (en x à gauche et en y à droite) en utilisant une corrélation sous-pixellique (méthode itérative de Blanc).*

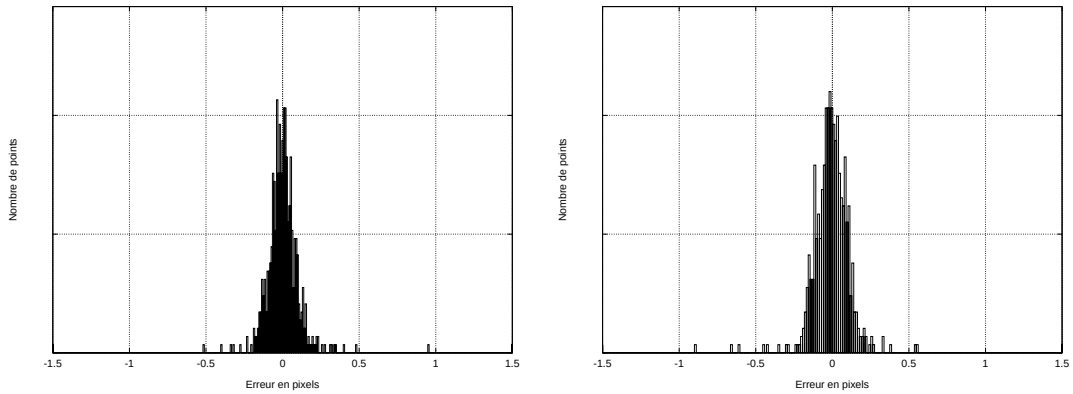


FIG. 1.9: *Erreur d'appariement dans l'image en pixels (en x à gauche et en y à droite) en utilisant une corrélation sous-pixellique (méthode de Lan).*

la scène est souvent plus important que la taille de la fenêtre de recherche. Le taux de trajectoires de points correctes sur les séquences traitées est en général supérieur à 98%.

Pour une étude plus exhaustive des différents paramètres du problème d'appariement, on pourra se réferrer à la thèse de Blanc [Bla 98] (mesures de corrélation, taille du masque, etc.).

1.2.2 Cas d'une tête stéréoscopique

1.2.2.1 Algorithme

Nous nous intéressons maintenant au cas d'une tête stéréoscopique. L'algorithme suivant est une extension de la méthode de poursuite de point pour une séquence d'images stéréoscopiques (voir figure 1.15 pour illustration).

1. Faire une extraction des points caractéristiques sur le premier couple d'images gauche-droite, les mettre en correspondance, estimer la matrice fondamentale de manière robuste, puis rejeter les faux appariements [Zha 95c].

2. Estimer de manière sous-pixellique la position des points de l'image de droite, les points de gauche étant fixés (utiliser une des interpolations précédentes).

3. Pour chaque nouveau couple d'images, faire une corrélation entre les points de l'image précédente de gauche et l'image courante de gauche (suivant le même principe que pour le cas d'une séquence monoculaire). Pour chaque point prédit dans l'image de gauche, estimer une précision sous-pixellique, puis effectuer une corrélation croisée pour valider les appariements.

4. Pour chaque point prédit dans la nouvelle image de gauche, estimer la ligne épipolaire correspondante dans l'image de droite, puis rechercher les points de l'image de droite (par corrélation à partir de l'image de droite précédente) sur une zone réduite autour de la ligne épipolaire.

5. Éventuellement, réestimer la géométrie épipolaire toutes les quelques images (typiquement 5 images).

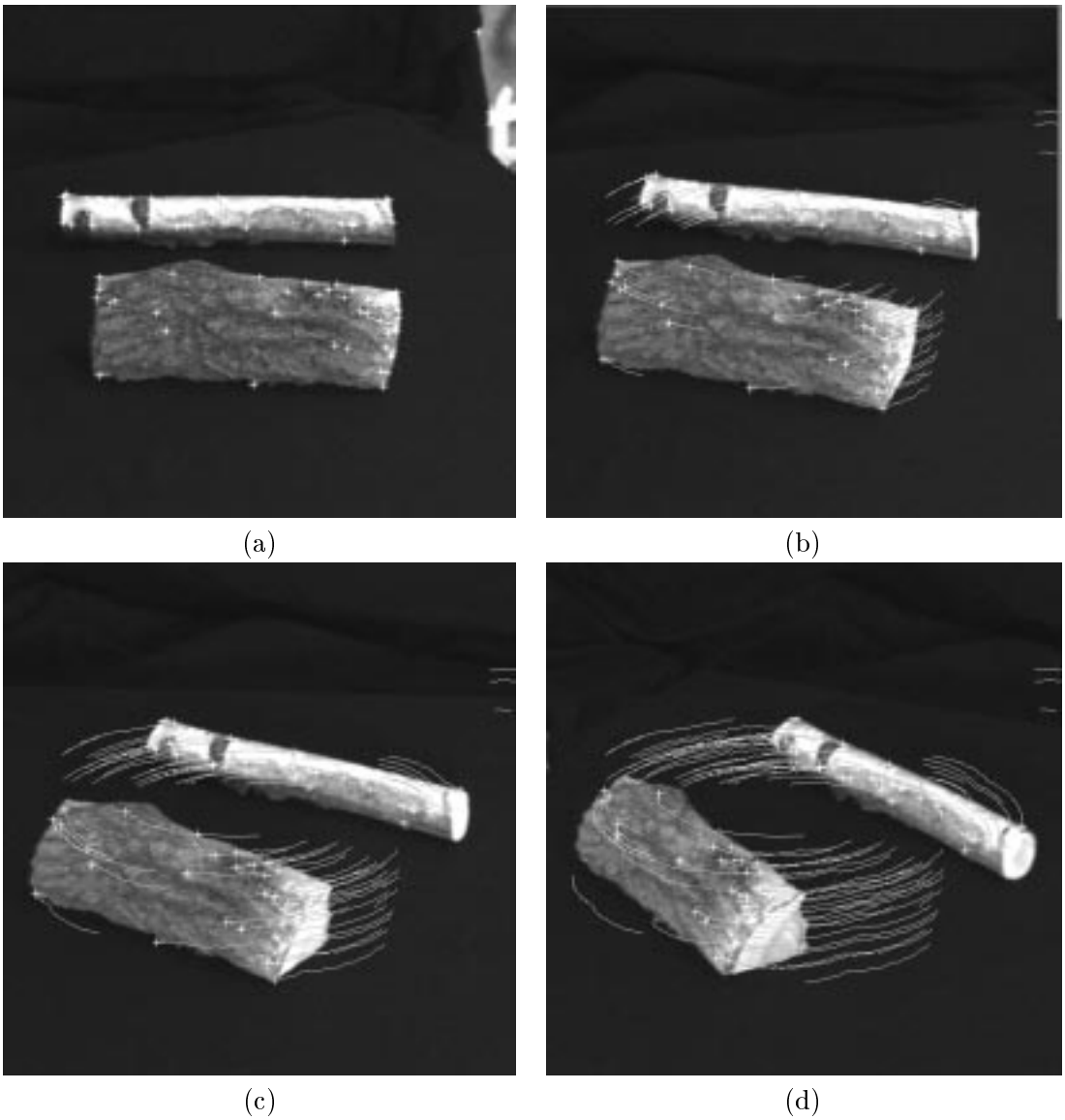
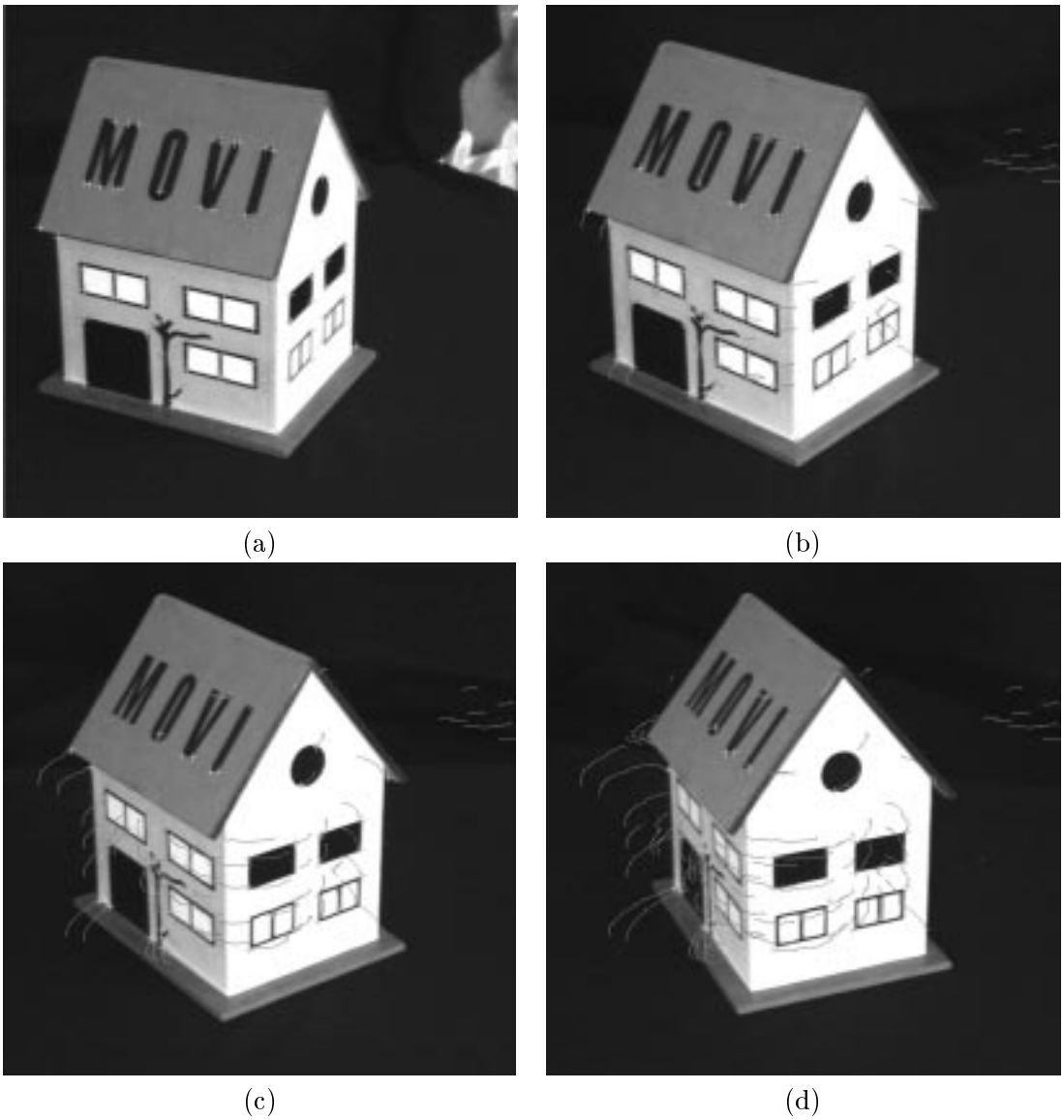


FIG. 1.10: *Séquence monoculaire 1 : (a) 92 points extraits, 1^{re} image, (b) 80 points, 15^{e} image, (c) 75 points, 30^{e} image, (d) 58 points, 50^{e} image.*



(a) (b)

(c) (d)

FIG. 1.11: *Séquence monoculaire 2 : (a) 94 points extraits, 1re image, (b) 72 points, 15e image, (c) 55 points, 25e image, (d) 36 points, 35e image.*

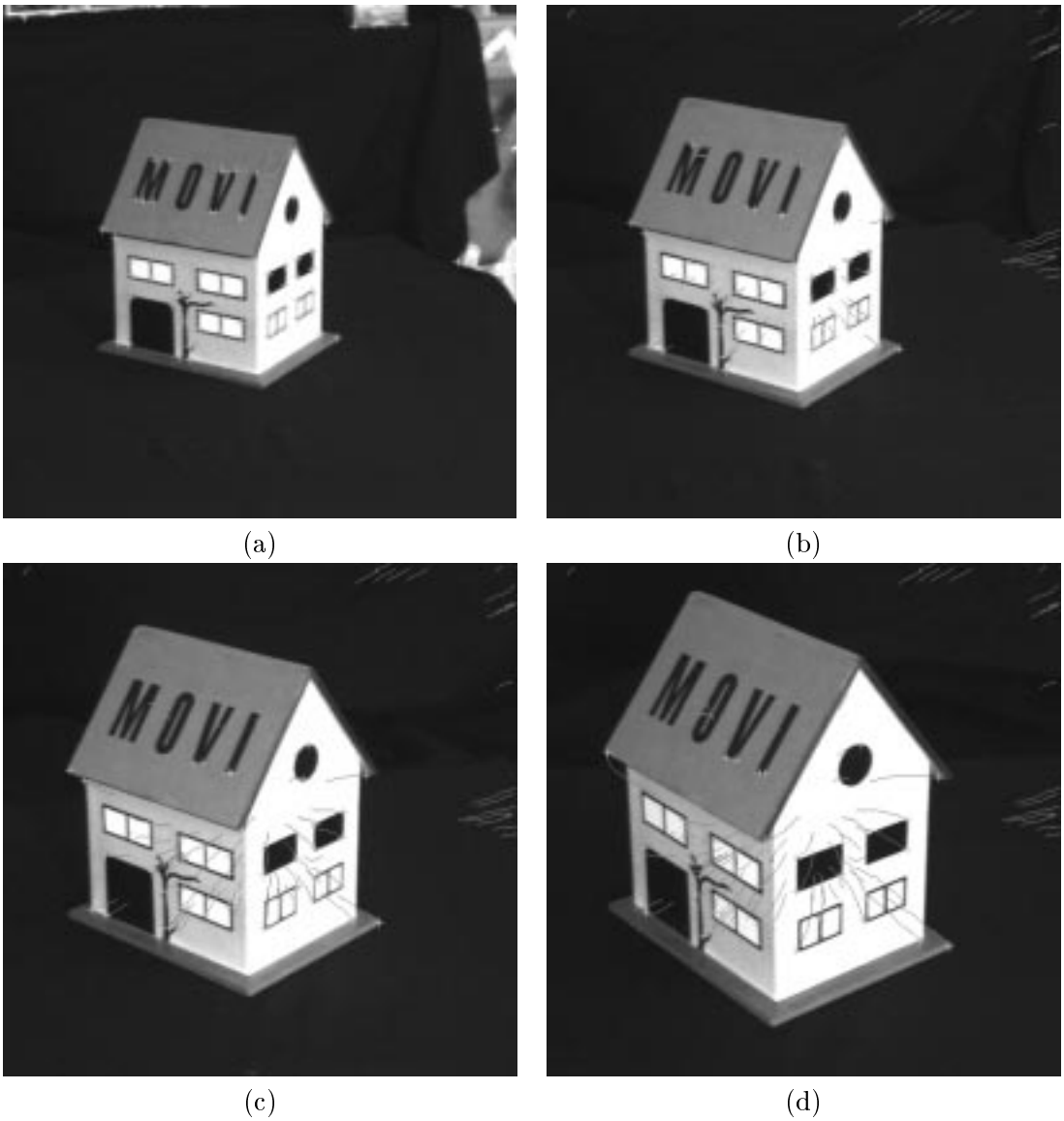


FIG. 1.12: *Séquence monoculaire 3 : (a) 96 points extraits, 1re image, (b) 63 points, 10e image, (c) 50 points, 20e image, (d) 35 points, 30e image.*

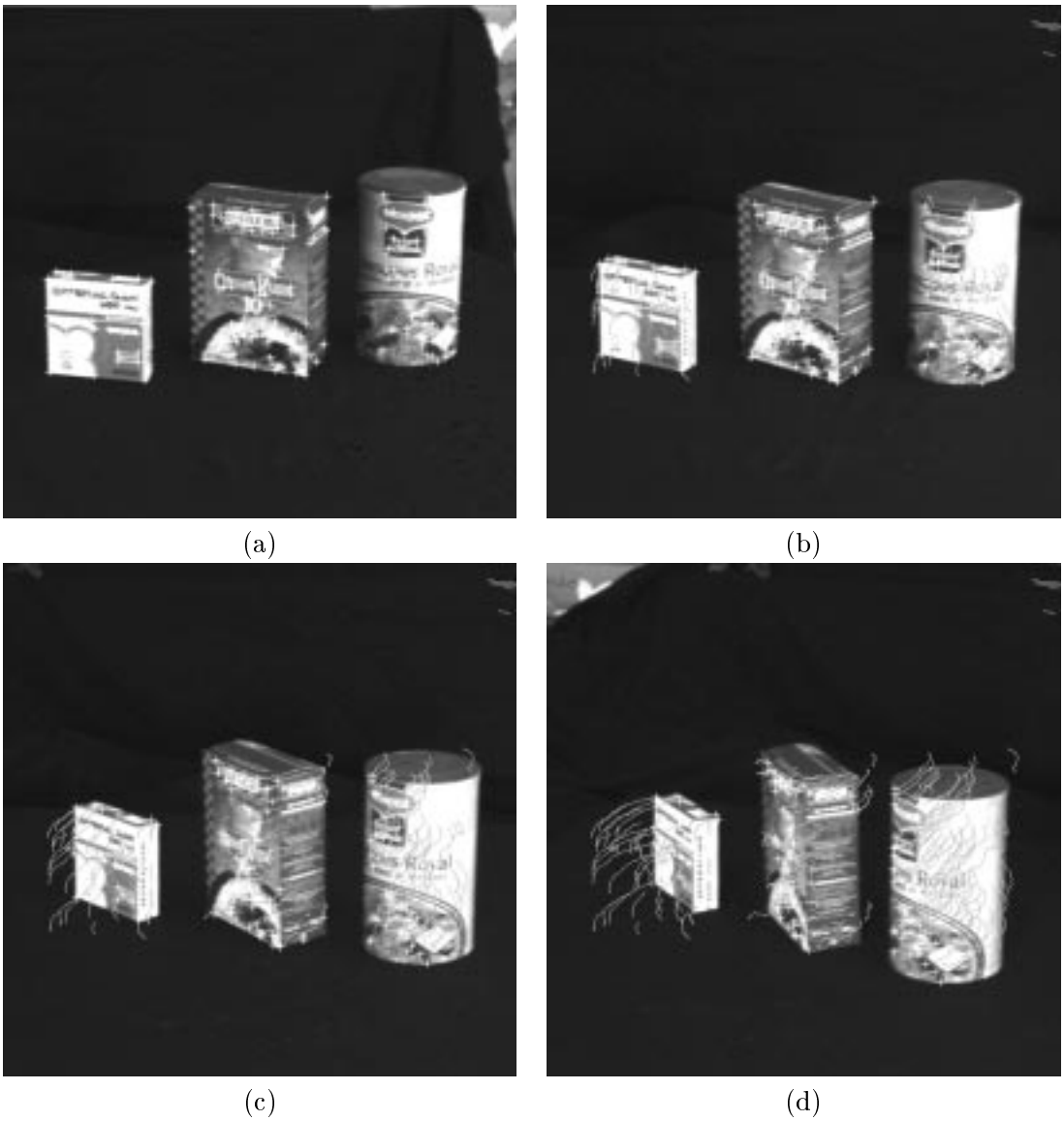
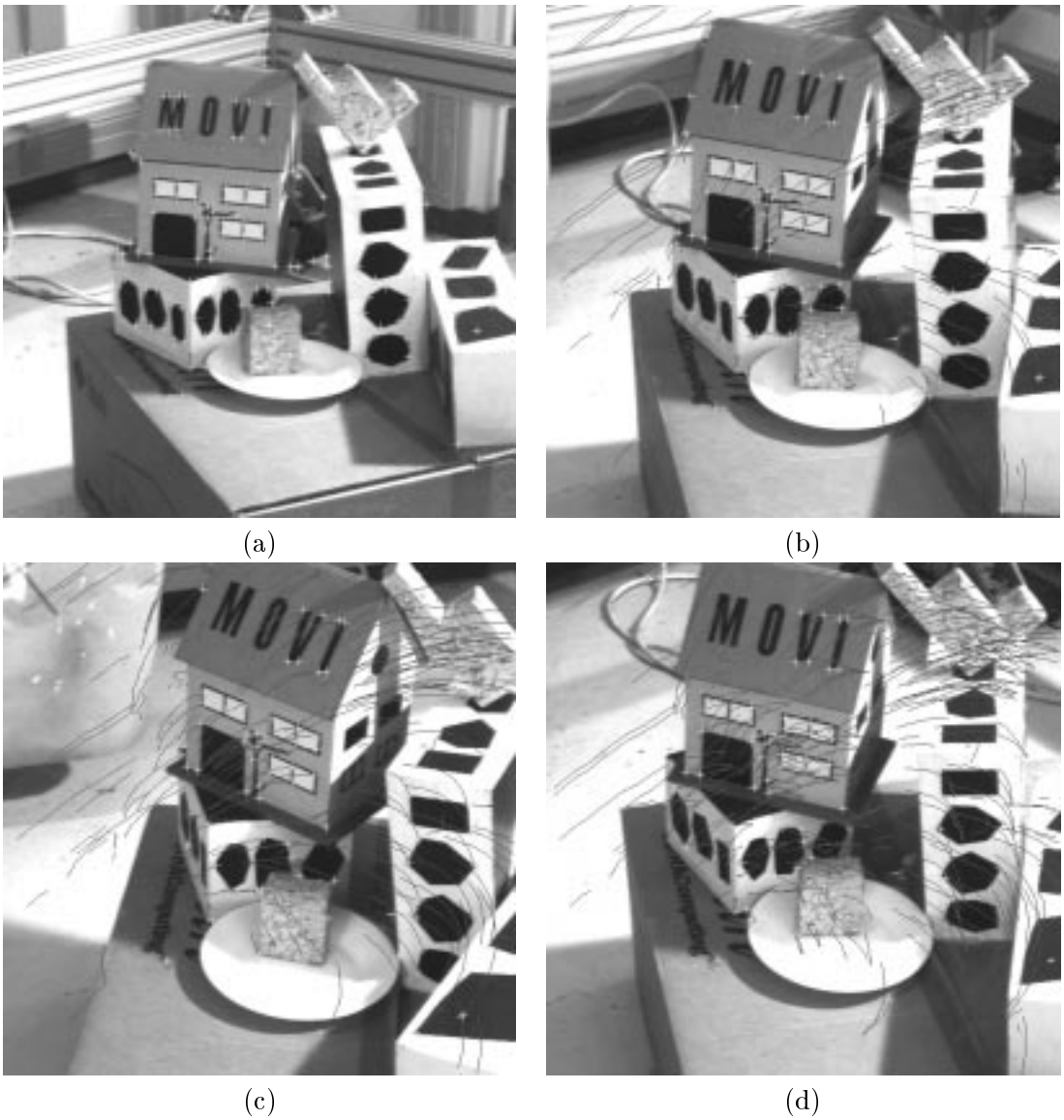


FIG. 1.13: *Séquence monoculaire 4 : (a) 228 points extraits, 1re image, (b) 194 points, 20e image, (c) 161 points, 35e image, (d) 122 points, 50e image.*



(a) (b)

(c) (d)

FIG. 1.14: *Séquence monoculaire 5 : (a) 230 points extraits, 1re image, (b) 174 points, 10e image, (c) 100 points, 20e image, (d) 88 points, 30e image.*

6. Si le nombre de points restant dans les images est inférieur à un certain seuil, refaire une extraction de points, ne conserver que les points suffisamment éloignés des points existants, puis faire une mise en correspondance sur les nouveaux points extraits.

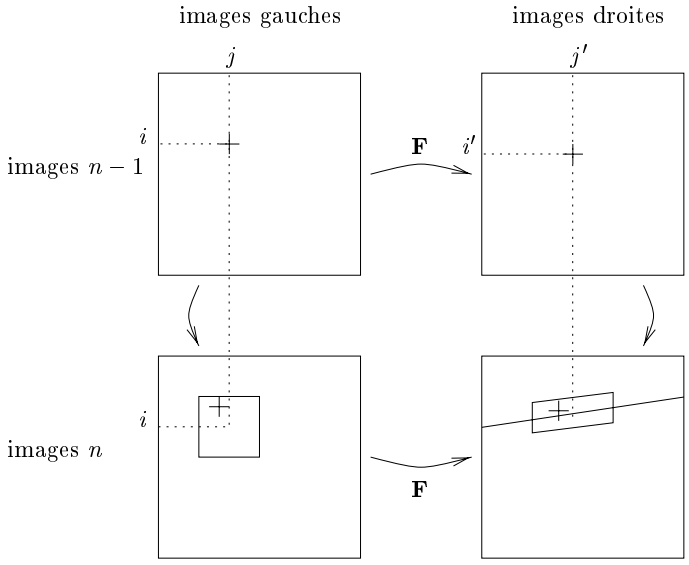


FIG. 1.15: *Principe de la poursuite de points pour des images stéréoscopiques (voir texte).*

1.2.2.2 Résultats expérimentaux

Les figures 1.16 et 1.17 illustrent les résultats obtenus pour deux séquences d'images stéréoscopiques :

- une séquence de 18 paires d'images ;
- une séquence de 25 paires d'images.

L'utilisation de la contrainte épipolaire permet d'éliminer la plupart des mauvais appariements prédits dans l'image de gauche.

1.3 Conclusion

Nous avons présenté un algorithme permettant d'effectuer la poursuite de points caractéristiques sur une séquence d'images, sans faire d'hypothèse sur le type de mouvement, et rapide en temps de calcul.

L'algorithme génère peu de mauvaises correspondances entre deux images successives : la fenêtre de recherche est réduite, les mauvaises correspondances (rares) sont en général dues à des motifs répétitifs dans l'image ou à des occultations. L'utilisation d'une corrélation inverse (vérification croisée) permet de valider les appariements et rejette de manière efficace les mauvais appariements.

Les mesures de corrélation utilisées, SAD ou SSD, sont très rapides par rapport aux autres mesures de corrélation (l'utilisation d'un tableau précalculé permet d'améliorer le

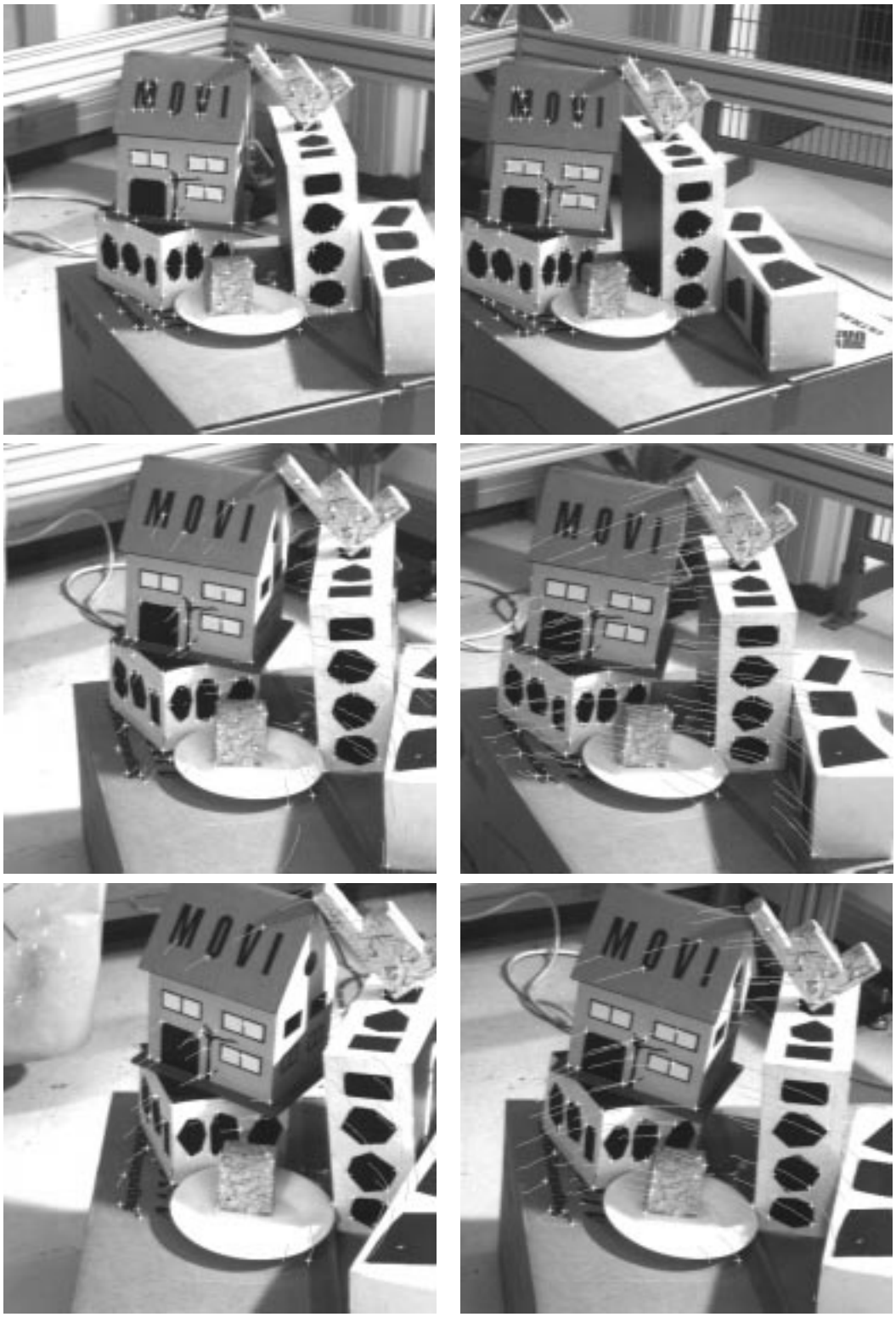


FIG. 1.16: *Séquence stéréoscopique 1 (corrélation* SSD*, fenêtre de recherche* ± 30 *pixels) – 1re ligne : 176 points extraits et mis en correspondance, 1re paire d'images – 2e ligne : 98 points suivis, 10e paire d'images – 3e ligne : 72 points suivis, 18e paire d'images.*

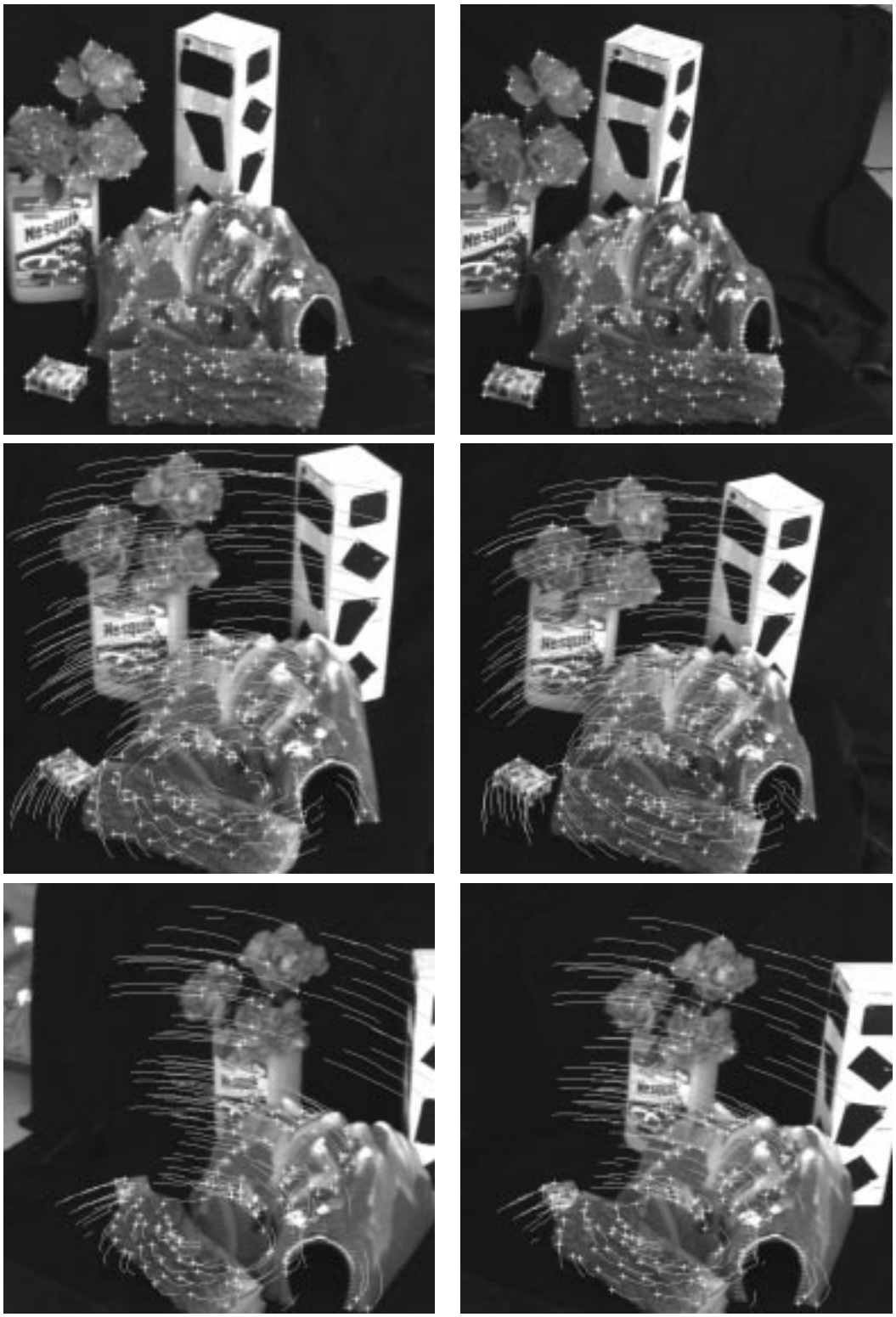


FIG. 1.17: *Séquence stéréoscopique 2 (corrélation* SSD*, fenêtre de recherche* ± 25 *pixels) – 1re ligne : 260 points extraits et mis en correspondance, 1re paire d'images – 2e ligne : 197 points suivis, 13e paire d'images – 3e ligne : 145 points suivis, 25e paire d'images.*

temps de calcul de SSD, même si ce gain est minime). Le choix de ces mesures est justifié par le fait que le changement d'éclairage est faible entre deux images consécutives, les deux images étant proches l'une de l'autre.

L'obtention d'une précision sous-pixellique est peu coûteuse en temps de calcul (multiplication par une matrice pseudo-inverse précalculée). Le fait d'avoir une précision sous-pixellique permet d'éviter d'avoir une déviation trop importante sur la localisation des points lorsque le nombre d'images augmente.

La méthode a été étendue au cas d'une séquence d'images stéréoscopiques. L'utilisation de la géométrie épipolaire permet d'accélérer la mise en correspondance et d'éliminer la plupart des faux appariements.

Chapitre 2

Caméra : le lien entre les modèles perspectif et affines

"La géométrie est la seule des «sciences humaines» à produire des démonstrations infaillibles fondées sur des définitions nominales."

BLAISE PASCAL, *Œuvres complètes*, 1963.

2.1 Modèles de caméra

2.1.1 Modèle perspectif de caméra

Considérons un modèle perspectif de caméra (voir figure 2.1). Soient ($\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$) l'orientation et $\boldsymbol{t} = {}^t(\ t_x \quad t_y \quad t_z\ )$ la translation de la caméra par rapport à l'objet. Ainsi, la matrice :

$$\mathbf{T} = \begin{pmatrix} {}^t\boldsymbol{i} & t_x \\ {}^t\boldsymbol{j} & t_y \\ {}^t\boldsymbol{k} & t_z \\ {}^t\boldsymbol{0} & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \boldsymbol{t} \\ {}^t\boldsymbol{0} & 1 \end{pmatrix}_{4\times 4}$$

représente la matrice de passage du repère objet au repère caméra.

Dans la suite de ce chapitre, nous supposerons que les paramètres intrinsèques de la caméra sont connus. Pour un point m_j appartenant à l'image, considérons les équations mathématiques liant les coordonnées caméra (x_j, y_j) et les coordonnées image en

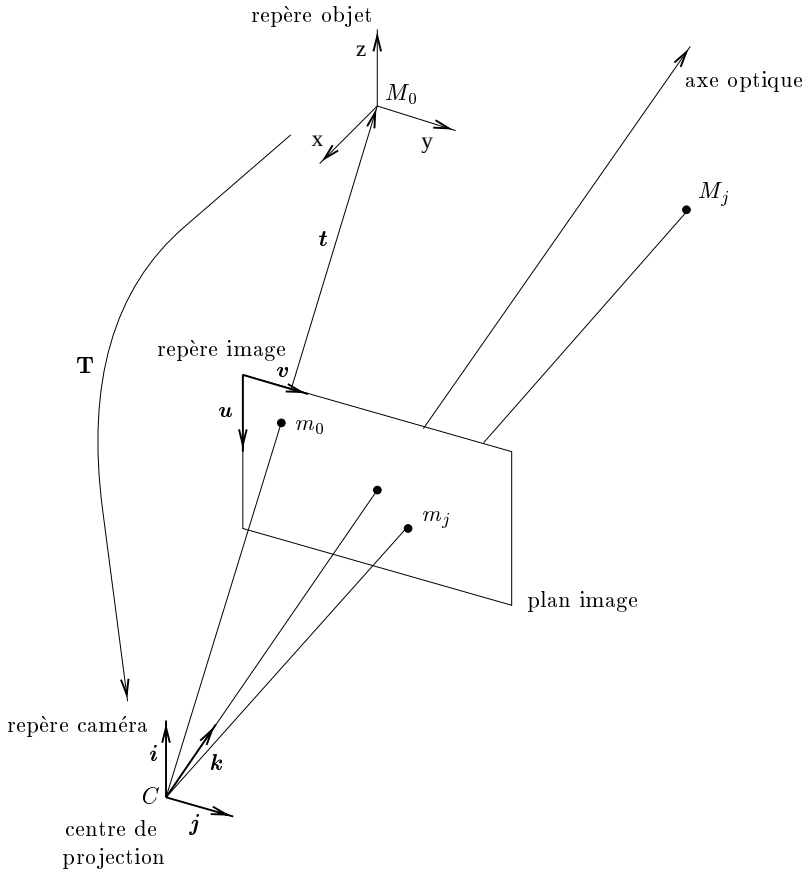


FIG. 2.1: *Cette figure représente la modélisation géométrique d'une caméra. Un point de l'objet M_0 (éventuellement point virtuel) est choisi comme étant l'origine du repère objet. Le problème consiste à calculer l'orientation $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ et la position de la caméra par rapport au repère de l'objet.*

pixels (u_j, v_j) :

$$x_j = \frac{u_j - u_c}{\alpha_u} \tag{2.1}$$

$$y_j = \frac{v_j - v_c}{\alpha_v} \tag{2.2}$$

Dans ces équations, α_u et α_v sont les facteurs d'échelle vertical et horizontal, et u_c et v_c sont les coordonnées du centre de l'image. Nous montrerons par la suite que la qualité de la pose ou de la reconstruction obtenues par la méthode proposée ne dépend fortement que du rapport α_u/α_v, et qu'elle est très peu sensible aux erreurs sur u_c et v_c.

Considérons maintenant un point M_j appartenant de l'espace, et ayant pour coordonnées euclidiennes X_j, Y_j et Z_j dans un repère lié à l'objet. Le point M_j se projette sur

l'image en un point m_j de coordonnées normalisées x_j et y_j, et vérifie :

$$\begin{pmatrix} sx_j \\ sy_j \\ s \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{R} & \boldsymbol{t} \\ {}^t\boldsymbol{0} & 1 \end{pmatrix} \begin{pmatrix} X_j \\ Y_j \\ Z_j \\ 1 \end{pmatrix} \tag{2.3}$$

$$= \begin{pmatrix} \mathbf{R} & \boldsymbol{t} \end{pmatrix} \begin{pmatrix} X_j \\ Y_j \\ Z_j \\ 1 \end{pmatrix} \tag{2.4}$$

Ainsi la matrice de projection s'écrit, pour un modèle perspectif de caméra étalonnée :

$$\mathbf{P}_p = \begin{pmatrix} \mathbf{R} & \boldsymbol{t} \end{pmatrix}_{3\times 4}$$

Introduisons les notations suivantes :

- $\boldsymbol{I} = \dfrac{\boldsymbol{i}}{t_z}$, $\boldsymbol{J} = \dfrac{\boldsymbol{j}}{t_z}$, $\boldsymbol{K} = \dfrac{\boldsymbol{k}}{t_z}$ sont respectivement la première, seconde et troisième lignes de la matrice de rotation à un facteur d'échelle près qui est la composante en z de la translation ;
- $x_0 = \dfrac{t_x}{t_z}$ et $y_0 = \dfrac{t_y}{t_z}$ sont les coordonnées caméra de m_0, projection du point M_0 – origine du repère de l'objet.

La matrice de projection étant définie à un facteur d'échelle près, nous pouvons écrire :

$$\mathbf{P}_p = \begin{pmatrix} {}^t\boldsymbol{I} & x_0 \\ {}^t\boldsymbol{J} & y_0 \\ {}^t\boldsymbol{K} & 1 \end{pmatrix}$$

Exprimons maintenant les équations liant un point 3D de l'objet et sa projection dans l'image ($\boldsymbol{M}_j$ est le vecteur allant du point M_0 au point M_j) :

$$\begin{pmatrix} s\boldsymbol{m}_j \\ s \end{pmatrix} = \begin{pmatrix} {}^t\boldsymbol{I} & x_0 \\ {}^t\boldsymbol{J} & y_0 \\ {}^t\boldsymbol{K} & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{M}_j \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} \boldsymbol{I}\cdot\boldsymbol{M}_j + x_0 \\ \boldsymbol{J}\cdot\boldsymbol{M}_j + y_0 \\ \boldsymbol{K}\cdot\boldsymbol{M}_j + 1 \end{pmatrix}$$

Nous en déduisons :

$$x_j = \frac{\boldsymbol{I}\cdot\boldsymbol{M}_j + x_0}{1+\varepsilon_j} \tag{2.5}$$

$$y_j = \frac{\boldsymbol{J}\cdot\boldsymbol{M}_j + y_0}{1+\varepsilon_j} \tag{2.6}$$

où

$$\varepsilon_j = \boldsymbol{K}\cdot\boldsymbol{M}_j = \frac{\boldsymbol{k}\cdot\boldsymbol{M}_j}{t_z} \tag{2.7}$$

Nous pouvons également écrire ces équations sous la même forme que Dementhon et Davis [Dem 92b, Dem 95] :

$$x_j(1+\varepsilon_j) - x_0 = \boldsymbol{I} \cdot \boldsymbol{M}_j \quad (2.8)$$
$$y_j(1+\varepsilon_j) - y_0 = \boldsymbol{J} \cdot \boldsymbol{M}_j \quad (2.9)$$

Si l'objet est loin de la caméra, alors les ε_j sont petits par rapport à 1. Nous pouvons donc introduire deux approximations des équations perspectives : les modèles perspectif faible et paraperspectif.

2.1.2 Modèle perspectif faible ou orthographique à l'échelle de caméra

Le modèle perspectif faible[1] suppose que les points de l'objet appartiennent à un plan parallèle au plan image et passant par l'origine du repère M_0. Cela revient à effectuer une approximation à l'ordre 0 :

$$\forall j \in \{1, \ldots, n\}, \ \frac{1}{1+\varepsilon_j} \approx 1$$

Avec cette approximation, les équations (2.5) et (2.6) deviennent :

$$x_j^{pf} - x_0 = \boldsymbol{I} \cdot \boldsymbol{M}_j \quad (2.10)$$
$$y_j^{pf} - y_0 = \boldsymbol{J} \cdot \boldsymbol{M}_j \quad (2.11)$$

Dans ces équations, x_j^{pf} et y_j^{pf} sont les coordonnées normalisées de la projection du point m_j selon un modèle perspectif faible. Par identification avec les équations (2.8) et (2.9), nous pouvons faire le lien entre les projections perspective et orthographique à l'échelle du point M_j :

$$x_j^{pf} = x_j(1+\varepsilon_j) \quad (2.12)$$
$$y_j^{pf} = y_j(1+\varepsilon_j) \quad (2.13)$$

Ces équations nous permettent de calculer l'erreur faite par l'approximation perspective faible par rapport à la projection perspective. Ainsi,

$$\Delta x^{pf} = |x_j^{pf} - x_j| = |x_j \varepsilon_j| \quad (2.14)$$
$$\Delta y^{pf} = |y_j^{pf} - y_j| = |y_j \varepsilon_j| \quad (2.15)$$

Ainsi, la qualité de l'approximation dépend à la fois de la valeur de ε_j et de la position du point dans l'image. Considérons par exemple une image de taille 512×512, et les paramètres intrinsèques suivants : $\alpha_u = \alpha_v = 1000$, et $u_c = v_c = 256$. Par suite, en utilisant les équations (2.1) et (2.2), nous avons :

$$0 \leq |x_j| \, ; \, |y_j| \leq 0,25$$

La borne inférieure correspond au point d'intersection de l'axe optique avec le plan image ($x_j = y_j = 0$), et la borne supérieure correspond aux bords de l'image :

$$x_j = \frac{u_j - u_c}{\alpha_u} = \frac{512 - 256}{1000} = 0,25$$

1. On dira indifféremment perspectif faible ou orthographique à l'échelle.

En conclusion, lorsque le rapport distance objet-caméra/taille de l'objet est petit, c'est-à-dire pour des grandes valeurs de ε_j, le modèle perspectif reste valide à condition que l'objet reste au voisinage de l'axe optique.

2.1.3 Modèle paraperspectif de caméra

Le modèle paraperspectif peut être vu comme une approximation à l'ordre 1 du modèle perspectif. Avec l'approximation :

$$\forall j \in \{1, \ldots, n\},\ \frac{1}{1+\varepsilon_j} \approx 1 - \varepsilon_j$$

on peut exprimer la projection paraperspective du point M_j :

$$\begin{aligned} x_j^{pp} &= (\boldsymbol{I} \cdot \boldsymbol{M}_j + x_0)(1 - \varepsilon_j) \\ &= (\boldsymbol{I} \cdot \boldsymbol{M}_j + x_0)(1 - \boldsymbol{K} \cdot \boldsymbol{M}_j) \\ &= \boldsymbol{I} \cdot \boldsymbol{M}_j + x_0 - x_0\ \boldsymbol{K} \cdot \boldsymbol{M}_j \\ &= (\boldsymbol{I} - x_0\ \boldsymbol{K}) \cdot \boldsymbol{M}_j + x_0 \end{aligned}$$

où le terme en $1/t_z^2$ a été négligé. On peut faire de même pour y_j^{pp}.

Finalement, les équations paraperspectives sont les suivantes :

$$x_j^{pp} - x_0 = \boldsymbol{I}_p \cdot \boldsymbol{M}_j \tag{2.16}$$

$$y_j^{pp} - y_0 = \boldsymbol{J}_p \cdot \boldsymbol{M}_j \tag{2.17}$$

où l'on a posé :

$$\boldsymbol{I}_p = \boldsymbol{I} - x_0\ \boldsymbol{K} = \frac{\boldsymbol{i} - x_0\ \boldsymbol{k}}{t_z} \tag{2.18}$$

$$\boldsymbol{J}_p = \boldsymbol{J} - y_0\ \boldsymbol{K} = \frac{\boldsymbol{j} - y_0\ \boldsymbol{k}}{t_z} \tag{2.19}$$

Comme précédemment, en identifiant les équations (2.16) et (2.17) avec les équations (2.8) et (2.9), on obtient la relation entre les projections paraperspective et perspective du point M_j :

$$x_j^{pp} = x_j(1 + \varepsilon_j) - x_0\varepsilon_j \tag{2.20}$$

$$y_j^{pp} = y_j(1 + \varepsilon_j) - y_0\varepsilon_j \tag{2.21}$$

Comme dans le paragraphe précédent, on peut facilement calculer l'erreur entre les projections paraperspective et perspective :

$$\Delta x^{pp} = |x_j^{pp} - x_j| = |(x_j - x_0)\varepsilon_j| \tag{2.22}$$

$$\Delta y^{pp} = |y_j^{pp} - y_j| = |(y_j - y_0)\varepsilon_j| \tag{2.23}$$

Lorsqu'un point M_j de l'objet est éloigné de l'axe optique, le modèle perspectif faible est une mauvaise approximation. Cependant, l'utilisation du modèle paraperspectif et un choix judicieux de l'origine – c'est-à-dire M_0 – permettent de rendre l'erreur petite. Dans ce cas, le modèle paraperspectif est une bonne approximation du modèle perspectif, même si les ε_j sont grands.

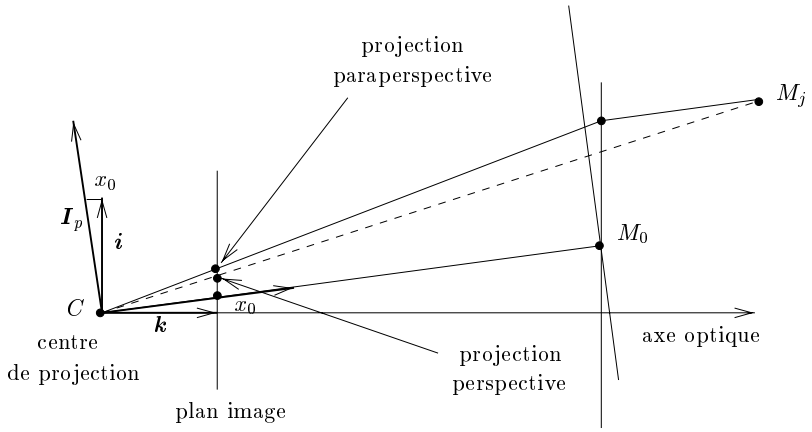


FIG. 2.2: *Cette figure montre le principe de projection d'un modèle paraperspectif de caméra. Nous considérons le plan passant par M_0 et parallèle au plan image. Un point* 3D *M_j se projette d'abord dans ce plan suivant la direction $\boldsymbol{CM}_0$, et se projette ensuite dans l'image suivant une droite passant par C. Remarquons que les deux vecteurs $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$ sont perpendiculaires à la direction de projection $\boldsymbol{CM}_0$ (seul $\boldsymbol{I}_p$ est représenté sur la figure).*

2.2 D'un modèle affine de caméra au modèle perspectif

2.2.1 Du modèle orthographique à l'échelle au modèle perspectif

Afin de résoudre le problème du calcul de pose [2], Dementhon et Davis [Dem 93, Dem 95] ont remarqué que les équations (2.10) et (2.11) sont similaires aux équations (2.8) et (2.9) lorsque les ε_j sont égaux à 0. Ils en déduisent :

- lorsque les ε_j sont fixés (pas nécessairement nuls), les équations de pose (2.8) et (2.9) deviennent linéaires en $\boldsymbol{I}$ et $\boldsymbol{J}$. Une solution peut être obtenue si l'on a au moins 4 points de l'objet non coplanaires (voir paragraphe 3.2) ;

- il est possible de résoudre les équations (2.8) et (2.9) de manière itérative par approximations linéaires successives.

Dans ce cas, l'algorithme proposé dans [Dem 93, Dem 95] effectue une initialisation avec la pose calculée avec un modèle orthographique à l'échelle. Cette pose approximative est ensuite améliorée de la manière suivante.

1. Pour tout j, $j \in \{1, \ldots, n\}$, $n \geq 3$, initialiser $\varepsilon_j = 0$.

2. Résoudre le système linéaire surcontraint formé par les équations (2.8) et (2.9). La solution fournit une estimation des vecteurs $\boldsymbol{I}$ et $\boldsymbol{J}$ (paragraphe 3.2).

2. Le calcul de pose consiste à déterminer la position et l'orientation d'une caméra par rapport à un objet, à partir d'une seule image et d'un modèle 3D de l'objet (cf. chapitre 3).

3. Calculer la position et l'orientation de la caméra par rapport à l'objet :

$$\begin{aligned} t_z &= \frac{1}{2}\left(\frac{1}{\|\boldsymbol{I}\|}+\frac{1}{\|\boldsymbol{J}\|}\right) & \boldsymbol{i} &= \frac{\boldsymbol{I}}{\|\boldsymbol{I}\|} \\ t_x &= x_0 t_z & \boldsymbol{j} &= \frac{\boldsymbol{J}}{\|\boldsymbol{J}\|} \\ t_y &= y_0 t_z & \boldsymbol{k} &= \boldsymbol{i}\wedge\boldsymbol{j} \end{aligned}$$

4. Pour tout j, calculer :

$$\varepsilon_j = \frac{\boldsymbol{k}\cdot\boldsymbol{M}_j}{t_z}$$

Si les ε_j calculés à cette itération sont égaux à ceux de l'itération précédente, alors arrêter l'algorithme, sinon retourner à l'étape 2.

Le comportement de l'algorithme itératif est illustré sur la figure 2.3. À la première itération, l'algorithme considère les *vraies* projections perspectives des points M_j, et tente d'effectuer un calcul de pose en supposant que les points image ont été projetés suivant un modèle orthographique à l'échelle de caméra (un raisonnement semblable peut être suivi pour un modèle paraperspectif). À la seconde itération, l'algorithme considère les positions modifiées des points image (en tenant compte des valeurs estimées pour les ε_j). À la dernière itération, les positions des points image ont été modifiées de sorte qu'elles correspondent à des projections suivant un modèle orthographique à l'échelle. Les positions relatives des projections perspectives et orthographiques à l'échelle vérifient les relations théoriques données par les équations (2.12) et (2.13). Remarquons que la projection perspective du point M_0 est identique à la projection obtenue avec un modèle orthographique à l'échelle (ou paraperspectif), donc la projection de ce point n'est pas modifiée.

Les équations (2.8) et (2.9) peuvent être interprétées de deux manières différentes :

- on peut considérer x_j et y_j comme étant la projection perspective du point M_j ;
- on peut considérer $x_j(1+\varepsilon_j)$ et $y_j(1+\varepsilon_j)$ comme étant la projection orthographique à l'échelle de M_j.

2.2.2 Du modèle paraperspectif au modèle perspectif

Dans ce paragraphe, nous étendons l'algorithme précédent pour un modèle paraperspectif de caméra. Considérons de nouveau les équations (2.8) et (2.9) :

$$\begin{aligned} x_j(1+\varepsilon_j) - x_0 &= \boldsymbol{I}\cdot\boldsymbol{M}_j \\ y_j(1+\varepsilon_j) - y_0 &= \boldsymbol{J}\cdot\boldsymbol{M}_j \end{aligned}$$

et remplaçons $\boldsymbol{I}$ et $\boldsymbol{J}$ par les expressions données par les équations (2.18) et (2.19) :

$$\begin{aligned} \boldsymbol{I} &= \boldsymbol{I}_p + x_0\,\boldsymbol{K}_j \\ \boldsymbol{J} &= \boldsymbol{J}_p + y_0\,\boldsymbol{K}_j \\ \varepsilon_j &= \boldsymbol{K}\cdot\boldsymbol{M}_j \end{aligned}$$

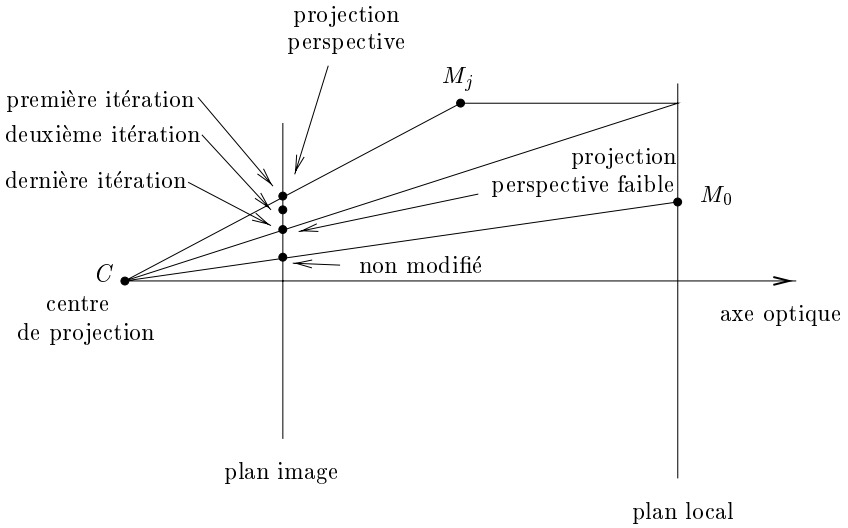


FIG. 2.3: *Le principe de la méthode consiste à modifier les coordonnées des points image correspondant à une projection perspective en des points correspondant à une projection perspective faible (ici sur la figure) ou paraperspective.*

Nous obtenons :

$$(x_j - x_0)(1 + \varepsilon_j) = \boldsymbol{I}_p \cdot \boldsymbol{M}_j \quad (2.24)$$

$$(y_j - y_0)(1 + \varepsilon_j) = \boldsymbol{J}_p \cdot \boldsymbol{M}_j \quad (2.25)$$

Remarquons que lorsque les ε_j sont nuls, les équations perspectives ci-dessus, (2.24) et (2.25), deviennent identiques aux équations du modèle paraperspectif, (2.16) et (2.17). Les équations (2.24) et (2.25) peuvent être interprétées de deux manières différentes :

- on peut considérer x_j et y_j comme étant la projection perspective du point M_j ;
- on peut considérer $x_j(1 + \varepsilon_j) - x_{0_i}\varepsilon_j$ et $y_j(1 + \varepsilon_j) - y_{0_i}\varepsilon_j$ comme étant la projection paraperspective de M_j.

Comme dans le cas du modèle orthographique à l'échelle, nous avons une méthode linéaire pour calculer la pose avec un modèle paraperspectif, et une méthode itérative pour calculer la pose avec un modèle perspectif par approximations successives par un modèle paraperspectif.

1. Pour tout j, $j \in \{1, \ldots, n\}$, $n \geq 3$, initialiser $\varepsilon_j = 0$.
2. Résoudre le système linéaire surcontraint formé par les équations (2.24) et (2.25). La solution fournit une estimation des vecteurs $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$ (paragraphe 3.2).
3. Calculer la position (t_x, t_y, et t_z) et l'orientation ($\boldsymbol{i}$, $\boldsymbol{j}$, et $\boldsymbol{k}$) de la caméra par rapport à l'objet.
4. Pour tout j, réestimer les ε_j. Si les ε_j calculés à cette itération sont égaux à ceux de l'itération précédente, alors arrêter l'algorithme, sinon retourner à l'étape 2.

Regardons plus en détails comment calculer la position t_x, t_y, t_z, et l'orientation $\boldsymbol{i}$, $\boldsymbol{j}$ et $\boldsymbol{k}$ à partir de $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$.

Déterminons d'abord la position de la caméra. Nous remarquons que :

$$\begin{aligned} \|\boldsymbol{I}_p\|^2 &= \frac{(\boldsymbol{i} - x_0\,\boldsymbol{k}) \cdot (\boldsymbol{i} - x_0\,\boldsymbol{k})}{t_z^2} \\ &= \frac{1 + x_0^2}{t_z^2} \end{aligned}$$

et

$$\|\boldsymbol{J}_p\|^2 = \frac{1 + y_0^2}{t_z^2}$$

Nous obtenons ainsi :

$$t_z = \frac{1}{2}\left(\frac{\sqrt{1 + x_0^2}}{\|\boldsymbol{I}_p\|} + \frac{\sqrt{1 + y_0^2}}{\|\boldsymbol{J}_p\|}\right) \tag{2.26}$$

et

$$\begin{aligned} t_x &= x_0 t_z \\ t_y &= y_0 t_z \end{aligned}$$

Dans un second temps, nous calculons les vecteurs $\boldsymbol{i}$, $\boldsymbol{j}$ et $\boldsymbol{k}$ de la façon suivante. En théorie, les trois vecteurs sont deux à deux orthogonaux, mais en pratique, les calculs ne garantissent pas qu'ils le soient réellement. La méthode proposée n'assure pas que $\boldsymbol{i}$ et $\boldsymbol{j}$ soient orthogonaux, mais assure que le troisième vecteur $\boldsymbol{k}$ est orthogonal aux deux premiers. Les équations (2.18) et (2.19) s'écrivent :

$$\boldsymbol{i} = t_z\,\boldsymbol{I}_p + x_0\,\boldsymbol{k} \tag{2.27}$$

$$\boldsymbol{j} = t_z\,\boldsymbol{J}_p + y_0\,\boldsymbol{k} \tag{2.28}$$

Le troisième vecteur, $\boldsymbol{k}$, est le produit vectoriel des deux premiers :

$$\begin{aligned} \boldsymbol{k} &= \boldsymbol{i} \wedge \boldsymbol{j} \\ &= t_z^2\,\boldsymbol{I}_p \wedge \boldsymbol{J}_p + t_z y_0\,\boldsymbol{I}_p \wedge \boldsymbol{k} - t_z x_0\,\boldsymbol{J}_p \wedge \boldsymbol{k} \end{aligned}$$

Soit $[\boldsymbol{u}]_\times$ la matrice antisymétrique associée au vecteur $\boldsymbol{u}$ de taille 3. Si $\boldsymbol{u} = {}^t(\ u_x \quad u_y \quad u_z\ )$, alors :

$$[\boldsymbol{u}]_\times = \begin{pmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{pmatrix}$$

et le produit vectoriel $\boldsymbol{u} \wedge \boldsymbol{v}$ peut s'écrire $[\boldsymbol{u}]_\times \boldsymbol{v}$. Posons également $\boldsymbol{\mathcal{I}}_3$ la matrice identité de taille 3×3. L'équation précédente s'écrit donc :

$$\underbrace{(\boldsymbol{\mathcal{I}}_3 - t_z y_0\,[\boldsymbol{I}_p]_\times + t_z x_0\,[\boldsymbol{J}_p]_\times)}_{\mathbf{B}}\,\boldsymbol{k} = t_z^2\,\boldsymbol{I}_p \wedge \boldsymbol{J}_p \tag{2.29}$$

Cette équation nous permet de calculer le vecteur $\boldsymbol{k}$ à condition que le système linéaire précédent soit de rang plein. En effet, nous pouvons remarquer que la matrice $\mathbf{B}$ est de la forme :

$$\mathbf{B} = \begin{pmatrix} 1 & c & -b \\ -c & 1 & a \\ b & -a & 1 \end{pmatrix}$$

Son déterminant est strictement positif:

$$\det(\mathbf{B}) = 1 + a^2 + b^2 + c^2$$

Par conséquent, la matrice $\mathbf{B}$ est de rang 3 et le vecteur $\boldsymbol{k}$ peut être calculé à partir de l'équation (2.29). Les vecteurs $\boldsymbol{i}$ et $\boldsymbol{j}$ peuvent ensuite être estimés à partir des équations (2.27) et (2.28). Il est donc toujours possible de calculer la pose, c'est-à-dire $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$, t_x, t_y, t_z, et d'estimer les ε_j pour chaque image et pour chaque point (équation (2.7)). Il est alors facile de calculer les ε_j.

2.3 Une approche unifiée pour les problèmes de calcul de pose et reconstruction

Le problème du calcul de pose et le problème de reconstruction font intervenir les mêmes équations (2.8) et (2.25), sous deux hypothèses différentes :

- le calcul de pose est effectué à partir d'une seule image, le modèle 3D de l'objet étant connu (i.e. les vecteurs $\boldsymbol{M}_j$ sont donnés) ;
- la reconstruction est obtenue à partir d'une séquence d'images, seuls les points images $(x_{ij},\, y_{ij})$ étant connus.

Réécrivons les équations (2.8) et (2.9) faisant le lien entre le modèle orthographique à l'échelle et perspectif, pour un ensemble d'images (l'indice i représente la i-ième image, et l'indice j représente le j-ième point) :

$$x_{ij}(1+\varepsilon_{ij}) - x_{0_i} = \boldsymbol{I}_i \cdot \boldsymbol{M}_j \tag{2.30}$$
$$y_{ij}(1+\varepsilon_{ij}) - y_{0_i} = \boldsymbol{J}_i \cdot \boldsymbol{M}_j \tag{2.31}$$

où

$$\boldsymbol{I}_i = \frac{\boldsymbol{i}_i}{t_{z_i}} \qquad \boldsymbol{J}_i = \frac{\boldsymbol{j}_i}{t_{z_i}} \qquad \boldsymbol{K}_i = \frac{\boldsymbol{k}_i}{t_{z_i}} \qquad x_{0_i} = \frac{t_{x_i}}{t_z} \qquad y_{0_i} = \frac{t_{y_i}}{t_{z_i}}$$

et les équations (2.24) et (2.25) faisant le lien entre le modèle paraperspectif et perspectif, pour un ensemble d'images :

$$(x_{ij} - x_{0_i})(1+\varepsilon_{ij}) = \boldsymbol{I}_{p_i} \cdot \boldsymbol{M}_j \tag{2.32}$$
$$(y_{ij} - y_{0_i})(1+\varepsilon_{ij}) = \boldsymbol{J}_{p_i} \cdot \boldsymbol{M}_j \tag{2.33}$$

où

$$\boldsymbol{I}_{p_i} = \boldsymbol{I}_i - x_{0_i}\boldsymbol{K}_i = \frac{\boldsymbol{i}_i - x_{0_i}\,\boldsymbol{k}_i}{t_{z_i}}$$
$$\boldsymbol{J}_{p_i} = \boldsymbol{J}_i - y_{0_i}\boldsymbol{K}_i = \frac{\boldsymbol{j}_i - y_{0_i}\,\boldsymbol{k}_i}{t_{z_i}}$$

Nous proposons un algorithme itératif permettant de calculer la pose (à partir d'une seule image) ou d'effectuer une reconstruction euclidienne (pour une séquence d'images), en utilisant un modèle perspectif de caméra. À chaque pas d'itération, on utilise un modèle affine, mais l'algorithme converge à la fin vers la solution obtenue avec un modèle perspectif de caméra. L'algorithme est le suivant.

1. Initialiser tous les ε_j (resp. ε_{ij}) à zéro.

2. Effectuer un calcul de pose ou une reconstruction euclidienne (suivant le cas) avec un modèle affine de caméra (orthographique à l'échelle ou paraperspectif). Plus précisément, l'enchaînement des étapes est le suivant (nous reprendrons plus en détails chacune des étapes aux chapitres 3, 4, et 5) :

   - *Pour le calcul de pose* : (i) estimer la translation dans l'image (x_0, y_0), (ii) calculer les vecteurs $\boldsymbol{I}$ et $\boldsymbol{J}$ (ou $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$), (iii) en déduire la translation dans l'espace (t_x, t_y, t_z), puis l'orientation $(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k})$, (iv) orthogonaliser la matrice de rotation.
   - *Pour la reconstruction* : (i) estimer la translation dans chaque image (x_{0_i}, y_{0_i}), (ii) effectuer une reconstruction affine puis le passage à une reconstruction euclidienne en utilisant les contraintes sur les vecteurs $\boldsymbol{I}_i$ et $\boldsymbol{J}_i$ (ou $\boldsymbol{I}_{p_i}$ et $\boldsymbol{J}_{p_i}$), (iii) en déduire les coordonnées euclidiennes des points 3D ($\boldsymbol{M}_j$), les translations dans l'espace $(t_{x_i}, t_{y_i}, t_{z_i})$ et les orientations $(\boldsymbol{i}_i, \boldsymbol{j}_i, \boldsymbol{k}_i)$ pour chaque image.

3. Réestimer les ε_j (resp. les ε_{ij}) :

$$\varepsilon_j = \frac{\boldsymbol{k} \cdot \boldsymbol{M}_j}{t_z} \quad \left(\text{resp. } \varepsilon_{ij} = \frac{\boldsymbol{k}_i \cdot \boldsymbol{M}_j}{t_{z_i}} \right)$$

   Si les epsilons calculés à cette itération sont égaux à ceux de l'itération précédente, alors arrêter l'algorithme, sinon retourner à l'étape 2.

Chapitre 3

Calcul de pose à partir de correspondances de points

"La vérité n'est pas seulement une connaissance, mais la vérification d'une connaissance par la pratique."

William James, *Le Pragmatisme*, 1968.

3.1 État de l'art

Le calcul de pose consiste à déterminer la position et l'orientation d'une caméra par rapport à un objet, c'est-à-dire les paramètres extrinsèques de la caméra. Dans ce chapitre nous supposerons donnés un modèle 3D d'un objet – modélisé par un ensemble de points ou de droites – ainsi qu'une image dans laquelle cet objet est visible.

Le problème du calcul de pose est un problème très actif dans le domaine de la vision par ordinateur et de la photogrammétrie, et a fait l'objet d'un grand nombre de publications. Haralick et al. [Har 89] citent une dissertation allemande de 1958 qui recense près de 80 solutions différentes de la littérature photogrammétrique qui ont été développées depuis 1841. Les différentes approches pour résoudre le problème du calcul de pose peuvent être classées en deux catégories distinctes : les solutions exactes et les solutions numériques (approchées).

Les solutions exactes ne peuvent être appliquées que pour un nombre limité de primitives mises en correspondance (Dhome et al. [Dho 89], Fischler et Bolles [Fis 81], Horaud et al. [Hor 89], Förstner [För 87]). Cependant, le fait d'utiliser le minimum de points (soit 3 points) conduit à une ambiguïté de la solution obtenue qui n'est pas unique

[Har 89, Fis 81, För 87]. Dans ce cas, il peut y avoir jusqu'à quatre solutions différentes, et des hypothèses supplémentaires sont nécessaires pour lever cette ambiguïté. Dès lors que l'on dispose d'au moins 4 points, nous obtenons en général une solution unique sauf pour quelques configurations particulières de points dans l'espace [Tho 66, Wro 92].

Cependant, lorsque le nombre de correspondances est supérieur à 4, les solutions exactes deviennent trop compliquées et il faut alors utiliser des méthodes numériques itératives (Haralick et al. [Har 89], Lowe [Low 91], Yuan [Yua 89]). Ces méthodes sont en général très robustes, et convergent vers la solution exacte à condition d'avoir une bonne estimation de la solution exacte. Phong et al. [Pho 95] proposent une méthode qui utilise une optimisation par régions de confiance, moins sensible à l'initialisation que les autres méthodes. Cependant, la méthode proposée par Phong et al. ne donne de bons résultats lorsque le nombre de correspondances est important. Lorsque ce nombre de correspondances est compris entre 3 et 10, alors la méthode de minimisation par région de confiance demande soit un grand nombre d'itérations, soit ne converge pas vers la bonne solution. D'un point de vue pratique, il est important d'avoir un algorithme pouvant résoudre le problème de pose ne nécessitant pas un trop grand nombre de correspondances et ne souffrant pas des limitations inhérentes aux méthodes de résolution exactes.

La méthode proposée récemment par Dementhon et Davis [Dem 92b, Dem 93, Dem 95] permet d'utiliser un nombre de correspondances quelconques. Il faut au minimum 4 points, et ces points ne doivent pas être coplanaires. La méthode est rapide et robuste par rapport aux données image et à l'étalonnage de la caméra. La méthode consiste à améliorer, de manière itérative, la pose obtenue avec un modèle orthographique à l'échelle de caméra pour converger, à la fin, vers la pose obtenue avec un modèle perspectif de caméra. À notre connaissance, cette méthode est l'une des premières tentatives de faire le lien entre des techniques de résolutions linéaires (associées aux modèles affines de caméra) et le modèle perspectif. En effet, la solution obtenue avec un modèle affine de caméra est simple à calculer mais n'est qu'une approximation de la solution précise, alors que les méthodes de résolutions non linéaires donnent une solution précise mais requièrent une initialisation correcte.

Une façon de faire le lien entre le modèle de perspectif de caméra et les modèles affines consiste à utiliser la solution obtenue avec un modèle affine pour initialiser le problème de minimisation non linéaire lié au modèle perspectif de caméra. Cependant, cette approche a plusieurs inconvénients. Premièrement, cette méthode de résolution ne prend pas en compte le lien existant entre le modèle perspectif et ses approximations linéaires. Ensuite, la solution obtenue avec un modèle affine peut être éloignée de la vraie solution. Dans ce cas, un grand nombre d'itérations est nécessaire afin d'obtenir une solution stable par le processus de minimisation non linéaire.

La projection perspective peut être modélisée comme étant une transformation d'un espace projectif 3D dans un espace projectif 2D. Les modèles perspectif faible et paraperspectif sont les approximations affines les plus courantes du modèle perspectif. La méthode proposée par Dementhon et Davis [Dem 92b, Dem 93, Dem 95] commence par calculer la pose avec un modèle orthographique à l'échelle, puis converge en quelques itérations vers la pose obtenue avec un modèle perspectif. La méthode est élégante, très rapide, et précise. Elle est cependant limitée aux cas où l'approximation perspective faible est valide. Si l'objet est proche de la caméra et/ou décalé de manière importante par rapport à l'axe optique, alors l'algorithme de Dementhon et Davis ne converge pas toujours.

Dans ce chapitre, nous montrons comment étendre la méthode initiale proposée par Dementhon et Davis pour un modèle paraperspectif. Nous étudions également le cas particulier d'un ensemble de points coplanaires dont la résolution est spécifique. De plus, nous introduisons un moyen simple de prendre en compte la contrainte d'orthogonalité de la matrice de rotation 3×3 représentant l'orientation entre l'objet 3D et la caméra. En effet, un algorithme de calcul de pose linéaire ne garantit pas que cette matrice est orthogonale. La méthode d'orthogonalisation proposée utilise des quaternions unitaires. Cette étape améliore la précision de la méthode et ne nécessite que peu de calculs supplémentaires.

Plan du chapitre

L'organisation du chapitre est le suivant. Le paragraphe 3.2 explique comment résoudre le système linéaire associé au problème du calcul de pose, dans le cas d'un modèle non planaire ou planaire. Le paragraphe 3.3 explique comment améliorer les performance des algorithmes en introduisant de manière simple l'orthogonalité de la matrice de rotation représentant l'orientation entre le modèle 3D et la caméra. Le paragraphe 3.4 compare les deux méthodes l'une par rapport à l'autre, puis par rapport à une méthode de minimisation non linéaire. De nombreux exemples d'application sont également présentés.

3.2 Résolution du système linéaire

Les deux algorithmes itératifs (orthographique à l'échelle ou paraperspectif) ont besoin de résoudre un système linéaire surcontraint, formé par les équations (2.8), (2.9) (modèle orthographique à l'échelle) :

$$x_j(1+\varepsilon_j) - x_0 = \boldsymbol{I} \cdot \boldsymbol{M}_j \tag{3.1}$$
$$y_j(1+\varepsilon_j) - y_0 = \boldsymbol{J} \cdot \boldsymbol{M}_j \tag{3.2}$$

ou (2.24), (2.25) (modèle paraperspectif) :

$$(x_j - x_0)(1+\varepsilon_j) = \boldsymbol{I}_p \cdot \boldsymbol{M}_j \tag{3.3}$$
$$(y_j - y_0)(1+\varepsilon_j) = \boldsymbol{J}_p \cdot \boldsymbol{M}_j \tag{3.4}$$

Sous forme matricielle, ces équations peuvent s'écrire :

$$\underbrace{\mathbf{M}}_{n\times 3}\underbrace{\boldsymbol{I}}_{3\times 1} = \underbrace{\boldsymbol{x}}_{n\times 1} \tag{3.5}$$
$$\underbrace{\mathbf{M}}_{n\times 3}\underbrace{\boldsymbol{J}}_{3\times 1} = \underbrace{\boldsymbol{y}}_{n\times 1} \tag{3.6}$$

où $\mathbf{M}$ est une matrice $n \times 3$ formée par les coordonnées 3D des n vecteurs $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_n$:

$$\mathbf{M} = \begin{pmatrix} X_1 & Y_1 & Z_1 \\ \vdots & \vdots & \vdots \\ X_n & Y_n & Z_n \end{pmatrix}$$

Soit $\boldsymbol{1} = {}^t(\ 1 \quad \ldots \quad 1\ )$ le vecteur de dimension n composé de 1. Multiplions à gauche les équations (3.5) et (3.6) par ${}^t\boldsymbol{1}$. Si l'on exprime les coordonnées 3D des points par rapport

au centre de gravité de l'objet, nous avons ${}^t\boldsymbol{1}\mathbf{M} = {}^t\boldsymbol{0}$. Nous obtenons ainsi :

$$\begin{aligned} {}^t\boldsymbol{1}\boldsymbol{x} &= {}^t\boldsymbol{0} \\ {}^t\boldsymbol{1}\boldsymbol{y} &= {}^t\boldsymbol{0} \end{aligned}$$

Nous en déduisons les coordonnées $(x_0,\, y_0)$ de la projection du centre de gravité dans l'image, pour un modèle orthographique à l'échelle de caméra :

$$\begin{aligned} x_0 &= \frac{1}{n}\sum_{j=1}^{n} x_j(1+\varepsilon_j) \\ y_0 &= \frac{1}{n}\sum_{j=1}^{n} y_j(1+\varepsilon_j) \end{aligned}$$

et pour un modèle paraperspectif de caméra :

$$x_0 = \frac{\sum_{j=1}^{n} x_j(1+\varepsilon_j)}{\sum_{j=1}^{n}(1+\varepsilon_j)} \qquad y_0 = \frac{\sum_{j=1}^{n} y_j(1+\varepsilon_j)}{\sum_{j=1}^{n}(1+\varepsilon_j)}$$

Il nous reste ensuite à déterminer l'orientation modélisée par les vecteurs $\boldsymbol{I}$ et $\boldsymbol{J}$ (ou $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$) des équations (3.5) et (3.6). Afin de résoudre ces équations linéaires, il nous faut distinguer deux cas : la cas où le modèle 3D est non planaire et le cas où il est planaire.

3.2.1 Objet non planaire

Si les points de l'objet ne sont pas coplanaires, le rang de $\mathbf{M}$ est 3, et par conséquent les vecteurs $\boldsymbol{I}$ et $\boldsymbol{J}$ (ou de manière équivalente $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$) sont calculés simplement en calculant la matrice pseudo-inverse de $\boldsymbol{M}$:

$$\begin{aligned} \boldsymbol{I} &= \mathbf{M}^\dagger\,\boldsymbol{x} \\ \boldsymbol{J} &= \mathbf{M}^\dagger\,\boldsymbol{y} \end{aligned}$$

où $\mathbf{M}^\dagger = ({}^t\mathbf{M}\mathbf{M})^{-1}\,{}^t\mathbf{M}$ est la matrice pseudo-inverse de $\mathbf{M}$. Notons que cette matrice pseudo-inverse peut être calculée une fois pour toute. De cette manière, le calcul de $\boldsymbol{I}$ et $\boldsymbol{J}$ est particulièrement rapide.

3.2.2 Objet planaire

Lorsque les points de l'objet sont coplanaires, le rang de la matrice $\mathbf{M}$ est 2, et le système ne peut plus être résolu de la même façon que précédemment. Considérons le plan formé par les points de l'objet, et $\boldsymbol{u}$ le vecteur unitaire orthogonal à ce plan. Les vecteurs $\boldsymbol{I}$ et $\boldsymbol{J}$ (ou $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$) peuvent être décomposés comme étant la somme d'un vecteur appartenant à ce plan et d'un vecteur perpendiculaire à ce plan. Plus précisément [Dem 93, Obe 93] (voir figure 3.1) :

$$\boldsymbol{I} = \boldsymbol{I}_0 + \alpha\,\boldsymbol{u} \tag{3.7}$$

$$\boldsymbol{J} = \boldsymbol{J}_0 + \beta\,\boldsymbol{u} \tag{3.8}$$

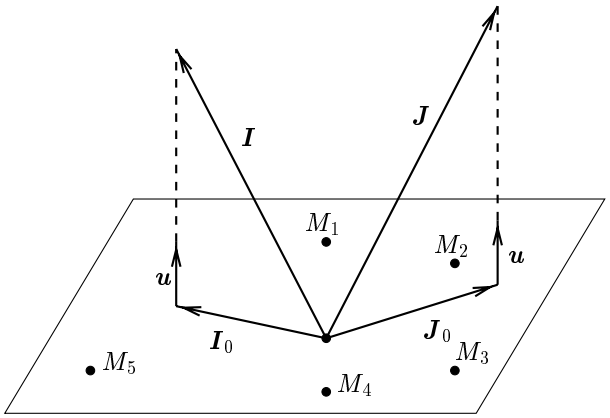


FIG. 3.1: *Cas d'un objet planaire.*

En remplaçant ces expressions de $\boldsymbol{I}$ et $\boldsymbol{J}$ dans les équations (3.5) et (3.6), nous obtenons :

$$\begin{aligned} \mathbf{M}\,\boldsymbol{I}_0 &= \boldsymbol{x} \\ \mathbf{M}\,\boldsymbol{J}_0 &= \boldsymbol{y} \end{aligned}$$

Ces équations peuvent être résolues si l'on ajoute les contraintes supplémentaires suivantes :

$$\begin{aligned} \boldsymbol{u}\cdot\boldsymbol{I}_0 &= 0 \\ \boldsymbol{u}\cdot\boldsymbol{J}_0 &= 0 \end{aligned}$$

Nous en déduisons ainsi $\boldsymbol{I}_0$ et $\boldsymbol{J}_0$:

$$\begin{aligned} \boldsymbol{I}_0 &= \mathbf{M}'^{\dagger}\begin{pmatrix} \boldsymbol{x} \\ 0 \end{pmatrix} \\ \boldsymbol{J}_0 &= \mathbf{M}'^{\dagger}\begin{pmatrix} \boldsymbol{y} \\ 0 \end{pmatrix} \end{aligned}$$

où $\mathbf{M}'$ est définie par :

$$\mathbf{M}' = \begin{pmatrix} \mathbf{M} \\ {}^t\boldsymbol{u} \end{pmatrix}$$

De manière évidente, le rang de $\mathbf{M}'$ est égal à 3. Afin d'estimer $\boldsymbol{I}$ et $\boldsymbol{J}$ (ou $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$), il nous reste à estimer les deux réels α et β.

- Dans le cas d'un modèle orthographique à l'échelle, Oberkampf, Dementhon et Davis [Obe 93] ont proposé une solution à partir des contraintes $\|\boldsymbol{I}\| = \|\boldsymbol{J}\|$ et $\boldsymbol{I}\cdot\boldsymbol{J} = 0$.
- Dans le cas d'un modèle paraperspectif, nous pouvons établir une solution de la même manière à partir des contraintes sur $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$ (déduites des équations (2.18) et (2.19)) :

$$\|\boldsymbol{I}_p\|^2 = \frac{1+x_0^2}{t_z^2}$$

$$\begin{aligned} \|\boldsymbol{J}_p\|^2 &= \frac{1+y_0^2}{t_z^2} \\ \boldsymbol{I}_p \cdot \boldsymbol{J}_p &= \frac{x_0 y_0}{t_z^2} \end{aligned}$$

En éliminant t_z, nous obtenons deux contraintes :

$$\begin{aligned} \boldsymbol{I}_p \cdot \boldsymbol{J}_p &= \frac{x_0 y_0}{1+x_0^2} \|\boldsymbol{I}_p\|^2 \\ \boldsymbol{I}_p \cdot \boldsymbol{J}_p &= \frac{x_0 y_0}{1+y_0^2} \|\boldsymbol{J}_p\|^2 \end{aligned}$$

En substituant dans ces équations les expressions de $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$ données par les équations (3.7) et (3.8), nous avons :

$$\begin{aligned} \boldsymbol{I}_0 \cdot \boldsymbol{J}_0 + \alpha\,\beta &= \frac{x_0 y_0}{1+x_0^2} (\|\boldsymbol{I}_0\|^2 + \alpha^2) \\ \boldsymbol{I}_0 \cdot \boldsymbol{J}_0 + \alpha\,\beta &= \frac{x_0 y_0}{1+y_0^2} (\|\boldsymbol{J}_0\|^2 + \beta^2) \end{aligned}$$

Et finalement, en éliminant β, nous obtenons une équation bicarrée avec une seule inconnue :

$$\mathcal{A}\,\alpha^4 + \mathcal{B}\,\alpha^2 + \mathcal{C} = 0 \tag{3.9}$$

où :

$$\begin{aligned} \mathcal{A} &= a^2 - g \\ \mathcal{B} &= 2\,a^2 d - g\,d + e - 2\,a\,c \\ \mathcal{C} &= a^2 d^2 + c^2 - 2\,a\,c\,d \\ a &= \frac{x_0 y_0}{1+x_0^2} \\ b &= \frac{x_0 y_0}{1+y_0^2} \\ c &= \boldsymbol{I}_0 \cdot \boldsymbol{J}_0 \\ d &= \|\boldsymbol{I}_0\|^2 \\ e &= \|\boldsymbol{J}_0\|^2 \\ g &= \frac{1+y_0^2}{1+x_0^2} \end{aligned}$$

Étudions le nombre de solutions réelles de l'équation (3.9). Posons $t = \alpha^2$:

$$\mathcal{A}\,t^2 + \mathcal{B}\,t + \mathcal{C} = 0$$

La quantité $\mathcal{C}/\mathcal{A}$ est égale au produit des racines. Nous avons :

$$\begin{aligned} \frac{\mathcal{C}}{\mathcal{A}} &= f(x_0, y_0, c, d) \\ &= \frac{-x_0^2 y_0^2\, d^2 - (1+x_0^2)^2\, c^2 + 2 x_0 y_0 (1+x_0^2)\, c\, d}{1+x_0^2+y_0^2} \\ &= -\frac{(x_0 y_0\, d - (1+x_0^2)\, c)^2}{1+x_0^2+y_0^2} \\ &\leq 0 \end{aligned}$$

Ainsi, $\mathcal{C}/\mathcal{A}$ est toujours négatif ou nul. Il y a donc une racine positive (ou nulle) et une racine négative (ou nulle) pour t. Il y a donc deux racines réelles pour α (une positive et une négative), et deux racines imaginaires.

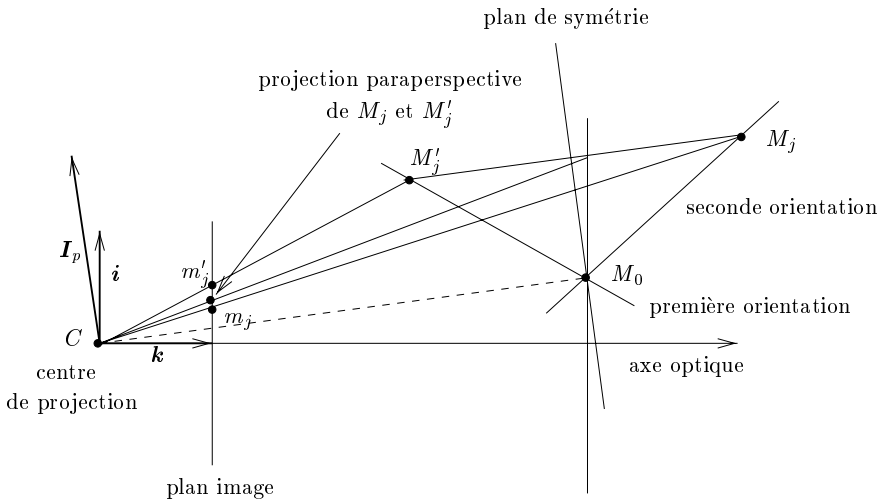


FIG. 3.2: *Un objet planaire conduit à deux orientations possibles. Cependant, les deux orientations conduisent à deux projections perspectives distinctes dans l'image – m_j et m'_j.*

Les deux racines réelles pour α fournissent deux solutions pour β, et par conséquent deux solutions pour $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$. Il y a donc deux solutions correspondant au problème connu d'ambiguïté de signe pour un modèle affine de caméra. Ces deux solutions sont représentées sur la figure 3.2. Les points M_j (appartenant au plan de l'objet dans une orientation) et M'_j (appartenant au plan de l'objet dans une autre orientation) ont les mêmes projections paraperspectives, mais ont des projections perspectives différentes. Remarquons que ces orientations sont symétriques par rapport à un plan perpendiculaire à la ligne de vue passant par M_0.

Ainsi, l'algorithme itératif proposé au paragraphe 2.3 conduit à deux poses à chaque itération. Ces deux poses ont les mêmes vecteurs de translation mais ont des orientations différentes. Ainsi, après n itérations, nous avons théoriquement 2^n solutions. Pour éviter la multiplication du nombre de solutions, nous procédons comme suit. À la première itération, nous conservons les deux solutions; aux itérations suivantes, nous ne conservons qu'une seule des solutions – la solution la plus proche de celle de l'itération précédente. À la convergence de l'algorithme, nous obtenons ainsi deux solutions pour l'orientation de la caméra : une solution associée à la branche gauche de l'arbre de recherche, et une autre associée à la branche droite de l'arbre (voir figure 3.3). Parmi les deux solutions finales, l'une coïncide généralement mieux avec les points image donnés, il est alors possible de choisir la solution correcte. Cependant, si le plan de l'objet est proche du plan de symétrie, alors l'ambiguïté n'est pas résolue.

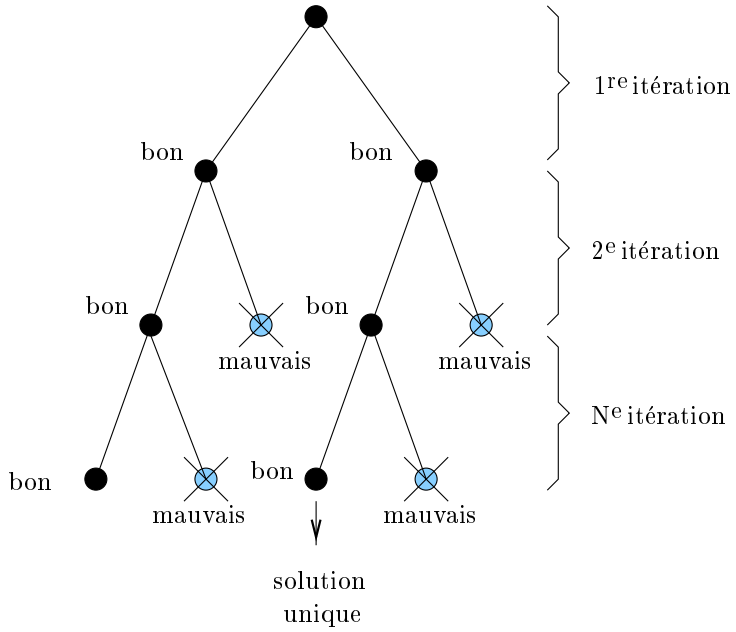


FIG. 3.3: *Arbre binaire de recherche pour le choix de la meilleure pose possible pour un objet planaire.*

3.3 Contrainte d'orthogonalité

Les algorithmes décrits jusqu'à maintenant ne garantissent pas que la matrice représentant l'orientation de la caméra par rapport à l'objet est une vraie matrice de rotation. Cette matrice est formée par les trois vecteurs lignes ${}^t\boldsymbol{i}$, ${}^t\boldsymbol{j}$, et ${}^t\boldsymbol{k}$. Ces vecteurs sont mis à jour à chaque pas d'itération des algorithmes, et nous voulons calculer une "vraie" matrice de rotation à partir de ces trois vecteurs. Mathématiquement, cela signifie que nous cherchons une matrice de rotation $\mathbf{R}$ qui vérifie :

$$\mathbf{R}^t \begin{pmatrix} {}^t\boldsymbol{i} \\ {}^t\boldsymbol{j} \\ {}^t\boldsymbol{k} \end{pmatrix} = \boldsymbol{\mathcal{I}}_3$$

Cette expression peut se réécrire :

$$\begin{aligned} \mathbf{R}\,\boldsymbol{i} &= \boldsymbol{e}_1 \\ \mathbf{R}\,\boldsymbol{j} &= \boldsymbol{e}_2 \\ \mathbf{R}\,\boldsymbol{k} &= \boldsymbol{e}_3 \end{aligned}$$

où $\boldsymbol{e}_i$ est le i-ième vecteur colonne de la matrice identité. La solution est donnée par la matrice de rotation qui minimise le critère :

$$\min_{\mathbf{R}} \sum_{i=1}^{3} \|\mathbf{R}\boldsymbol{v}_i - \boldsymbol{e}_i\|^2$$

où $\boldsymbol{v}_i$ représente l'un des vecteurs $\boldsymbol{i}$, $\boldsymbol{j}$, ou $\boldsymbol{k}$. Il est bien connu que ce problème de minimisation possède une solution exacte [Fau 93]. En effet, si la matrice de rotation est

représentée par un quaternion unitaire $\boldsymbol{q}$, alors le problème de minimisation devient :

$$\min_{\boldsymbol{q}}({}^t\boldsymbol{q}\mathbf{B}\boldsymbol{q})$$

où $\mathbf{B}$ est une matrice 4×4 symétrique, semi-définie positive. Par conséquent, le quaternion $\boldsymbol{q}$ qui minimise la forme quadratique est le vecteur propre associé à la plus petite valeur propre de $\mathbf{B}$. Remarquons que deux vecteurs (c'est-à-dire $\boldsymbol{i}$ et $\boldsymbol{j}$) sont suffisants pour calculer la matrice orthogonale $\mathbf{R}$. Un rappel sur les quaternions ainsi que la méthode détaillée sont présentés dans l'annexe C.

3.4 Résultats expérimentaux

Dans ce paragraphe, nous étudions les performances des algorithmes itératifs orthographique à l'échelle et paraperspectif. Nous nous intéressons en particulier à deux critères :

- la précision de la pose en position et orientation en fonction de la distance de l'objet par rapport à la caméra ;
- la convergence (nombre d'itérations) des algorithmes itératifs de pose en fonction de la distance de l'objet par rapport à la caméra.

Pour la première classe d'expérimentations (précision), nous comparons les résultats obtenus à partir de trois algorithmes : les deux algorithmes itératifs linéaires décrits ci-dessus, et une méthode basée sur une minimisation non linéaire. Pour la seconde classe d'expérimentations (convergence et nombre d'itérations), nous comparons les deux algorithmes linéaires. Dans chacun des cas, nous utilisons des données simulées. L'objet considéré représente une maison formée par 14 points.

Pour chaque expérience, nous plaçons l'objet à différentes positions de la caméra, et pour chacune de ces positions, nous plaçons l'objet dans 500 orientations aléatoires. Les matrices de rotation définissant ces 500 orientations sont calculées à partir des angles d'Euler choisis de manière aléatoire dans l'intervalle $[0, 2\pi]$. La position de l'objet par rapport à la caméra est représentée par un vecteur de translation du centre de projection de la caméra à l'origine du repère objet (choisie comme étant le centre de gravité de l'objet). D'un point de vue quantitatif, nous calculons l'erreur entre la pose théorique et la pose calculée par l'un des algorithmes. Pour chaque position, nous traçons l'erreur moyenne à partir des 500 orientations. Les critères d'erreur sont l'erreur en position et l'erreur en orientation. L'erreur en orientation est définie par l'angle de rotation en degrés nécessaire pour aligner le repère de l'objet dans l'orientation calculée avec l'orientation théorique. L'erreur en position est définie par la norme du vecteur représentant la différence entre les deux vecteurs de translation (le vecteur calculé et le vecteur théorique), divisée par la taille de l'objet. Les axes des abscisses des graphiques des figures 3.4, 3.5 et 3.6 représentent la composante en z du vecteur de translation divisée par la taille de l'objet.

Les figures 3.4 et 3.5 représentent respectivement l'erreur en position et orientation en présence de bruit gaussien (d'écart-type $\sigma = 1$ pixel) en fonction du rapport entre la distance caméra-objet et la taille de l'objet. Nous nous sommes successivement placés dans deux configurations : lorsque l'objet est centré sur l'axe optique (figures de gauche), et lorsque l'objet est décalé par rapport à l'axe optique (figures de droites). Dans le premier cas, le centre de l'objet se projette au point de coordonnées (256, 256) dans l'image;

dans le second cas, il se projette en (100, 100). Les deux algorithmes itératifs (orthographique à l'échelle et paraperspectif) donnent des résultats similaires. Ceci s'explique car ils convergent tous les deux vers la même solution correspondant au modèle perspectif de caméra. L'erreur augmente en fonction de la distance caméra-objet car la présence de bruit dans les images devient plus significative lorsque la taille de l'objet est petite dans l'image. La méthode de minimisation non linéaire a été initialisée à partir des résultats fournis par une des méthodes itératives. On constate une amélioration des résultats, mais cette amélioration est relativement faible en pourcentage.

La figure 3.6 représente le nombre d'itérations (nombre moyen et maximum) en fonction du rapport entre la distance caméra-objet et la taille de l'objet. L'algorithme paraperspectif itératif converge en moins d'itérations que l'algorithme orthographique à l'échelle lorsque l'objet est décalé par rapport à l'axe optique. En effet, la pose obtenue après la première itération avec un modèle paraperspectif est plus précise (plus proche du modèle perspectif) que la pose obtenue avec un modèle orthographique à l'échelle. Notons que les deux algorithmes ont convergé dans 100 % des configurations.

Une des applications du calcul de pose est l'asservissement visuel. Dans ce contexte, la caméra et le traitement d'image associé sont insérés dans une boucle temps réelle afin de déplacer un robot manipulateur. Celui-ci doit être déplacé vers une position finale en tenant compte du mouvement apparent dans l'image. Il s'agit de convertir les variations dans l'image en commande 3D du robot. Dans le cas où l'on utilise une seule caméra, le calcul de pose devient capital [Esp 92] et doit être réalisé en quelques millisecondes.

Une application plus particulière de l'asservissement visuel est la saisie d'un objet polyédrique par une pince montée sur un robot et composée de deux mâchoires parallèles [Hor 97b]. Une seule caméra observe à la fois l'objet et la pince. Afin de commander le robot, nous devons localiser la position et l'orientation de la caméra par rapport à l'objet, c'est-à-dire calculer la pose. La figure 3.8 montre une image d'un objet polyédrique devant être localisé (en haut à gauche) et sa représentation en fil de fer (en haut à droite). Nous faisons l'hypothèse d'une connaissance grossière de la position et orientation de l'objet. L'image et le modèle sont modélisés par un ensemble de segments de droites et de jonctions qui sont mis en correspondance en utilisant une méthode similaire à celle proposée dans [Gro 93a]. Nous avons ainsi obtenu 10 jonctions correctement mises en correspondance (au milieu). Le résultat obtenu à la première itération est représenté en bas à gauche correspondant à la pose obtenue avec un modèle paraperspectif de caméra. Après seulement trois itérations, l'algorithme détermine de manière correcte la position et l'orientation de l'objet par rapport à la caméra (en bas à droite).

Le second exemple, figure 3.9 est une situation où seules 4 jonctions ont été mises en correspondance. Ces jonctions trouvées sont planaires (en haut). La première itération trouve une pose approchée (en bas à gauche), et le résultat correct est obtenu au bout de trois itérations (en bas à droite).

Sur ces deux exemples, l'algorithme paraperspectif itératif a convergé en $0,7 \times 10^{-3}$ secondes sur une Sun/Sparc10.

Le troisième exemple, figure 3.10, représente l'image d'une scène où l'on a extrait et mis en correspondance 11 points (à gauche), et le résultat obtenu avec l'algorithme paraperspectif itératif (à droite).

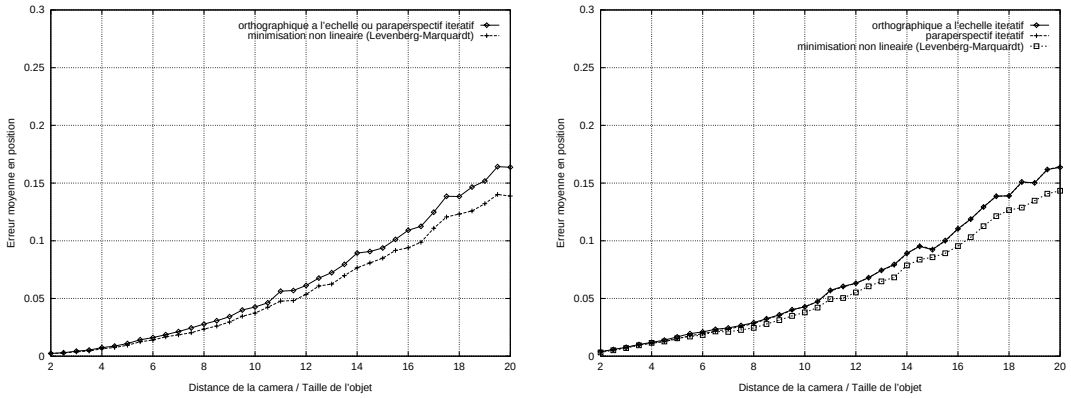


FIG. 3.4: *Erreur en position en fonction de la distance caméra-objet en présence de bruit gaussien à partir de correspondances de points : (a) l'objet est centré sur l'axe optique, (b) l'objet est décalé par rapport à l'axe optique.*

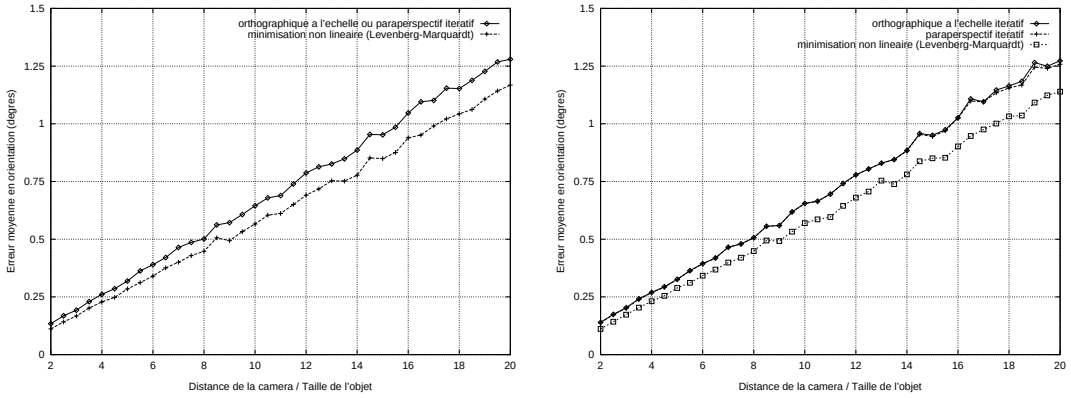


FIG. 3.5: *Erreur en orientation en fonction de la distance caméra-objet en présence de bruit gaussien à partir de correspondances de points : (a) l'objet est centré sur l'axe optique, (b) l'objet est décalé par rapport à l'axe optique.*

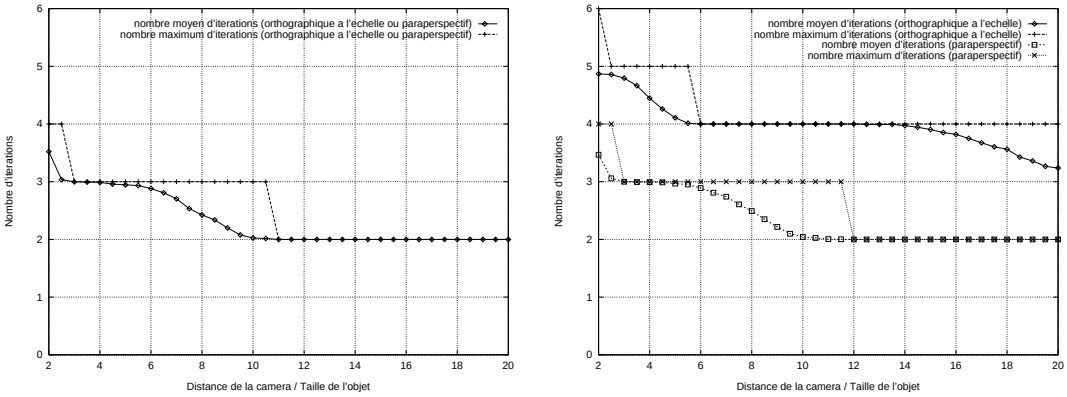


FIG. 3.6: *Rapidité de convergence en fonction de la distance caméra-objet en présence de bruit gaussien à partir de correspondances de points : (a) l'objet est centré sur l'axe optique, (b) l'objet est décalé par rapport à l'axe optique.*

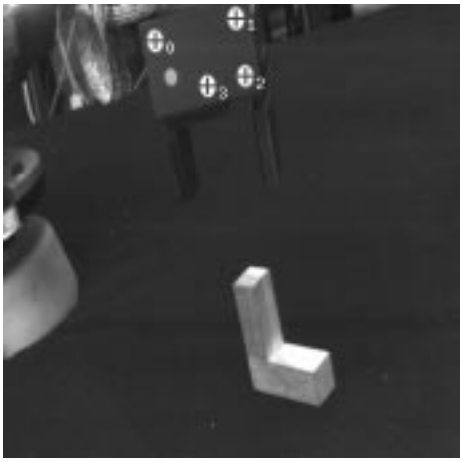


FIG. 3.7: *Les quatre points de la pince sont planaires. Comme l'objet est décalé par rapport à l'axe optique, l'algorithme paraperspectif itératif donne de meilleurs résultats que l'algorithme orthographique à l'échelle itératif.*

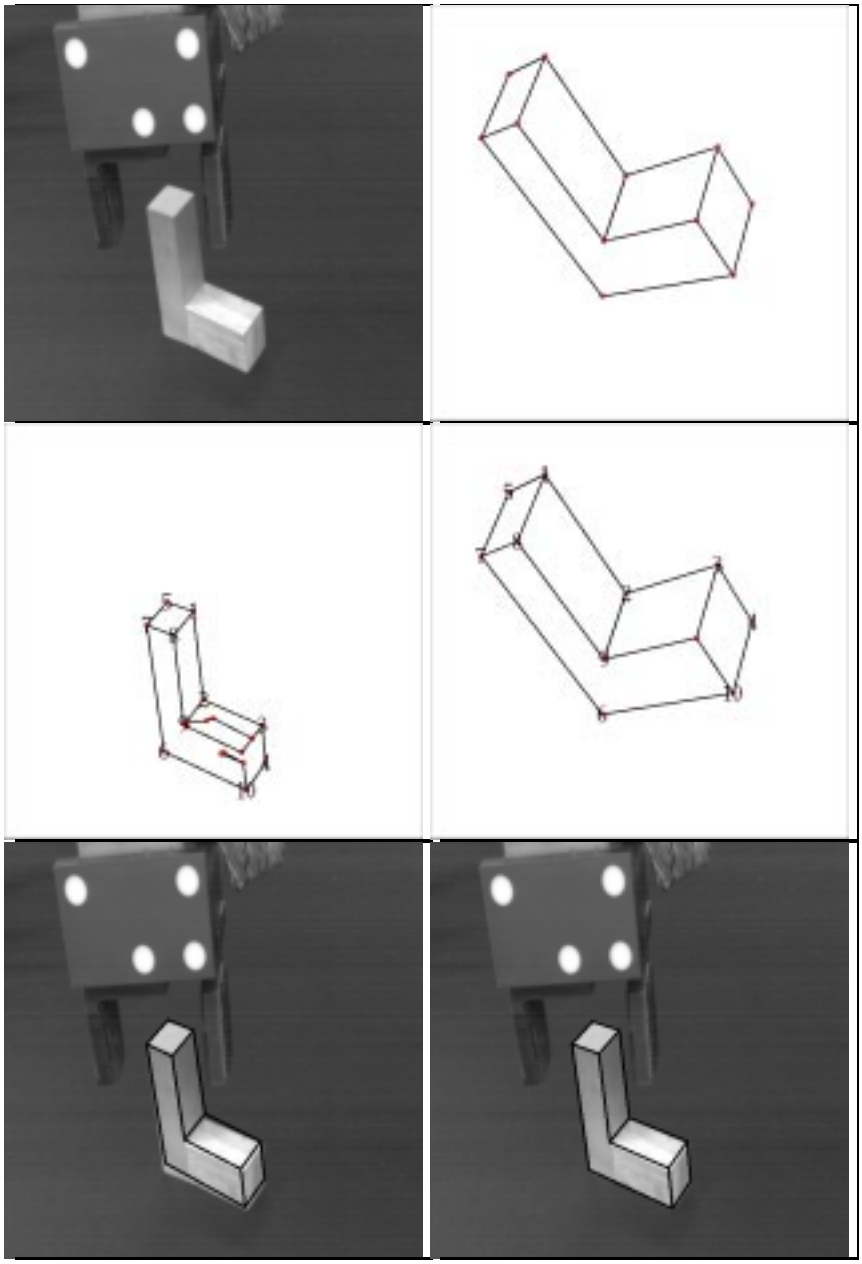


FIG. 3.8: *Exemple d'application de l'algorithme paraperspectif itératif pour un ensemble de points non coplanaires.*

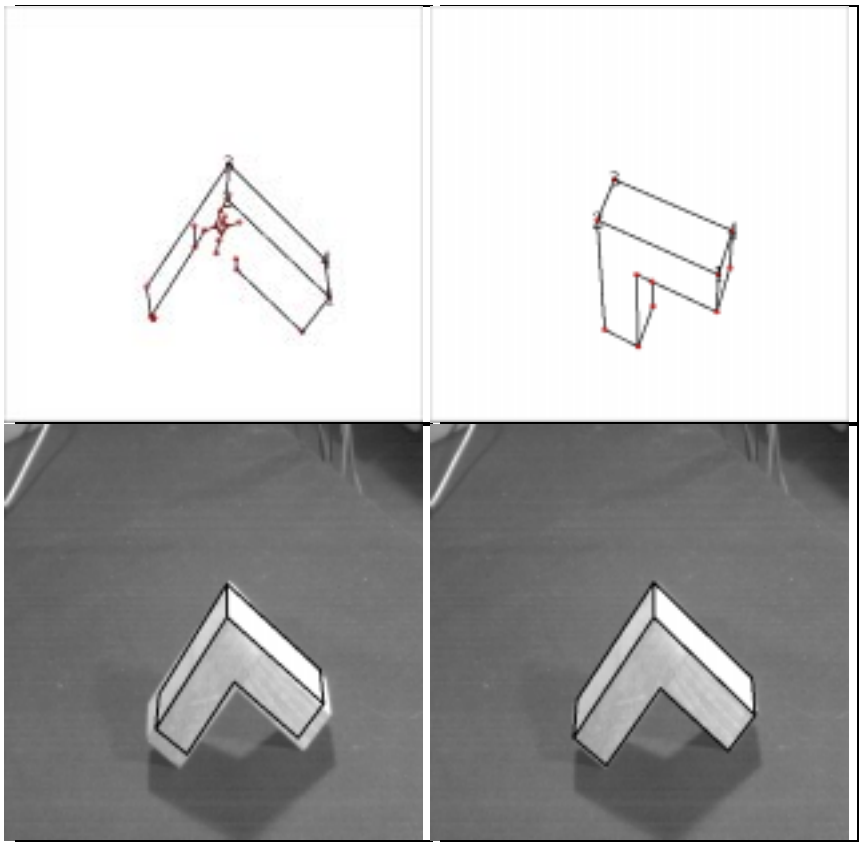


FIG. 3.9: *Exemple d'application de l'algorithme paraperspectif itératif pour un ensemble de 4 points coplanaires.*

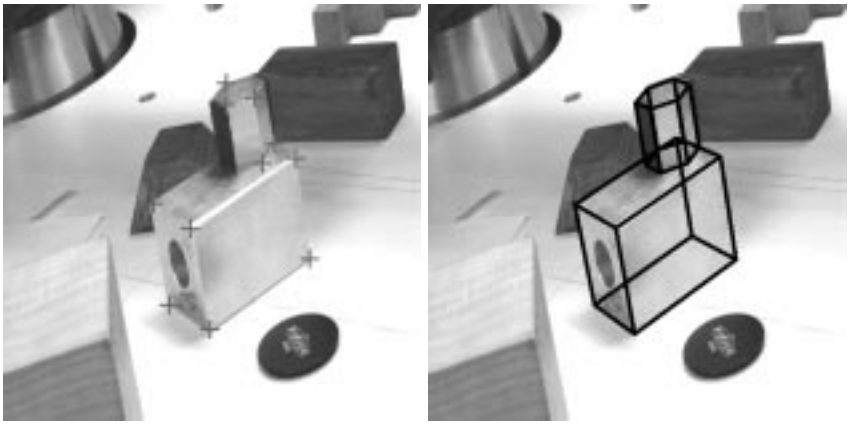


FIG. 3.10: *Le temps de calcul est de 1,47 millisecondes sur une Sun-Sparc en utilisant l'algorithme paraperspectif itératif.*

3.5 Conclusion

Nous avons proposé une extension pour le cas paraperspectif de l'algorithme de calcul de pose itératif développé par Dementhon et Davis [Dem 92b, Dem 93], à partir de correspondances de points. Nous avons établi le lien entre le modèle paraperspectif et le modèle perspectif de caméra. La méthode proposée calcule la pose de manière linéaire grâce à un algorithme itératif qui effectue une succession d'approximations par un modèle paraperspectif, pour converger, à la limite, vers la solution obtenue avec un modèle perspectif. Lorsque l'objet est relativement près de la caméra et éloigné de l'axe optique, l'algorithme paraperspectif itératif a plus de chance de converger que l'algorithme orthographique à l'échelle. De plus, il converge généralement en moins d'itérations. Nous avons également présenté un moyen simple de prendre en compte la contrainte d'orthogonalité de la matrice de rotation entre l'objet et la caméra.

Nous avons comparé la qualité des résultats obtenus avec une méthode itérative et une méthode de minimisation non linéaire. La première méthode est beaucoup plus rapide que la seconde et pratiquement aussi précise. Cependant, en présence de bruit dans l'image, les performances des méthodes linéaires se dégradent plus vite lorsque l'amplitude du bruit augmente, par rapport à une méthode non linéaire. Ceci s'explique par le fait que les méthodes de minimisation numériques non linéaires sont beaucoup plus robustes au bruit que les techniques linéaires algébriques.

Lorsque la vitesse est primordiale, on préférera les algorithmes itératifs linéaires. Si l'on désire avoir une meilleure estimation, on aura intérêt à utiliser un algorithme de minimisation non linéaire, en prenant pour initialisation le résultat fourni par un des algorithmes itératifs. La valeur donnée pour l'initialisation étant proche de la solution finale, le nombre d'itérations est faible (généralement 1 à 3 itérations) et donc peu pénalisante en temps de calcul. Le problème de tomber dans un minimum local est également moindre.

Chapitre 4

Calcul de pose à partir de correspondances de droites

"Une démonstration n'est pas autre chose que la résolution d'une vérité en d'autres vérités déjà connues."

LEIBNITZ.

Nous venons de présenter au chapitre précédent deux algorithmes itératifs permettant de résoudre le problème du calcul de pose à partir de correspondances de points. Cependant, dans certains contextes, il peut être utile de travailler avec d'autres primitives – des droites par exemple. Les droites, par rapport aux points, ont l'avantage d'être extraites de manière plus précises dans les images, et sont également plus robustes face au problème de la mise en correspondance 2D/3D en présence d'occultations. Pour des applications temps réel, il est généralement plus facile de considérer des droites, en particulier pour effectuer la mise en correspondance sur une séquence d'images. Dans certains cas où l'extraction des contours est difficile, les points extraits peuvent être localisés de manière imprécise à cause de segments manquants. Ce problème conduit à de mauvaises mises en correspondance qui pourraient être évitées si l'on considérait des droites.

Dans ce paragraphe, nous étendons les algorithmes précédents pour résoudre le problème du calcul de pose à partir de correspondances de droites. Nous faisons le lien entre les modèles affines de caméra (orthographique à l'échelle et paraperspectif) et le modèle perspectif pour des droites, et nous proposons deux algorithmes itératifs correspondants. Nous étudions également les configurations de droites dégénérées. Enfin, nous comparons les algorithmes proposés avec les algorithmes présentés dans le paragraphe précédent pour des correspondances de points.

Plan du chapitre

Au paragraphe 4.1, nous établissons les équations de projections pour le cas des droites pour les modèles perspectif, orthographique à l'échelle et paraperspectif. Nous détaillons l'algorithme au paragraphe 4.2, et la résolution du système au paragraphe 4.3. Le paragraphe 4.4 évalue les performances des algorithmes.

Enfin, le paragraphe 4.5 explique comment mélanger des correspondances de points et de droites, et le paragraphe 4.6 résume les résultats obtenus.

4.1 Équations de base

La figure 4.1 représente le schéma général pour le calcul de pose à partir de droites. Nous reprenons les notations des chapitres précédents.

- L'origine du repère scène 3D est un point de l'objet M_0.
- Soient ${}^t\boldsymbol{i}$, ${}^t\boldsymbol{j}$, ${}^t\boldsymbol{k}$ les lignes de la matrice de rotation $\mathbf{R}$ et $\boldsymbol{t} = {}^t(\ t_x \quad t_y \quad t_z\ )$ le vecteur de translation représentant la transformation rigide entre le repère de l'objet et le repère caméra :
$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \boldsymbol{t} \\ {}^t\boldsymbol{0} & 1 \end{pmatrix}$$
- Une droite 3D (de l'objet) est notée D_j. Celle-ci est représentée par un point de référence Ω_j et un vecteur directeur $\boldsymbol{D}_j$.

4.1.1 Modèle perspectif de caméra

Nous établissons ici les équations de base pour un modèle perspectif de caméra. Soit M_j un point 3D appartenant à une ligne j de l'espace. On peut alors écrire ($\boldsymbol{\Omega}_j$ est le vecteur allant de l'origine du repère au point Ω_j) :

$$\boldsymbol{M}_j = \boldsymbol{\Omega}_j + \lambda \boldsymbol{D}_j$$

Nous supposons que la caméra est étalonnée. La matrice de projection 3×4 décrivant la projection de l'espace 3D euclidien dans l'image 2D est donnée par :

$$\mathbf{P}_p = \begin{pmatrix} {}^t\boldsymbol{I} & x_0 \\ {}^t\boldsymbol{J} & y_0 \\ {}^t\boldsymbol{K} & 1 \end{pmatrix}$$

La projection de ce point dans l'image vérifie donc :

$$\begin{aligned} \begin{pmatrix} s\boldsymbol{m}_j \\ s \end{pmatrix} &= \mathbf{P}_p \begin{pmatrix} \boldsymbol{M}_j \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} \boldsymbol{I} \cdot \boldsymbol{\Omega}_j + x_0 \\ \boldsymbol{J} \cdot \boldsymbol{\Omega}_j + y_0 \\ 1 + \eta_j \end{pmatrix} + \lambda \begin{pmatrix} \boldsymbol{I} \cdot \boldsymbol{D}_j \\ \boldsymbol{J} \cdot \boldsymbol{D}_j \\ \mu_j \end{pmatrix} \end{aligned} \tag{4.1}$$

avec

$$\eta_j = \boldsymbol{K} \cdot \boldsymbol{\Omega}_j \quad \text{et} \quad \mu_j = \boldsymbol{K} \cdot \boldsymbol{D}_j$$

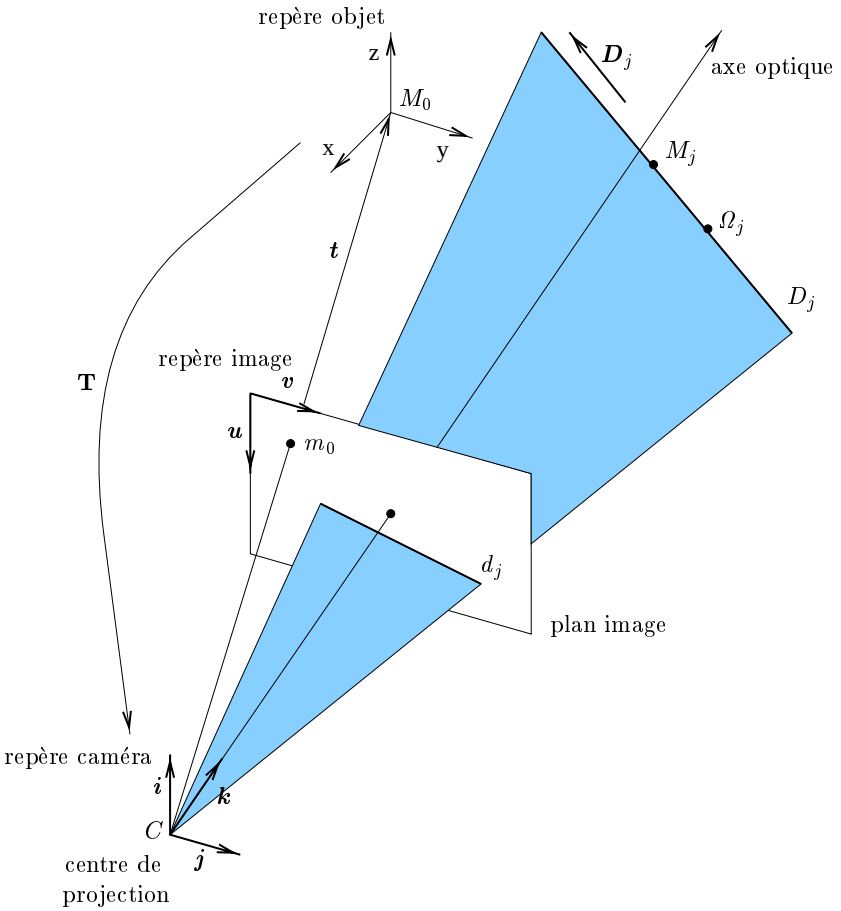


FIG. 4.1: *Cette figure représente le schéma général pour le calcul de pose à partir de droites. Le point M_0 est l'origine du repère de l'objet, et il se projette en m_0. Une droite* 3D *est représentée par un point de référence Ω_j et un vecteur directeur $\boldsymbol{D}_j$ exprimés dans le repère de l'objet ($\boldsymbol{\Omega}_j \perp \boldsymbol{D}_j$). La droite D_j se projette en d_j dans l'image.*

D'autre part, le point m_j doit appartenir à la droite dans l'image ayant pour équation $a_j x + b_j y + c_j = 0$. En remplaçant x et y par les expressions données par l'équation (4.1), et après avoir regroupé les termes, on obtient :

$$a_j \boldsymbol{I} \cdot \boldsymbol{\Omega}_j + b_j \boldsymbol{J} \cdot \boldsymbol{\Omega}_j + a_j x_0 + b_j y_0 + c_j(1+\eta_j) + \lambda(a_j \boldsymbol{I} \cdot \boldsymbol{D}_j + b_j \boldsymbol{J} \cdot \boldsymbol{D}_j + c_j \mu_j) = 0$$

Cette équation est vérifiée pour tout point M_j appartenant à la droite du modèle, elle est donc vraie quel que soit la valeur de λ. Par conséquent, on obtient finalement deux équations :

$$a_j \boldsymbol{I} \cdot \boldsymbol{\Omega}_j + b_j \boldsymbol{J} \cdot \boldsymbol{\Omega}_j + a_j x_0 + b_j y_0 + c_j(1+\eta_j) = 0 \quad (4.2)$$

$$a_j \boldsymbol{I} \cdot \boldsymbol{D}_j + b_j \boldsymbol{J} \cdot \boldsymbol{D}_j + c_j \mu_j = 0 \quad (4.3)$$

Les inconnues sont les paramètres de pose $\boldsymbol{I}$, $\boldsymbol{J}$, x_0 et y_0, ainsi que les η_j et μ_j représentant l'effet de perspective. Pour n correspondances de droites, on a $2n$ équations.

De manière équivalente, on peut écrire ces équations en faisant intervenir $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$. À partir des équations (2.18) et (2.19), on en déduit :

$$\begin{aligned} \boldsymbol{I} &= \boldsymbol{I}_p + x_0 \boldsymbol{K} \\ \boldsymbol{J} &= \boldsymbol{J}_p + y_0 \boldsymbol{K} \end{aligned}$$

Enfin, en remplaçant dans les équations (4.2) et (4.3), on obtient finalement :

$$a_j \boldsymbol{I}_p \cdot \boldsymbol{\Omega}_j + b_j \boldsymbol{J}_p \cdot \boldsymbol{\Omega}_j + (a_j x_0 + b_j y_0 + c_j)(1 + \eta_j) = 0 \quad (4.4)$$

$$a_j \boldsymbol{I}_p \cdot \boldsymbol{D}_j + b_j \boldsymbol{J}_p \cdot \boldsymbol{D}_j + (a_j x_0 + b_j y_0 + c_j)\mu_j = 0 \quad (4.5)$$

4.1.2 Modèle perspectif faible de caméra

Les équations reliant une droite de la scène et sa projection avec un modèle perspectif faible de caméra découlent du paragraphe précédent. En effet, il suffit de reprendre les calculs avec la matrice de projection qui s'écrit, dans ce cas :

$$\mathbf{P}_{pf} = \begin{pmatrix} {}^t\boldsymbol{I} & x_0 \\ {}^t\boldsymbol{J} & y_0 \\ {}^t\boldsymbol{0} & 1 \end{pmatrix} \quad (4.6)$$

Nous obtenons ainsi :

$$a_j \boldsymbol{I} \cdot \boldsymbol{\Omega}_j + b_j \boldsymbol{J} \cdot \boldsymbol{\Omega}_j + a_j x_0 + b_j y_0 + c_j = 0 \quad (4.7)$$

$$a_j \boldsymbol{I} \cdot \boldsymbol{D}_j + b_j \boldsymbol{J} \cdot \boldsymbol{D}_j = 0 \quad (4.8)$$

Par conséquent, en prenant $\eta_j = \mu_j = 0$ dans les équations (4.2) and (4.3), on retrouve les équations caractérisant les correspondances de droites 2D/3D.

4.1.3 Modèle paraperspectif de caméra

De manière similaire, en utilisant la matrice de projection pour un modèle paraperspectif de caméra :

$$\mathbf{P}_{pp} = \begin{pmatrix} {}^t\boldsymbol{I}_p & x_0 \\ {}^t\boldsymbol{J}_p & y_0 \\ {}^t\boldsymbol{0} & 1 \end{pmatrix} \quad (4.9)$$

où

$$\boldsymbol{I}_p = \boldsymbol{I} - x_0 \boldsymbol{K} \quad (4.10)$$

$$\boldsymbol{J}_p = \boldsymbol{J} - y_0 \boldsymbol{K} \quad (4.11)$$

nous obtenons les équations suivantes :

$$a_j \boldsymbol{I}_p \cdot \boldsymbol{\Omega}_j + b_j \boldsymbol{J}_p \cdot \boldsymbol{\Omega}_j + a_j x_0 + b_j y_0 + c_j = 0 \quad (4.12)$$

$$a_j \boldsymbol{I}_p \cdot \boldsymbol{D}_j + b_j \boldsymbol{J}_p \cdot \boldsymbol{D}_j = 0 \quad (4.13)$$

Par conséquent, en prenant $\eta_j = \mu_j = 0$ dans les équations (4.4) and (4.5), on retrouve les équations caractérisant les correspondances de droites 2D/3D.

4.2 Algorithme

Les équations que nous venons d'établir nous permettent de calculer les paramètres de pose pour chacun des modèles de projection.

- *Perspectif faible* : les équations (4.7) et (4.8) sont linéaires en $\boldsymbol{I}$, $\boldsymbol{J}$, x_0 et y_0. Puisqu'il y a huit inconnues, et que chaque correspondance de droite fournit deux équations, il faut au minimum quatre droites pour résoudre le système à partir desquels les paramètres de pose (matrice de rotation et vecteur de translation) peuvent être estimés.

- *Paraperspectif* : les équations (4.12) et (4.13) sont linéaires en $\boldsymbol{I}_p$, $\boldsymbol{J}_p$, x_0 et y_0. Sous les mêmes conditions que précédemment, les paramètres de pose peuvent être estimés.

- *Perspectif* : dans ce cas, les paramètres de pose peuvent être estimés de manière linéaire en résolvant soit les équations (4.2) et (4.3) (itérations avec un modèle perspectif faible) ou les équations (4.4) et (4.5) (itérations avec un modèle paraperspectif). Dans chacun de ces deux cas, l'algorithme itératif est le suivant.

  1. Pour tout j, $j \in \{1, \ldots, n\}$, initialiser $\eta_j = \mu_j = 0$.
  2. Résoudre le système linéaire surcontraint formé par les équations (4.2) et (4.3), ou (4.4) et (4.5).
  3. Estimer le vecteur de translation (t_x, t_y, et t_z) et la matrice de rotation formée par les vecteurs $\boldsymbol{i}$, $\boldsymbol{j}$, et $\boldsymbol{k}$ en lignes ; orthogonaliser cette matrice afin d'estimer la rotation $\mathbf{R}$.
  4. Pour tout j, calculer :

$$\eta_j = \frac{\boldsymbol{k} \cdot \boldsymbol{\Omega}_j}{t_z} \quad \text{et} \quad \mu_j = \frac{\boldsymbol{k} \cdot \boldsymbol{D}_j}{t_z}$$

     Si les η_j et μ_j calculés à cette itération sont égaux à ceux calculés à l'itération précédente, alors arrêter l'algorithme, sinon retourner à l'étape 2.

4.3 Résolution du système linéaire

Les deux algorithmes itératifs (orthographique à l'échelle et paraperspectif) doivent résoudre un système linéaire surcontraint formé par les équations (4.2), (4.3) (orthographique à l'échelle) ou (4.4), (4.5) (paraperspectif). Sous forme matricielle, ces équations peuvent s'écrivent :

$$\underbrace{\mathbf{A}}_{2n\times 8}\ \underbrace{\boldsymbol{x}}_{8\times 1} = \underbrace{\boldsymbol{b}}_{2n\times 1} \tag{4.14}$$

Plus précisément, nous avons :

$$\begin{pmatrix} \cdots & \cdots & \cdots & \cdots \\ a_j{}^t\boldsymbol{\Omega}_j & b_j{}^t\boldsymbol{\Omega}_j & a_j & b_j \\ a_j{}^t\boldsymbol{D}_j & b_j{}^t\boldsymbol{D}_j & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots \end{pmatrix} \begin{pmatrix} {}^t\boldsymbol{I} \\ {}^t\boldsymbol{J} \\ x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} \cdots \\ -c_j(1+\eta_j) \\ -c_j\mu_j \\ \cdots \end{pmatrix}$$

ou de manière équivalente :

$$\begin{pmatrix} \dots & \dots & \dots & \dots \\ a_j{}^t\boldsymbol{\Omega}_j & b_j{}^t\boldsymbol{\Omega}_j & a_j(1+\eta_j) & b_j(1+\eta_j) \\ a_j{}^t\boldsymbol{D}_j & b_j{}^t\boldsymbol{D}_j & a_j\mu_j & b_j\mu_j \\ \dots & \dots & \dots & \dots \end{pmatrix} \begin{pmatrix} {}^t\boldsymbol{I}_p \\ {}^t\boldsymbol{J}_p \\ x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} \dots \\ -c_j(1+\eta_j) \\ -c_j\mu_j \\ \dots \end{pmatrix}$$

Nous rappelons qu'une droite 3D est représentée par un point de référence Ω_j arbitraire de cette droite, et par un vecteur directeur $\boldsymbol{D}_j$. La droite image 2D correspondante est représentée par les scalaires a_j, b_j, et c_j.

Le vecteur inconnu $\boldsymbol{x}$ a 8 composantes. Par conséquent, la matrice $\mathbf{A}$ de taille $2n \times 8$ doit être de rang égal à 8. Remarquons que pour le modèle orthographique à l'échelle, cette matrice ne dépend pas des "distorsions perspectives" η_j et μ_j.

4.3.1 Analyse du rang

Nous analysons ici les configurations géométriques pour lesquelles la matrice $\mathbf{A}$ vérifie la contrainte du rang donnée ci-dessus, et nous identifions les configurations pour lesquelles la pose ne peut pas être calculée. Du fait que l'analyse du rang est basée sur des configurations géométriques 3D, il est facile d'éviter les configurations dégénérées.

Dans un premier temps, nous pouvons remarquer que les deux équations données par une correspondance de droites, (4.2) et (4.3) [1], sont indépendantes. Le seul cas pour lequel ces deux équations ne sont pas indépendantes est lorsque la droite passe par le centre de projection. Dans ce cas, nous avons $\boldsymbol{\Omega}_j = \alpha \boldsymbol{D}_j$ et la droite est réduite à un point dans l'image.

Considérons maintenant deux droites 3D. Ces droites se projettent en deux droites images distinctes, et nous avons deux plans de projection distincts associés (voir figure 4.1). Par conséquent, deux correspondances de droites donnent 4 équations indépendantes.

Enfin, considérons trois droites 3D, figure 4.2 (a) et (b). Si ces droites s'intersectent en un point commun, alors les plans de projection forment un faisceau de plans, et l'un de ces plans est combinaison linéaire des deux autres plans. Un faisceau de trois droites ou plus, concourantes ou parallèles, ne seront équivalentes qu'à deux droites.

La figure 4.3 montre quelques exemples de configurations de droites qui peuvent être utilisées avec l'équation (4.14).

Considérons finalement le cas d'un objet planaire. Nous allons montrer ci-dessous que la contrainte de coplanarité peut être prise en compte de manière explicite, et que 3 droites coplanaires sont suffisantes pour calculer la pose.

4.3.2 Cas d'un objet formé de lignes coplanaires

Lorsque les droites de l'objet sont coplanaires, la contrainte de coplanarité peut être explicitement utilisée pour calculer la pose. Comme nous allons le montrer ci-dessous, l'avantage pratique est que le nombre minimum de droites nécessaires est dans ce cas égal à 3 (au lieu de 4).

Considérons le plan passant par les droites de l'objet, et soit $\boldsymbol{u}$ le vecteur unitaire orthogonal à ce plan. Comme dans le cas des points, les vecteurs $\boldsymbol{I}$ et $\boldsymbol{J}$ s'écrivent comme

1. Les équations (4.4) et (4.5) sont équivalentes aux équations (4.2) et (4.3).

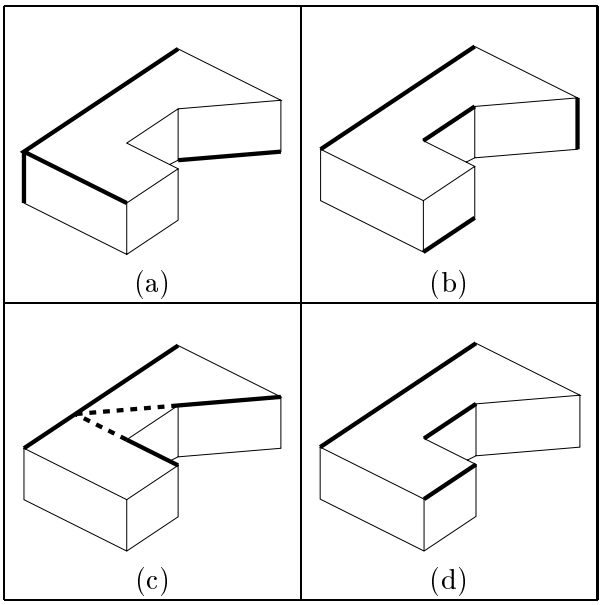


FIG. 4.2: *Cette figure montre toutes les configurations géométriques qui font échouer l'algorithme proposé. Dans le cas d'un modèle non planaire, le seul cas interdit est un faisceau de trois droites ou plus, que ces droites s'intersectent en un même point (a) ou qu'elles soient parallèles (b). Dans le cas d'une configuration planaire, trois droites sont suffisantes, mais celles-ci ne doivent pas être concourantes (c) ou parallèles (d).*

combinaison linéaire d'un vecteur appartement à ce plan, et du vecteur $\boldsymbol{u}$ (un raisonnement similaire est valable pour les vecteurs $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$) :

$$\boldsymbol{I} = \boldsymbol{I}_0 + \alpha\,\boldsymbol{u} \tag{4.15}$$

$$\boldsymbol{J} = \boldsymbol{J}_0 + \beta\,\boldsymbol{u} \tag{4.16}$$

L'idée consiste à remplacer les inconnues $\boldsymbol{I}$ et $\boldsymbol{j}$ par $\boldsymbol{I}_0$ et $\boldsymbol{J}_0$, et à ajouter au système linéaire deux contraintes supplémentaires, $\boldsymbol{I}_0 \cdot \boldsymbol{u} = 0$ et $\boldsymbol{J}_0 \cdot \boldsymbol{u} = 0$. En substituant les équations (4.15) et (4.16) dans les équations (4.2) et (4.3), et en remarquant que $\boldsymbol{\Omega}_j \cdot \boldsymbol{u} = 0$ et $\boldsymbol{D}_j \cdot \boldsymbol{u} = 0$, nous obtenons :

$$\mathbf{A}'\boldsymbol{x}' = \boldsymbol{b}'$$

où

$$\begin{pmatrix} & \mathbf{A} & & \\ {}^t\boldsymbol{u} & {}^t\boldsymbol{0} & 0 & 0 \\ {}^t\boldsymbol{0} & {}^t\boldsymbol{u} & 0 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{I}_0 \\ \boldsymbol{J}_0 \\ x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} \boldsymbol{b} \\ 0 \\ 0 \end{pmatrix}$$

Comme le vecteur $\boldsymbol{u}$ est perpendiculaire aux lignes de l'objet, les deux lignes supplémentaires de la matrice $\mathbf{A}'$ sont indépendantes des lignes de la matrice $\mathbf{A}$. Puisque la matrice $\mathbf{A}'$ doit être de rang 8, il est suffisant que le rang de $\mathbf{A}$ est égal à 6. Ainsi, trois droites (non toutes parallèles ou concourantes) sont suffisantes pour calculer $\boldsymbol{x}'$. Nous obtenons ainsi une solution pour $\boldsymbol{I}_0$, $\boldsymbol{J}_0$, x_0 et y_0 :

$$\boldsymbol{x}' = \mathbf{A}^{\dagger}\boldsymbol{b}'$$

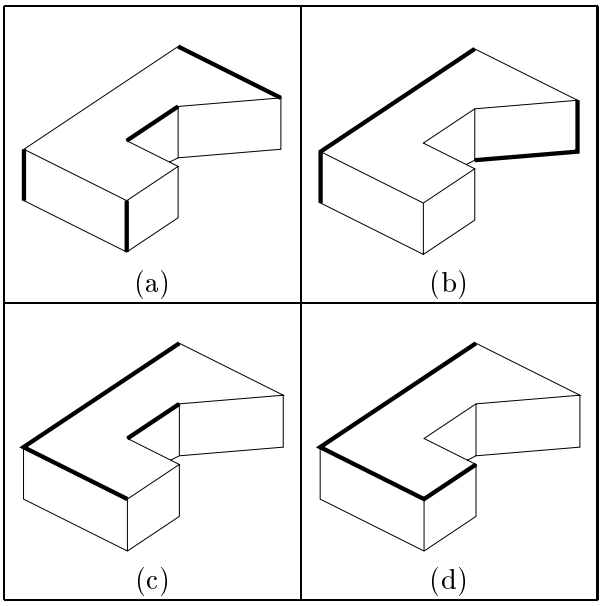


FIG. 4.3: *Cette figure représente quelques exemples de configurations non planaires (a) et (b), et planaires (c) et (d) qui peuvent être utilisées pour calculer la pose avec l'un des deux modèles de caméra proposés.*

Pour finir, afin d'estimer $\boldsymbol{I}$ et $\boldsymbol{J}$ à partir des vecteurs $\boldsymbol{I}_0$ et $\boldsymbol{J}_0$ estimés, il reste à déterminer α et β. Ceci peut être fait facilement en combinant les équations (4.15) et (4.16) avec les contraintes $\|\boldsymbol{I}\| = \|\boldsymbol{J}\|$ et $\boldsymbol{I} \cdot \boldsymbol{J} = 0$. Pour plus de détails sur l'estimation de α et β, voir [Obe 93] et le paragraphe 3.2.2.

4.4 Résultats expérimentaux

Afin de valider la méthode, nous avons testé l'algorithme sur un grand nombre de configurations. Nous avons lancé les algorithmes orthographique à l'échelle et paraperspectif itératifs en présence de bruit gaussien sur les données image, et nous avons analysé les résultats en fonction de la distance relative de l'objet par rapport à la caméra. Nous avons étudié :

- la précision de la pose en fonction de la position et de l'orientation de la caméra par rapport à l'objet ;
- la convergence des algorithmes itératifs.

Nous avons utilisé les paramètres suivants :

- les paramètres intrinsèques de la caméra sont $u_c = v_c = 256$, $\alpha_u = \alpha_v = 1000$;
- nous avons ajouté un bruit gaussien d'écart-type $\sigma = 1$ pixel sur les données image, et nous avons effectué 500 mesures pour chaque expérience (l'objet a été tourné dans 500 orientations aléatoires) ;

– le modèle 3D est une maison synthétique composée de 18 droites.

Les figures 4.4 et 4.5 représentent respectivement l'erreur en position et en orientation de la caméra en présence de bruit gaussien en fonction du rapport entre la distance caméra-objet et la taille de l'objet. Comme dans le cas des points, les deux algorithmes itératifs (orthographique à l'échelle et paraperspectif) donnent des résultats similaires car ils convergent vers la même solution perspective.

La figure 4.6 représente le nombre d'itérations en fonction de la distance relative. Le nombre d'itérations nécessaires est généralement inférieur de une à deux itérations avec le modèle paraperspectif par rapport au modèle orthographique à l'échelle.

Les performances de ces algorithmes basés sur les correspondances de droites sont comparables (très légèrement moins bonnes) par rapport aux algorithmes basés sur les points. Cependant, nous n'avons pas tenu compte du fait que l'on peut extraire les droites avec une plus grande précision que les points.

Convergence : Dans toutes ces configurations, les deux algorithmes ont convergé dans 100% des configurations, en 3 à 5 itérations.

Temps d'exécution : Sans aucune optimisation, le temps d'exécution est de 0,04 seconde par itération sur une UltraSparc 1/170 (pour 18 droites).

Reprenons l'application de la saisie d'un objet polyédrique par une pince montée sur un robot. L'image acquise est segmentée en un ensemble de jonctions et segments de droites, et ces jonctions et segments sont ensuite mis en correspondance avec un modèle 3D en fils de fer de l'objet. Le résultat de cette étape est une liste de points et une liste de droites mis en correspondance (voir figure 4.7). Le calcul de pose peut être ensuite effectué à partir de points ou de droites. La figure 4.8 montre des résultats obtenus avec différents algorithmes :

– (a) – orthographique à l'échelle à partir de correspondances de droites, c'est-à-dire le résultat obtenu après la première itération de l'algorithme orthographique à l'échelle itératif ;

– (b) – paraperspectif à partir de correspondances de droites, c'est-à-dire le résultat obtenu après la première itération de l'algorithme paraperspectif itératif ;

– (c) – perspectif à partir de correspondances de droites en utilisant l'algorithme orthographique à l'échelle itératif ;

– (d) – perspectif à partir de correspondances de points en utilisant l'algorithme orthographique à l'échelle itératif ;

– (e) – perspectif à partir de correspondances de droites en utilisant la méthode non linéaire proposée dans [Pho 95] ;

– (f) – perspectif à partir de correspondances de points en utilisant la même méthode non linéaire.

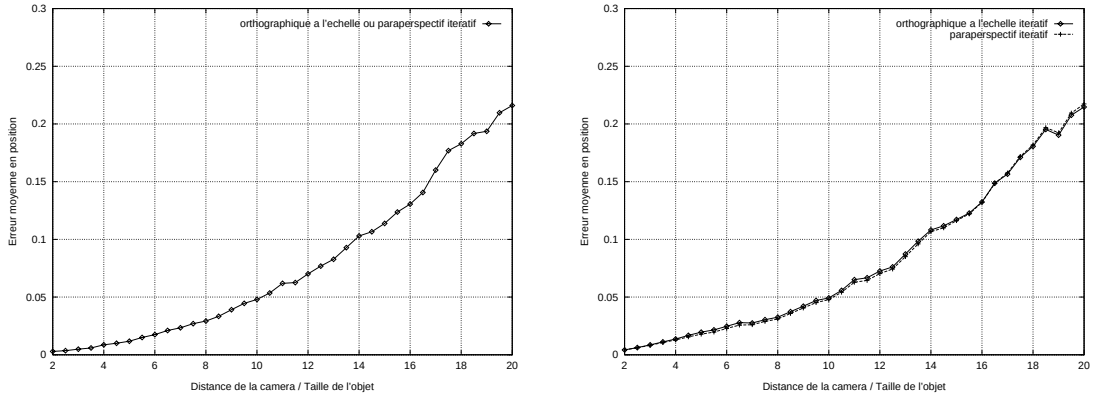


FIG. 4.4: *Erreur en position en fonction de la distance caméra-objet en présence de bruit gaussien à partir de correspondances de droites : (a) l'objet est centré sur l'axe optique, (b) l'objet est décalé par rapport à l'axe optique.*

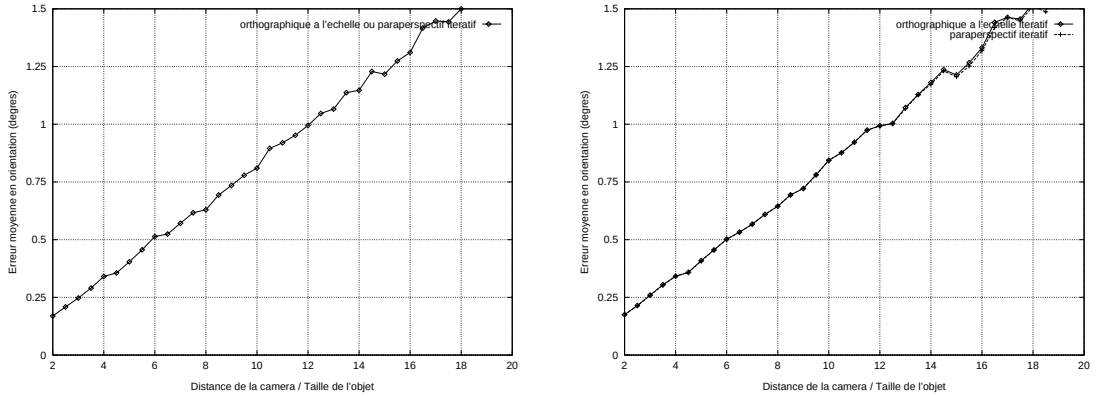


FIG. 4.5: *Erreur en orientation en fonction de la distance caméra-objet en présence de bruit gaussien à partir de correspondances de droites : (a) l'objet est centré sur l'axe optique, (b) l'objet est décalé par rapport à l'axe optique.*

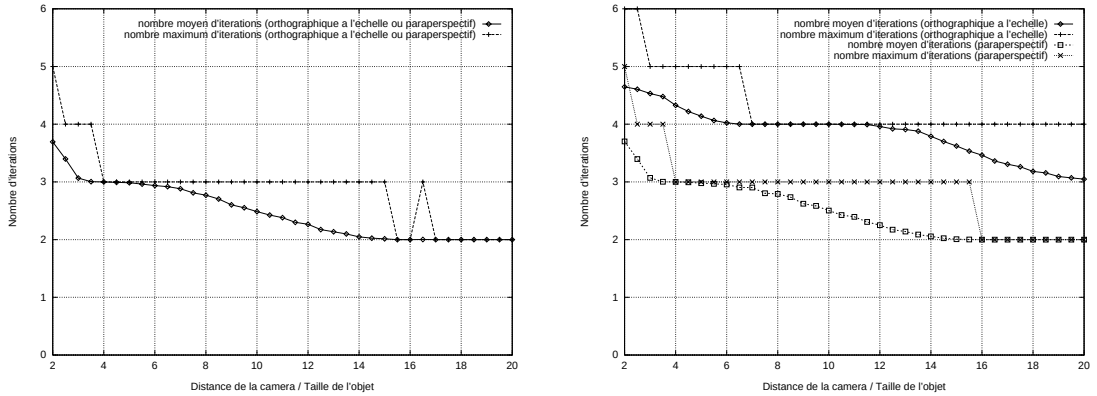


FIG. 4.6: *Rapidité de convergence en fonction de la distance caméra-objet en présence de bruit gaussien à partir de correspondances de droites : (a) l'objet est centré sur l'axe optique, (b) l'objet est décalé par rapport à l'axe optique.*

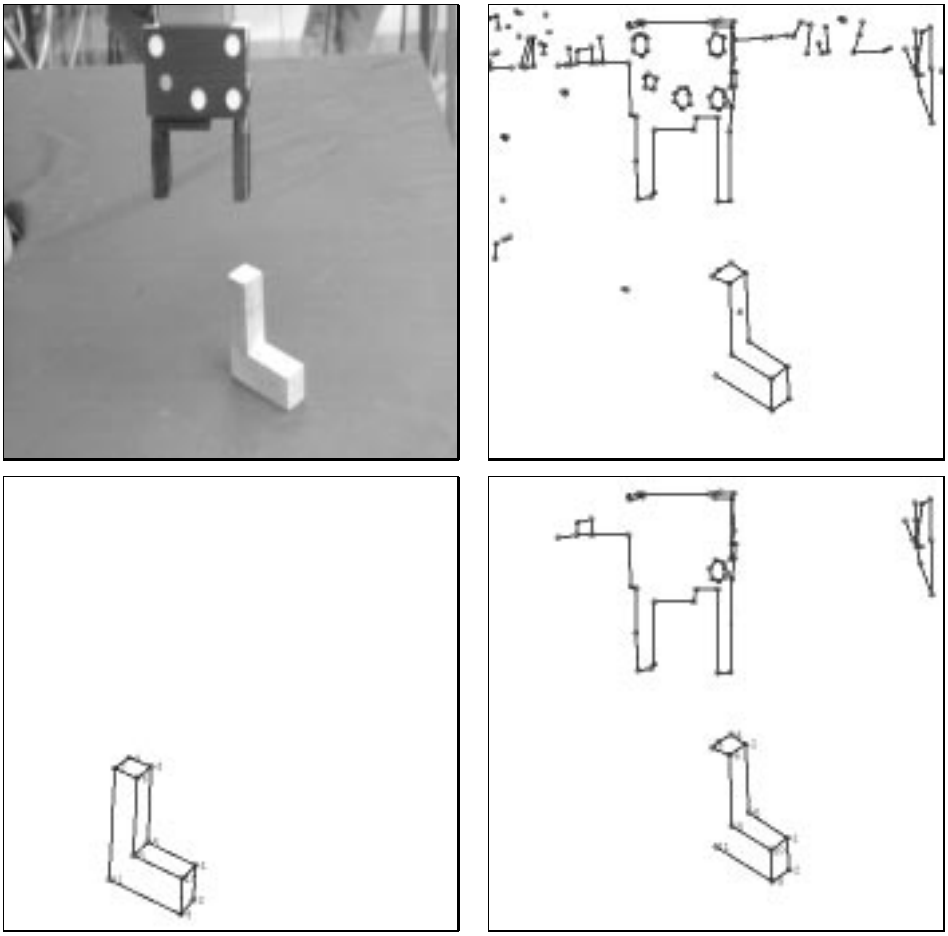


FIG. 4.7: *Droites et jonctions extraites à partir de l'image initiale (en haut). Un modèle en fil de fer de l'objet est mis en correspondance avec les droites et jonctions extraites (en bas). Le résultat est un ensemble de correspondances de droites et un ensemble de correspondances de jonctions.*

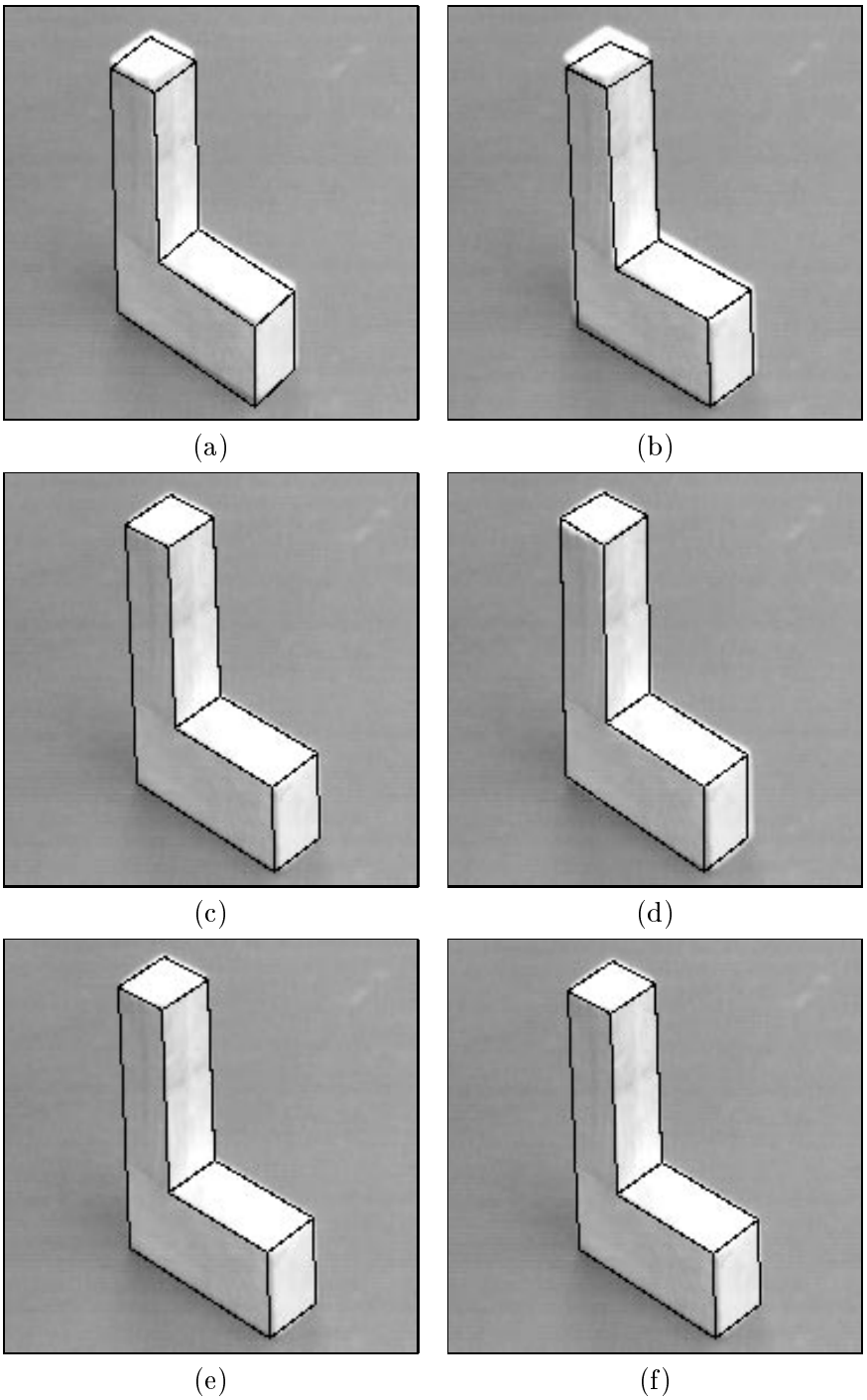


FIG. 4.8: *Pose calculée suivant: (a) – orthographique à l'échelle et droites, (b) – para-perspectif et droites, (c) – algorithme orthographique à l'échelle itératif et droites, (d) – algorithme orthographique à l'échelle itératif et points, (e) – méthode de minimisation non linéaire et droites, (f) – méthode de minimisation non linéaire et points.*

4.5 Mélanger points et droites

Dans ce paragraphe, nous montrons comment étendre les algorithmes précédents pour calculer la pose à partir de correspondances de points et de droites.

4.5.1 Modèle perspectif faible de caméra

Considérons dans un premier temps le cas de l'algorithme orthographique à l'échelle itératif. En combinant les équations obtenues pour les cas de points (équations (3.1) et (3.2)) et de droites (équations (4.2) et (4.3)), le système d'équations peut s'écrire :

$$\begin{aligned}
\boldsymbol{I} \cdot \boldsymbol{M}_j &= x_j(1+\varepsilon_j) - x_0 \\
\boldsymbol{J} \cdot \boldsymbol{M}_j &= y_j(1+\varepsilon_j) - y_0 \\
a_j \boldsymbol{I} \cdot \boldsymbol{\Omega}_j + b_j \boldsymbol{J} \cdot \boldsymbol{\Omega}_j &= -a_j x_0 - b_j y_0 - c_j(1+\eta_j) \\
a_j \boldsymbol{I} \cdot \boldsymbol{D}_j + b_j \boldsymbol{J} \cdot \boldsymbol{D}_j &= -c_j \mu_j
\end{aligned}$$

Lorsque les ε_j, η_j et μ_j sont fixés, ce système est linéaire en $\boldsymbol{I}$, $\boldsymbol{J}$, x_0 et y_0. Nous utilisons donc un algorithme itératif similaire à ceux des paragraphes précédents. Le point 3D de référence M_0 peut être soit l'un des points, soit le centre de gravité des points. Si M_0 est un des points de l'objet, alors sa projection (x_0, y_0) dans l'image est connue; si M_0 est le centre de gravité des points, (x_0, y_0) peut être facilement calculé comme étant le centre de gravité des points dans l'image. À chaque pas d'itération, nous calculons l'orientation de la caméra $\boldsymbol{I}$ et $\boldsymbol{J}$ (6 inconnues). Ensuite, on peut déduire $\boldsymbol{i}$, $\boldsymbol{j}$ et t_z, puis réestimer les valeurs des ε_j, η_j et μ_j. Pour finir, on teste la convergence de l'algorithme. Sous forme matricielle, les équations précédentes peuvent être écrites de la façon suivante :

$$\begin{pmatrix}
\cdots & \cdots \\
{}^t\boldsymbol{M}_j & {}^t\boldsymbol{0} \\
{}^t\boldsymbol{0} & {}^t\boldsymbol{M}_j \\
a_j\,{}^t\boldsymbol{\Omega}_j & b_j\,{}^t\boldsymbol{\Omega}_j \\
a_j\,{}^t\boldsymbol{D}_j & b_j\,{}^t\boldsymbol{D}_j \\
\cdots & \cdots
\end{pmatrix}
\begin{pmatrix} \boldsymbol{I} \\ \boldsymbol{J} \end{pmatrix} =
\begin{pmatrix}
\cdots \\
x_j(1+\varepsilon_j) - x_0 \\
y_j(1+\varepsilon_j) - y_0 \\
-a_j x_0 - b_j y_0 - c_j(1+\eta_j) \\
-c_j \mu_j \\
\cdots
\end{pmatrix}$$

4.5.2 Modèle paraperspectif de caméra

Dans ce cas, en combinant les équations pour les correspondances de points (équations (3.1) et (3.2)) et de droites (équations (4.4) et (4.5)), nous obtenons :

$$\begin{aligned}
\boldsymbol{I}_p \cdot \boldsymbol{M}_j &= (x_j - x_0)(1+\varepsilon_j) \\
\boldsymbol{J}_p \cdot \boldsymbol{M}_j &= (y_j - y_0)(1+\varepsilon_j) \\
a_j \boldsymbol{I}_p \cdot \boldsymbol{\Omega}_j + b_j \boldsymbol{J}_p \cdot \boldsymbol{\Omega}_j &= -(a_j x_0 + b_j y_0 + c_j)(1+\eta_j) \\
a_j \boldsymbol{I}_p \cdot \boldsymbol{D}_j + b_j \boldsymbol{J}_p \cdot \boldsymbol{D}_j &= -(a_j x_0 + b_j y_0 + c_j)\mu_j
\end{aligned}$$

Comme précédemment, nous calculons (à chaque pas d'itération) d'abord les valeurs de x_0 et y_0 (si nécessaire), et ensuite l'orientation de la caméra représentée par les vecteurs $\boldsymbol{I}_p$

et $\boldsymbol{J}_p$. Sous forme matricielle :

$$\begin{pmatrix} \dots & \dots \\ {}^t\boldsymbol{M}_j & {}^t\boldsymbol{0} \\ {}^t\boldsymbol{0} & {}^t\boldsymbol{M}_j \\ a_j{}^t\boldsymbol{\Omega}_j & b_j{}^t\boldsymbol{\Omega}_j \\ a_j{}^t\boldsymbol{D}_j & b_j{}^t\boldsymbol{D}_j \\ \dots & \dots \end{pmatrix} \begin{pmatrix} \boldsymbol{I}_p \\ \boldsymbol{J}_p \end{pmatrix} = \begin{pmatrix} \dots \\ (x_j - x_0)(1+\varepsilon_j) \\ (y_j - y_0)(1+\varepsilon_j) \\ -(a_j x_0 + b_j y_0 + c_j)(1+\eta_j) \\ -(a_j x_0 + b_j y_0 + c_j)\mu_j \\ \dots \end{pmatrix}$$

4.6 Conclusion

Nous avons étendu les deux algorithmes itératifs (orthographique à l'échelle et paraperspectif) pour le cas de correspondances de droites. Notons cependant que dans le cas des droites, la matrice pseudo-inverse ne peut être calculée une fois pour toute que dans le cas orthographique à l'échelle et non dans le cas paraperspectif. En effet, dans ce dernier cas, la matrice dépend des distorsions perspectives (η_j et μ_j) qui varient à chaque pas d'itération.

Les résultats obtenus dans le cas des points ou de droites sont comparables, avec en moyenne un léger avantage pour les points (mais cela dépend des configurations). Ceci s'explique par le fait qu'un point image donne plus d'information (plus de contrainte) sur le point 3D correspondant (le point 3D appartient à la droite passant par le centre de projection et le point image), qu'une droite image (la droite 3D appartient au plan passant par le centre de projection et la droite image). Enfin, l'étude (et le test) des configurations dégénérées est plus simple dans le cas des points que dans le cas des droites.

Chapitre 5

Reconstruction

"Un phénomène n'est pas une simple apparence : il représente réellement l'objet tel qu'il nous apparaît dans l'espace et le temps, mais non tel qu'il est en lui-même."

EMMANUEL KANT, *Critique de la raison pure*, XVIII^e s.

5.1 État de l'art

Le problème de reconstruction 3D d'un objet à partir d'une séquence d'images a suscité beaucoup d'attention ces dernières années. Les approches existantes peuvent être classées en différentes catégories suivant le nombre d'images utilisées (2, 3, ou n images), suivant que la caméra est étalonnée ou non (i.e. les paramètres intrinsèques de la caméra sont connus ou non), suivant le modèle de caméra utilisé (affine ou perspectif), suivant le type de primitives considérées (points, droites, ellipses, contours, contours occultant...), et suivant le type de reconstruction obtenue : projective, affine ou euclidienne.

Reconstruction à partir de 2 images. Si l'on considère deux vues à partir de caméras non étalonnées, il est possible d'obtenir une reconstruction projective à partir de points appariés (Faugeras [Fau 92a], Hartley [Har 92b], Kara et al. [Kar 94], Longuet-Higgins [LH 81]). Si les caméras sont étalonnées, il est alors possible d'obtenir une reconstruction affine [Koe 91] ou euclidienne [Zhu 86].

Le problème de reconstruction à partir de 2 images peut être vu comme un problème de triangulation [Tos 87, Bea 94, Bea 95, Wen 88]. Hartley et Sturm ont proposé différentes méthodes de triangulation qui ont la propriété d'être invariantes soit à

des transformations affines, soit à des transformation projectives [Har 94a, Har 97, Stu 97].

Shashua [Sha 93] effectue une reconstruction projective via un invariant projectif appelé "profondeur projective". Cet invariant est calculé à partir de 4 points de référence.

Enfin, Quan [Qua 96a] s'est intéressé au problème de la reconstruction projective et euclidienne de coniques à partir de deux vues.

Reconstruction à partir de 3 images. Shashua [Sha 94, Sha 95b] a établi une relation trilinéaire reliant les coordonnées d'un point visible dans 3 images et le point tridimensionnel correspondant. Ces relations (valables pour chaque point) peuvent être utilisées pour effectuer une reconstruction projective.

Hartley [Har 94c] propose d'aligner le repère de reconstruction avec la première image pour simplifier les matrices de projection.

Quan [Qua 95a] effectue une reconstruction projective en utilisant les invariants projectifs à partir de 6 points et 3 images. Quan [Qua 97c] s'est également intéressé au problème de reconstruction affine à partir de droites, avec une caméra non étalonnée.

Reconstruction à partir de n images. Le problème de reconstruction à partir d'une séquence d'images avec un modèle perspectif de caméra non étalonnée conduit à des équations non linéaires. Mohr et al. [Moh 93c, Moh 93a], Boufama [Bou 94b, Bou 94a], Szeliski et Keng [Sze 94, Sze 95a] ont proposé des méthodes non linéaires (ajustement de faisceaux) pour effectuer une reconstruction projective avec une caméra non étalonnée. Boufama [Bou 93] a montré comment introduire des connaissances sur la scène pour effectuer le passage d'une reconstruction projective à une reconstruction euclidienne. Il existe également de nombreuses approches basées sur l'utilisation d'un filtre de Kalman initialisé à partir de 2 ou 3 images [Aya 87, Bro 86, Mat 89, May 90, Vi 94, Vi 95a, Zha 90b, Wen 93, Zha 90a].

D'autres méthodes de résolution du problème de reconstruction utilisent un modèle affine de caméra qui est un modèle approché (plus simple) que le modèle perspectif. Weinshall [Wei 93] a utilisé les invariants affines pour effectuer une reconstruction affine, puis a montré comment faire le passage à une reconstruction euclidienne. Tomasi, Poelman et Kanade effectuent une reconstruction euclidienne avec un modèle affine de caméra (orthographique, orthographique à l'échelle et paraperspectif). La méthode, basée sur la factorisation d'une matrice contenant les mesures images, sera reprise au paragraphe 5.3.2.2. La reconstruction est effectuée en deux étapes : (i) reconstruction affine, et (ii) passage d'une reconstruction affine à une reconstruction euclidienne. La méthode a ensuite été reprise par Quan [Qua 95b, Qua 96b] pour une caméra affine non étalonnée, par Morita [Mor 94, Mor 97] pour effectuer la reconstruction de manière incrémentale au fur et à mesure de l'acquisition des images, puis par Costeira [Cos 95] pour le cas de plusieurs objets ayant des mouvements indépendants. Hu et Ahuja [Hu 94] ont analysé les cas d'ambiguïté pour la reconstruction en utilisant un modèle affine de caméra. Debrunner et Ahuja [Deb 92] ont proposé une autre méthode de factorisation à partir d'un modèle orthographique à l'échelle. La différence, par rapport à Tomasi et Kanade, provient de la modélisation

du mouvement. Sturm et Triggs [Stu 96, Stu 97] ont étendu la méthode de factorisation pour effectuer une reconstruction projective à partir d'un modèle perspectif de caméra non étalonnée. L'algorithme se déroule en deux étapes : (i) calcul des facteurs d'échelle λ_{ij} (tels que $\lambda_{ij}\boldsymbol{m}_{ij} = \mathbf{P}_i\boldsymbol{M}_j$), et (ii) factorisation de la matrice de mesures normalisée.

Ainsi, avec une caméra étalonnée, il est possible d'effectuer une reconstruction euclidienne à un facteur d'échelle près en utilisant un modèle perspectif de caméra [Cui 90, Tay 91, Tay 95, Bro 91] ou affine [Deb 92, Tom 91a, Tom 92, Wei 93, Koe 91, Poe 94, Poe 95]. Avec une caméra non étalonnée, la reconstruction est déterminée à une transformation projective près [Fau 92a, Bou 93, Bou 94a, Har 93b, Vi 94], ou à une transformation affine près [Fau 92a, Har 93b]. Dans le cadre de ce rapport, nous nous intéressons au problème de la reconstruction euclidienne avec une caméra étalonnée.

Le modèle perspectif de caméra conduit généralement à des techniques de reconstruction non linéaires. Le problème conduit à utiliser des méthodes de minimisation non linéaires pour lesquelles une étape d'initialisation est nécessaire [Cui 90, Tay 91, Tay 95, Bro 91, Sze 94, Bou 93, Bou 94a, Har 93b, Vi 94]. Si l'estimation initiale est trop éloignée de la vraie solution, alors le processus de minimisation est lent ou converge vers une mauvaise solution. Les modèles affines de caméra conduisent, en général, à des méthodes de résolution linéaires [Deb 92, Tom 91a, Tom 92, Wei 93, Koe 91, Poe 94, Poe 95], mais la solution est définie à une transformation miroir près (ambiguïté de signe), et ces solutions ne sont que des approximations de la solution exacte. La figure 5.1 représente quatre formes 3D euclidiennes. Afin de les visualiser et de se rendre compte des côtés parallèles, celles-ci ont été projetées orthogonalement dans l'image. La première (en haut à gauche) représente le modèle théorique. La seconde (en haut à droite) est la reconstruction obtenue en utilisant un modèle perspectif de caméra. La troisième et la quatrième (en bas à gauche et en bas à droite) sont les deux reconstructions symétriques obtenues avec un modèle perspectif faible de caméra.

Les méthodes de reconstruction non linéaires classiques ont besoin d'une initialisation. Celle-ci est, en général, effectuée à partir d'une solution obtenue avec un modèle affine de caméra. Cependant, cette approche conduit à plusieurs problèmes : (i) elle ne prend pas en compte le lien existant entre le modèle perspectif et ses approximations linéaires (nous l'avons déjà mentionné au début du chapitre 2 sur le calcul de pose) ; (ii) rien ne prouve mathématiquement que le système non linéaire est initialisé correctement avec une solution obtenue par une méthode linéaire ; (iii) enfin, nous obtenons deux solutions avec un modèle affine et il n'est pas facile de savoir comment choisir l'une des deux solutions pour l'initialisation.

Nous proposons d'étendre l'algorithme itératif utilisé pour le calcul de pose, au problème de la reconstruction euclidienne à partir d'une séquence d'images, sans aucune connaissance à priori sur le mouvement de la caméra ou sur l'objet à reconstruire.

La méthode proposée dans ce chapitre effectue une reconstruction euclidienne avec un modèle affine de caméra à chaque pas d'itération, mais converge vers la solution obtenue avec un modèle perspectif. L'originalité de la méthode proposée est double : (i) elle étend l'algorithme de calcul de pose itératif présenté dans [Dem 93, Dem 95] et dans le chapitre précédent pour résoudre le problème de reconstruction à partir d'une séquence d'images, et (ii) c'est une généralisation de la méthode de factorisation [Tom 91a, Tom 92, Poe 94, Poe 95] et de la méthode des invariants affines [Wei 93]. La méthode itérative

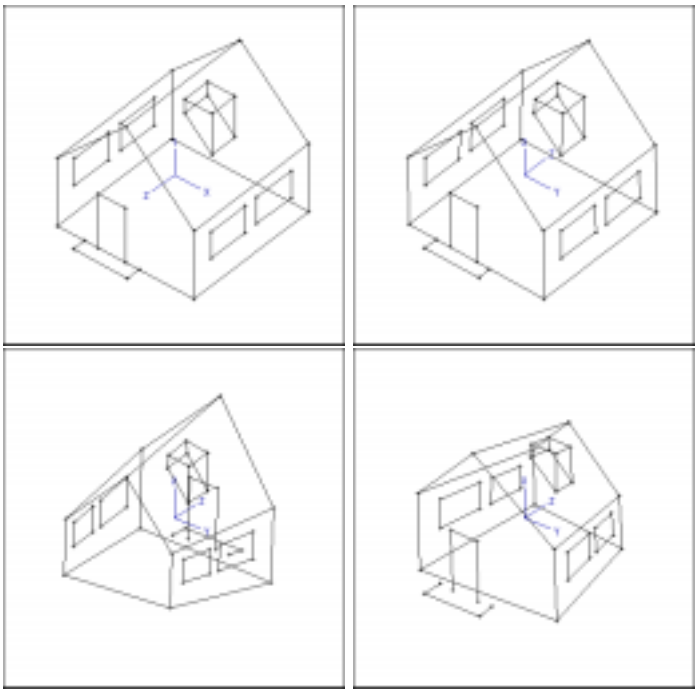


FIG. 5.1: *Cette figure représente un modèle* 3D *théorique (en haut à gauche), et trois reconstructions obtenues à partir d'une séquence de 10 images. La première reconstruction (en haut à droite) est obtenue à partir d'un modèle perspectif de caméra, en utilisant l'algorithme présenté dans la suite. La seconde reconstruction (en bas à gauche), et son inverse (en bas à droite) sont obtenues en utilisant un modèle perspectif faible de caméra et la méthode de factorisation proposée par Poelman et Kanade.*

de reconstruction que nous proposons ici possède un certain nombre de caractéristiques intéressantes.

- La méthode résout le problème d'ambiguïté de signe (solution miroir) inhérent aux modèles affines de caméra.
- Elle est rapide car elle converge en quelques itérations (en général 3 à 5 itérations), chaque itération ne nécessitant que des calculs algébriques simples. En particulier, il n'y a pas d'inversion de matrice comme dans le cas des techniques itératives de minimisation non linéaires.
- Nous démontrons que la qualité de reconstruction euclidienne obtenue n'est que faiblement influencée par l'erreur d'étalonnage de la caméra. Le paramètre le plus important est le rapport d'échelle entre la taille des pixels horizontaux et verticaux – rapport qui est donné par le constructeur et qui est connu comme étant très stable [Tsa 87].
- Nous pouvons utiliser, au choix, le modèle orthographique à l'échelle ou paraperspectif comme approximation à chaque pas d'itération.
- La méthode peut être combinée avec n'importe quelle méthode de reconstruction

affine. En particulier, nous montrons comment utiliser la méthode des invariants affines [Wei 93] ou la méthode de factorisation [Tom 91a, Tom 92, Poe 94, Poe 95].

Plan du chapitre

Le paragraphe 5.2 explique comment effectuer une reconstruction en utilisant un modèle perspectif de caméra, par itérations successives d'un algorithme de reconstruction utilisant un modèle orthographique à l'échelle ou paraperspectif. Le paragraphe 5.3 rappelle comment effectuer une reconstruction affine avec un modèle affine de caméra, en utilisant la méthode des invariants affines ou la méthode de factorisation. Il explique également comment passer d'une reconstruction affine à une reconstruction euclidienne. Le paragraphe 5.4 explique comment résoudre le problème de l'ambiguïté de la reconstruction (ambiguïté de signe) associé à un modèle affine de caméra. Le paragraphe 5.5 compare de manière théorique la complexité de la méthode itérative proposée par rapport à une méthode de minimisation non linéaire. Le paragraphe 5.6 explique comment traiter le problème des occultations. Enfin, le paragraphe 5.7 évalue la méthode sur des données synthétiques sur différents types de mouvements, puis sur des images réelles.

5.2 Reconstruction avec un modèle perspectif de caméra

Reprenons les notations des chapitres précédents, et considérons de nouveau un modèle perspectif de caméra. Soient $(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k})$ l'orientation et $\boldsymbol{t} = {}^t(\ t_x \quad t_y \quad t_z\ )$ la position de la caméra par rapport à l'objet.

Nous avons établi au paragraphe 2.2 deux couples d'équations équivalentes faisant intervenir les coordonnées normalisées des points images, les premières permettant de faire le lien entre le modèle orthographique à l'échelle de caméra et le modèle perspectif:

$$x_{ij}(1+\varepsilon_{ij}) - x_{0_i} = \boldsymbol{I}_i \cdot \boldsymbol{M}_j \tag{5.1}$$
$$y_{ij}(1+\varepsilon_{ij}) - y_{0_i} = \boldsymbol{J}_i \cdot \boldsymbol{M}_j \tag{5.2}$$

où

$$\boldsymbol{I}_i = \frac{\boldsymbol{i}_i}{t_{z_i}} \qquad \boldsymbol{J}_i = \frac{\boldsymbol{j}_i}{t_{z_i}} \qquad \boldsymbol{K}_i = \frac{\boldsymbol{k}_i}{t_{z_i}} \qquad x_{0_i} = \frac{t_{x_i}}{t_z} \qquad y_{0_i} = \frac{t_{y_i}}{t_{z_i}}$$

les secondes permettant de faire le lien entre le modèle paraperspectif de caméra et le modèle perspectif:

$$(x_{ij} - x_{0_i})(1+\varepsilon_{ij}) = \boldsymbol{I}_{p_i} \cdot \boldsymbol{M}_j \tag{5.3}$$
$$(y_{ij} - y_{0_i})(1+\varepsilon_{ij}) = \boldsymbol{J}_{p_i} \cdot \boldsymbol{M}_j \tag{5.4}$$

où

$$\boldsymbol{I}_{p_i} = \boldsymbol{I}_i - x_{0_i}\boldsymbol{K}_i = \frac{\boldsymbol{i}_i - x_{0_i}\,\boldsymbol{k}_i}{t_{z_i}} \tag{5.5}$$

$$\boldsymbol{J}_{p_i} = \boldsymbol{J}_i - y_{0_i}\boldsymbol{K}_i = \frac{\boldsymbol{j}_i - y_{0_i}\,\boldsymbol{k}_i}{t_{z_i}} \tag{5.6}$$

Comme dans le cas du calcul de pose, l'idée de base de notre méthode consiste à estimer les valeurs des ε_{ij} de manière incrémentale de façon à calculer les projections perspectives faibles ou paraperspectives des points M_j à partir des points images donnés qui

correspondent aux projections perspectives. Ainsi, le problème de reconstruction avec un modèle perspectif de caméra se réduit à effectuer une reconstruction itérative en utilisant un modèle perspectif faible ou paraperspectif de caméra.

Les deux équations (5.1) et (5.2), ou (5.3) et (5.4) peuvent s'écrire :

$$\underbrace{\boldsymbol{s}_{ij}}_{2\times 1} = \underbrace{\mathbf{R}_i}_{2\times 3}\underbrace{\boldsymbol{M}_j}_{3\times 1} \tag{5.7}$$

Pour le modèle orthographique à l'échelle, nous avons :

$$\boldsymbol{s}_{ij} = \begin{pmatrix} x_{ij}(1+\varepsilon_{ij}) - x_{0_i} \\ y_{ij}(1+\varepsilon_{ij}) - y_{0_i} \end{pmatrix} \tag{5.8}$$

$$\mathbf{R}_i = \begin{pmatrix} {}^t\boldsymbol{I}_i \\ {}^t\boldsymbol{J}_i \end{pmatrix}$$

et pour le modèle paraperspectif :

$$\boldsymbol{s}_{ij} = \begin{pmatrix} (x_{ij} - x_{0_i})(1+\varepsilon_{ij}) \\ (y_{ij} - y_{0_i})(1+\varepsilon_{ij}) \end{pmatrix} \tag{5.9}$$

$$\mathbf{R}_i = \begin{pmatrix} {}^t\boldsymbol{I}_{p_i} \\ {}^t\boldsymbol{J}_{p_i} \end{pmatrix}$$

Dans les équations précédentes, ε_{ij} est défini pour chaque point et pour chaque image :

$$\varepsilon_{ij} = \frac{\boldsymbol{k}_i \cdot \boldsymbol{M}_j}{t_{z_i}} \tag{5.10}$$

Le problème de reconstruction consiste à résoudre simultanément $2 \times n \times k$ équations de la forme de l'équation (5.7). Les inconnues sont les suivantes :

- les coordonnées 3D des points M_j ($3 \times n$ variables) ;
- les coefficients des matrices de projection $\mathbf{R}_i$ ($2 \times 3 \times k$ variables) ;
- les termes de corrections perspectives ε_{ij} ($n \times k$ variables).

Afin de pouvoir réestimer les ε_{ij}, il nous faut d'abord effectuer une reconstruction euclidienne avec un modèle affine de caméra. Cette étape peut se décomposer en deux phases : (i) effectuer une reconstruction affine, (ii) faire le passage de la reconstruction affine obtenue en une reconstruction euclidienne.

Le problème de la reconstruction affine revient à déterminer $\mathbf{R}_i$ et $\boldsymbol{M}_j$, pour tout i et pour tout j, cf. équation (5.7), lorsque les $\boldsymbol{s}_{ij}$ sont connus. Réécrivons l'équation (5.7) sous forme matricielle pour l'ensemble des points et des images :

$$\begin{pmatrix} \boldsymbol{s}_{11} & \dots & \boldsymbol{s}_{1n} \\ \vdots & & \vdots \\ \boldsymbol{s}_{k1} & \dots & \boldsymbol{s}_{kn} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_1 \\ \vdots \\ \mathbf{R}_k \end{pmatrix} \begin{pmatrix} \boldsymbol{M}_1 & \dots & \boldsymbol{M}_n \end{pmatrix}$$

c'est-à-dire :

$$\mathbf{\Sigma} = \mathbf{RM}$$

Cette étape détermine la reconstruction et le mouvement (rotation) à une transformation affine 3D près. En effet, pour toute matrice $\mathbf{U}$ inversible de taille 3×3, nous avons :

$$\begin{aligned} \mathbf{\Sigma} &= \mathbf{RM} && (5.11)\\ &= (\mathbf{RU})(\mathbf{U}^{-1}\mathbf{M}) && (5.12)\\ &= \mathbf{NS} && (5.13) \end{aligned}$$

Nous utiliserons les conventions suivantes :

- $\mathbf{R}$ et $\mathbf{M}$ représentent respectivement le mouvement et la reconstruction euclidiennes ;
- $\mathbf{N}$ et $\mathbf{S}$ représentent respectivement le mouvement et la reconstruction affines.

Pour effectuer le passage d'une reconstruction affine à une reconstruction euclidienne, il nous faut considérer des contraintes euclidiennes liées au mouvement de la caméra ou à l'objet visible dans les images. Comme nous utilisons ici une caméra étalonnée, nous pouvons utiliser les contraintes liées au mouvement rigide et au modèle de caméra (orthographique à l'échelle ou paraperspectif) [Tom 92], [Poe 94]. Par conséquent, l'étape 3 de l'algorithme fournit à la fois la reconstruction euclidienne ($\boldsymbol{M}_1 \quad \ldots \quad \boldsymbol{M}_n$) et le mouvement euclidien représenté par k matrices de la forme :

$$\begin{pmatrix} {}^t\boldsymbol{i}_i & t_{x_i} \\ {}^t\boldsymbol{j}_i & t_{y_i} \\ {}^t\boldsymbol{k}_i & t_{z_i} \\ {}^t\boldsymbol{0} & 1 \end{pmatrix}$$

Il existe plusieurs méthodes pour effectuer le passage d'une reconstruction affine à une reconstruction euclidienne, soit avec une caméra calibrée ou partiellement calibrée [Tom 92], [Wei 95] (orthographique à l'échelle), [Poe 94] (paraperspective), soit avec une caméra non calibrée [Qua 95b, Qua 96b]. Une fois la reconstruction euclidienne obtenue, nous pouvons réestimer les ε_{ij} pour tout i et pour tout j à partir de l'équation (5.10).

L'algorithme proposé effectue une reconstruction euclidienne avec un modèle perspectif de caméra par itérations successives d'une reconstruction euclidienne avec un modèle affine de caméra. Par conséquent, il est nécessaire d'avoir un aperçu du problème de reconstruction euclidienne avec un modèle affine de caméra comme nous l'expliquons au paragraphe suivant.

5.3 Reconstruction avec un modèle affine de caméra

Dans ce paragraphe, nous expliquons plus en détails comment effectuer une reconstruction euclidienne avec un modèle affine de caméra. Nous montrons que la méthode de reconstruction en utilisant les invariants affines est similaire à la méthode de factorisation. Ces deux méthodes utilisent un modèle linéaire de caméra et donnent une reconstruction 3D affine si nous avons au moins 2 vues et 4 points non coplanaires, et si le mouvement de la caméra n'est pas une translation pure. Bien que la méthode des invariants affines

nous permette de mieux analyser le problème (voir ci-dessous), la méthode de factorisation est plus commode d'utilisation car elle ne nécessite pas un choix de base explicite. La méthode des invariants affines a déjà été décrite dans [Koe 91] et [Wei 93]. La méthode de factorisation a été introduite par Tomasi et Kanade [Tom 92] (modèle orthographique de caméra), puis a été reprise par Poelman et Kanade [Poe 94] (modèle orthographique à l'échelle et paraperspectif).

5.3.1 Estimation de la translation

La première étape consiste à déterminer la position du point de référence (x_{0_i}, y_{0_i}) dans chaque image afin d'avoir une estimation de la matrice $\mathbf{\Sigma}$. Reprenons l'équation (5.11) :

$$\mathbf{\Sigma} = \mathbf{RM}$$

et introduisons $\boldsymbol{1} = {}^t(\ 1 \quad \dots \quad 1\ )$ le vecteur de dimension n composé de 1. En multipliant les deux membres de l'équation précédente à droite par le vecteur $\boldsymbol{1}$, et en remarquant que $\boldsymbol{M\,1} = \boldsymbol{0}$ (car les coordonnées 3D des points sont exprimées par rapport au centre de gravité de l'objet), nous obtenons :

$$\mathbf{\Sigma}\,\boldsymbol{1} = \boldsymbol{0}$$

Cette équation signifie que la somme des coefficients de chaque ligne de la matrice $\mathbf{\Sigma}$ est nulle. Nous en déduisons les coordonnées (x_{0_i}, y_{0_i}) de la projection du centre de gravité dans l'image, dont les formules sont identiques à celles obtenues dans le cadre du calcul de pose (voir paragraphe 3.2).

5.3.2 Reconstruction affine

5.3.2.1 Méthode des invariants affines

Considérons quatre points non coplanaires de la scène, M_0, M_1, M_2, et M_3 définissant ainsi une base affine de l'espace. Introduisons $\boldsymbol{S}_1$, $\boldsymbol{S}_2$, et $\boldsymbol{S}_3$ les trois vecteurs unitaires associés à la base affine : $\boldsymbol{S}_1 = {}^t(\ 1 \quad 0 \quad 0\ )$, $\boldsymbol{S}_2 = {}^t(\ 0 \quad 1 \quad 0\ )$, et $\boldsymbol{S}_3 = {}^t(\ 0 \quad 0 \quad 1\ )$. Soit $\boldsymbol{S}_j$ les coordonnées affines du point M_j dans cette base affine. $\boldsymbol{S}_j$ peut donc s'écrire comme combinaison linéaire des trois vecteurs de base :

$$\boldsymbol{S}_j = \alpha_j \boldsymbol{S}_1 + \beta_j \boldsymbol{S}_2 + \gamma_j \boldsymbol{S}_3$$

En combinant cette équation avec l'équation (5.13), nous obtenons :

$$\begin{aligned} \boldsymbol{s}_{ij} &= \alpha_j \mathbf{N}_i \boldsymbol{S}_1 + \beta_j \mathbf{N}_i \boldsymbol{S}_2 + \gamma_j \mathbf{N}_i \boldsymbol{S}_3 \\ &= \alpha_j \boldsymbol{s}_{i1} + \beta_j \boldsymbol{s}_{i2} + \gamma_j \boldsymbol{s}_{i3} \end{aligned}$$

Pour k images et pour chaque point j, avec $j \geq 4$, nous obtenons l'équation matricielle suivante :

$$\begin{pmatrix} \boldsymbol{s}_{11} & \boldsymbol{s}_{12} & \boldsymbol{s}_{13} \\ \vdots & \vdots & \vdots \\ \boldsymbol{s}_{k1} & \boldsymbol{s}_{k2} & \boldsymbol{s}_{k3} \end{pmatrix} \begin{pmatrix} \alpha_j \\ \beta_j \\ \gamma_j \end{pmatrix} = \begin{pmatrix} \boldsymbol{s}_{1j} \\ \vdots \\ \boldsymbol{s}_{kj} \end{pmatrix}$$

ou de manière plus compacte :

$$\underbrace{\mathbf{B}}_{2k\times 3}\ \underbrace{\boldsymbol{S}_j}_{3\times 1} = \underbrace{\boldsymbol{\Sigma}_j}_{2k\times 1}$$

La matrice pseudo-inverse de $\mathbf{B}$ peut être calculée si le rang de $\mathbf{B}$ est égal à 3. Cette condition est vérifiée si :

- le nombre d'images (k) est supérieur ou égal à 2 ;
- les projections des vecteurs de base ne sont pas colinéaires dans l'image ;
- le mouvement de la caméra n'est pas réduit à un mouvement de translation pure ou à une rotation autour de l'axe optique.

Si ces conditions sont vérifiées, nous pouvons calculer les coordonnées affines de tous les points j de l'espace ($j \geq 4$) :

$$\boldsymbol{S}_j = \mathbf{B}^\dagger \boldsymbol{\Sigma}_j$$

L'exposant † indique qu'il s'agit de la pseudo-inverse de la matrice. L'expression précédente peut s'écrire pour tout j, $j \geq 4$:

$$\mathbf{S}^* = \mathbf{B}^\dagger \boldsymbol{\Sigma}^*$$

où

$$\mathbf{S}^* = \begin{pmatrix} \boldsymbol{S}_4 & \dots & \boldsymbol{S}_n \end{pmatrix}$$
$$\boldsymbol{\Sigma}^* = \begin{pmatrix} \boldsymbol{\Sigma}_4 & \dots & \boldsymbol{\Sigma}_n \end{pmatrix}$$

Ainsi, la reconstruction affine est définie par la matrice $\mathbf{S}$ de taille $3 \times n$:

$$\mathbf{S} = \begin{pmatrix} \boldsymbol{S}_1 & \boldsymbol{S}_2 & \boldsymbol{S}_3 & \mathbf{S}^* \end{pmatrix}$$

Il ne reste qu'à déterminer les matrices de transformations affines $\mathbf{N}_i$. En reprenant l'équation (5.13) :

$$\boldsymbol{\Sigma} = \mathbf{NS}$$

nous en déduisons les matrices de projections affines :

$${}^t\mathbf{N} = ({}^t\mathbf{S})^\dagger {}^t\boldsymbol{\Sigma}$$

5.3.2.2 Méthode de factorisation

D'après l'équation (5.13), la matrice $\boldsymbol{\Sigma}$ peut s'écrire comme étant le produit de deux matrices, l'une représentant l'orientation de la caméra, l'autre dépendant des points de la scène :

$$\boldsymbol{\Sigma} = \mathbf{NS}$$

Le problème de calcul de $\mathbf{N}$ et $\mathbf{S}$ se pose comme étant un problème de factorisation. Tomasi et Kanade [Tom 91a, Tom 92] ont remarqué que les matrices $\mathbf{N}$ et $\mathbf{S}$ peuvent être calculées simultanément en effectuant une décomposition en valeurs singulières de la matrice $\boldsymbol{\Sigma}$ de taille $2k \times n$:

$$\boldsymbol{\Sigma} = \mathbf{O}_1 \mathbf{D} \mathbf{O}_2$$

telle que ${}^t\mathbf{O}_1\mathbf{O}_1 = {}^t\mathbf{O}_2\mathbf{O}_2 = \mathbf{O}_2{}^t\mathbf{O}_2 = \boldsymbol{\mathcal{I}}_n$ et les valeurs singulières soient triées par ordre décroissant sur la diagonale de la matrice $\mathbf{D}$: $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n$.

La matrice $\mathbf{N}$ est de taille $2k \times 3$, et la matrice $\mathbf{S}$ est de taille $3 \times n$. Par conséquent, la matrice de mesures $\mathbf{\Sigma}$ est théoriquement de rang inférieur ou égal à trois :

$$\text{rang } \mathbf{\Sigma} \leq 3$$

En pratique, le rang de $\mathbf{\Sigma}$ dépend du rang de la matrice $\mathbf{D}$ qui est de taille $n \times n$. À cause d'instabilités numériques et du bruit dans les images, le rang de $\mathbf{\Sigma}$ peut être supérieur à 3. Tomasi et Kanade ont proposé de résoudre le problème du rang en tronquant la matrice $\mathbf{D}$ et en ne conservant que les trois plus grandes valeurs singulières, les autres valeurs singulières étant dues au bruit dans la matrice de mesures. En partitionnant les matrices $\mathbf{O}_1$, $\mathbf{D}$, et $\mathbf{O}_2$ de la manière suivante :

$$\begin{aligned}
\mathbf{O}_1 &= \underbrace{\begin{pmatrix} \mathbf{O}'_1 \end{pmatrix}}_{3} \underbrace{\begin{pmatrix} \mathbf{O}''_1 \end{pmatrix}}_{n-3} \Big\}_{2k} \\
\mathbf{D} &= \underbrace{\begin{pmatrix} \mathbf{D}' \\ \mathbf{0} \end{pmatrix}}_{3} \underbrace{\begin{pmatrix} \mathbf{0} \\ \mathbf{D}'' \end{pmatrix}}_{n-3} \begin{matrix} \}_3 \\ \}_{n-3} \end{matrix} \\
\mathbf{O}_2 &= \underbrace{\begin{pmatrix} \mathbf{O}'_2 \\ \mathbf{O}''_2 \end{pmatrix}}_{n} \begin{matrix} \}_3 \\ \}_{n-3} \end{matrix}
\end{aligned}$$

la matrice $\mathbf{\Sigma}$ se décompose en :

$$\mathbf{\Sigma} = \mathbf{O}'_1 \mathbf{D}' \mathbf{O}'_2 + \mathbf{O}''_1 \mathbf{D}'' \mathbf{O}''_2$$

où $\mathbf{D}'$ est une matrice diagonale de taille 3×3 contenant les trois plus grandes valeurs singulières de $\mathbf{D}$. Nous en déduisons ainsi le mouvement et la reconstruction affines :

$$\begin{aligned}
\mathbf{N} &= \mathbf{O}'_1 (\mathbf{D}')^{1/2} \\
\mathbf{S} &= (\mathbf{D}')^{1/2} \mathbf{O}'_2
\end{aligned}$$

Notons que Tomasi et Kanade ne furent pas les premiers à proposer une méthode consistant à factoriser une matrice, puis à tronquer les matrices obtenues. Hartley [Har 95a] rappelle que la méthode est due à Tsai et Huang [Tsa 84] qui ont introduit une méthode de factorisation pour extraire la translation et la rotation à partir de la matrice essentielle – matrice de taille 3×3 et de rang 2. Hartley [Har 95a] rappelle également que la matrice tronquée obtenue (qui revient à fixer un certain nombre de valeurs singulières à zéro) est la meilleure approximation de la matrice initiale par une matrice de rang donnée au sens de la norme de Frobenius.

5.3.3 D'une reconstruction affine à une reconstruction euclidienne

Nous venons de présenter dans le paragraphe précédent comment effectuer une reconstruction affine. Le problème est maintenant de faire le passage à une reconstruction euclidienne en introduisant des contraintes euclidiennes liées au modèle de caméra utilisé.

Nous cherchons donc à déterminer une matrice $\mathbf{U}$ de taille 3×3 inversible qui transforme la reconstruction affine en une reconstruction euclidienne :

$$\begin{pmatrix} \boldsymbol{M}_1 & \dots & \boldsymbol{M}_n \end{pmatrix} = \mathbf{U}^{-1} \begin{pmatrix} \boldsymbol{S}_1 & \dots & \boldsymbol{S}_n \end{pmatrix}$$

et le mouvement affine en un mouvement rigide :

$$\begin{pmatrix} \mathbf{R}_1 \\ \vdots \\ \mathbf{R}_k \end{pmatrix} = \begin{pmatrix} \mathbf{N}_1 \\ \vdots \\ \mathbf{N}_k \end{pmatrix} \mathbf{U}$$

Pour éviter toute confusion, $\mathbf{S}$ et $\mathbf{N}$ représentent la reconstruction et le mouvement affines, et $\mathbf{M}$ et $\mathbf{R}$ représentent la reconstruction et le mouvement euclidiens.

La méthode de factorisation décrite dans le paragraphe précédent ne fournit pas une décomposition unique de la matrice $\mathbf{\Sigma}$. Tomasi & Kanade [Tom 92] ont proposé une solution pour un modèle orthographique de caméra. Poelman et Kanade [Poe 95], et Weinshall et Tomasi [Wei 95] ont proposé une solution pour un modèle orthographique à l'échelle de caméra. Enfin, Poelman et Kanade [Poe 94, Poe 95] ont proposé une solution pour un modèle paraperspectif. La méthode que nous décrivons pour retrouver une reconstruction euclidienne avec un modèle paraperspectif est une approche différente de celle décrite dans [Poe 94, Poe 95].

5.3.3.1 Avec un modèle perspectif faible de caméra

Ce paragraphe explique comment calculer la matrice $\mathbf{U}$ pour un modèle orthographique à l'échelle de caméra. Rappelons l'expression des matrices $\mathbf{R}_i$ pour ce modèle :

$$\mathbf{R}_i = \begin{pmatrix} {}^t\boldsymbol{I}_i \\ {}^t\boldsymbol{J}_i \end{pmatrix}$$

où $\boldsymbol{I}_i = \boldsymbol{i}_i / t_{z_i}$ et $\boldsymbol{J}_i = \boldsymbol{j}_i / t_{z_i}$. Par conséquent, les lignes de la matrice $\mathbf{R}_i$ vérifient, pour tout i :

$$\|\boldsymbol{I}_i\| = \|\boldsymbol{J}_i\| \tag{5.14}$$

$$\boldsymbol{I}_i \cdot \boldsymbol{J}_i = 0 \tag{5.15}$$

Notons ${}^t\boldsymbol{a}_i$ et ${}^t\boldsymbol{b}_i$ les vecteurs lignes de la matrice $\mathbf{N}_i$. Nous avons donc les relations :

$$\begin{aligned} {}^t\boldsymbol{I}_i &= {}^t\boldsymbol{a}_i\mathbf{U} \\ {}^t\boldsymbol{J}_i &= {}^t\boldsymbol{b}_i\mathbf{U} \end{aligned}$$

Les contraintes (5.14) et (5.15) se réécrivent :

$${}^t\boldsymbol{a}_i\mathbf{U}{}^t\mathbf{U}\boldsymbol{a}_i - {}^t\boldsymbol{b}_i\mathbf{U}{}^t\mathbf{U}\boldsymbol{b}_i = 0 \tag{5.16}$$

$${}^t\boldsymbol{a}_i\mathbf{U}{}^t\mathbf{U}\boldsymbol{b}_i = 0 \tag{5.17}$$

Ces contraintes sont homogènes par rapport aux coefficients de la matrice $\mathbf{U}$. Pour éviter la solution nulle triviale, nous fixons le facteur d'échelle en imposant :

$$t_{z_1} = 1$$

On obtient ainsi deux contraintes supplémentaires (en fait l'une des deux est redondante avec la contrainte (5.16)) :

$$ {}^t\boldsymbol{a}_1 \mathbf{U}\,{}^t\mathbf{U}\boldsymbol{a}_1 = 1 \tag{5.18} $$

$$ {}^t\boldsymbol{b}_1 \mathbf{U}\,{}^t\mathbf{U}\boldsymbol{b}_1 = 1 \tag{5.19} $$

Ces contraintes sont non linéaires par rapport aux coefficients de la matrice $\mathbf{U}$. En posant :

$$ \mathbf{Q} = \mathbf{U}\,{}^t\mathbf{U} $$

les équations (5.16), (5.17), (5.18) et (5.18) deviennent linéaires. $\mathbf{Q}$ étant une matrice 3×3 symétrique semi-définie positive, nous avons donc 6 inconnues. D'autre part, nous avons $2k + 1$ contraintes indépendantes, il faut donc au moins 3 images afin d'estimer la matrice $\mathbf{Q}$. Après avoir calculé cette matrice, nous pouvons déduire $\mathbf{U}$ en factorisant la matrice $\mathbf{Q}$ (décomposition de Choleski ou décomposition en valeurs propres). Nous montrerons cependant au paragraphe 5.4 qu'il y a une ambiguïté liée à la factorisation d'une matrice semi-définie positive. Ce problème est la cause de l'ambiguïté de la reconstruction associée à un modèle affine de caméra.

Après avoir déterminé la matrice $\mathbf{U}$, nous pouvons en déduire la reconstruction euclidienne $\boldsymbol{M}_j = \mathbf{U}^{-1}\boldsymbol{S}_j$ et le mouvement rigide $\mathbf{R}_i = \mathbf{N}_i\mathbf{U}$. Pour finir, nous calculons l'orientation $(\boldsymbol{i}_i,\ \boldsymbol{j}_i,\ \boldsymbol{k}_i)$ et la position $(t_{x_i},\ t_{y_i},\ t_{z_i})$ en utilisant les formules données au paragraphe 2.2.1.

5.3.3.2 Avec un modèle paraperspectif de caméra

Considérons maintenant un modèle paraperspectif de caméra. Dans ce cas, les matrices $\mathbf{R}_i$ s'écrivent :

$$ \mathbf{R}_i = \begin{pmatrix} {}^t\boldsymbol{I}_{p_i} \\ {}^t\boldsymbol{J}_{p_i} \end{pmatrix} $$

où

$$ \boldsymbol{I}_{p_i} = \frac{\boldsymbol{i}_i - x_{0_i}\boldsymbol{k}_i}{t_{z_i}} \quad \text{et} \quad \boldsymbol{J}_{p_i} = \frac{\boldsymbol{j}_i - y_{0_i}\boldsymbol{k}_i}{t_{z_i}} $$

Les contraintes euclidiennes nous permettant de calculer la matrice $\mathbf{U}$ sont donc les suivantes :

$$ \frac{\|\boldsymbol{I}_{p_i}\|^2}{1 + x_{0_i}^2} = \frac{\|\boldsymbol{J}_{p_i}\|^2}{1 + y_{0_i}^2} $$

et

$$ \boldsymbol{I}_{p_i} \cdot \boldsymbol{J}_{p_i} = \frac{x_{0_i} y_{0_i}}{2} \left(\frac{\|\boldsymbol{I}_{p_i}\|^2}{1 + x_{0_i}^2} + \frac{\|\boldsymbol{J}_{p_i}\|^2}{1 + y_{0_i}^2} \right) $$

Comme précédemment, en notant $\boldsymbol{a}_i$ et $\boldsymbol{b}_i$ les vecteurs lignes de $\mathbf{N}_i$, nous obtenons $2k$ contraintes pour la matrice $\mathbf{U}$:

$$ \frac{{}^t\boldsymbol{a}_i \mathbf{U}\,{}^t\mathbf{U}\boldsymbol{a}_i}{1 + x_{0_i}^2} - \frac{{}^t\boldsymbol{b}_i \mathbf{U}\,{}^t\mathbf{U}\boldsymbol{b}_i}{1 + y_{0_i}^2} = 0 $$

$$ {}^t\boldsymbol{a}_i \mathbf{U}\,{}^t\mathbf{U}\boldsymbol{b}_i = \frac{x_{0_i} y_{0_i}}{2} \left(\frac{{}^t\boldsymbol{a}_i \mathbf{U}\,{}^t\mathbf{U}\boldsymbol{a}_i}{1 + x_{0_i}^2} + \frac{{}^t\boldsymbol{b}_i \mathbf{U}\,{}^t\mathbf{U}\boldsymbol{b}_i}{1 + y_{0_i}^2} \right) $$

Nous pouvons également fixer le facteur d'échelle $t_{z_1} = 1$ qui donne les contraintes supplémentaires suivantes :

$$\begin{aligned} \|\boldsymbol{I}_{p_1}\|^2 &= 1 + x_{0_1}^2 \\ \|\boldsymbol{J}_{p_1}\|^2 &= 1 + y_{0_1}^2 \end{aligned}$$

que l'on peut également écrire :

$$\begin{aligned} {}^t\boldsymbol{a}_1 \mathbf{U}\,{}^t\mathbf{U} \boldsymbol{a}_1 &= 1 + x_{0_1}^2 \\ {}^t\boldsymbol{b}_1 \mathbf{U}\,{}^t\mathbf{U} \boldsymbol{b}_1 &= 1 + y_{0_1}^2 \end{aligned}$$

On obtient ainsi $2k + 1$ contraintes indépendantes pour 6 inconnues. Par conséquent, comme pour un modèle orthographique à l'échelle, il nous faut au moins 3 images pour pouvoir effectuer le passage d'une reconstruction affine à une reconstruction euclidienne.

Nous pouvons ensuite déterminer l'orientation ($\boldsymbol{i}_i$, $\boldsymbol{j}_i$, $\boldsymbol{k}_i$) et la position (t_{x_i}, t_{y_i}, t_{z_i}) de chaque caméra comme nous l'avons expliqué au paragraphe 2.2.2.

5.4 Résolution de l'ambiguïté de signe

L'algorithme décrit dans le paragraphe 5.3.3 effectue une reconstruction euclidienne avec un modèle perspectif de caméra de manière itérative, en supposant à chaque pas d'itération un modèle orthographique à l'échelle ou paraperspectif de caméra. Dans ce paragraphe, nous montrons comment modifier cet algorithme itératif de manière à tenir compte du problème d'ambiguïté de la reconstruction propre aux modèles affines de caméra. En effet, considérons de nouveau la reconstruction et le mouvement affines décrits dans le paragraphe précédent. Le principe consiste à estimer la transformation $\mathbf{U}$ permettant de convertir la structure affine en une structure euclidienne. Cette transformation doit être calculée à partir de la décomposition d'une matrice symétrique semi-définie positive $\mathbf{Q}$:

$$\mathbf{Q} = \mathbf{U}\,{}^t\mathbf{U}$$

Nous avons au moins deux manières pour déterminer la matrice $\mathbf{U}$:

1. Par une décomposition en valeurs propres, $\mathbf{Q}$ peut s'écrire :

   $$\mathbf{Q} = \mathbf{O}\mathbf{D}\,{}^t\mathbf{O}$$

   où $\mathbf{O}$ est une matrice orthogonale contenant les vecteurs propres de $\mathbf{Q}$, et $\mathbf{D}$ est une matrice diagonale contenant les valeurs propres de $\mathbf{Q}$. Puisque les valeurs propres d'une matrice symétrique semi-définie positive sont toutes réelles et positives (ou nulles), on peut écrire $\mathbf{Q}$ de la façon suivante :

   $$\mathbf{Q} = (\mathbf{O}\mathbf{D}^{1/2})^t(\mathbf{O}\mathbf{D}^{1/2}) = \mathbf{K}\,{}^t\mathbf{K}$$

2. De manière équivalente, nous pouvons faire une décomposition Choleski de la matrice $\mathbf{Q}$:

   $$\mathbf{Q} = \mathbf{L}\,{}^t\mathbf{L}$$

   où $\mathbf{L}$ est une matrice triangulaire inférieure.

Soit $\mathbf{H}$ une matrice non singulière telle que :

$$\mathbf{L} = \mathbf{K}\mathbf{H}$$

alors, nous avons :

$$\begin{aligned} \mathbf{Q} &= \mathbf{L}^t\mathbf{L} \\ &= \mathbf{K}\mathbf{H}^t\mathbf{H}^t\mathbf{K} \\ &= \mathbf{K}^t\mathbf{K} \end{aligned}$$

Nous en concluons que la matrice $\mathbf{H}$ est orthogonale. L'orthogonalité de $\mathbf{H}$ a aussi été mentionnée dans [Wei 95], mais sans preuve formelle. Par conséquent, la matrice $\mathbf{H}$ représente soit une rotation, soit un antidéplacement (son déterminant est $+1$ ou -1). Il y a donc deux classes de formes reconstruites possibles :

- une reconstruction *directe* définie à une rotation près ;
- une reconstruction *inversée* obtenue en appliquant un antidéplacement sur la reconstruction directe.

Puisque la reconstruction est définie à une rotation près, nous pouvons supposer, sans perte de généralité, que la matrice d'antidéplacement est $(-\boldsymbol{\mathcal{I}})$ où $\boldsymbol{\mathcal{I}}$ est la matrice identité. Par conséquent, les deux solutions pour la reconstruction s'écrivent :

$$\boldsymbol{\Sigma} = \mathbf{N}\mathbf{S} = (-\mathbf{N})(-\mathbf{S})$$

À cause de cette ambiguïté de signe, nous avons deux solutions pour les ε_{ij} à chaque pas d'itération.

Considérons un modèle paraperspectif (un raisonnement similaire s'applique pour un modèle orthographique à l'échelle). Les vecteurs $\boldsymbol{k}_i$ sont calculés à partir de l'équation (2.29). Cette équation peut être utilisée soit avec les vecteurs $\boldsymbol{I}_p$ et $\boldsymbol{J}_p$ (la première solution) ou $-\boldsymbol{I}_p$ et $-\boldsymbol{J}_p$ (la seconde solution). Par conséquent, nous obtenons deux solutions distinctes, soient $\boldsymbol{k}_i^1$ et $\boldsymbol{k}_i^2$. Les deux solutions pour ε_{ij} correspondent à $\boldsymbol{k}_i^1$ et $\boldsymbol{M}_j$, et à $\boldsymbol{k}_i^2$ et $-\boldsymbol{M}_j$:

$$\varepsilon_{ij}^{1,2} = \pm\frac{\boldsymbol{k}_i^{1,2} \cdot \boldsymbol{M}_j}{t_{z_i}}$$

À chaque pas d'itération de l'algorithme, nous calculons deux valeurs pour ε_{ij}. Par conséquent, après N itérations, il y aura 2^N solutions possibles. Ces solutions ne sont cependant pas forcément cohérentes avec les données images, et une technique simple de vérification permet de vérifier cette cohérence afin d'éviter l'explosion du nombre de solutions. Finalement, une seule solution est retenue.

La première itération de l'algorithme induit deux solutions – une reconstruction "positive" $\mathbf{S}^{(1)}$ et une solution "négative" $\mathbf{R}^{(1)}$ ($\mathbf{R}^{(1)} = -\mathbf{S}^{(1)}$) – qui sont toutes les deux retenues. Pour chaque itération suivante, nous obtenons quatre formes reconstruites possibles : $\mathbf{S}_1^{(i)}$ et $\mathbf{S}_2^{(i)}$ issues de la solution positive, et $\mathbf{R}_1^{(i)}$ et $\mathbf{R}_2^{(i)}$ issues de la solution négative. Nous conservons les reconstructions $\mathbf{S}$ et $\mathbf{R}$ les plus cohérentes avec les reconstructions obtenues aux itérations précédentes. La solution finale est obtenue en choisissant la reconstruction (parmi les deux choix possibles) dont les points reprojetés dans les images

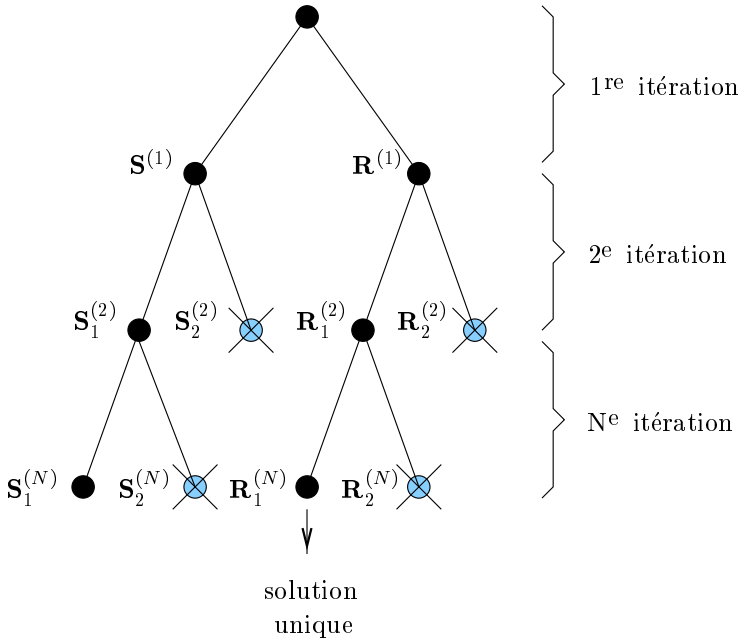


FIG. 5.2: *Stratégie pour le choix de la solution correcte.*

sont les plus proches des points images donnés. Le processus de sélection de la solution est illustré figure 5.2.

Le processus de sélection des solutions que nous venons de décrire est illustré figures 5.1 (en bas), et 5.3. L'algorithme calcule d'abord deux reconstructions à partir d'un modèle affine de caméra (figure 5.1 en bas) : une reconstruction "positive" $\mathbf{S}^{(1)}$ (figure 5.1 en bas à gauche) et une reconstruction "négative" $\mathbf{R}^{(1)}$ (figure 5.1 en bas à droite). À la convergence, nous obtenons deux reconstructions possibles issues de $\mathbf{S}^{(i)}$ (figure 5.3 à gauche) et de $\mathbf{R}^{(i)}$ (figure 5.3 à droite). La solution issue des reconstructions de la famille $\mathbf{R}$ (à droite) est la plus cohérente avec les points images donnés, elle est donc sélectionnée comme étant la solution unique pour le problème de reconstruction avec un modèle perspectif de caméra.

5.5 Comparaison avec une méthode de minimisation non linéaire

Par le passé, un grand nombre d'auteurs ont essayé d'effectuer une reconstruction euclidienne en utilisant des techniques non linéaires. Dans sa forme la plus générale, le problème revient à minimiser la fonction d'erreur [Sze 94] :

$$f(\boldsymbol{X}) = \sum_{ij} \left((x_{ij} - \widehat{x}_{ij})^2 + (y_{ij} - \widehat{y}_{ij})^2 \right)$$

où x_{ij} et y_{ij} sont les coordonnées d'un point dans l'image et $\widehat{x}_{ij}$ et $\widehat{y}_{ij}$ sont les coordonnées de la projection du point reconstruit en utilisant un modèle perspectif de caméra. $\widehat{x}_{ij}$ et $\widehat{y}_{ij}$ sont données par les équations (2.5) et (2.6).

Pour n points et k images, la fonction d'erreur ci-dessus a $2 \times n \times k$ termes positifs au carré. Le vecteur $\boldsymbol{X}$ regroupe les inconnues du problème : $3 \times n$ coordonnées et $6 \times k$

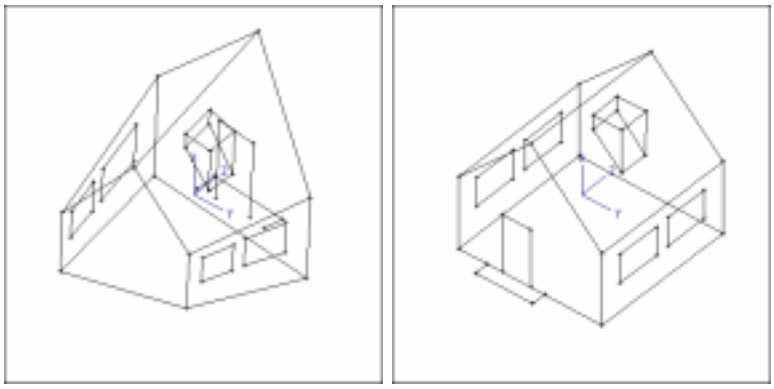


FIG. 5.3: *Résolution du problème d'inversion de la solution : les deux reconstructions initiales (obtenues à la première itération) donnent chacune, à la convergence de l'algorithme, une reconstruction correspondant à un modèle perspectif de caméra, mais seule l'une d'entre elles coïncide avec les points images donnés.*

paramètres liés au mouvement. Nous recherchons donc une valeur de $\boldsymbol{X}$ qui minimise la fonction d'erreur. Le jacobien de $f(\boldsymbol{X})$ est une matrice $m \times p$, et nous avons :

$$\begin{aligned} m &= 2 \times n \times k \\ p &= 3 \times n + 6 \times k \end{aligned}$$

Les méthodes non linéaires recherchent le minimum de façon incrémentale. À chaque itération, le système suivant doit être résolu de manière à calculer $d\boldsymbol{X}$, afin de remplacer ensuite $\boldsymbol{X}$ par $\boldsymbol{X} + d\boldsymbol{X}$:

$$^t\mathbf{J}(\boldsymbol{X})\,\mathbf{J}(\boldsymbol{X})\,d\boldsymbol{X} = \boldsymbol{b}$$

Par conséquent, la complexité à chaque itération est dominée par la complexité d'inversion d'une matrice symétrique définie positive – le hessien. La taille de cette matrice est $p \times p$ et dépend de n et k. De plus, en prenant en compte le fait que le hessien est une matrice bande, la complexité de l'inversion est de l'ordre de p^3+8^2+p opérations flottantes [Gol 89]. En remplaçant p par son expression, on obtient la complexité suivante :

$$27\,n^3 + 162\,n^2k + 324\,nk^2 + 216\,k^3 + 72\,n^2 + 288\,nk + 288\,k^2 + 3\,n + 6\,k$$

En ne gardant que les termes à l'ordre 3, on obtient finalement :

$$27\,n^3 + 162\,n^2k + 324\,nk^2 + 216\,k^3 \tag{5.20}$$

Afin de comparer notre méthode itérative avec les méthodes non linéaires, calculons la complexité d'une itération de notre méthode. La partie la plus consommatrice de temps est la décomposition en valeurs singulières de la matrice $\boldsymbol{\Sigma}$ de taille $2k \times n$. La complexité de la décomposition en valeurs singulières est [Gol 89] :

$$22\,n^3 + 8\,nk^2 \tag{5.21}$$

La complexité de la méthode de factorisation et des méthodes de minimisation non linéaires sont récapitulées dans le tableau 5.1 pour trois cas : le nombre d'images (k) est

grand par rapport au nombre de points (n), le nombre d'images est sensiblement égal au nombre de points, et le nombre d'images est petit par rapport au nombre de points.

Les comparaisons sont données pour une itération. Nous pouvons conclure que la méthode proposée est intrinsèquement plus rapide qu'une méthode de minimisation non linéaire. Remarquons que la complexité des méthodes non linéaires augmente très rapidement lorsque le nombre d'images est plus grand que le nombre de points.

Méthode	$k \approx n/10$	$k \approx n$	$k \approx 10n$
Factorisation	$22n^3$	$30n^3$	$822n^3$
Non linéaire (une itération)	$46n^3$	$729n^3$	$250000n^3$

TAB. 5.1: *Ce tableau indique le nombre d'opérations flottantes en fonction du nombre de points (n) et du nombre d'images (k). Si le nombre d'images est petit par rapport au nombre de points, alors les deux méthodes sont de complexités comparables. Cependant, si le nombre d'images augmente, les méthodes de minimisation non linéaires demandent beaucoup plus de calculs.*

5.6 Traitement des occultations

Le principal problème des méthodes de factorisation est la nécessité d'avoir une matrice de mesures $\mathbf{\Sigma}$ complète, sans données manquantes. Ces points manquants sont dus soit aux occultations, soit à un échec lors de la mise en correspondance. En effet, il est nécessaire d'avoir une matrice pleine pour pouvoir effectuer une décomposition en valeurs singulières. Résoudre ce problème n'est pas simple et les méthodes existantes ont chacune des inconvénients.

- Tomasi et Kanade [Tom 91a] proposent d'effectuer la factorisation d'une sous-matrice de la matrice de mesures, sans données manquantes. On obtient ainsi la reconstruction d'un sous-ensemble de points et la détermination de la position et orientation d'un sous-ensemble d'images. On peut ensuite estimer les points manquants dans les autres images de proche en proche. L'inconvénient de cette approche vient du fait que la qualité du résultat provient de la factorisation initiale. Le choix de la sous-matrice a factoriser – qui doit être la plus grande possible – n'est pas simple : il s'agit d'un problème NP-complet dans le cas général [Jac 97a].

- Wiberg [Wib 76], Shum et al. [Shu 94, Shu 95] ont proposé une méthode basée sur une analyse en composantes principales des données incomplètes. Il est cependant nécessaire de disposer d'une initialisation de la structure ou du mouvement.

- Récemment, Jacobs [Jac 97a] a montré comment déterminer une matrice de rang r approchant au mieux une matrice donnée incomplète et de même rang r. L'avantage de la méthode est la prise en compte simultanée de toutes les données et l'absence d'initialisation. Cependant, cette méthode est complexe en temps de calcul.

Nous proposons ici une solution similaire à celle proposée par Tomasi et Kanade [Tom 91a], en utilisant un modèle perspectif de caméra. Le principe de la méthode est

le suivant.

1. Effectuer une factorisation, puis le passage à une reconstruction euclidienne, à partir de la plus grande sous-matrice de $\mathbf{\Sigma}$ possible (maximiser le nombre de coefficients de la matrice). Le choix de la sous-matrice peut être effectué de manière simple en tenant compte des propriétés de l'algorithme de poursuite de points présenté au chapitre 1. Si l'extraction de points n'est effectuée qu'à la première image, la première image est celle qui contient le plus de points. On extrait donc une sous-matrice de $\mathbf{\Sigma}$ en commençant à la première image, et en considérant le plus grand nombre possible d'images consécutives.

2. Estimer de proche en proche les coordonnées 3D de chaque point, et la position et l'orientation des images qui n'ont pas été estimées à la première étape.

   - *Estimation d'un point* 3D *visible dans un certain nombre d'images* : résoudre un système d'équations linéaires dont les équations sont de la forme :

$$\begin{aligned}(\boldsymbol{I}_i - x_{ij}\boldsymbol{K}_i)\cdot\boldsymbol{M}_j &= x_{ij} - x_{0_i}\\ (\boldsymbol{J}_i - y_{ij}\boldsymbol{K}_i)\cdot\boldsymbol{M}_j &= y_{ij} - y_{0_i}\end{aligned}$$

   - *Estimation de la position et orientation d'une image* : effectuer un calcul de pose.

5.7 Résultats expérimentaux

5.7.1 Données synthétiques

Dans ce paragraphe, nous étudions les performances des méthodes itératives et nous les comparons avec la méthode de factorisation. Deux types de résultats sont analysés :

1. la qualité de la reconstruction tridimensionnelle en fonction de différents types de mouvement en présence de bruit gaussien sur les pixels ;

2. la convergence des algorithmes itératifs en fonction de différents types de mouvement.

Dans toutes les expérimentations de ce paragraphe, nous nous plaçons dans les conditions suivantes.

- Les paramètres intrinsèques de la caméra sont : $u_c = v_c = 256$, $\alpha_u = \alpha_v = 1000$.

- On effectue la reconstruction à partir de 15 images et 42 points (les données synthétiques utilisées sont représentées sur la figure 5.1).

- Pour tous les types de mouvements considérés, la variation angulaire entre deux vues consécutives est de 2^o : la variation totale est donc de 28^o.

- Du bruit gaussien a été ajouté sur les points des images, avec un écart-type $\sigma = 1$ pixel. 200 expériences ont été effectuées pour chaque cas (c'est-à-dire pour chaque point des graphiques) avec des bruits différents.

- Un paramètre important pour chacune des expérimentations est la distance moyenne entre les points par rapport à la caméra. Soit D la distance du centre de gravité des points 3D au centre de projection, divisé par le diamètre de l'ensemble des points 3D. D est donc sans unité et nous l'appellerons *distance relative*. Remarquons que D est approximativement égal à $1/\overline{\varepsilon_j}$ où $\overline{\varepsilon_j}$ est la valeur moyenne des ε_j pour tous les points.

- Pour les expérimentations où D varie, la variation est égale à 5.

- La qualité des résultats de reconstruction est évaluée en calculant les erreurs euclidiennes moyenne et maximum entre les points 3D théoriques et les points 3D estimés (le facteur d'échelle est ajusté en conséquence), et les erreurs angulaires moyenne et maximum entre deux paires de segments 3D théoriques et estimés.

Les figures 5.4 et 5.5 étudient la qualité de reconstruction (erreur en distance et erreur en orientation) lorsque la caméra reste à distance constante de l'objet et tourne autour de celui-ci. L'objet reste centré sur l'axe optique de la caméra.

Les figures 5.6 et 5.7 étudient la qualité de reconstruction lorsque la distance caméra-objet varie.

Les figures 5.8 et 5.9 s'intéressent au cas où l'objet est décalé par rapport à l'axe optique.

5.7.2 Donnés réelles

Dans ce paragraphe, nous considérons plusieurs exemples de séquences composées d'images réelles et nous étudions le comportement de l'algorithme paraperspectif itératif :

- une séquence de 5 images d'un cube sur laquelle 38 points ont été suivis sur la séquence (figure 5.10) ;

- une séquence de 5 images d'une maison sur laquelle 36 points ont été suivis (figure 5.11) ;

- une séquence de 10 images représentant une pièce en bois sur laquelle 10 points ont été suivis (figure 5.12).

Dans chaque cas, le centre de l'image a été fixé à $u_c = v_c = 256$, et les facteurs d'échelles verticale et horizontale sont $\alpha_u = 1500$ et $\alpha_v = 1000$. Le mouvement de la caméra est très général. Remarquons la différence de qualité de la reconstruction obtenue avec la méthode de factorisation et la méthode itérative. L'écart est particulièrement visible lorsque l'objet reconstruit est vu de dessus.

Le tableau 5.2 récapitule les performances de l'algorithme itératif obtenues avec des données synthétiques et réelles.

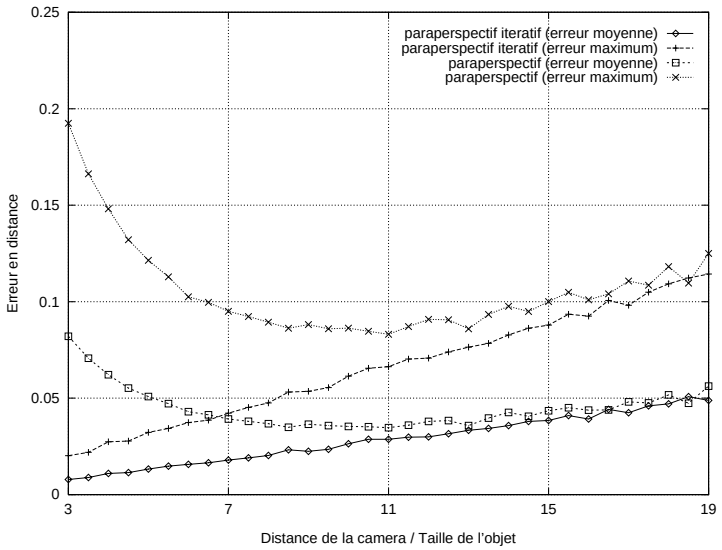


FIG. 5.4: *Qualité de reconstruction en distance en fonction de D (voir texte) lorsque le mouvement de la caméra est parallèle au plan image.*

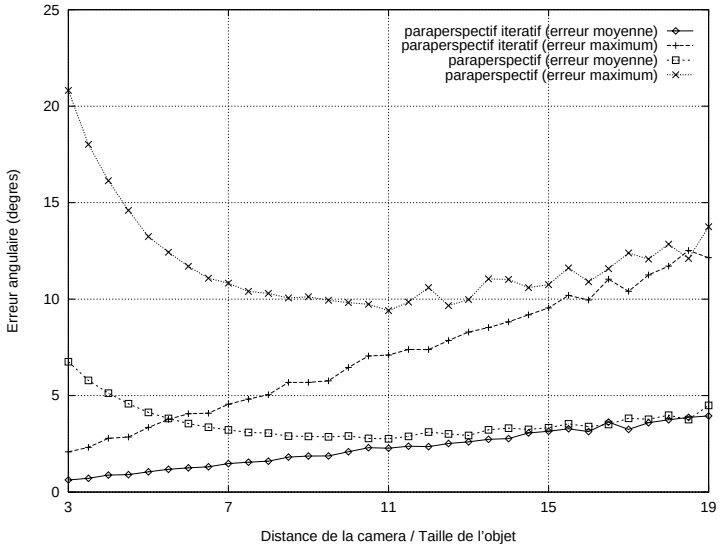


FIG. 5.5: *Qualité de reconstruction (erreur angulaire) en fonction de D (voir texte) lorsque le mouvement de la caméra est parallèle au plan image.*

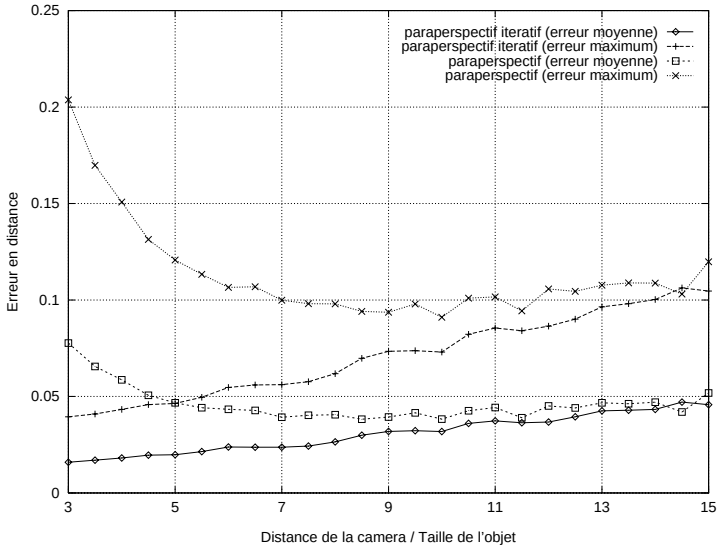


FIG. 5.6: *Qualité de reconstruction en distance en fonction de D lorsque la distance caméra-objet varie. La variation de profondeur entre deux images consécutives est 0,5.*

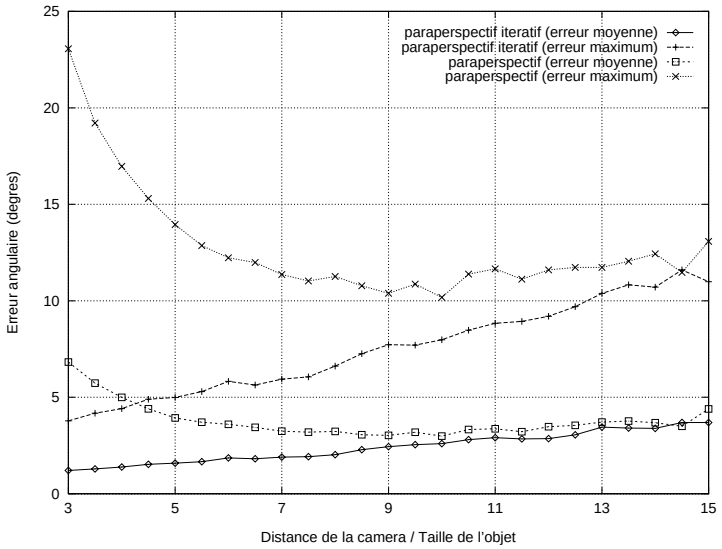


FIG. 5.7: *Qualité de reconstruction (erreur angulaire) en fonction de D lorsque la distance caméra-objet varie. La variation de profondeur entre deux images consécutives est 0,5.*

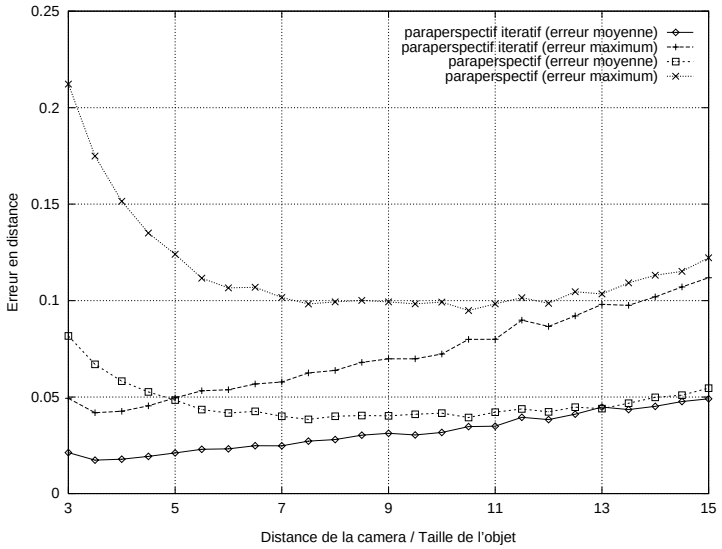


FIG. 5.8: *Qualité de reconstruction en fonction de D lorsque la distance caméra-objet varie, et lorsque le centre de gravité est décalé par rapport à l'axe optique.*

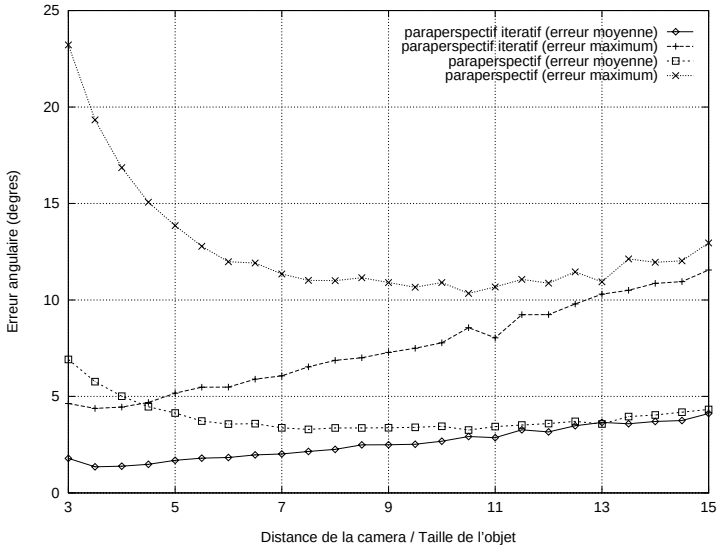


FIG. 5.9: *Comme ci-dessus mais pour l'erreur angulaire.*

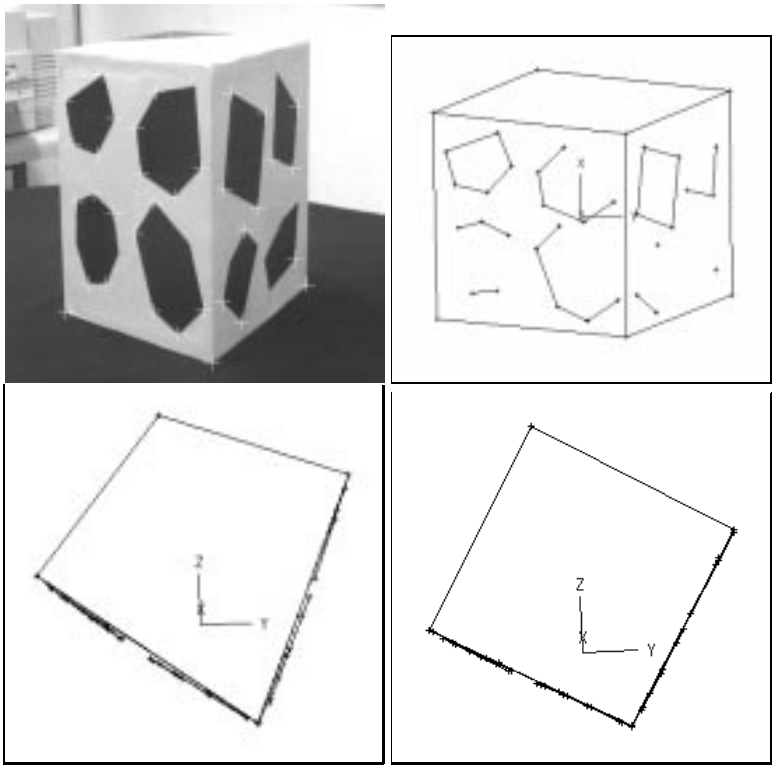


FIG. 5.10: *Cette figure montre respectivement une image (en haut à gauche) sur une séquence de 5 images acquises avec une caméra en mouvement, et le résultat de la reconstruction par la méthode itérative à partir de 38 points extraits (en haut à droite). Les vues de dessus de la scène reconstruite permettent de comparer quantitativement les résultats de la méthode de factorisation (en bas à gauche) avec les résultats obtenus avec la méthode itérative décrite précédemment (en bas à droite).*

Nb points	Nb images	Temps CPU/ itération	Nb itérations	Séquences d'images
38	5	0,31	5	"cube"
10	10	0,37	4	"pièce en bois"
42	15	1,50	4	images synthétiques

TAB. 5.2: *Récapitulation des résultats obtenus par la méthode de reconstruction itérative. Le temps* CPU *est exprimé en secondes et a été obtenus sur une station Sun/Sparc10/SunOs.*

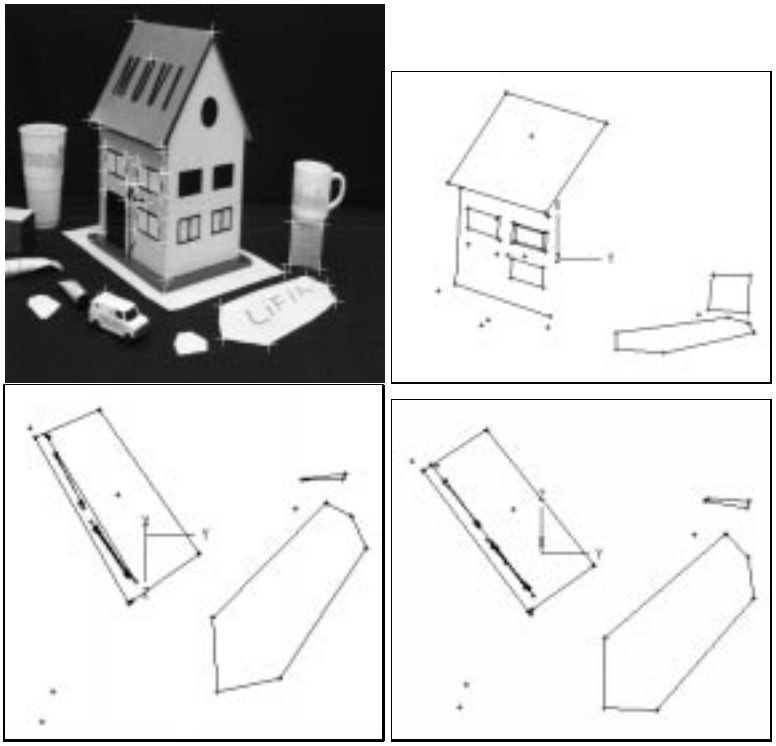


FIG. 5.11: *Mêmes commentaires que la figure précédente pour une autre séquence de 5 images et 46 points.*

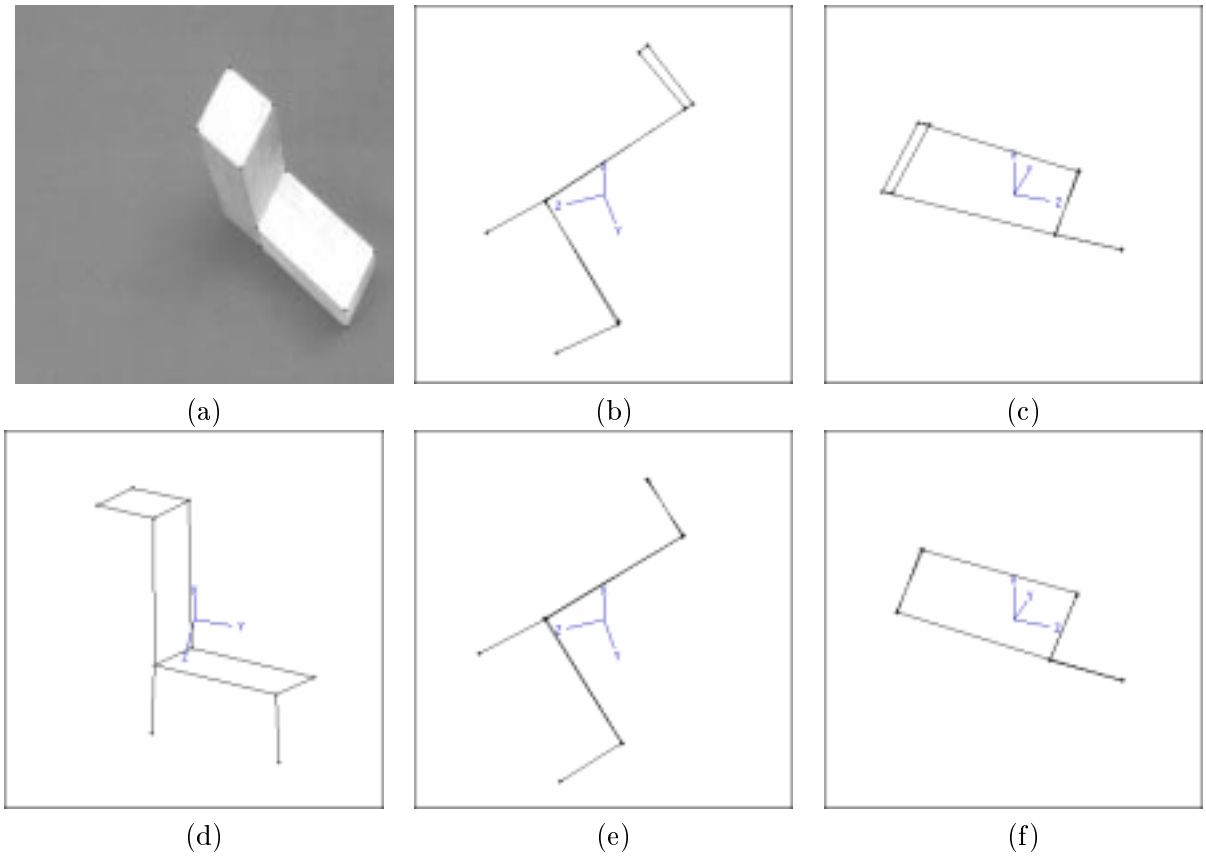


FIG. 5.12: *Cette figure montre (a) une image parmi une séquence de 10 images où 10 points ont été mis en correspondance et reconstruits. La première ligne (b) et (c) représente le résultat de la reconstruction obtenu avec la méthode de factorisation, alors que la seconde ligne (d), (e) et (f) représente le résultat de la reconstruction obtenu avec la méthode itérative et un modèle perspectif de caméra. Dans cet exemple, l'algorithme a convergé en 4 itérations.*

5.8 Conclusion

Nous avons proposé une méthode pour effectuer une reconstruction euclidienne à partir d'une séquence d'images en utilisant un modèle perspectif de caméra. La reconstruction est effectuée de manière incrémentale, en utilisant à chaque pas d'itération un modèle orthographique à l'échelle ou paraperspectif. La méthode peut être vue comme une extension de la méthode de factorisation proposée par Tomasi et Kanade [Tom 91a, Tom 92], mais pour un modèle perspectif de caméra. Par rapport aux méthodes existantes antérieures, l'avantage de l'algorithme est d'être beaucoup plus efficace, du point de vue temps de calcul, par rapport aux méthodes de minimisation non linéaires. L'algorithme converge en moyenne en 5 itérations, et celui-ci peut utiliser un nombre quelconque d'images ($k \geq 3$) et de points ($n \geq 4$ non coplanaires), sans privilégier une image ou un point par rapport aux autres.

À chaque pas d'itération, deux étapes principales sont effectuées : nous faisons d'abord une reconstruction affine, puis le passage à une reconstruction euclidienne. Notons qu'il est nécessaire d'avoir au moins un mouvement de rotation pour pouvoir effectuer le passage d'une reconstruction affine à une reconstruction euclidienne (voir paragraphe 5.3.3). En pratique, et dans nos conditions de prise de vues (voir paragraphe 5.7), nous avons constaté qu'il faut avoir un mouvement de rotation d'au moins une vingtaine de degrés sur l'ensemble de la séquence d'images pour avoir une reconstruction euclidienne correcte (information sur la profondeur). Cette valeur dépend cependant des paramètres de la caméra, ainsi que du bruit dans les images. Plus le bruit est important, plus le mouvement de la caméra doit être important pour compenser ce bruit.

Les résultats obtenus sont bien sûr légèrement moins bons que ceux obtenues avec une méthode de minimisation non linéaire. Comme dans le cas du calcul de pose, nous pouvons utiliser la solution obtenue par la méthode itérative pour initialiser une méthode de minimisation non linéaire. Dans ce cas, le nombre d'itérations de l'algorithme non linéaire est très faible, mais cette étape reste lente (comparativement beaucoup plus lente que dans le cas du calcul de pose).

Bien que les résultats expérimentaux nous aient montré qu'il y a peu de problèmes de convergence, l'étude de la convergence de l'algorithme est difficile d'un point de vue théorique. Néanmoins, il n'existe pas non plus de justification de convergence des algorithmes de reconstruction basés sur des méthodes de minimisation non linéaires. La comportement de la méthode proposée a l'avantage de pouvoir être expliqué de manière géométrique. Nous avons étudié la convergence sur des considérations numériques et pratiques afin de pouvoir déterminer à l'avance les conditions optimales d'utilisation de la méthode.

Nous avons également montré comment traiter les problèmes d'occultations, et résoudre le problème de l'ambiguïté de la solution associé à un modèle affine de caméra. Notons que dans le cas où le bruit dans les images est important, et lorsque l'effet de perspective est faible (i.e. la caméra est loin de l'objet), l'ambiguïté sur le choix de la bonne solution demeure : les deux solutions obtenues se projettent sur des points très proches dans les images.

Chapitre 6

Analyse de l'algorithme

> "La science commence par une rupture avec l'objet de la connaissance sensible. Elle cherche l'évidence rationnelle et non la satisfaction intime et échappe ainsi aux fausses évidences désirées."
>
> GASTON BACHELARD,
> *La formation de l'esprit scientifique*, 1970.

Le paragraphe 6.1 analyse la convergence des deux variantes de l'algorithme itératif à partir de considérations numériques, et le paragraphe 6.2 explique pourquoi la méthode proposée est peu sensible aux erreurs d'étalonnage de la caméra.

6.1 Analyse de la convergence

Afin d'analyser la convergence des algorithmes itératifs de calcul de pose ou de reconstruction, nous considérons séparément les équations du modèle orthographique à l'échelle et celles du modèle paraperspectif. L'analyse de la convergence est difficile d'un point de vue théorique. Par conséquent, nous baserons notre analyse sur des considérations numériques en comparant la valeur des termes négligés par rapport à ceux conservés.

Considérons les équations (2.8), (2.9) et (2.24), (2.25). Ces deux ensembles d'équations sont celles du modèle perspectif d'une caméra étalonnée. Le premier ensemble exprime le problème du calcul de pose (ou de la reconstruction) par un algorithme orthographique à l'échelle itératif, alors que le second ensemble exprime le problème avec un algorithme paraperspectif itératif. Si de bonnes estimations pour les ε_j sont disponibles, alors le problème du calcul de pose se réduit au problème du calcul de pose avec un modèle affine de caméra. Bien sûr, en pratique, il n'est pas possible d'avoir de telles estimations.

Nous montrons ici qu'en initialisant ε_j à zéro, nous obtenons un bon comportement de l'algorithme, même dans le cas où l'objet est proche de la caméra. Ceci explique que l'algorithme converge en quelques itérations. Nous regardons dans un premier temps le cas orthographique à l'échelle, puis le cas paraperspectif.

Soient, pour n points, $\varepsilon_j^\star$ les vraies valeurs que l'algorithme est sensé calculer. À la première itération, l'algorithme effectue un calcul de pose avec un modèle orthographique à l'échelle. Par conséquent, l'erreur est proportionnelle à l'erreur induite par le modèle orthographique à l'échelle $|x_j\varepsilon_j^\star|$ et $|y_j\varepsilon_j^\star|$, d'après les équations (2.14) et (2.15). Si ces erreurs sont importantes, la solution obtenue avec un modèle orthographique à l'échelle sera très différente de la solution recherchée, et la convergence de l'algorithme ne pourra être garantie. Dans leur travaux sur l'estimation de pose, Dementhon et Davis [Dem 95] ont remarqué que l'algorithme orthographique à l'échelle itératif converge même dans le cas où les valeurs des ε_j sont proches de 1, *à condition que les points de la scène soient proches de l'axe optique*. En effet, lorsque les points de la scène sont proches de l'axe optique, les coordonnées image de leur projection, x_j et y_j, sont petites (l'origine du repère caméra appartient à l'axe optique) et elles compensent les grandes valeurs des $\varepsilon_j^\star$. De plus, comme nous l'avons vu dans le paragraphe précédent, des valeurs plus réalistes pour $\varepsilon_j^\star$ sont inférieures à 0,5.

L'algorithme paraperspectif itératif est capable de traiter des configurations pour lesquelles l'algorithme orthographique à l'échelle itératif diverge. En effet, pour le modèle paraperspectif, les erreurs initiales sont $|(x_j - x_0)\varepsilon_j^\star|$ et $|(x_j - x_0)\varepsilon_j^\star|$, d'après les équations (2.22) et (2.23). Lorsque le point M_0 est proprement choisi (ce devrait être typiquement le centre de gravité des points image), alors les différences $(x_j - x_0)$ et $(y_j - y_0)$ sont petites et compensent de grandes valeurs pour $\varepsilon_j^\star$. Par conséquent, l'algorithme paraperspectif est capable de converger dans un plus grand nombre de configurations que l'algorithme orthographique à l'échelle.

Cas de la reconstruction

La figure 6.1 indique le nombre d'itérations moyen pour les deux méthodes (seuls les cas ayant convergés sont pris en compte).

La figure 6.2 représente le pourcentage de convergence des deux variantes de l'algorithme itératif.

Enfin, les figures 6.3 et 6.4 comparent l'algorithme orthographique à l'échelle itératif et l'algorithme paraperspectif itératif. Les performances obtenues par ces deux variantes sont comparables car les deux algorithmes convergent vers la même solution correspondant au modèle perspectif de caméra.

6.2 Sensibilité à l'étalonnage de la caméra

Nous avons supposé jusqu'à maintenant que la caméra est étalonnée, ce qui signifie que les paramètres intrinsèques α_u, α_v, u_c, v_c sont connus dans les équations (2.1) et (2.2). Le problème de l'étalonnage d'une caméra est difficile et les paramètres de la caméra sont instables avec le temps ou le mouvement. Nous savons que le seul paramètre intrinsèque stable est le rapport entre la taille des pixels horizontale et verticale :

$$\gamma = \frac{\alpha_u}{\alpha_v}$$

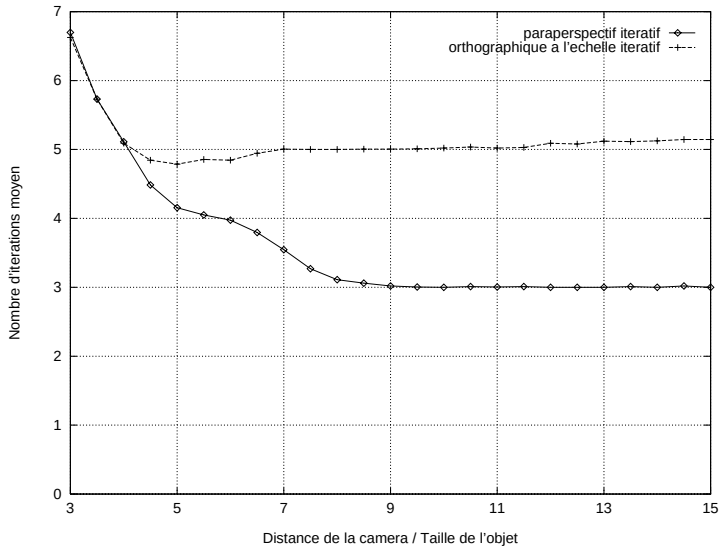


FIG. 6.1: *Nombre d'itérations (algorithmes itératifs perspectif faible et paraperspectif) en fonction de D lorsque la distance caméra-objet varie, et lorsque le centre de gravité est décalé par rapport à l'axe optique.*

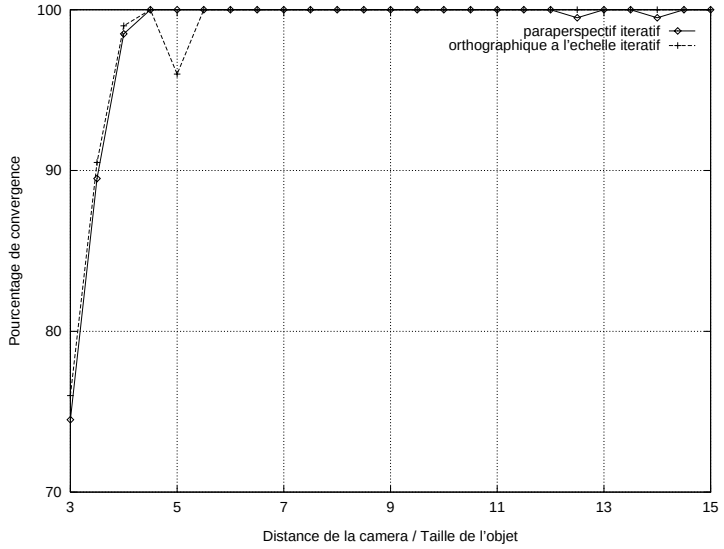


FIG. 6.2: *Pourcentage de convergence des deux algorithmes itératifs en fonction de D lorsque la distance caméra-objet varie, et lorsque le centre de gravité est décalé par rapport à l'axe optique.*

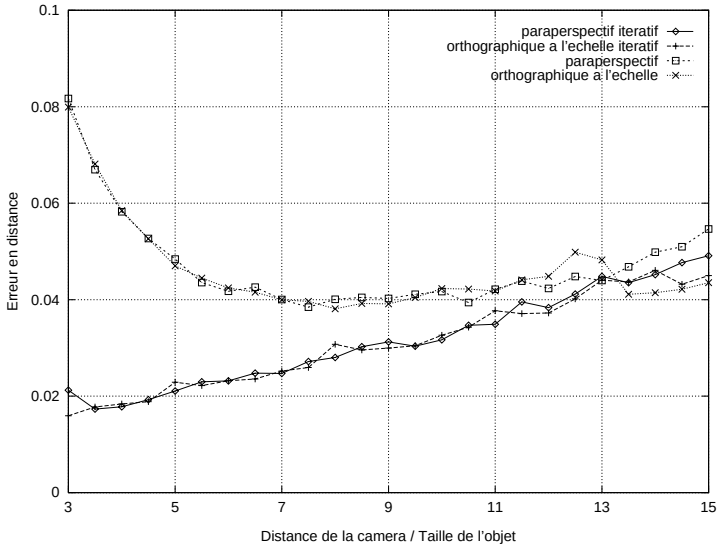


FIG. 6.3: *Comparaison de la précision obtenue avec les deux algorithmes itératifs, le mouvement de la caméra étant parallèle au plan image.*

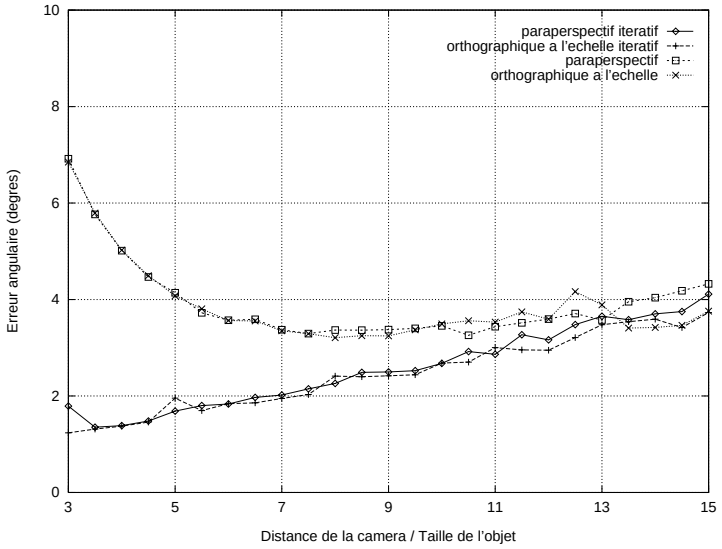


FIG. 6.4: *Comparaison de la précision obtenue avec les deux algorithmes itératifs (erreur angulaire).*

Considérons de nouveau les équations (2.8) et (2.9). En les combinant avec les équations (2.1) et (2.2), nous obtenons :

$$\begin{aligned} u_j(1+\varepsilon_j) - u_0 - u_c\varepsilon_j &= \gamma\alpha_v \boldsymbol{I} \cdot \boldsymbol{M}_j \\ v_j(1+\varepsilon_j) - v_0 - v_c\varepsilon_j &= \alpha_v \boldsymbol{J} \cdot \boldsymbol{M}_j \end{aligned}$$

Lorsque la caméra est étalonnée, nous pouvons obtenir des valeurs relativement précises pour les facteurs d'échelles vertical et horizontal α_u et α_v, alors que les coordonnées du centre de l'image u_c et v_c peuvent varier jusqu'à 10% de leur vraies valeurs (Tsai [Tsa 87], Faugeras [Fau 93]). En analysant les équations précédentes, nous pouvons remarquer que l'influence de u_c et v_c est pondéré par ε_j. Pour tout point M_j de l'objet, ε_j représente la projection du vecteur $\boldsymbol{M}_0\boldsymbol{M}_j$ suivant l'axe optique, divisé par la composante en z du vecteur de translation $\boldsymbol{t}$.

Si l'objet est loin de la caméra, l'influence de u_c et v_c peut être négligée, mais les points image doivent être extraits avec une grande précision. Si l'objet est proche de la caméra, l'influence de u_c et v_c devient importante. En pratique, la valeur maximum pour ε_j peut monter jusqu'à 0,5, mais des valeurs plus réalistes sont dans l'intervalle $[0,1;0,2]$. Dans ce cas, les erreurs sur u_c et v_c sont divisées par un facteur allant de 5 à 10.

En examinant les équations ci-dessus, nous pouvons remarquer que le paramètre intrinsèque α_v agit comme un facteur d'échelle sur la reconstruction. Cependant, comme la reconstruction est effectuée à un facteur d'échelle près, le connaissance exacte de α_v affecte peu la reconstruction. Dans le cadre de la reconstruction, le seul paramètre intrinsèque qu'il est nécessaire de connaître est le rapport entre la taille horizontale des pixels et la taille verticale.

Chapitre 7

Reconstruction euclidienne et étalonnage affine d'une caméra montée sur le bras d'un robot

"Ce n'est pas parce que les choses sont difficiles que nous n'osons pas, c'est parce que nous n'osons pas qu'elles sont difficiles."

Sénèque.

7.1 Contexte

Dans ce chapitre, nous nous intéressons au problème de la reconstruction avec un modèle affine de caméra montée sur le bras d'un robot. Les mouvements effectués par le robot sont connus, mais nous ne connaissons pas la position de la caméra par rapport au bras du robot. Des travaux récents ont montrés que l'utilisation des mouvements du robot en utilisant un modèle perspectif de caméra ne simplifie pas le problème reconstruction [Hor 95c]. Le modèle de caméra que nous utilisons est un modèle affine. Ce modèle est un modèle simplifié par rapport au modèle projectif. Il existe cependant de nombreuses applications pour lesquelles la caméra est loin de l'objet ; dans ce cas, le modèle affine est une bonne approximation.

Comme la caméra est montée sur le bras d'un robot, il existe une transformation rigide fixée entre le repère de la caméra et le repère du bras. Cette transformation n'est pas connue à l'avance et par conséquent, le mouvement de la caméra n'est connu qu'à une rotation et une translation près.

Plus formellement, le problème que l'on se propose de résoudre ici est le suivant : étant donnés un modèle affine de caméra montée sur le bras d'un robot, un certain nombre de points mis en correspondance sur une séquence d'images, et le mouvement du robot entre ces images, le problème est (i) de déterminer une reconstruction euclidienne de la scène, (ii) d'étalonner la caméra, et (iii) d'estimer la transformation caméra-pince. Ce dernier problème est également connu comme étant le problème de l'étalonnage caméra-pince. Des approches antérieures du problème de l'étalonnage caméra-pince proposent de résoudre le problème en deux étapes : dans un premier temps, le robot est déplacé en des positions prédéterminées, et en chacune de ces positions, les paramètres intrinsèques et extrinsèques sont calculés en utilisant une mire 3D d'étalonnage connue ; dans un second temps un système d'équations homogènes est résolu de manière à déterminer la transformation caméra-pince [Hor 95b]. La méthode proposée peut être vue comme une méthode d'étalonnage en ligne qui ne demande aucune grille d'étalonnage ni la connaissance des paramètres intrinsèques. Cependant, comme nous considérons ici un modèle affine de caméra, seule la matrice de rotation de la transformation caméra-pince peut être calculée.

Les principales contributions sont les suivantes :

- nous montrons que le problème de reconstruction euclidienne avec un modèle affine de caméra non étalonnée conduit à une formulation linéaire simple si la caméra est montée sur un robot, et si le robot effectue des mouvements connus ;
- nous effectuons à la fois une reconstruction euclidienne de la scène, l'étalonnage de

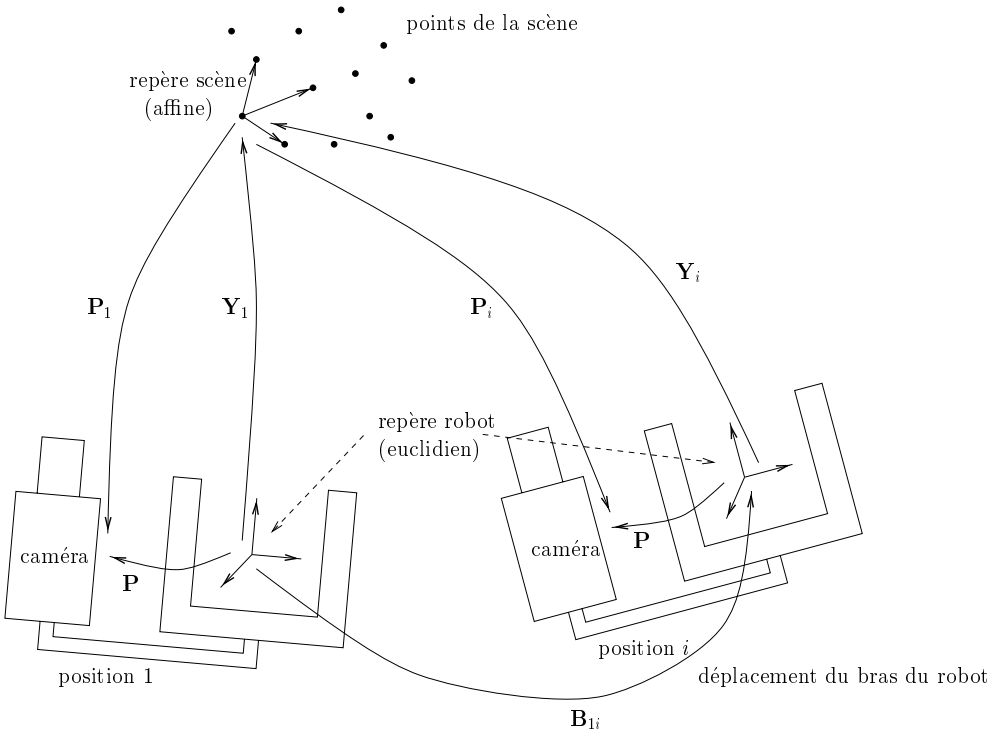


FIG. 7.1: *Cette figure montre la caméra fixée sur le bras d'un robot. Le bras effectue un déplacement rigide (ainsi que la caméra), et la caméra observe une scène* 3D *matérialisée par en ensemble de points.*

la caméra et l'étalonnage caméra-pince.

7.2 Modèle affine de caméra

Rappelons la modélisation mathématique d'un modèle affine de caméra :

$$\boldsymbol{m} \simeq \mathbf{P}_a \boldsymbol{M} \tag{7.1}$$

où $\boldsymbol{m}$ est un point image exprimé en coordonnées image, $\boldsymbol{M}$ représente les coordonnées d'un point 3D exprimé dans un repère euclidien, et $\mathbf{P}_a$ est une matrice 3×4 de la forme :

$$\mathbf{P}_a = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

L'équation (7.1) peut également s'écrire :

$$\begin{pmatrix} u \\ v \end{pmatrix} = \underbrace{\begin{pmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \end{pmatrix}}_{\mathbf{N}} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \underbrace{\begin{pmatrix} p_{14} \\ p_{24} \end{pmatrix}}_{\boldsymbol{n}} \tag{7.2}$$

Nous pouvons éliminer la translation en faisant un changement d'origine du repère image :

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} u - p_{14} \\ v - p_{24} \end{pmatrix} = \mathbf{N} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{7.3}$$

Le vecteur $\boldsymbol{n}$ de dimension 2 représente l'origine du nouveau repère image et celui-ci est en fait la projection de l'origine du repère 3D de l'objet.

De manière à faire apparaître de manière explicite les paramètres intrinsèques et extrinsèques d'un modèle affine de caméra, une méthode consiste à effectuer une décomposition QR de la matrice $\mathbf{N}$ [Qua 95b, Qua 96b] :

$$\mathbf{N} = \underbrace{\begin{pmatrix} a & 0 \\ b & c \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} {}^t\boldsymbol{i} \\ {}^t\boldsymbol{j} \end{pmatrix}}_{\mathbf{R}_{2\times 3}} \tag{7.4}$$

Cette décomposition est unique si le rang de la matrice $\mathbf{N}$ est égal à 2. La matrice $\mathbf{A}$ dépend de 3 paramètres associés à un modèle affine de caméra, et $\mathbf{R}_{2\times 3}$ contient les deux premières lignes d'une matrice de rotation, la troisième pouvant facilement être calculée par la formule :

$$\boldsymbol{k} = \boldsymbol{i} \wedge \boldsymbol{j}$$

Nous avons introduit au chapitre 2 deux cas particuliers de modèles affines de caméra : orthographique à l'échelle et paraperspectif. Cependant, dans la suite, nous ne ferons aucune supposition spécifique sur l'un ou l'autre de ces deux modèles.

7.3 Formulation du problème

Considérons maintenant une caméra montée rigidement sur le bras d'un robot, et nous associons un repère euclidien au bras du robot. Soit ($\mathbf{R}$ $\boldsymbol{t}$) la transformation rigide (rotation et translation) entre le repère du bras du robot et le repère caméra. Les matrices $\mathbf{R}$ et $\boldsymbol{t}$ représentent l'étalonnage caméra-pince. Cependant, avec un modèle affine de caméra, seuls les paramètres de rotation (la matrice $\mathbf{R}$) peuvent être calculés.

La matrice de projection s'écrit :

$$\mathbf{P}_a = \begin{pmatrix} \mathbf{A}\mathbf{R}_{2\times 3} & \boldsymbol{n} \\ {}^t\boldsymbol{0} & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{N} & \boldsymbol{n} \\ {}^t\boldsymbol{0} & 1 \end{pmatrix} \tag{7.5}$$

où $\mathbf{A}$ contient les 3 paramètres intrinsèques du modèle affine de caméra et $\mathbf{R}_{2\times 3}$ contient les deux premiers vecteurs de la matrice de rotation $\mathbf{R}$.

Par conséquent, la détermination de la matrice $\mathbf{P}_a$ dépend à la fois du problème de l'étalonnage de la caméra et du problème de l'étalonnage caméra-pince. Cette matrice $\mathbf{P}_a$ ne peut être estimée de manière directe. Nous proposons une méthode qui effectue d'abord une reconstruction de la scène, et ensuite un étalonnage de la caméra ainsi que l'estimation de la transformation caméra-pince.

Nous supposons maintenant que la caméra et le robot subissent une succession de mouvements rigides, et que les matrices $\mathbf{A}$, $\mathbf{R}$ et $\boldsymbol{t}$ restent constantes pendant ces déplacements. Ainsi, la matrice $\mathbf{P}_a$ reste également fixe. La caméra observe une scène 3D et la mise en correspondances des points sur la séquence d'images permet d'effectuer une reconstruction affine de la scène. Pour ce faire, il est possible d'utiliser la méthode de factorisation ou la méthode des invariants affines. Afin d'effectuer cette reconstruction affine, nous associons un repère affine à la scène. Soient $\boldsymbol{S}_j$ les coordonnées affines des points de la scène exprimées dans ce repère et $\mathbf{P}_i$ la matrice de passage de ce repère affine au repère euclidien associé à la i-ième image (voir figure 7.1). Par conséquent, $\boldsymbol{S}_j$ et $\boldsymbol{M}_j$ représentent le même point physique 3D exprimé dans deux repères différents : un repère affine et un repère euclidien.

Soit $\mathbf{Y}_i$ une matrice 4×4 inversible représentant la transformation entre le repère euclidien associé à la pince (à la position i) et le repère affine de la scène. La matrice $\mathbf{Y}_i$ est une transformation 3D affine. Ainsi, la matrice $\mathbf{P}_a$ peut s'écrire comme combinaison de deux transformations affines (voir figure 7.1) :

$$\begin{aligned} \mathbf{P}_a &= \mathbf{P}_1\mathbf{Y}_1 \\ &= \dots \\ &= \mathbf{P}_i\mathbf{Y}_i \\ &= \dots \\ &= \mathbf{P}_k\mathbf{Y}_k \end{aligned} \tag{7.6}$$

Dans ces équations, $\mathbf{P}_1$, …, $\mathbf{P}_i$, …, $\mathbf{P}_k$ sont des matrices de taille 3×4 décrivant la transformation affine entre le repère affine de la scène et le repère euclidien associé à chaque image. De plus, posons $\mathbf{B}_{1i}$ la matrice 4×4 représentant le mouvement rigide du bras du robot entre la position initiale – 1 – et n'importe quel autre position – i. La relation entre $\mathbf{Y}_1$, $\mathbf{Y}_i$, et $\mathbf{B}_{1i}$ est simplement :

$$\mathbf{Y}_i = \mathbf{Y}_1\mathbf{B}_{1i}^{-1}$$

En substituant $\mathbf{Y}_i$ dans l'équation (7.6), nous obtenons $k-1$ équations matricielles :

$$\left\{\begin{array}{rcl} \mathbf{P}_1\mathbf{Y}_1\mathbf{B}_{12} &=& \mathbf{P}_2\mathbf{Y}_1 \\ & \vdots & \\ \mathbf{P}_1\mathbf{Y}_1\mathbf{B}_{1k} &=& \mathbf{P}_k\mathbf{Y}_1 \end{array}\right. \tag{7.7}$$

Chacune de ces équations peut s'écrire de la façon suivante :

$$\underbrace{\begin{pmatrix} \mathbf{N}_1 & \boldsymbol{n}_1 \\ {}^t\boldsymbol{0} & 1 \end{pmatrix}}_{3\times4}\underbrace{\begin{pmatrix} \mathbf{X}_1 & \boldsymbol{x}_1 \\ {}^t\boldsymbol{0} & 1 \end{pmatrix}}_{4\times4}\underbrace{\begin{pmatrix} \mathbf{R}_i & \boldsymbol{t}_i \\ {}^t\boldsymbol{0} & 1 \end{pmatrix}}_{4\times4} = \underbrace{\begin{pmatrix} \mathbf{N}_i & \boldsymbol{n}_i \\ {}^t\boldsymbol{0} & 1 \end{pmatrix}}_{3\times4}\underbrace{\begin{pmatrix} \mathbf{X}_1 & \boldsymbol{x}_1 \\ {}^t\boldsymbol{0} & 1 \end{pmatrix}}_{4\times4}$$

Cette équation matricielle peut se décomposer par :

$$\mathbf{N}_1\mathbf{X}_1\mathbf{R}_i = \mathbf{N}_i\mathbf{X}_1 \tag{7.8}$$

$$\mathbf{N}_1\mathbf{X}_1\boldsymbol{t}_i + (\mathbf{N}_1 - \mathbf{N}_i)\boldsymbol{x}_1 = \boldsymbol{n}_i - \boldsymbol{n}_1 \tag{7.9}$$

La première de ces équations est composée de matrices de taille 2×3 et est homogène par rapport aux éléments de la matrice $\mathbf{X}_1$ de taille 3×3. La seconde équation est composée de vecteurs de dimension 2 et les inconnus sont la matrice $\mathbf{X}_1$ et le vecteur $\boldsymbol{x}_1$:

$$\mathbf{Y}_1 = \begin{pmatrix} \mathbf{X}_1 & \boldsymbol{x}_1 \\ {}^t\boldsymbol{0} & 1 \end{pmatrix}$$

Nous avons donc 12 inconnues, et chaque déplacement de caméra conduit à $6+2$ contraintes. Pour k positions de la caméra, nous avons $k-1$ déplacements, soit $8(k-1)$ équations linéaires. Par conséquent, k doit être au moins égal à 3 afin de pouvoir calculer les éléments de la matrice $\mathbf{Y}_1$.

Avant d'aller plus loin et de développer la solution du problème, récapitulons les principales étapes de la méthode.

1. Effectuer un certain nombre (au moins 2) de déplacements du robot sur lequel une caméra a été fixée. Ces déplacements fournissent les éléments des matrices $\mathbf{B}_{1i} = (\ \mathbf{R}_i \quad \boldsymbol{t}_i\ )$. Faire l'acquisition d'une image à chaque position et mettre les points en correspondance entre les images.

2. Déterminer une reconstruction affine et le déplacement affine à partir des points extraits des images, c'est-à-dire déterminer les coordonnées affines $\boldsymbol{S}_i$ et les matrices de projection $\mathbf{P}_i$ qui projettent ces points $\boldsymbol{S}_i$ dans les images. Cette étape estime la matrice $\mathbf{N}_i$ et le vecteur $\boldsymbol{n}_i$ des équations (7.8) et (7.9).

3. Transformer la reconstruction affine en une reconstruction euclidienne. Il s'agit de résoudre un système linéaire surcontraint formé par les équations (7.8) et (7.9). Nous calculons ainsi $\mathbf{X}_1$ et $\boldsymbol{x}_1$, c'est-à-dire la matrice $\mathbf{Y}_1$. Après avoir déterminé $\mathbf{Y}_1$, la transformation des coordonnées affines des points en coordonnées euclidiennes est donnée par la formule :

$$\forall j \in \{1,\ldots,n\},\quad \boldsymbol{M}_j = \mathbf{Y}_1^{-1}\boldsymbol{S}_j \tag{7.10}$$

4. L'étalonnage de la caméra et l'étalonnage caméra-pince sont calculés à partir de la matrice $\mathbf{P}_a = (\ \mathbf{N} \quad \boldsymbol{n}\ )$ qui peut être estimée à partir de n'importe quelle expression donnée par l'équation (7.6). Comme nous l'avons expliqué au paragraphe 7.2, les paramètres intrinsèques et extrinsèques du modèle affine de caméra sont calculés en faisant une décomposition QR de la matrice $\mathbf{N}$ de taille 2×3. L'étalonnage caméra-pince est donnée par la matrice de rotation 3×3.

Unicité de la solution Horaud et al. [Hor 95a] ont démontré une condition suffisante pour pouvoir résoudre le système (7.7) : *parmi l'ensemble des déplacements* $\mathbf{B}_{1i}$ *du robot, (i) au moins deux mouvements doivent avoir un axe de rotation distinct, et (ii) au moins un déplacement doit avoir son vecteur de translation non orthogonal à l'axe de rotation.* Remarquons que la première condition est identique à celle pour l'unicité de la solution de l'étalonnage euclidien caméra-pince [Che 91b]. Si la dernière condition n'est pas vérifiée, la reconstruction euclidienne est effectuée à un facteur d'échelle près.

7.4 Résolution du problème

Soient $\{M_1, \ldots, M_n\}$ un ensemble de points de la scène dont nous cherchons les coordonnées euclidiennes représentées par les vecteurs $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_n$. Comme nous l'avons expliqué précédemment, nous calculons d'abord les coordonnées affines des points, puis nous convertissons ces coordonnées en coordonnées euclidiennes.

7.4.1 Reconstruction et mouvements affines

Considérons une base affine de l'espace, et soient $\boldsymbol{S}_1$, ..., $\boldsymbol{S}_n$ les coordonnées affines des points de la scène dans ce repère. Reprenons les notations du paragraphe 5.7, et notons $\boldsymbol{s}_{ij}$ le vecteur allant du point m_{0_i} au point m_{ij}. Ainsi, pour tout i et pour tout j ($i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, n\}$), l'équation (7.2) devient :

$$\boldsymbol{s}_{ij} = \mathbf{N}_i \boldsymbol{S}_j \tag{7.11}$$

$$\boldsymbol{m}_{0_i} = \boldsymbol{n}_i \tag{7.12}$$

La première équation ci-dessus peut s'écrire pour k images et n points :

$$\begin{pmatrix} \boldsymbol{s}_{11} & \ldots & \boldsymbol{s}_{1n} \\ \vdots & & \vdots \\ \boldsymbol{s}_{k1} & \ldots & \boldsymbol{s}_{kn} \end{pmatrix} = \begin{pmatrix} \mathbf{N}_1 \\ \vdots \\ \mathbf{N}_k \end{pmatrix} \begin{pmatrix} \boldsymbol{S}_1 & \ldots & \boldsymbol{S}_n \end{pmatrix}$$

ou sous forme matricielle :

$$\boldsymbol{\Sigma} = \mathbf{N}\mathbf{S} \tag{7.13}$$

Nous avons vu dans le paragraphe 5.3.2 deux méthodes pour résoudre cette équation : la méthode des invariants affines et la méthode de factorisation. La méthode de factorisation est la méthode utilisée en pratique car elle ne nécessite pas de choix explicite d'une base affine formée par 4 points non coplanaires. La méthode de factorisation calcule la reconstruction et le déplacement affine en même temps en effectuant une décomposition en valeurs singulières de la matrice $\boldsymbol{\Sigma}$ de taille $2k \times n$ (équation (7.13)).

7.4.2 Reconstruction et déplacements euclidiens

Nous sommes maintenant capables de calculer $\mathbf{Y}_1$ figurant dans les équations (7.7), où chaque contrainte matricielle peut se décomposer en deux équations (7.8) et (7.9) :

$$\begin{aligned} \mathbf{N}_1\mathbf{X}_1\mathbf{R}_i &= \mathbf{N}_i\mathbf{X}_1 \\ \mathbf{N}_1\mathbf{X}_1\boldsymbol{t}_i + (\mathbf{N}_1 - \mathbf{N}_i)\boldsymbol{x}_1 &= \boldsymbol{n}_i - \boldsymbol{n}_1 \end{aligned}$$

Dans la première équation, $\mathbf{X}_1$ est inconnue. En combinant $k-1$ équations de cette forme, nous obtenons un système linéaire homogène surcontraint de la forme suivante ($\boldsymbol{\mathcal{X}}_1$ est un vecteur de dimension 9 formé avec les éléments de $\mathbf{X}_1$) :

$$\underbrace{\boldsymbol{\mathcal{A}}}_{6(k-1)\times 9}\ \underbrace{\boldsymbol{\mathcal{X}}_1}_{9\times 1} = \boldsymbol{0} \tag{7.14}$$

Ce système est homogène. En imposant $\|\boldsymbol{\mathcal{X}}_1\| = 1$, la solution optimale pour $\boldsymbol{\mathcal{X}}_1$ est obtenue en résolvant :

$$\min_{\boldsymbol{\mathcal{X}}_1} \left(\|\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{X}}_1\|^2 + \lambda(1 - \|\boldsymbol{\mathcal{X}}_1\|^2) \right)$$

La solution est le vecteur propre associé à la plus petite valeur propre de la matrice semi-définie positive ${}^t\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{A}}$. Ainsi, les éléments de $\mathbf{X}_1$ sont définis à un facteur multiplicatif près : $\mu\mathbf{X}_1$.

En substituant $\mu\mathbf{X}_1$ dans la seconde équation, nous obtenons :

$$\mu\mathbf{N}_1\mathbf{X}_1\boldsymbol{t}_i + (\mathbf{N}_1 - \mathbf{N}_i)\boldsymbol{x}_1 = \boldsymbol{n}_i - \boldsymbol{n}_1$$

Pour k positions de la caméra (c'est-à-dire $k-1$ déplacements), nous avons un système linéaire d'équations de la forme :

$$\underbrace{\boldsymbol{\mathcal{B}}}_{2(k-1)\times 4} \underbrace{\begin{pmatrix} \mu \\ \boldsymbol{x}_1 \end{pmatrix}}_{4\times 1} = \boldsymbol{b}$$

Ce système linéaire a une solution évidente si le rang de $\boldsymbol{\mathcal{B}}$ est égal à 4. Les coordonnées euclidiennes des points de la scène sont ensuite calculées en utilisant l'équation (7.10) :

$$\boldsymbol{M}_j = \mathbf{Y}_1^{-1}\boldsymbol{S}_j$$

7.4.3 Étalonnage de la caméra et étalonnage caméra-pince

L'étalonnage de la caméra, ainsi que l'étalonnage caméra-pince sont incluses dans la matrice de projection $\mathbf{P}_a$ (équation (7.5)), qui est obtenue en multipliant les deux matrices que nous venons de calculer :

$$\mathbf{P}_a = \mathbf{P}_1\mathbf{Y}_1$$

Cette équation est la première du système (7.6). Par la suite, les paramètres intrinsèques de la caméra et la matrice de rotation entre la caméra et le bras du robot sont calculés en effectuant une décomposition QR de $\mathbf{N}$ – sous-matrice 2×3 de $\mathbf{P}_a$.

Les paramètres intrinsèques sont extraits sous la forme d'une matrice $\mathbf{A}$ triangulaire basse de taille 2×2, d'après l'équation (7.4). Cette matrice correspond au modèle général d'un modèle affine de caméra, et les paramètres (a, b, et c) ont la signification suivante :

- Si l'élément b est petit par rapport aux éléments diagonaux, alors celui-ci peut être négligé et le modèle de caméra est équivalent au modèle orthographique à l'échelle.

– Dans le cas contraire (si b est grand), le modèle de caméra peut s'identifier au modèle paraperspectif.

7.5 Résultats expérimentaux

La caméra est montée sur un robot et observe un objet 3D sous 12 positions différentes. La figure 7.2 montre la première (a) et la dixième (b) image de cette séquence. L'objet est un parallélépipède sur lequel figurent des rectangles noirs sur les faces. La position exacte et la taille de ces rectangles noirs ne sont pas connus. La taille de ce parallélépipède est de $5 \times 4 \times 4$ cm et celui-ci se trouve à une distance approximative de 50 cm de la caméra. Ainsi, l'effet de perspective est faible, et la projection des points dans l'image suit une projection affine.

Les points d'intérêt ont été détectés et poursuivis sur la séquence d'images : 132 points ont été détectés. Sur cette séquence, les appariements ont été effectués à la main à cause des motifs trop répétitifs, mais pour d'autres séquences d'images réelles, nous avons utilisé la méthode développée au chapitre 1.

La figure 7.2 montre les résultats obtenus pour la reconstruction 3D de l'objet en utilisant deux méthodes différentes : la méthode que nous venons de décrire et qui utilise le mouvement du robot, et la méthode itérative que nous avons présentée au début du chapitre (cette méthode ne nécessite aucune connaissance sur le déplacement, mais les paramètres intrinsèques doivent être connus). La figure 7.2 montre une vue de dessus et une vue latérale de la reconstruction pour chacune des méthodes : (c) et (d) correspondent à la méthode utilisant le mouvement du robot, et (e) et (f) correspondent à la méthode itérative. Il y a deux différences importantes entre ces deux méthodes : (i) la première utilise une caméra non étalonnée, alors que la seconde utilise une caméra étalonnée, et (ii) la première méthode suppose un modèle affine de caméra alors que la seconde utilise un modèle perspectif de caméra.

Il n'est pas facile de comparer les résultats des deux reconstructions car les repères 3D utilisés pour chaque méthode sont différents. Dans le cas de la première méthode, le repère 3D euclidien est celui du bras du robot. Pour la seconde méthode, le repère 3D est centré par rapport à l'objet et est aligné avec le repère caméra à la première position.

Les paramètres obtenus pour la caméra avec la séquence précédente sont les suivants :

$$\begin{aligned} \mathbf{N} &= \gamma \mathbf{A} \mathbf{R}_{2\times 3} \\ &= 1,36 \begin{pmatrix} 0,58 & 0 \\ 0,06 & 1 \end{pmatrix} \begin{pmatrix} -0,97 & -0,12 & +0,18 \\ +0,12 & -0,99 & -0,02 \end{pmatrix} \end{aligned}$$

Le facteur d'échelle inclut à la fois la distance moyenne de l'objet par rapport à la caméra et la distance focale de la caméra. Comme l'élément en dessous de la diagonale de $\mathbf{A}$ peut être négligé, le modèle de caméra peut dans ce cas être approché par un modèle orthographique à l'échelle.

7.6 Conclusion

Nous avons proposé une méthode pour calculer la structure 3D euclidienne d'une scène en utilisant un modèle affine de caméra non étalonnée montée sur le bras d'un robot. L'information euclidienne est fournie par le déplacement connu du robot. Nous avons proposé

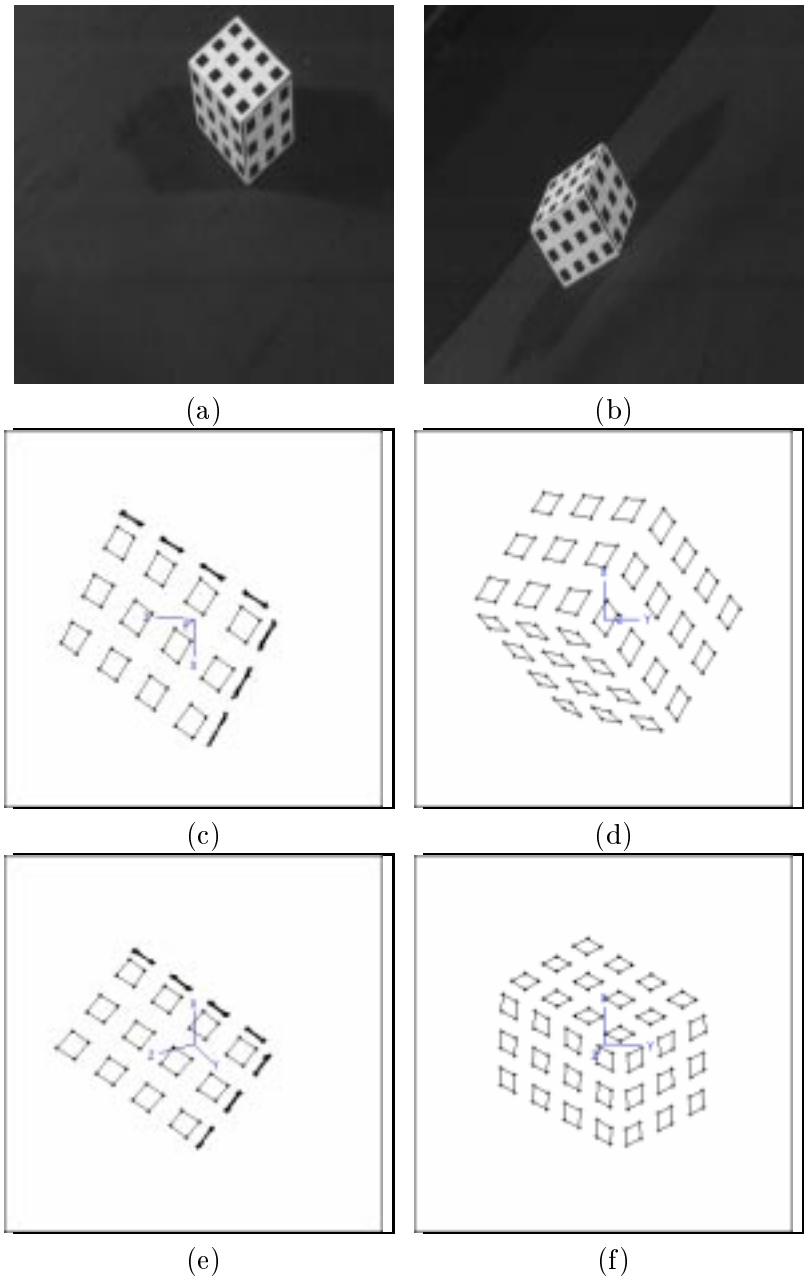


FIG. 7.2: *Cette figure montre deux images (a) et (b) sur une séquence de 12 images acquises avec une caméra fixée sur le bras d'un robot, et le résultat de la reconstruction : la vue de dessus (c) et générale (d) de la reconstruction obtenue avec la méthode ci-dessus, et la vue de dessus (e) et générale (f) de la reconstruction obtenue avec un modèle perspectif de caméra.*

une méthode de résolution linéaire très simple pour effectuer la reconstruction et déterminer les paramètres intrinsèques et extrinsèques de la caméra. Les paramètres intrinsèques permettent de déterminer s'il s'agit d'un modèle perspectif faible ou paraperspectif de caméra. Les paramètres extrinsèques (matrice de rotation) correspondent à l'orientation relative entre le repère associé au bras du robot et le repère caméra.

La méthode proposée peut s'appliquer dès que l'on peut effectuer un ensemble de déplacements contrôlés. C'est le cas lorsque l'on dispose d'une caméra montée sur un robot ou d'une tête active stéréo. De manière similaire, un certain nombre d'auteurs ont montré que le problème de l'étalonnage d'une caméra est plus simple si l'on effectue des déplacements particuliers dans l'espace [Har 94a]. Il n'est cependant pas toujours facile d'effectuer une rotation d'une caméra autour d'un axe aligné avec l'axe optique.

Une extension de ces travaux serait de généraliser la méthode pour un modèle perspectif de caméra, avec une méthode itérative modifiant de manière incrémentale la position des points dans les images.

Chapitre 8

Transfert technologique

"Toute connaissance débute avec l'expérience mais n'en dérive pas."

EMMANUEL KANT, *Critique de la raison pure*, XVIII^{e}S.

Ce chapitre présente quelques travaux effectués en collaboration avec la Société AÉROSPATIALE.

La stratégie de guidage d'un missile est un des sujets les plus importants dans le domaine de la Défense. Un missile est dirigé par un système de centrale inertielle utilisant des senseurs et des gyroscopes permettant de mesurer les accélérations et les changements de direction. Ainsi, le mouvement relatif entre deux images consécutives est connu mis à part le fait que les données inertielles sont affectées par du bruit et un certain biais.

La vie d'un missile peut se décomposer en trois parties : la lancement, le vol à vitesse de croisière, et la phase terminale. Cette phase de guidage terminal commence typiquement dans les derniers deux/trois kilomètres avant d'atteindre la cible. Au début de cette dernière phase, la cible peut ne pas être visible dans l'image à cause des erreurs sur les données inertielles fournies par le missile. La première étape consiste donc à retrouver la position de la cible et à réestimer la position du missile : il s'agit de la phase d'acquisition.

La seconde étape est la poursuite de la cible. Une fois la position de la cible estimée pendant la phase d'acquisition, nous prédisons sa position d'image en image en tenant compte du mouvement relatif du missile.

La connaissance de la position exacte du missile et de la distance missile-cible augmente de manière considérable la précision du guidage du missile. Ces valeurs sont aussi utiles afin de projeter correctement le modèle 3D de la scène sur l'image, et de pouvoir prédire la position d'un point de l'image – la position de la cible par exemple. Le calcul de pose

est donc primordial pour effectuer ces corrections. De plus, deux contraintes doivent être respectées :

- les calculs doivent pouvoir être effectués en temps réel ;
- l'algorithme doit pouvoir s'accommoder de tout type de scène : non planaire ou planaire.

8.1 Acquisition

La première étape de la phase terminale du guidage d'un missile est la préacquisition/-acquisition qui consiste à estimer la position initiale de la cible dans l'image et réestimer la position du missile.

Le principe de l'algorithme est de calculer la position du missile sur les premières images, pour finalement mettre à jour les données inertielles du missile. Ainsi, à chaque image, nous calculons la position du missile, et nous utilisons un filtre de Kalman pour fusionner les résultats. Le filtre de Kalman nous donne également l'incertitude sur le résultat, ce qui nous permet de décider du moment du passage de la position donnée par la centrale inertielle à la position donnée par le filtre de Kalman.

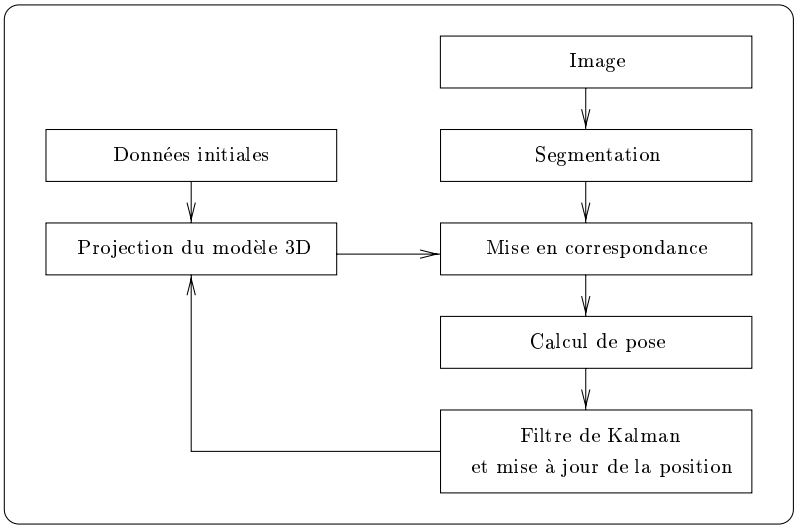


FIG. 8.1: *Principe du calcul de pose.*

Le calcul de pose est effectué à partir de correspondances 2D-3D de points, de la manière suivante (voir figure 8.1).

1. Extraire des points dans l'image courante.
2. En même temps, projeter le modèle 3D sur l'image.
3. Mettre en correspondance les deux ensembles de points. Pour ce faire, calculer dans un premier temps la taille maximum de la fenêtre de recherche en tenant compte des erreurs inertielles.

4. La mise en correspondance fournit des correspondances 2D-3D de points (nous pouvons faire le lien entre les points 3D du modèle et les points 2D du modèle projeté).

5. Effectuer un calcul de pose afin de calculer la position du missile et les attitudes à partir de ces correspondances.

8.2 Poursuite

Une fois la phase d'acquisition réussie, la seconde étape est la poursuite de la cible jusqu'au point d'impact. La principe consiste à prédire la position de la cible dans chaque image à partir du mouvement relatif entre deux images consécutives, puis à améliorer la position 2D par une corrélation locale (voir figure 8.2). Afin de valider la cible poursuivie, nous effectuons des réacquisitions en tâche de fond. Un superviseur se charge de vérifier la cohérence entre la réacquisition et la poursuite.

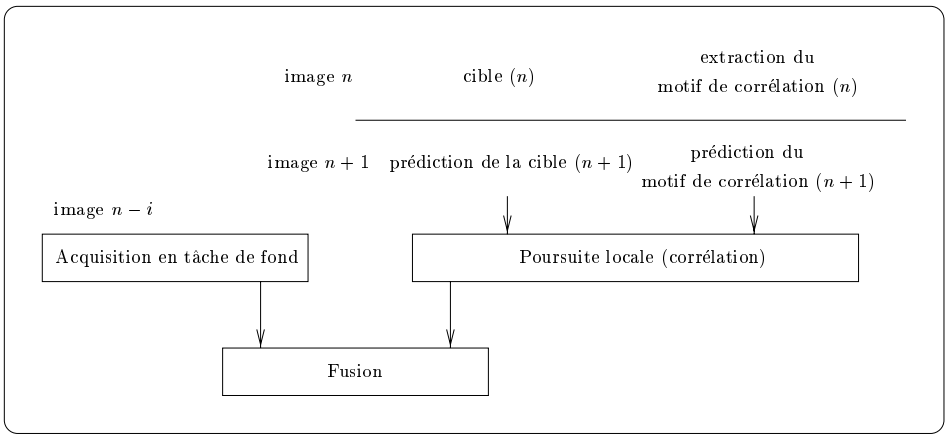


FIG. 8.2: *Principe de la poursuite.*

8.2.1 Prédiction de la position de la cible

À chaque image, nous prédisons la localisation de la cible à partir de l'image précédente. La prédiction d'un point est effectuée en deux étapes.

1. Nous effectuons d'abord une reconstruction du point à partir de l'image précédente en utilisant un modèle orthographique à l'échelle de caméra. Rappelons que ce modèle est valide lorsque la distance caméra-objet est importante par rapport à la taille de l'objet. Ce modèle présuppose que tous les points 3D se trouvent dans un plan parallèle à l'image, et qu'ils sont situés à une distance t_z du centre de projection. Ainsi, un point de coordonnées pixelliques (u, v) a pour coordonnées 3D :

$$\boldsymbol{M} = {}^{t}\left(\begin{array}{ccc} \dfrac{u-u_0}{\alpha_u}\, t_z & \dfrac{v-v_0}{\alpha_v}\, t_z & t_z \end{array} \right)$$

où $(u_0, v_0, \alpha_u, \alpha_v)$ sont les quatre paramètres intrinsèques de la caméra (centre de l'image et facteurs d'échelle).

2. Ensuite, le point est reprojeté dans l'image courante :

$$\boldsymbol{m} = \begin{pmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{R} & \boldsymbol{t} \\ {}^t\boldsymbol{0} & 1 \end{pmatrix} \boldsymbol{M}$$

où $\mathbf{R}$ et $\boldsymbol{t}$ représentent le mouvement relatif entre les deux images.

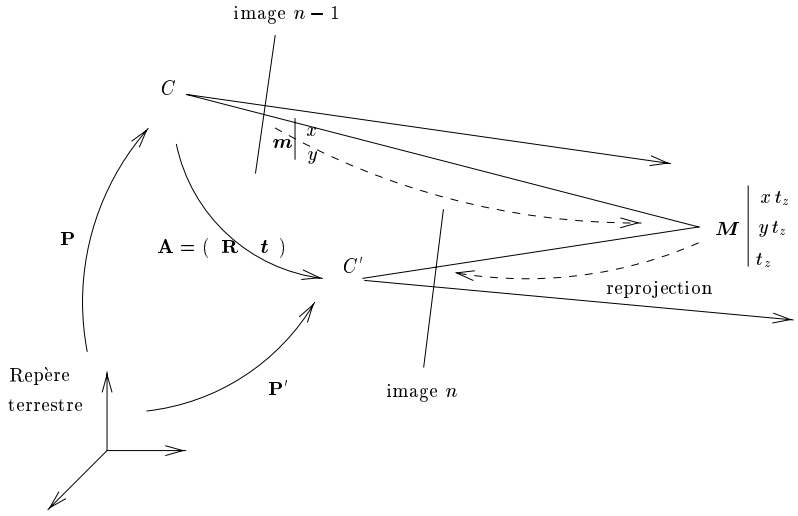


FIG. 8.3: *Prédiction d'un point de l'image : le point est d'abord reconstruit dans l'image précédente en utilisant un modèle orthographique à l'échelle de caméra, et est ensuite reprojeté dans l'image courante en tenant compte du mouvement relatif entre les deux images.*

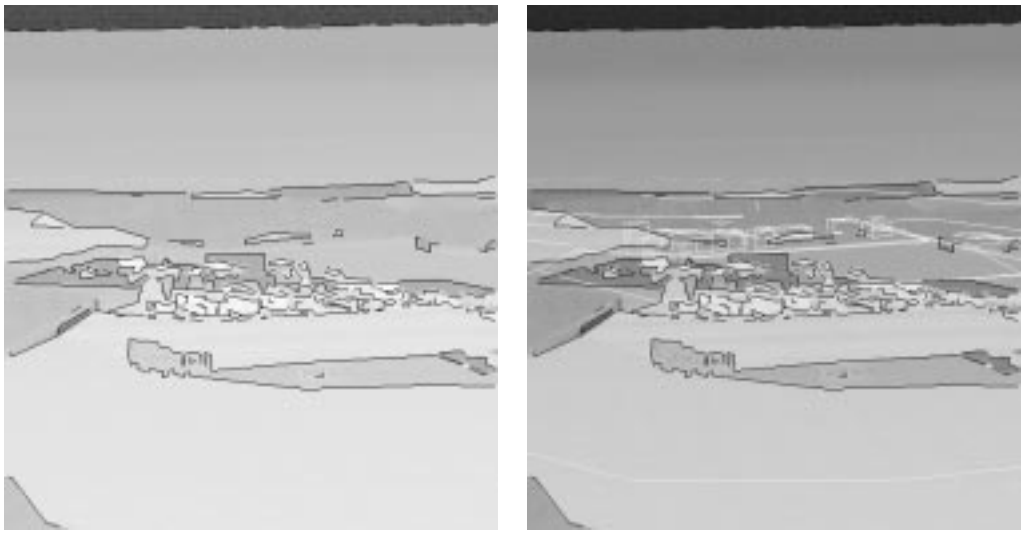
8.2.2 Corrélation

La corrélation permet de localiser de manière précise la position dans l'image. Le motif de corrélation est calculé en effectuant une prédiction sur chaque pixel. La mesure de corrélation utilisée est SAD (somme des carrés des différences de niveaux de gris). Cette mesure donne des résultats satisfaisants car le changement d'intensité lumineuse est faible entre deux images consécutives.

8.3 Résultats expérimentaux

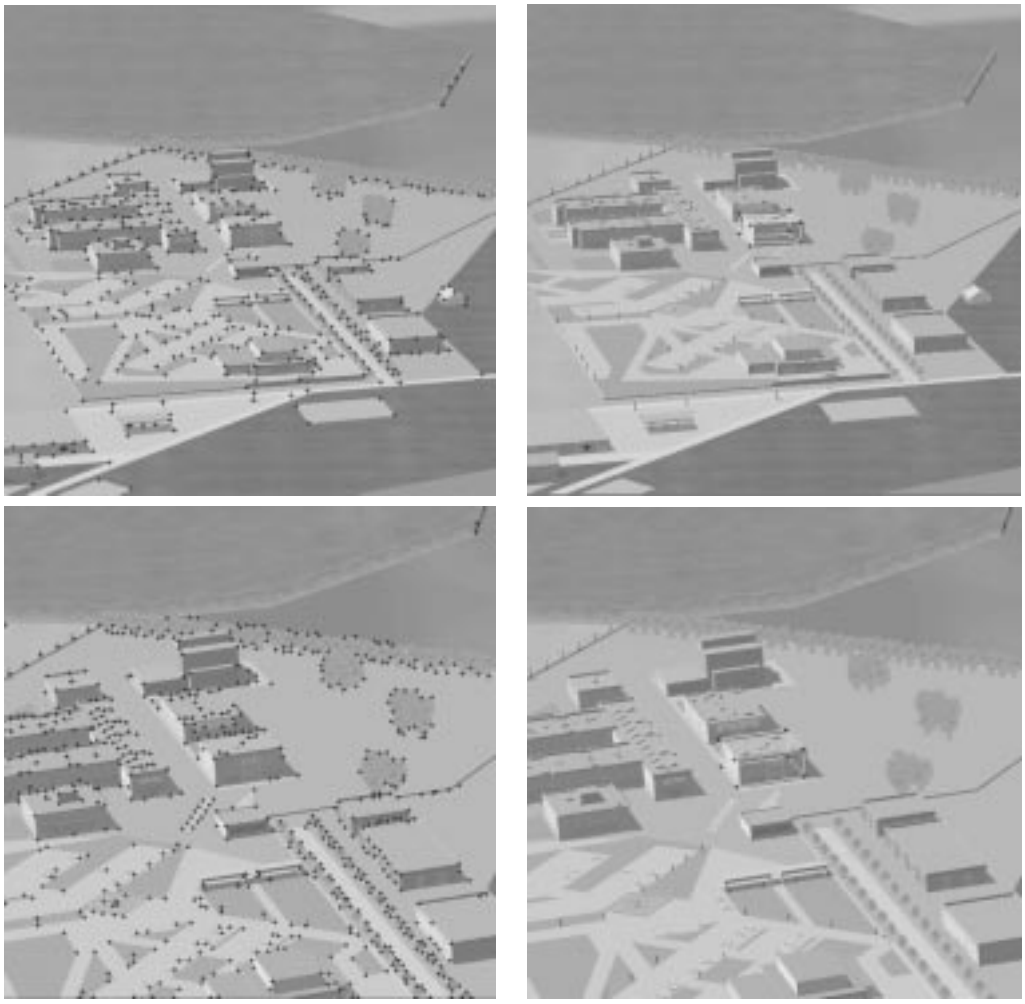
L'algorithme de calcul de pose a été utilisé sur un grand nombre de scènes : scènes rurales et urbaines. Cet algorithme calcule la position et les attitudes du missile en temps réel, et permet d'améliorer la distance entre le point visé et le point d'impact de 30 %.

Les figures 8.4 et 8.5 montrent les performances de l'algorithme sur quelques images. Remarquons que le modèle projeté coïncide exactement avec les points extraits après le calcul de pose (ce n'est pas le cas avec les données inertielles). L'erreur sur la distance missile-cible estimée par l'algorithme est environ 5 %.



(a) (b)

FIG. 8.4: *(a) Contours extraits (b) Image and segments projetés du modèle suivant les données inertielles du missile.*



(a) (b)

FIG. 8.5: *(a) Points extraits (b) Projection du modèle* 3D *après calcul de pose. Remarquons que le modèle coïncide avec les points extraits.*

Conclusion et perspectives

> "Quelque incertitude et quelque variété qui paraisse dans le monde, on y remarque néanmoins un certain enchaînement secret, et un ordre réglé de tout temps par la Providence, qui fait que chaque chose marche en son rang, et suit le cours de sa destinée."
>
> La Rochefoucauld, *Maximes*, 1678.

Dans le cadre de cette thèse, nous nous sommes intéressés successivement :

- à la poursuite de points (mise en correspondance) sur une séquence d'images (monoculaire et stéréoscopique) ;
- au problème du calcul de pose à partir d'une image et d'un modèle 3D, à partir de correspondances de points ou de droites ;
- au problème de reconstruction euclidienne à partir de points sur une séquence d'images, sous deux approches différentes :
  1. avec un modèle perspectif de caméra étalonnée, le mouvement de la caméra étant inconnu ;
  2. avec un modèle affine de caméra non étalonnée montée sur le bras d'un robot.

Poursuite de points

Le problème de mise en correspondance de points sur une séquence d'images est une étape requise pour de nombreux problèmes en vision par ordinateur utilisant plusieurs images. Le principe de la méthode proposée consiste à effectuer une extraction de points caractéristiques sur la première image, puis à prédire, uniquement par corrélation, la position de ces points sur les images suivantes.

Afin d'obtenir une précision sous-pixellique des positions des points dans les images, nous avons proposé une méthode basée sur l'interpolation des scores de corrélation avec les points voisins, et analysé la précision obtenue avec les méthodes proposées (interpolation par deux paraboles ou un paraboloïde). L'avantage de la méthode est le faible coût en

temps de calcul (la méthode se réduit à la multiplication d'une matrice et d'un vecteur pour chaque point).

Nous avons étendu la méthode de poursuite de points au cas d'une séquence d'images stéréoscopiques. L'utilisation de la géométrie épipolaire permet de réduire la fenêtre de recherche pour la localisation des points, et d'éliminer de mauvaises mises en correspondance gauche-droite. L'obtention de mise en correspondance de points sur des couples d'images est utile pour pouvoir effectuer l'auto-étalonnage d'une tête stéréoscopique par exemple (Horaud et Csurka [Hor 98], Csurka et al. [Csu 98]).

Calcul de pose et reconstruction

Nous avons proposé une approche unifiée pour les problèmes de calcul de pose et de reconstruction dont les équations de base sont identiques. La différence provient du nombre d'équations (pour le problème de reconstruction, on s'intéresse à plusieurs images), et des inconnues. Nous avons supposé que les paramètres intrinsèques de la caméra sont connus (en pratique, on peut utiliser les paramètres du constructeur). Nous avons utilisé le lien existant entre les modèles affines de caméra (orthographique à l'échelle ou paraperspectif) et le modèle perspectif. Nous avons étendu l'algorithme itératif proposé par Dementhon faisant le lien entre le modèle orthographique à l'échelle et perspectif, pour le cas de la reconstruction à partir d'une séquence d'images. Nous avons proposé une variante faisant le lien entre le modèle paraperspectif et perspectif. Ces algorithmes font l'hypothèse, à chaque pas d'itération, d'un modèle affine de caméra, mais convergent, à la fin, vers la solution obtenue avec un modèle perspectif. L'intérêt de cette approche est d'obtenir la solution suivant une bonne modélisation géométrique d'une caméra, en utilisant une succession d'approximations du modèle afin de simplifier les calculs. La méthode ne fait intervenir que des calculs algébriques simples: le calcul de pose se limite à la résolution d'un système linéaire, et la complexité de la méthode de reconstruction se réduit à la décomposition en valeur singulière d'une matrice. Le comportement (et la convergence) des algorithmes s'explique également d'un point de vue géométrique, contrairement aux méthodes de minimisation non linéaires pour lesquelles se pose aussi le problème de l'initialisation. Les résultats obtenus par ces méthodes itératives sont proches des solutions obtenues avec une méthode de minimisation non linéaire.

Les algorithmes itératifs proposés peuvent être vus comme des extensions de travaux existants :

- de la méthode itérative pour le calcul de pose proposée par Dementhon (lien entre le modèle orthographique à l'échelle et perspectif) [Dem 92a, Dem 93] ;
- de la méthode de factorisation pour la reconstruction proposée par Tomasi, Poelman et Kanade (modèle affine de caméra) [Tom 91a, Tom 92, Poe 94, Poe 95].

Calcul de pose

L'algorithme complet de calcul de pose peut se résumer de la façon suivante.

1. Effectuer une segmentation de l'image: extraction des points caractéristiques et des segments.

2. Projeter le modèle dans l'image (suivant une orientation approximative connue) et mettre en correspondance les points ou les segments de droites entre le modèle projeté et les primitives extraites dans l'image (appariement 2D/2D).

3. Effectuer un calcul de pose en utilisant un des algorithmes itératifs ; en déduire la position et l'orientation de la caméra par rapport à l'objet (ou de manière équivalente la position de l'objet par rapport à la caméra).

4. Éventuellement, effectuer une minimisation non linéaire pour améliorer la qualité de la reconstruction obtenue (méthode de Levenberg-Marquardt).

Reconstruction

Nous disposons d'une chaîne de traitement complète pour effectuer une reconstruction euclidienne à partir d'une séquence d'images, sans connaissance à priori du mouvement de la caméra. Les étapes peuvent être résumées de la façon suivante.

1. Faire l'acquisition d'une séquence d'images. Extraire les points caractéristiques sur la première image, et retrouver leur position sur les images suivantes (algorithme de poursuite de points).

2. Effectuer une reconstruction euclidienne sur le plus grand sous-ensemble de points visibles sur le plus grand sous-ensemble d'images. Pour ce faire, on maximise le nombre de coefficients de la matrice extraite de la matrice de mesure contenant les points, en commençant à la première image. La reconstruction est effectuée en deux étapes : nous effectuons d'abord une reconstruction affine, puis le passage de la reconstruction affine à une reconstruction euclidienne.

3. Calculer les points manquants dans les images à partir de la reconstruction précédente par propagation des données.

4. Effectuer une reconstruction euclidienne sur la matrice complète.

5. Éventuellement, effectuer une minimisation non linéaire pour améliorer la qualité de la reconstruction obtenue (méthode de Levenberg-Marquardt).

6. Visualiser la reconstruction obtenue.

L'algorithme de reconstruction calcule non seulement la structure 3D de la scène (les coordonnées 3D des points), mais également la position et orientation de la caméra par rapport à l'objet pour chaque image.

Notons que l'extension de la méthode de reconstruction itérative à partir de droites a peu d'intérêt d'un point de vue pratique. Quan a montré qu'il faut au moins 7 orientations différentes pour utiliser la méthode de reconstruction euclidienne avec un modèle affine de caméra proposée dans [Qua 97b, Qua 97c].

Nous avons également présenté une seconde approche du problème de reconstruction : reconstruction euclidienne avec un modèle affine de caméra non étalonnée montée sur le bras d'un robot, l'information euclidienne étant fournie par le déplacement euclidien du robot.

Perspectives

Ces travaux ouvrent plusieurs voies de recherche dont certaines sont en cours de développement.

- La mise en correspondance de points possède de nombreuses applications (robotique...)
- Il serait intéressant de pouvoir effectuer une reconstruction euclidienne sur une séquence d'images avec un modèle perspectif de caméra non étalonnée (i.e. les paramètres intrinsèques sont inconnus). Sturm et Triggs ont proposé une méthode de factorisation pour effectuer une reconstruction projective avec un modèle perspectif de caméra non étalonnée. Il serait cependant intéressant de pouvoir effectuer une reconstruction euclidienne en faisant l'hypothèse que les paramètres intrinsèques restent constants le long de la séquence d'images.
- Le problème de reconstruction euclidienne avec une caméra montée sur un robot peut être repris en supposant un modèle perspectif (et non plus affine), et en s'intéressant également à l'estimation de la transformation caméra-pince.
- La méthode de reconstruction proposée est bien adaptée pour des séquences d'images pas trop longues. Le problème de reconstruction à partir d'une longue séquence d'images n'est pas simple car chaque point n'est visible que sur un petit sous-ensemble d'images : dans ce cas, la matrice de mesures est creuse, et l'estimation des points manquants n'est pas fiable.

Annexe A

Modèle paraperspectif : Interprétation géométrique

Cette annexe apporte quelques détails supplémentaires sur le modèle paraperspectif d'une caméra. En effet, nous avons considéré au paragraphe 2.1.3 que le modèle paraperspectif d'une caméra est une approximation à l'ordre 1 du modèle perspectif :

$$\frac{1}{1+\varepsilon} \approx 1 - \varepsilon$$

Nous justifions ici l'interprétation géométrique du modèle paraperspectif (voir figure A.2).

Rappel mathématique Plaçons-nous dans un espace 3D muni d'un repère euclidien, et considérons un plan π ayant pour normale le vecteur $\boldsymbol{n}$ et situé à une distance d de l'origine du repère. Soit M un point de l'espace et M' la projection de ce point dans le plan π suivant la direction $\boldsymbol{r}$. Les coordonnées du point M' vérifient :

$$\boldsymbol{OM'} = \boldsymbol{OM} - \frac{\boldsymbol{OM} \cdot \boldsymbol{n} - d}{\boldsymbol{r} \cdot \boldsymbol{n}} \boldsymbol{r} \tag{A.1}$$

□

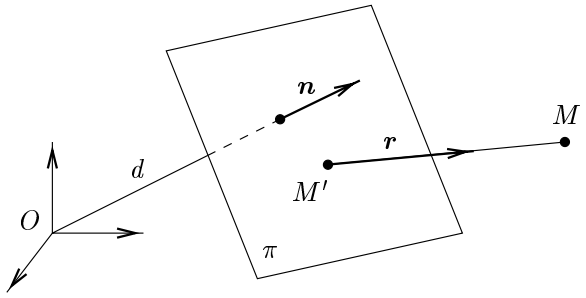


FIG. A.1: *Projection du point M de l'espace dans le plan π suivant la direction $\boldsymbol{r}$. Le plan π est situé à une distance d de l'origine du repère et a pour normale le vecteur $\boldsymbol{n}$.*

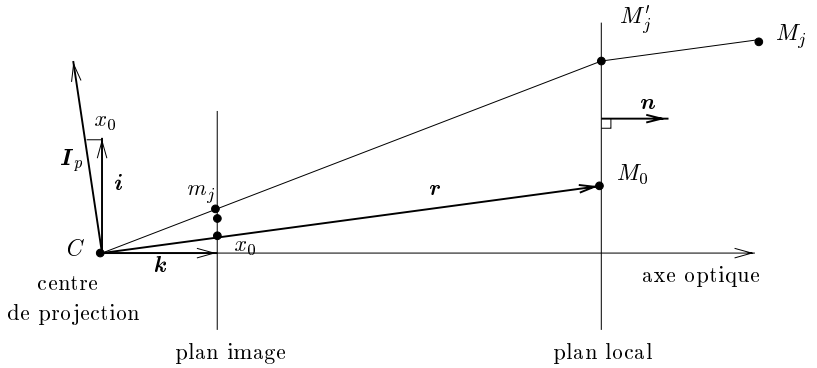


FIG. A.2: *Modélisation géométrique d'un modèle paraperspectif de caméra.*

Considérons un modèle paraperspectif de caméra (voir figure A.2) et exprimons les coordonnées du point M'_j en utilisant la formule précédente. Nous nous plaçons dans un espace muni d'un repère euclidien ayant pour origine M. Le plan de projection est le plan local représenté sur la figure A.2. Comme celui-ci passe par l'origine du repère, nous avons $d = 0$. La direction de projection dans ce plan est définie par le vecteur $\boldsymbol{r} = \boldsymbol{CM}_0$, et la normale à ce plan est $\boldsymbol{n} = \boldsymbol{k}$. Remarquons également que le dénominateur de l'équation (A.1) se simplifie : $\boldsymbol{r} \cdot \boldsymbol{n} = \boldsymbol{k} \cdot \boldsymbol{CM}_0 = t_z$. Ainsi, nous obtenons :

$$\begin{aligned} \boldsymbol{M}_0\boldsymbol{M}'_j &= \boldsymbol{M}_0\boldsymbol{M}_j - \frac{\boldsymbol{k} \cdot \boldsymbol{M}_0\boldsymbol{M}_j}{t_z} \boldsymbol{CM}_0 \\ &= \boldsymbol{M}_0\boldsymbol{M}_j - (\boldsymbol{K} \cdot \boldsymbol{M}_0\boldsymbol{M}_j) \boldsymbol{CM}_0 \end{aligned} \tag{A.2}$$

D'autre part, en utilisant le théorème de Thalès en faisant intervenir le plan local et le plan image de la figure A.2, nous avons :

$$\begin{aligned} x_j - x_0 &= \frac{\boldsymbol{i} \cdot \boldsymbol{M}_0\boldsymbol{M}'_j}{t_z} \\ &= \boldsymbol{I} \cdot \boldsymbol{M}_0\boldsymbol{M}'_j \end{aligned}$$

Enfin, en remplaçant $\boldsymbol{M}_0\boldsymbol{M}'_j$ par l'expression donnée par l'équation (A.2) :

$$x_j - x_0 = \boldsymbol{I} \cdot (\boldsymbol{M}_0\boldsymbol{M}_j - (\boldsymbol{K} \cdot \boldsymbol{M}_0\boldsymbol{M}_j) \boldsymbol{CM}_0)$$

En remarquant que

$$\boldsymbol{I} \cdot \boldsymbol{CM}_0 = x_0$$

nous avons finalement :

$$\begin{aligned} x_j - x_0 &= (\boldsymbol{I} - x_0\boldsymbol{K}) \cdot \boldsymbol{M}_0\boldsymbol{M}_j \\ &= \boldsymbol{I}_p \cdot \boldsymbol{M}_0\boldsymbol{M}_j \end{aligned}$$

De manière similaire, nous pouvons montrer que :

$$y_j - y_0 = \boldsymbol{J}_p \cdot \boldsymbol{M}_0\boldsymbol{M}_j$$

Nous retrouvons ainsi les expressions données par les équations (2.16) et (2.17). Ce calcul montre ainsi que le modèle paraperspectif est bien une approximation à l'ordre 1 du modèle perspectif.

Annexe B

Décomposition QL *d'une matrice*

Une matrice peut se décomposer en un produit de deux matrices : une matrice orthogonale $\mathbf{Q}$ suivie d'une matrice triangulaire basse $\mathbf{L}$. Cette décomposition est une variante de la décomposition QR, et elle est utile pour extraire les paramètres intrinsèques et extrinsèques d'une caméra à partir d'une matrice de projection.

Mettons en évidence le lien existant entre une décomposition QR et une décomposition QL, et montrons comment obtenir une décomposition QL d'une matrice $\mathbf{A}$ de taille $m \times n$ en utilisant un algorithme de décomposition QR.

Soit $\mathbf{P}$ la matrice de permutation carrée $n \times n$ composée de 1 sur la seconde diagonale :

$$\mathbf{P} = \begin{pmatrix} 0 & & 1 \\ & \diagup & \\ 1 & & 0 \end{pmatrix}$$

et effectuons tout d'abord une décomposition QR du produit $\mathbf{AP}$. Nous obtenons :

$$\mathbf{AP} = \mathbf{QR}$$

Comme nous avons :

$$\mathbf{PP} = \mathcal{I}_n$$

il s'en suit :

$$\mathbf{A} = (\mathbf{QP})(\mathbf{PRP})$$

Posons :

$$\begin{aligned} \mathbf{Q}' &= \mathbf{QP} \\ \mathbf{L} &= \mathbf{PRP} \end{aligned}$$

On a bien $\mathbf{A} = \mathbf{Q}'\mathbf{L}$ avec $\mathbf{Q}'$ orthogonale et $\mathbf{L}$ triangulaire inférieure. Pour finir, nous pouvons nous arranger pour que $\det \mathbf{Q}' = +1$ et choisir le signe des coefficients diagonaux de la matrice $\mathbf{L}$ en multipliant $\mathbf{Q}'$ à droite et $\mathbf{L}$ à gauche par une matrice $\mathbf{P}'$ de la forme :

$$\mathbf{P} = \begin{pmatrix} \pm 1 & & 0 \\ & \diagdown & \\ 0 & & \pm 1 \end{pmatrix}$$

car on a l'égalité :

$$\mathbf{A} = \mathbf{Q}'\mathbf{L} = (\mathbf{Q}'\mathbf{P}')(\mathbf{P}'\mathbf{L})$$

Annexe C

Quaternions et rotations

Les quaternions, inventés en 1843 par le mathématicien irlandais William Hamilton, jouent en dimension 3 et 4, vis à vis des groupes orthogonaux, un rôle analogue à celui des nombres complexes en dimension 2.

C.1 Rappel sur les quaternions

Il existe une algèbre $\mathcal{H}$ de dimension 4 sur $\mathbb{R}$, appelée algèbre des quaternions, munie d'une base 1, i, j, k, telle que :

- 1 est élément neutre pour la multiplication ;
- $i^2 = j^2 = k^2 = -1$;
- $jk = -kj = i$, $ki = -ik = j$, $ij = -ji = k$.

Soient $\boldsymbol{q}$ et $\boldsymbol{q}'$ deux quaternions :

$$\begin{aligned} \boldsymbol{q} &= q_0 + q_x i + q_y j + q_z k = (q_0, \boldsymbol{w}) \\ \boldsymbol{q}' &= q'_0 + q'_x i + q'_y j + q'_z k = (q'_0, \boldsymbol{w}') \end{aligned}$$

Grâce à ces formules, nous pouvons calculer le produit de deux quaternions, noté "*" :

$$\begin{aligned} \boldsymbol{q} * \boldsymbol{q}' &= (q_0, \boldsymbol{w}) * (q'_0, \boldsymbol{w}') \\ &= (q_0 q'_0 - \boldsymbol{w} \cdot \boldsymbol{w}', \boldsymbol{w} \wedge \boldsymbol{w}' + a\boldsymbol{w}' + a'\boldsymbol{w}) \end{aligned}$$

Ce produit, non commutatif, peut également s'écrire sous forme matricielle :

$$\begin{aligned} \boldsymbol{q} * \boldsymbol{q}' &= \mathbf{Q}(\boldsymbol{q})\boldsymbol{q}' \\ &= \mathbf{W}(\boldsymbol{q}')\boldsymbol{q} \end{aligned}$$

où

$$\mathbf{Q}(\boldsymbol{q}) = \begin{pmatrix} q_0 & -q_x & -q_y & -q_z \\ q_x & q_0 & -q_z & q_y \\ q_y & q_z & q_0 & -q_x \\ q_z & -q_y & q_x & q_0 \end{pmatrix} \quad \text{et} \quad \mathbf{W}(\boldsymbol{q}') = \begin{pmatrix} q'_0 & -q'_x & -q'_y & -q'_z \\ q'_x & q'_0 & q'_z & -q'_y \\ q'_y & -q'_z & q'_0 & q'_x \\ q'_z & q'_y & -q'_x & q'_0 \end{pmatrix}$$

On définit le conjugué $\overline{\boldsymbol{q}}$ de $\boldsymbol{q}$ par :

$$\overline{\boldsymbol{q}} = q_0 - q_x i - q_y j - q_z k$$

et la norme par :

$$\|\boldsymbol{q}\| = \sqrt{\boldsymbol{q}\overline{\boldsymbol{q}}} = \sqrt{\overline{\boldsymbol{q}}\boldsymbol{q}} = \sqrt{q_0^2 + q_x^2 + q_y^2 + q_z^2}$$

Nous avons les propriétés suivantes :

$$\begin{aligned} \overline{\boldsymbol{q}\boldsymbol{q}'} &= \overline{\boldsymbol{q}}'\overline{\boldsymbol{q}} \\ \overline{\overline{\boldsymbol{q}}} &= \boldsymbol{q} \\ \|\boldsymbol{q}\boldsymbol{q}'\| &= \|\boldsymbol{q}\|\,\|\boldsymbol{q}'\| \\ \boldsymbol{q}^{-1} &= \frac{\overline{\boldsymbol{q}}}{\|\boldsymbol{q}\|^2} \end{aligned}$$

C.2 Représentation des rotations par les quaternions unitaires

Soit $\boldsymbol{q}$ un quaternion unitaire, et soit l'application :

$$\begin{array}{rccc} S_{\boldsymbol{q}} : & Q & \longrightarrow & Q \\ & \boldsymbol{p} & \longmapsto & \boldsymbol{p}' = \boldsymbol{q} * \boldsymbol{p} * \overline{\boldsymbol{q}} \end{array}$$

Nous avons les propriétés suivantes :

- $S_{\boldsymbol{q}}$ est bijective, et $(S_{\boldsymbol{q}})^{-1} = S_{\overline{\boldsymbol{q}}}$;
- $S_{\boldsymbol{q}_1 * \boldsymbol{q}_2} = S_{\boldsymbol{q}_1} S_{\boldsymbol{q}_2}$;
- $S_{\boldsymbol{q}}$ conserve la norme : $\|S_{\boldsymbol{q}}(\boldsymbol{p})\| = \|\boldsymbol{p}\|$;
- l'image d'un quaternion imaginaire pur est un quaternion imaginaire pur.

Exprimons l'image $\boldsymbol{p}'$ de $\boldsymbol{p}$ sous forme matricielle :

$$\begin{aligned} \boldsymbol{p}' &= \boldsymbol{q} * \boldsymbol{p} * \overline{\boldsymbol{q}} \\ &= (\boldsymbol{q} * \boldsymbol{p}) * \overline{\boldsymbol{q}} \\ &= (\mathbf{Q}(\boldsymbol{q})\boldsymbol{p}) * \overline{\boldsymbol{q}} \\ &= \mathbf{W}(\overline{\boldsymbol{q}})(\mathbf{Q}(\boldsymbol{q})\boldsymbol{p}) \\ &= ({}^t\mathbf{W}(\overline{\boldsymbol{q}})\mathbf{Q}(\boldsymbol{q}))\boldsymbol{p} \end{aligned}$$

On vérifie que la matrice ${}^t\mathbf{W}(\overline{\boldsymbol{q}})\mathbf{Q}(\boldsymbol{q})$ est une matrice orthonormée :

$$ {}^t\mathbf{W}(\overline{\boldsymbol{q}})\mathbf{Q}(\boldsymbol{q}) = \begin{pmatrix} 1 & {}^t\boldsymbol{0} \\ \boldsymbol{0} & \mathbf{R} \end{pmatrix}_{4\times 4} $$

où

$$ \mathbf{R} = \begin{pmatrix} q_0^2+q_x^2-q_y^2-q_z^2 & 2(q_xq_y-q_0q_z) & 2(q_xq_z+q_0q_y) \\ 2(q_xq_y+q_0q_z) & q_0^2-q_x^2+q_y^2-q_z^2 & 2(q_yq_z-q_0q_x) \\ 2(q_xq_z-q_0q_y) & 2(q_yq_z+q_0q_x) & q_0^2-q_x^2-q_y^2+q_z^2 \end{pmatrix} \tag{C.1} $$

La matrice $\mathbf{R}$ est la matrice de rotation associée au quaternion unitaire $\boldsymbol{q}$

Propriétés

- $S_{\boldsymbol{q}}$ laisse $\mathbb{R}^3$ (assimilé à l'ensemble des quaternions purs) invariants et la restriction de $S_{\boldsymbol{q}}$ à $\mathbb{R}^3$ est une rotation.
- Toute rotation est représentée par un quaternion unitaire.
- Les quaternions $\boldsymbol{q}$ et $\boldsymbol{q}'$ représentent la même rotation ($S_{\boldsymbol{q}} = S_{\boldsymbol{q}'}$) équivaut à $\boldsymbol{q} = \pm\boldsymbol{q}'$
- Le produit de deux rotations est décrit par le produit des quaternions associées.
- Une rotation $\mathbf{R}$ d'axe $\boldsymbol{n}$ et d'angle θ est représentée par les quaternions :

$$ \boldsymbol{q} = \pm\left(\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)\boldsymbol{n}\right) $$

C.3 Problème de minimisation

Dans l'algorithme du calcul de pose, nous avons introduit une contrainte d'orthogonalité afin d'assurer que la matrice représentant l'orientation de la caméra par rapport à l'objet soit une "vraie" matrice de rotation. Dans le cadre de la reconstruction d'un objet, nous alignons le repère de l'objet avec la première image pour faire le choix des solutions à conserver ou à rejeter. Ces deux cas conduisent à résoudre le même problème : il s'agit d'estimer la "meilleure" matrice de rotation $\mathbf{R}$ telle que :

$$ \mathbf{R}\,{}^t\begin{pmatrix} {}^t\boldsymbol{i} \\ {}^t\boldsymbol{j} \\ {}^t\boldsymbol{k} \end{pmatrix} = \boldsymbol{\mathcal{I}}_3 $$

Cette expression peut se réécrire :

$$ \forall i \in \{1,2,3\},\ \mathbf{R}\boldsymbol{v}_i = \boldsymbol{e}_i $$

où $\boldsymbol{e}_i$ est le i-ième vecteur colonne de la matrice identité, et $\boldsymbol{v}_1 = \boldsymbol{i}$, $\boldsymbol{v}_2 = \boldsymbol{j}$, $\boldsymbol{v}_3 = \boldsymbol{k}$. Nous cherchons donc la matrice de rotation qui minimise le critère :

$$ \min_{\mathbf{R}} \left(\sum_{i=1}^{3} \|\mathbf{R}\boldsymbol{v}_i - \boldsymbol{e}_i\|^2 \right) $$

En représentant la matrice de rotation par un quaternion unitaire $\boldsymbol{q}$, alors le problème de minimisation devient :

$$\min_{\|\boldsymbol{q}\|=1} \left(\sum_{i=1}^{3} \|\boldsymbol{q} * \boldsymbol{v}_i * \overline{\boldsymbol{q}} - \boldsymbol{e}_i\|^2 \right) \tag{C.2}$$

En utilisant le fait que $\|\boldsymbol{q}\| = 1$:

$$\begin{aligned} \|\boldsymbol{q} * \boldsymbol{v}_i * \overline{\boldsymbol{q}} - \boldsymbol{e}_i\|^2 &= \|\boldsymbol{q} * \boldsymbol{v}_i * \overline{\boldsymbol{q}} - \boldsymbol{e}_i\| \|\boldsymbol{q}\|^2 \\ &= \|\boldsymbol{q} * \boldsymbol{v}_i * \overline{\boldsymbol{q}} * \boldsymbol{q} - \boldsymbol{e}_i * \boldsymbol{q}\|^2 \\ &= \|\boldsymbol{q} * \boldsymbol{v}_i - \boldsymbol{e}_i * \boldsymbol{q}\|^2 \\ &= \|(\mathbf{Q}(\boldsymbol{e}_i) - \mathbf{W}(\boldsymbol{v}_i))\boldsymbol{q}\|^2 \\ &= {}^t\boldsymbol{q}\,{}^t[\mathbf{Q}(\boldsymbol{e}_i) - \mathbf{W}(\boldsymbol{v}_i)][\mathbf{Q}(\boldsymbol{e}_i) - \mathbf{W}(\boldsymbol{v}_i)]\boldsymbol{q} \\ &= {}^t\boldsymbol{q}\mathbf{A}_i\boldsymbol{q} \end{aligned}$$

où $\mathbf{A}_i$ est la matrice symétrique de taille 4×4 définie par :

$$\mathbf{A}_i = {}^t[\mathbf{Q}(\boldsymbol{e}_i) - \mathbf{W}(\boldsymbol{v}_i)][\mathbf{Q}(\boldsymbol{e}_i) - \mathbf{W}(\boldsymbol{v}_i)]$$

Le critère à minimiser (C.2) devient :

$$\begin{aligned} \min_{\boldsymbol{q}} \left(\sum_{i=1}^{3} {}^t\boldsymbol{q}\mathbf{A}_i\boldsymbol{q} \right) &= \min_{\boldsymbol{q}} \left({}^t\boldsymbol{q} \left(\sum_{i=1}^{3} \mathbf{A}_i \right) \boldsymbol{q} \right) \\ &= \min_{\boldsymbol{q}} \left({}^t\boldsymbol{q}\mathbf{B}\boldsymbol{q} \right) \end{aligned}$$

avec

$$\mathbf{B} = \sum_{i=1}^{3} \mathbf{A}_i$$

Sous la contrainte que le quaternion doit être unitaire, le critère à minimiser devient :

$$\min_{\boldsymbol{q}} Q = \min_{\boldsymbol{q}} \left({}^t\boldsymbol{q}\mathbf{B}\boldsymbol{q} + \lambda(1 - {}^t\boldsymbol{q}\boldsymbol{q}) \right) \tag{C.3}$$

En dérivant Q par rapport à $\boldsymbol{q}$, et en annulant la dérivée, nous obtenons :

$$\frac{dQ}{d\boldsymbol{q}} = 2(\mathbf{B}\boldsymbol{q} - \lambda\boldsymbol{q}) = 0$$

Nous en déduisons :

$$\lambda\boldsymbol{q} = \mathbf{B}\boldsymbol{q}$$

puis

$$\lambda\,{}^t\boldsymbol{q}\boldsymbol{q} = {}^t\boldsymbol{q}\mathbf{B}\boldsymbol{q}$$

En substituant cette solution dans l'équation (C.3), nous avons :

$$Q = \lambda$$

Le quaternion unitaire qui minimise Q est donc le vecteur propre unitaire de la matrice $\mathbf{B}$ associé à la plus petite valeur propre de $\mathbf{B}$, soit λ.

La matrice de rotation correspondant au quaternion unitaire $\boldsymbol{q} = (q_0, q_x, q_y, q_z)$ est alors donnée par l'équation (C.1).

Démonstration La matrice $\mathbf{B}$ est symétrique, réelle, et définie positive, elle est donc diagonalisable et ses valeurs propres sont positives. Soient $\boldsymbol{u}_1$, $\boldsymbol{u}_2$, $\boldsymbol{u}_3$, $\boldsymbol{u}_4$ les vecteurs propres unitaires de $\mathbf{B}$ associés aux valeurs propres λ_1, λ_2, λ_3, λ_4 avec $\lambda_1 < \lambda_2 < \lambda_3 < \lambda_4$. Nous avons :

$$\forall i \in \{1, \ldots, 4\},\ \mathbf{B}\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i$$

Les vecteurs $\boldsymbol{u}_i$ forment une base orthonormée, et le quaternion $\boldsymbol{q}$ peut s'écrire dans cette base :

$$\boldsymbol{q} = \sum_{i=1}^{4} \mu_i \boldsymbol{u}_i$$

En tenant compte du fait que les vecteurs propres forment une base orthonormée, nous pouvons écrire :

$${}^t\boldsymbol{q}\mathbf{B}\boldsymbol{q} = \sum_{i=1}^{4} \mu_i^2 \lambda_i$$

Comme le quaternion est unitaire, nous avons $\sum_{i=1}^{4} \mu_i^2 = 1$. L'expression précédente est donc minimale lorsque $\mu_1 = 1$ et $\mu_2 = \mu_3 = \mu_4 = 0$. Nous obtenons ainsi :

$$\begin{aligned} \boldsymbol{q} &= \boldsymbol{u}_1 \\ {}^t\boldsymbol{q}\mathbf{B}\boldsymbol{q} &= \lambda_1 \end{aligned}$$

Le quaternion unitaire optimal est donc le vecteur propre unitaire de $\mathbf{B}$ associé à sa plus petite valeur propre.

Annexe D

Méthode de reconstruction non linéaire

D.1 Présentation de la méthode

La méthode de reconstruction non linéaire estime à la fois les coordonnées tridimensionnelles des points observés, et les matrices de projection entre l'objet et chaque image (cela revient à estimer une matrice de rotation et un vecteur de translation pour chaque position de la caméra). Elle minimise la somme des carrés des distances entre les projections dans les images des points reconstruits et les positions des points observés. Écrivons l'expression de la projection d'un point j dans une image i :

$$\begin{aligned} \widehat{x}_{ij} &= \frac{\boldsymbol{i}_i \cdot \boldsymbol{M}_j + t_{x_i}}{\boldsymbol{k}_i \cdot \boldsymbol{M}_j + t_{z_i}} \\ \widehat{y}_{ij} &= \frac{\boldsymbol{j}_i \cdot \boldsymbol{M}_j + t_{y_i}}{\boldsymbol{k}_i \cdot \boldsymbol{M}_j + t_{z_i}} \end{aligned}$$

En notant x_{ij} et y_{ij} les positions des points extraits dans les images, nous cherchons à minimiser la quantité :

$$D = f(\ldots, \mathbf{R}_i, \boldsymbol{t}_i, \ldots, \boldsymbol{M}_j, \ldots) = \sum_{ij} \left((x_{ij} - \widehat{x}_{ij})^2 + (y_{ij} - \widehat{y}_{ij})^2 \right)$$

sous la contrainte que chaque matrice $\mathbf{R}_i$ doit être une matrice de rotation. La fonction à minimiser devient :

$$\min \left\{ f(\ldots, \mathbf{R}_i, \boldsymbol{t}_i, \ldots, \boldsymbol{M}_j, \ldots) + \lambda \sum_i \| \mathbf{R}_i {}^t\mathbf{R}_i - \boldsymbol{\mathcal{I}}_3 \|^2 \right\}$$

Cette fonction est non linéaire, il faut donc utiliser des méthodes d'optimisation non linéaires. Nous avons choisi d'utiliser la méthode de Levenberg-Marquardt qui est une

méthode dite à région de confiance, et qui a un comportement amélioré par rapport aux méthodes de Newton classiques. L'algorithme de Levenberg-Marquardt consiste à passer continûment de la méthode de descente de gradient à la méthode de quasi-Newton au fur et à mesure que l'on s'approche du minimum. On trouvera une description détaillée de la méthode dans [Pre 92]. Cependant, pour pouvoir utiliser une méthode de minimisation non linéaire, il est indispensable de disposer d'une solution approchée pour s'assurer de la convergence de l'algorithme. Nous utilisons, pour initialisation, la solution obtenue par une des deux méthodes itératives présentées au chapitre 5.

D.2 Paramétrisation de la matrice de rotation

Chaque matrice de rotation $\mathbf{R}_i$ possède 9 coefficients (matrice 3×3), mais n'a cependant que 3 degrés de liberté. Ceci implique qu'il existe de nombreuses contraintes sur cette matrice (les vecteurs lignes et colonnes sont normés et orthogonaux). Il est donc préférable d'utiliser une paramétrisation d'une matrice de rotation avec moins de paramètres.

Une paramétrisation minimale est d'utiliser les angles d'Euler. Dans ce cas, une matrice de rotation s'écrit comme étant le produit de trois matrices de rotation par rapport à chacun des axes, sous la forme suivante :

$$\begin{aligned}\mathbf{R}_{(\varphi,\theta,\psi)} &= \begin{pmatrix}\cos\varphi & -\sin\varphi & 0\\ \sin\varphi & \cos\varphi & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}\cos\theta & 0 & \sin\theta\\ 0 & 1 & 0\\ -\sin\theta & 0 & \cos\theta\end{pmatrix}\begin{pmatrix}1 & 0 & 0\\ 0 & \cos\psi & -\sin\psi\\ 0 & \sin\psi & \cos\psi\end{pmatrix}\\ &= \begin{pmatrix}\cos\varphi\cos\theta & -\sin\varphi\cos\psi+\cos\varphi\sin\theta\sin\psi & \sin\phi\sin\psi+\cos\varphi\sin\theta\cos\psi\\ \sin\varphi\cos\theta & \cos\varphi\cos\psi+\sin\varphi\sin\theta\sin\psi & -\cos\varphi\sin\psi+\sin\varphi\sin\theta\cos\psi\\ -\sin\theta & \cos\theta\sin\psi & \cos\theta\cos\psi\end{pmatrix}\end{aligned}$$

Le problème avec cette paramétrisation est qu'elle est parfois ambiguë : il peut y avoir plusieurs valeurs pour les angles possibles pour une matrice de rotation donnée.

Afin de de pas être confronté à ce problème, nous procédons comme suit. Soit $\widehat{\mathbf{R}}_i$ la matrice de rotation donnée pour initialisation, et $\mathbf{R}_i$ la vraie matrice de rotation. Nous pouvons alors écrire :

$$\mathbf{R}_i = \widehat{\mathbf{R}}_i\widetilde{\mathbf{R}}_i \tag{D.1}$$

où $\widetilde{\mathbf{R}}_i$ représente la rotation relative entre la vraie rotation et l'orientation estimée. Si $\widehat{\mathbf{R}}_i$ est une bonne estimation de $\mathbf{R}_i$, la matrice $\widetilde{\mathbf{R}}_i = \mathbf{R}_{(\varphi_i,\theta_i,\psi_i)}$ peut être approchée au premier ordre par la matrice suivante :

$$\widetilde{\mathbf{R}}_i = \begin{pmatrix}1 & -\varphi_i & \theta_i\\ \varphi_i & 1 & -\psi_i\\ -\theta_i & \psi_i & 1\end{pmatrix}$$

Ainsi, nous supposons que les matrices $\widehat{\mathbf{R}}_i$ sont fixées, et nous estimons les matrices $\widetilde{\mathbf{R}}_i$ dont les inconnues s'écrivent : $\{\varphi_i, \theta_i, \psi_i\}$. L'expression à minimiser s'écrit maintenant :

$$\min\{f(\ldots,\varphi_i,\theta_i,\psi_i,\boldsymbol{t}_i,\ldots,\boldsymbol{M}_j,\ldots)\}$$

Notons cependant les matrices $\widetilde{\mathbf{R}}_i$ ne sont pas exactement des matrices de rotation. Pour y remédier, à chaque pas pas d'itération de l'algorithme de Levenberg-Marquardt,

nous multiplions la matrice $\widehat{\mathbf{R}}_i$ dans l'équation (D.1) à droite par $\mathbf{R}_{(\varphi_i,\theta_i,\psi_i)}$ et non pas $\widetilde{\mathbf{R}}_i$, puis les variables sont ensuite remises à zéro. On s'assure ainsi que les matrices $\mathbf{R}_i$ dont toujours des matrices de rotation. L'approximation par les matrices $\widetilde{\mathbf{R}}_i$ n'est utilisée que dans l'algorithme de minimisation afin de simplifier la fonction à minimiser (ainsi que les dérivées à calculer). De plus, comme les matrices $\widehat{\mathbf{R}}_i$ sont mises à jour à chaque pas d'itération, nous nous assurons que les changements sur ces matrices sont réalisés par des petites rotations qui peuvent être approchées pendant la minimisation. L'algorithme peut se résumer de la manière suivante.

1. Initialiser $\varphi_i = \theta_i = \psi_i = 0$.
2. Effectuer une itération de l'algorithme de minimisation de Levenberg-Marquardt (estimation de $\varphi_i, \theta_i, \psi_i, \boldsymbol{t}_i, \boldsymbol{M}_j$).
3. Mettre à jour les matrices de rotation $\widehat{\mathbf{R}}_i = \widehat{\mathbf{R}}_i \mathbf{R}_{(\varphi_i,\theta_i,\psi_i)}$.
4. Tant que l'algorithme n'a pas convergé, retourner à l'étape 1.

Bibliographie de l'auteur

Revues

- **Euclidean Shape and Motion from Multiple Perspective Views by Affine Iterations.**
  Stéphane Christy, Radu Horaud. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, Volume 18, pages 1098–1104, Novembre 1996.

- **Object Pose : The Link between Weak Perspective, Paraperspective and Perspective.**
  Radu Horaud, Fadi Dornaika, Bart Lamiroy, Stéphane Christy. *International Journal on Computer Vision (IJCV)*, Volume 22(2), pages 173–189, Mars 1997.

- **Iterative Pose Computation from Line Correspondences.**
  Stéphane Christy, Radu Horaud. *Computer Vision and Image Understanding (CVIU).*

Conférences internationales avec comité de lecture

- **Object Pose : Links between Paraperspective and Perspective.**
  Radu Horaud, Stéphane Christy, Fadi Dornaika, Bart Lamiroy. *International Conference on Computer Vision (ICCV '95)*, Juin 1995, Cambridge, Massachussetts, USA.

- **A Quasi Linear Reconstruction Method from Multiple Perspective Views.**
  Stéphane Christy, Radu Horaud. *International Conference on Intelligent Robots and Systems (IROS '95)*, Pittsburgh, Pennsylvanie, USA, pages 374–380, Août 1995.

- **Euclidean Reconstruction : from Paraperspective to Perspective.**
  Stéphane Christy, Radu Horaud. *4th European Conference on Computer Vision (ECCV '96)*, Cambridge, Angleterre, pages 129–140, Avril 1996.

- **Euclidean Reconstruction and Affine Camera Calibration Using Controlled Robot Motions.**
  Radu Horaud, Stéphane Christy, Roger Mohr. *International Conference on Intelligent Robots and Systems (IROS '97)*, Grenoble, France, pages 1575–1582, Septembre 1997.

- **Fast and Reliable Methods to Compute Object Pose from Line Correspondences.**
  Stéphane Christy, Radu Horaud. *7th International Conference on Computer Analysis of Images and Patterns (CAIP '97)*, Kiel, Allemagne, Septembre 1997.

Autres conférences

- **Terminal Air-to-Ground Missile Guidance by Infrared Seeker.**
  Stéphane Christy, Bruno Mazar, Radu Horaud. *SPIE Aerosense '97, Conference on Acquisition Tracking and Pointing XI*, Orlando, Floride, USA, Avril 1997.

Rapports de recherche

- **Object Pose : The Link between Weak Perspective, Paraperspective and Perspective.**
  Radu Horaud, Fadi Dornaika, Bart Lamiroy, Stéphane Christy. *Rapport de recherche INRIA n° 2356*, Septembre 1994.

- **Euclidean Shape and Motion from Multiple Perspective Views by Affine Iterations.**
  Stéphane Christy, Radu Horaud. *Rapport de recherche INRIA n° 2421*, Décembre 1994.

Communications

- **Modélisation Tridimensionnelle d'Objets Quelconques par Vision Dynamique.**
  Stéphane Christy. *Forum Aérospatiale-Synergimage* (Groupe de concertation Aérospatiale sur le traitement d'images) réunissant doctorants de l'Aérospatiale et industriels, Cannes La Bocca, France, Septembre 1996.

Références bibliographiques

[Alo 90] J.Y. Aloimonos. Perspective approximations. *Image and Vision Computing*, 8(3): 179–192, August 1990.

[Ana 89] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2: 283–310, 1989.

[Arm 96] M. Armstrong, A. Zisserman, and R. Hartley. Self-calibration from image triplets. In B. Buxton and R. Cipolla, editors, *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, volume 1064 of *Lecture Notes in Computer Science*, pages 3–16. Springer-Verlag, April 1996.

[Asa 86] H. Asada and M. Brady. The curvature primal sketch. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 8(1): 2–14, 1986.

[Asc 92] P. Aschwanden and W. Guggenbühl. Experimental results from a comparative study on correlation-type registration algorithms. In Förstner and Ruwiedel, editors, *Robust Computer Vision*, pages 268–282. Wichmann, 1992.

[Aya 87] N. Ayache and O.D. Faugeras. Building, registrating, and fusing noisy visual maps. In *Proceedings of the 1st International Conference on Computer Vision, London, England*, pages 73–82, 1987.

[Bar 94] J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1): 43–77, 1994.

[Bea 78] P.R. Beaudet. Rotationally invariant image operators. In *Proceedings of the 4th International Joint Conference on Pattern Recognition, Tokyo*, pages 579–583, 1978.

[Bea 94] P. Beardsley, A. Zisserman, and D. Murray. Sequential update of projective and affine structure from motion. Technical Report 2012/94, University of Oxford, England, August 1994.

[Bea 95] P.A. Beardsley and A. Zisserman. Affine calibration of mobile vehicles. In R. Mohr and C. Wu, editors, *Europe-China Workshop on Geometrical Modelling and Invariants for Computer Vision, Xian, China*, pages 214–221. Xidan University Press, April 1995.

[Bha 96] D.N. Bhat and S.K. Nayar. Ordinal measure for visual correspondence. In *Proceedings of the Conference on Computer Vision and Pattern Recognition,*

San Francisco, California, USA, pages 351 – 357, San Francisco, California, June 1996.

[Bla 94] T. Blaszka and R. Deriche. Recovering and characterizing image features using an efficient model based approach. Technical Report 2422, INRIA, November 1994.

[Bla 98] J. Blanc. *Synthèse de nouvelles vues d'une scène 3D à partir d'images existantes.* Thèse de doctorat, Institut National Polytechnique de Grenoble, January 1998.

[Boe 94] W. Boehm and H. Prautzsch. *Geometric Concepts for Geometric Design.* A.K. Peters, 1994.

[Bou 93] B. Boufama, R. Mohr, and F. Veillon. Euclidean constraints for uncalibrated reconstruction. In *Proceedings of the 4th International Conference on Computer Vision, Berlin, Germany*, pages 466–470, May 1993.

[Bou 94a] B. Boufama. *Reconstruction tridimensionnelle en vision par ordinateur : cas des caméra non étalonnées.* Thèse de doctorat, Institut National Polytechnique de Grenoble, December 1994.

[Bou 94b] B. Boufama, D. Weinshall, and M. Werman. Shape from motion algorithms: a comparative analysis of scaled orthography and perspective. In J.O. Eklundh, editor, *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 199–204. Springer-Verlag, May 1994.

[Bou 95] B. Boufama and R. Mohr. Epipole and fundamental matrix estimation using the virtual parallax property. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 1030–1036, June 1995.

[Bra 94] P. Brand and R. Mohr. Accuracy in image measure. In S.F. El-Hakim, editor, *Proceedings of the* SPIE *Conference on Videometrics III, Boston, Massachusetts, USA*, volume 2350, pages 218–228, November 1994.

[Bra 95] P. Brand. *Reconstruction tridimensionnelle d'une scène à partir d'une caméra en mouvement : de l'influence de la précision.* Thèse de doctorat, Université Claude Bernard, Lyon I, October 1995.
ftp://ftp.imag.fr/pub/MOVI/theses/brand.ps.gz.

[Bro 86] T.J. Broida and R. Chellappa. Kinematics and structure of a rigid object from a sequence of noisy images. In *Workshop on Motion: Representation and Analysis*, pages 95–100, 1986.

[Bro 91] T. Broida and R. Chellapa. Estimating the kinematics and structure of a rigid object from a sequence of monocular images. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 13(6): 497–513, June 1991.

[Che 91a] H. Chen. Pose determination from line-to-plane correspondences: Existence solutions and closed-form solutions. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 13(6): 530–541, June 1991.

[Che 91b] H. Chen. A screw motion approach to uniqueness analysis of head-eye geometry. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Maui, Hawaii, USA*, pages 145–151, June 1991.

[Chr 94] S. Christy and R. Horaud. Euclidean shape and motion from multiple perspective views by affine iterations. Technical Report 2421, Inria, December 1994.

[Chr 95] S. Christy and R. Horaud. A quasi linear reconstruction method from multiple perspective views. In *Proceedings of the* IEEE/RSJ *International Conference on Intelligent Robots and Systems, Pittsburgh, Pennsylvania, USA*, pages 374–380. IEEE Computer Society Press, August 1995.

[Chr 96a] S. Christy and R. Horaud. Euclidean reconstruction: from paraperspective to perspective. In B. Buxton and R. Cipolla, editors, *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, pages 129–140. Springer-Verlag, April 1996.

[Chr 96b] S. Christy and R. Horaud. Euclidean shape and motion from multiple perspective views by affine iterations. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 18(11): 1098–1104, November 1996.

[Cos 95] J. Costeira and T. Kanade. A multi-body factorization method for motion analysis. In E. Grimson, editor, *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 1071–1076. IEEE, IEEE Computer Society Press, June 1995.

[Cot 94] J.C. Cottier. Extraction et appariements robustes des points d'intérêt de deux images non etalonnées, September 1994.

[Cox 93] I.J. Cox. A review of statistical data association techniques for motion correspondence. *International Journal of Computer Vision*, 10(1): 53–66, 1993.

[Csu 95] G. Csurka, C. Zeller, Z. Zhang, and O. Faugeras. Characterizing the uncertainty of the fundamental matrix. Technical Report 2560, INRIA, June 1995.

[Csu 96] G. Csurka. *Modelisation projective des objets tridimensionnels en vision par ordinateur.* Thèse de doctorat, Université de Nice – Sophia Antipolis, April 1996.

[Csu 98] G. Csurka, D. Demirdjian, A. Ruf, and R. Horaud. Closed-form solutions for the euclidean calibration of a stereo rig. In *Proceedings of the 5th European Conference on Computer Vision, Freiburg, Germany*, 1998. submitted.

[Cui 90] N. Cui, J. Weng, and P. Cohen. Extended structure from motion analysis from monocular image sequences. In *Proceedings of the 3rd International Conference on Computer Vision, Osaka, Japan*, pages 222–229. IEEE Computer Society Press, December 1990.

[Dau 95] N. Daucher, M. Dhome, J.T. Lapresté, and G. Rives. Robust tracking by monocular vision in grey-level images. Technical report, LASMEA, Blaise Pascal University, January 1995.

[Deb 92] C. Debrunner and N. Ahuja. Motion and structure factorization and segmentation of long multiple motion image sequences. In *Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy*, pages 217–221, 1992.

[Dem 92a] D. Dementhon and L.S. Davis. Exact and approximate solutions of the perspective-three-point problem. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 14(11): 1100–1105, November 1992.

[Dem 92b] D. Dementhon and L.S. Davis. Model-based object pose in 25 lines of code. In *Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy*, pages 335–343. Springer-Verlag, 1992.

[Dem 93] D. Dementhon. *De la vision artificielle à la réalité synthétique : système d'intéraction avec un ordinateur utilisant l'analyse d'images vidéo.* Thèse de doctorat, Université Joseph Fourier, Grenoble, October 1993.

[Dem 95] D. Dementhon and L.S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1/2): 123–141, 1995.

[Der 90a] R. Deriche and O. Faugeras. Tracking line segments. In *Proceedings of the 1st European Conference on Computer Vision, Antibes, France*, pages 259–267. Springer-Verlag, April 1990.

[Der 90b] R. Deriche and G. Giraudon. Accurate corner detection: An analytical study. In *Proceedings of the 3rd International Conference on Computer Vision, Osaka, Japan*, 1990.

[Der 93a] R. Deriche and T. Blaszka. Recovering and characterizing image features using an efficient model based approach. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, New York, USA*, pages 530–535, June 1993.

[Der 93b] R. Deriche and G. Giraudon. A computational approach for corner and vertex detection. *International Journal of Computer Vision*, 10(2): 101–124, 1993.

[Dev 95] F. Devernay and O. Faugeras. From projective to euclidean reconstruction. Rapport de Recherche 2725, INRIA, November 1995.

[Dev 96] F. Devernay and O. Faugeras. From projective to euclidean reconstruction. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Francisco, California, USA*, pages 264–269, June 1996.

[Dho 89] M. Dhome, M. Richetin, J.T. Lapresté, and G. Rives. Determination of the attitude of 3D objects from sigle perspective view. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 11(12): 1265–1278, December 1989.

[Dor 95] F. Dornaika. *Contributions à l'intégration vision/robotique : calibrage, localisation et asservissement.* Thèse de doctorat, Institut National Polytechnique de Grenoble, LIFIA–IMAG–INRIA Rhône-Alpes, September 1995.

[Dre 82] L. Dreschler and H.H. Nagel. Volumetric model and 3D trajectory of a moving car derived from monocular tv frame sequences of a street scene. *Computer Graphics and Image Processing*, 20: 199–228, 1982.

[Esp 92] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. IEEE *Transactions on Robotics and Automation*, 8(3): 313–326, June 1992.

[FÖ87] W. Förstner and E. Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Intercommission conference on fast processing of photogrammetric data, Interlaken, Switzerland*, pages 281–305, June 1987.

[Fau 87] O.D. Faugeras and G. Toscani. Camera calibration for 3D computer vision. In *Proceedings of International Workshop on Machine Vision and Machine Intelligence, Tokyo, Japan*, 1987.

[Fau 92a] O. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In G. Sandini, editor, *Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy*, pages 563–578. Springer-Verlag, May 1992.

[Fau 92b] O.D. Faugeras, Q.T. Luong, and S.J. Maybank. Camera self-calibration: Theory and experiments. In G. Sandini, editor, *Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy*, pages 321–334. Springer-Verlag, May 1992.

[Fau 93] O. Faugeras. *Three-Dimensional Computer Vision - A Geometric Viewpoint*. Artificial intelligence. The MIT Press, Cambridge, MA, USA, Cambridge, MA, 1993.

[Fau 95a] O. Faugeras. Stratification of three-dimensional vision: Projective, affine and metric representations. *Journal of the Optical Society of America*, 12: 465–484, 1995.

[Fau 95b] O. Faugeras, S. Laveau, L. Robert, G. Csurka, and C. Zeller. 3D reconstruction of urban scenes from sequences of images. Technical Report 2572, INRIA, June 1995.

[Fau 95c] O. Faugeras, S. Laveau, L. Robert, G. Csurka, and C. Zeller. 3D reconstruction of urban scenes from sequences of images. In A. Gruen, O. Kuebler, and P. Agouris, editors, *Automatic Extraction of Man-Made Objects from Aerial and Space Images*, pages 145–168. Birkhäuser Verlag, 1995.

[Fau 95d] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 951–956, June 1995.

[Fau 95e] O. Faugeras and B. Mourrain. On the geometry and algebra of the point and line correspondences between n images. Technical Report 2665, INRIA, October 1995.

[Fis 81] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Graphics and Image Processing*, 24(6): 381 – 395, June 1981.

[För 87] W. Förstner. Reliability analysis of parameter estimation in linear models with applications to mensuration problems in computer vision. *Computer Vision, Graphics and Image Processing*, 40: 273–310, 1987.

[Fua 91] P. Fua. Combining stereo and monocular information to compute dense depth maps that preserve discontinuities. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence, Sydney, Australia*, August 1991.

[Gol 89] G.H. Golub and C.F. van Loan. *Matrix Computation.* The Johns Hopkins University Press, Baltimore, 1989.

[Gro 93a] P. Gros. Matching and clustering: Two steps towards automatic model generation in computer vision. In *Proceedings of the AAAI Fall Symposium Series: Machine Learning in Computer Vision: What, Why, and How?, Raleigh, North Carolina, USA*, pages 40–44, October 1993.

[Gro 93b] P. Gros. *Outils géométriques pour la modélisation et la reconnaissance d'objets polyédriques.* Thèse de doctorat, Institut National Polytechnique de Grenoble, July 1993.

[Gru 85] A.W. Gruen. Adaptative least squares correlation: a powerful image matching technique. *S. Afr. Journal of Photogrammetry, Remote Sensing and Cartography*, 14(3): 175–187, 1985.

[Har 88] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, pages 147–151, 1988.

[Har 89] R.M. Haralick, H. Joo, C. Lee, X. Zhuang, V.G. Vaidya, and M.B. Kim. Pose estimation from corresponding point data. IEEE *Transactions on Systems, Man and Cybernetics*, 6(19): 1426–1446, November/December 1989.

[Har 92a] C. Harris. Tracking with rigid models. In A. Blake and A. Yuille, editors, *Active Vision.* The MIT Press, 1992.

[Har 92b] R. Hartley, R. Gupta, and T. Chang. Stereo from uncalibrated cameras. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Urbana-Champaign, Illinois, USA*, pages 761–764, 1992.

[Har 92c] R.I. Hartley. Estimation of relative camera positions for uncalibrated cameras. In G. Sandini, editor, *Proceedings of the 2nd European Conference on Computer Vision, Santa Margherita Ligure, Italy*, pages 579–587. Springer-Verlag, 1992.

[Har 93a] R.I. Hartley. Cheirality invariants. In *Proceedings of* DARPA *Image Understanding Workshop*, pages 745–753, 1993.

[Har 93b] R.I. Hartley. Euclidean reconstruction from uncalibrated views. In *Proceeding of the* DARPA–ESPRIT *workshop on Applications of Invariants in Computer Vision, Azores, Portugal*, pages 187–202, October 1993.

[Har 94a] R. Hartley and P. Sturm. Triangulation. In *Proceedings of* ARPA *Image Understanding Workshop, Monterey, California, USA*, pages 957–966, November 1994.

[Har 94b] R.I. Hartley. An algorithm for self calibration from several views. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Seattle, Washington, USA*, pages 908–912, 1994.

[Har 94c] R.I. Hartley. Lines and points in three views - an integrated approach. In *Proceedings of* ARPA *Image Understanding Workshop, Monterey, California, USA*, pages 1009–1016, November 1994.

[Har 94d] R.I. Hartley. Projective reconstruction and invariants from multiple images. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 16(10): 1036–1041, October 1994.

[Har 94e] R.I. Hartley. Projective reconstruction from line correspondences. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Seattle, Washington, USA*, pages 903–907, 1994.

[Har 94f] R.I. Hartley. Self-calibration from multiple views with a rotating camera. In *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 471–478. Springer-Verlag, May 1994.

[Har 95a] R. Hartley. In defence of the 8-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 1064–1070, June 1995.

[Har 95b] R.I. Hartley. A linear method for reconstruction from lines and points. In E. Grimson, editor, *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 882–887. IEEE, IEEE Computer Society Press, June 1995.

[Har 97] R. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2): 146–157, 1997.

[Hei 92] F. Heitger, L. Rosenthaler, R. von der Heydt, E. Peterhans, and O. Kuebler. Simulation of neural contour mechanism: from simple to end-stopped cells. *Vision Research*, 32(5): 963–981, 1992.

[Hor 89] R. Horaud, B. Conio, O. Leboulleux, and B. Lacolle. An analytic solution for the perspective 4-point problem. *Computer Vision, Graphics and Image Processing*, 47: 33–44, 1989.

[Hor 90] R. Horaud, T. Skordas, and F. Veillon. Finding geometric and relational structures in an image. In *Proceedings of the 1st European Conference on Computer Vision, Antibes, France*, Lecture Notes in Computer Science, pages 374–384. Springer-Verlag, April 1990.

[Hor 93] R. Horaud, R. Mohr, and B. Lorecki. On single-scanline camera calibration. IEEE *Transactions on Robotics and Automation*, 1(9): 71–75, 1993.

[Hor 94] R. Horaud, F. Dornaika, B. Boufama, and R. Mohr. Self calibration of a stereo head mounted onto a robot arm. In J.O. Eklundh, editor, *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 455–462. Springer-Verlag, 1994.

[Hor 95a] R. Horaud, S. Christy, F. Dornaika, and B. Lamiroy. Object pose: Links between paraperspective and perspective. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 426–433, Cambridge, Mass., June 1995. IEEE Computer Society Press.

[Hor 95b] R. Horaud and F. Dornaika. Hand-eye calibration. *The International Journal of Robotics Research*, 14(3): 195–210, June 1995.

[Hor 95c] R. Horaud, R. Mohr, F. Dornaika, and B. Boufama. The advantage of mounting a camera onto a robot arm. In *Europe-China Workshop on Geometrical Modelling and Invariants for Computer Vision, Xian, China*, pages 206–213, April 1995.

[Hor 95d] R. Horaud and O. Monga. *Vision par ordinateur: outils fondamentaux.* Éditions Hermès, Paris, 1995. *Deuxième édition revue et augmentée.*

[Hor 97a] R. Horaud, S. Christy, and R. Mohr. Euclidean reconstruction and affine camera calibration using controlled robot motions. In *Proceedings of the* IEEE/RSJ *International Conference on Intelligent Robots and Systems, Grenoble, France*, September 1997.

[Hor 97b] R. Horaud, F. Dornaika, and B. Espiau. Visually guided object grasping. IEEE *Transactions on Robotics and Automation*, 1997.

[Hor 97c] R. Horaud, F. Dornaika, B. Lamiroy, and S. Christy. Object pose: The link between weak perspective, paraperspective and full perspective. *International Journal of Computer Vision*, 22(2): 173–189, March 1997.

[Hor 98] R. Horaud and G. Csurka. Self-calibration and euclidean reconstruction using motions of a stereo rig. In *Proceedings of the 6th International Conference on Computer Vision, Bombay, India*, pages 96–103, January 1998.

[Hu 94] X. Hu and N. Ahuja. Mirror uncertainty and uniqueness conditions for determining shape and motion from orthographic projection. *International Journal of Computer Vision*, 13(3): 295–309, 1994.

[Hua 95] T.S. Huang, A.M. Bruckstein, R.J. Holt, and A.N. Netravali. Uniqueness of 3d pose under weak perspective: A geometrical proof. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 17(12): 1220–1221, December 1995.

[Hut 90] D.P. Huttenlocher and S. Ullman. Recognizing solid objects by algnment with an image. *International Journal of Computer Vision*, 5(2): 195–212, 1990.

[Jac 97a] D. Jacobs. Linear fitting with missing data: Applications to structure-from-motion and to characterizing intensity images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Puerto Rico, USA*, pages 206–212. IEEE Computer Society Press, June 1997.

[Jac 97b] D.W. Jacobs. Matching 3-D models to 2-D images. *International Journal of Computer Vision*, 21(1/2): 123–153, 1997.

[Kal 60] R.E. Kalman. A new approach to linear filtering and prediction problems. *Trans. of ASME Journal of Basic Engineering*, pages 35–45, 1960.

[Kar 94] A. Kara, D.M. Wilkes, and K. Kawamura. 3D structure reconstruction from point correspondences between two perspective projections. *Computer Vision, Graphics and Image Processing: Image Understanding*, 60(3): 392–397, November 1994.

[Kit 82] L. Kitchen and A. Rosenfeld. Gray-level corner detection. *Pattern Recognition Letters*, 1: 95–102, 1982.

[Koe 91] J. Koenderink and A. van Doorn. Affine structure from motion. *Journal of the Optical Society of America A*, 8(2): 377–385, 1991.

[Lan 95] Z.D. Lan and R. Mohr. Robust matching by partial correlation. Technical Report 2643, INRIA, August 1995.

[Lan 97] Z.D. Lan. *Méthodes robustes en vision: application aux appariements visuels.* Thèse de doctorat, Institut National Polytechnique de Grenoble, 1997.

[Lav 96] S. Laveau. *Géométrie d'un système de N caméras. Théorie, estimation, et applications.* Thèse de doctorat, École Polytechnique, May 1996.

[LH 81] H.C. Longuet-Higgins. A computer program for reconstructing a scene from two projections. *Nature*, 293: 133–135, September 1981.

[Liu 90] Y. Liu, T.S. Huang, and O.D. Faugeras. Determination of camera location from 2D to 3D line and point. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 12(1): 28–37, January 1990.

[Low 91] D. Lowe. Fitting parameterized three-dimensional models to images. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 13(5): 441–450, May 1991.

[Luo 92] Q.T. Luong. *Matrice fondamentale et autocalibration en vision par ordinateur.* Thèse de doctorat, Université de Paris-Sud, Orsay, France, December 1992.

[Luo 93] Q.T. Luong and T. Vieville. Canonic representations for the geometries of multiple projective views. Technical report, University of California, Berkeley, EECS, Cory Hall 211-215, University of California, Berkeley, CA 94720, October 1993.

[Luo 94] Q.T. Luong and T. Vieville. Canonic representations for the geometries of multiple projective views. In *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 589–599, May 1994.

[Mat 89] L. Matthies, T. Kanade, and R. Szeliski. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3: 209–236, 1989.

[May 90] S.J. Maybank. Filter based estimates of depth. In *Proceedings of the British Machine Vision Conference, Oxford, England*, pages 349–354, September 1990.

[Med 87] G. Medioni and Y. Yasumoto. Corner detection and curve representation using cubic B-splines. In *Computer Vision, Graphics and Image Processing*, volume 39, pages 267–278. 1987.

[Möb 85] A.F. Möbius. *Gesammelte Werke*, volume 1. Hirzel, Leipzig, 1885.

[Moh 93a] R. Mohr, B. Boufama, and P. Brand. Accurate projective reconstruction. In *Proceeding of the* DARPA–ESPRIT *workshop on Applications of Invariants in Computer Vision, Azores, Portugal*, pages 203–227, October 1993.

[Moh 93b] R. Mohr, B. Boufama, and P. Brand. Accurate projective reconstruction. In J.L. Mundy, A. Zisserman, and D. Forsyth, editors, *Proceeding of the* DARPA–ESPRIT *workshop on Applications of Invariants in Computer Vision, Azores, Portugal*, Lecture Notes in Computer Science, pages 257–276. Springer-Verlag, 1993.

[Moh 93c] R. Mohr, F. Veillon, and L. Quan. Relative 3D reconstruction using multiple uncalibrated images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, New York, USA*, pages 543–548, June 1993.

[Mor 94] T. Morita and T. Kanade. A sequential factorization method for recovering shape and motion from image streams. In *Proceedings of* ARPA *Image Understanding Workshop, Monterey, California, USA*, pages 1177–1188, November 1994.

[Mor 97] T. Morita and T. Kanade. A sequential factorization method for recovering shape and motion from image streams. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 19(8): 858–867, August 1997.

[Nag 83] H.H. Nagel. Displacement vectors derived from second order intensity variations in image sequences. *Computer Vision, Graphics and Image Processing*, 21: 85–117, 1983.

[Nag 87] H.H. Nagel. On the estimation of optical flow : Relations between different approches and some new results. *Artificial Intelligence*, 33: 299–324, 1987.

[Nob 88] J.A. Noble. Finding corners. *Image and Vision Computing*, 6(2): 121–128, May 1988.

[Obe 93] D. Oberkampf, D.F. Dementhon, and L.S. Davis. Iterative pose estimation using coplanar feature points. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, New York, USA*, pages 626–627. IEEE Computer Society Press, June 1993.

[Obe 96] D. Oberkampf, D.F. Dementhon, and L.S. Davis. Iterative pose estimation using coplanar feature points. *Computer Vision, Graphics and Image Processing*, 63(3): 495–511, May 1996.

[Ora 93] C.M. Orange and F.C.A Groen. Model based corner detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, New York, USA*, pages 690–691, June 1993.

[Pho 93] T.Q. Phong, R. Horaud, A. Yassine, and D.T. Pham. Optimal estimation of object pose from a single perspective view. In *Proceedings of the 4th International Conference on Computer Vision, Berlin, Germany*, May 1993.

[Pho 95] T.Q. Phong, R. Horaud, A. Yassine, and P.D. Tao. Object pose from 2D to 3D point and line correspondences. In *International Journal of Computer Vision*, pages 225–243. Kluwer Academic Publishers, July 1995.

[Poe 94] C.J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. In J.O. Eklundh, editor, *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 97–108. Springer-Verlag, May 1994.

[Poe 95] C.J. Poelman. *The Paraperspective and Projective Factorization Methods for Recovering Shape and Motion.* PhD thesis, Carnegie Mellon University, July 1995.

[Pol 89] S.B. Pollard, T.P. Pridmore, J. Porrill, J.E.W. Mayhew, and J.P. Frisby. Geometric modeling from multiple stereo views. *The International Journal of Robotics Research*, 8(4): 3–32, August 1989.

[Pol 98] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proceedings of the 6th International Conference on Computer Vision, Bombay, India*, pages 90–95, January 1998.

[Pre 92] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C - The Art of Scientific Computing.* Cambridge University Press, 2nd edition, 1992.

[Qua 92] L. Quan and R. Mohr. Affine shape representation from motion through reference points. *Journal of Mathematical Imaging and Vision*, 1: 145–151, 1992. also in IEEE Workshop on Visual Motion, New Jersey, pages 249–254, 1991.

[Qua 95a] L. Quan. Invariants of six points and projective reconstruction from three uncalibrated images. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 17(1): 34–46, January 1995.

[Qua 95b] L. Quan and R. Mohr. Self-calibration of an affine camera from multiple views. In *Proceedings of the 6th International Conference on Computer Analysis of Images and Patterns, Prague, Czech Republic*, pages 448–455, September 1995.

[Qua 96a] L. Quan. Conic reconstruction and correspondence from two views. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 18(2): 151–160, February 1996.

[Qua 96b] L. Quan. Self-calibration of an affine camera from multiple views. *International Journal of Computer Vision*, 19(1): 93–105, May 1996.

[Qua 97a] L. Quan. Uncalibrated 1D projective camera and 3D affine reconstruction of lines. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Puerto Rico, USA*, pages 60–65, June 1997.

[Qua 97b] L. Quan and T. Kanade. Affine structure from line correspondences with uncalibrated affine cameras. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 19(8): 834–845, August 1997.

[Qua 97c] L. Quan and R. Mohr. Uniqueness of 3d affine reconstruction of lines with affine cameras. In *Proceedings of the 7th International Conference on Computer Analysis of Images and Patterns, Kiel, Germany*, pages 231–238. Springer-Verlag, 1997.

[Qua 98a] L. Quan and Z.D. Lan. Linear $n \geq 4$-point pose determination. In *Proceedings of the 6th International Conference on Computer Vision, Bombay, India*, January 1998.

[Qua 98b] L. Quan and Y. Ohta. A new linear method for euclidean motion/structure from three calibrated affine views. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Santa Barbara, California, USA*, June 1998.

[Roh 92] K. Rohr. Recognizing corners by fitting parametric models. *International Journal of Computer Vision*, 9(3): 213–230, December 1992.

[Rot 95] C. Rothwell, G. Csurka, and O. Faugeras. A comparison of projective reconstruction methods for pairs of views. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 932–937, June 1995.

[Sch 96a] C. Schmid. *Appariement d'images par invariants locaux de niveaux de gris.* Thèse de doctorat, Institut National Polytechnique de Grenoble, GRAVIR – IMAG – INRIA Rhône–Alpes, July 1996.

[Sch 96b] C. Schmid and R. Mohr. Combining greyvalue invariants with local constraints for object recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Francisco, California, USA*, June 1996. [1]

[Sch 98] C. Schmid, R. Mohr, and Ch. Bauckhage. Comparing and evaluating interest points. In *Proceedings of the 6th International Conference on Computer Vision, Bombay, India.* IEEE Computer Society Press, January 1998.

1. ftp://ftp.imag.fr/pub/MOVI/publications/Schmid_cvpr96.ps.gz.

[Sha 89] T. Shakunaga and H. Kaneko. Perspective angle transform: Principle of shape from angle. *International Journal of Computer Vision*, 3(3): 239–254, September 1989.

[Sha 93] A. Shashua. Projective depth: A geometric invariant for 3D reconstruction from two perspective/orthographic views and for visual recognition. In *Proceedings of the 4th International Conference on Computer Vision, Berlin, Germany*, pages 583–590. IEEE Computer Society Press, May 1993.

[Sha 94] A. Shashua. Trilinearity in visual recognition by alignment. In J.O. Eklundh, editor, *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 479–484. Springer-Verlag, May 1994.

[Sha 95a] L.S. Shapiro, A. Zisserman, and M. Brady. 3D motion recovery via affine epipolar geometry. *International Journal of Computer Vision*, 16(2): 147–182, 1995.

[Sha 95b] A. Shashua and M. Werman. Trilinearity and the 3D-from-2D reconstruction problem. In R. Mohr and C. Wu, editors, *Europe-China Workshop on Geometrical Modelling and Invariants for Computer Vision, Xian, China*, pages 110–117. Xidan University Press, April 1995.

[Shu 94] H.Y. Shum, K. Ikeuchi, and R. Reddy. Principal component analysis with missing data and its application to object modeling. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Seattle, Washington, USA*, pages 560–565. IEEE, IEEE Computer Society Press, June 1994.

[Shu 95] H.Y. Shum, K. Ikeuchi, and R. Reddy. Principal component analysis with missing data and its application to polyhedral object modeling. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 17(9): 854–867, September 1995.

[Spe 92] M.E. Spetsakis and Y. Aloimonos. Optimal visual motion estimation: a note. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 14(9): 959–964, September 1992.

[Stu 95] P. Sturm and L. Quan. Affine stereo calibration. In *Proceedings of the 6th International Conference on Computer Analysis of Images and Patterns, Prague, Czech Republic*, pages 838–843, September 1995.

[Stu 96] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In B. Buxton and R. Cipolla, editors, *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, volume 1065 of *Lecture Notes in Computer Science*, pages 709–720. Springer-Verlag, April 1996.

[Stu 97] P. Sturm. *Vision 3D non calibrée : contributions à la reconstruction projective et étude des mouvements critiques pour l'auto-calibrage.* Thèse de doctorat, Institut National Polytechnique de Grenoble, December 1997.

[Sze 94] R. Szeliski and S.B. Kang. Recovering 3D shape and motion from image streams using nonlinear least squares. *Journal of Visual Communication and Image Representation*, 1(5): 10–28, March 1994.

[Sze 95a] R. Szeliski and S.B. Kang. Direct methods for visual scene reconstruction. In *Workshop on Representation of Visual Scenes, Cambridge, Massachusetts, USA*, pages 26–33, June 1995.

[Sze 95b] R. Szeliski and H.Y. Shum. Motion estimation with quadtree splines. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 757–763. IEEE Computer Society Press, June 1995.

[Tay 91] C.T. Taylor, D.J. Kriegman, and P. Anandan. Structure and motion in two dimensions from multiple images: A least squares approach. In *Proceedings of the* IEEE *Workshop on Visual Motion, Princeton, New Jersey*, pages 242–248. IEEE, IEEE Computer Society Press, October 1991.

[Tay 95] C.J. Taylor and D.J. Kriegman. Structure and motion from line segments in multiple images. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 17(11): 1021–1032, November 1995.

[Tho 66] E.H. Thompson. Space resection: Failure cases. *Photogrammetric Record*, X(27): 201–204, 1966.

[Tom 91a] C. Tomasi. *Shape and Motion from Image Streams: a Factorization Method.* PhD thesis, Carnegie Mellon University, USA, 1991.

[Tom 91b] C. Tomasi and T. Kanade. Factoring image sequences into shape and motion. In *Proceedings of the* IEEE *Workshop on Visual Motion, Princeton, New Jersey*, pages 21–28, Los Alamitos, California, USA, October 1991. IEEE Computer Society Press.

[Tom 92] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2): 137–154, November 1992.

[Tos 87] G. Toscani and O.D. Faugeras. Structure from motion using the reconstruction & reprojection technique. In *Proc. of the* Ieee *Computer Society Workshop on Computer Vision*, pages 345–348. IEEE Computer Society Press, November 1987.

[Tri 95] B. Triggs. Matching constraints and the joint image. In E. Grimson, editor, *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 338–343. IEEE, IEEE Computer Society Press, June 1995.

[Tsa 84] R.Y. Tsai and T.S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 6(1): 13–27, January 1984.

[Tsa 87] R.Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. IEEE *Journal of Robotics and Automation*, 3(4): 323–344, August 1987.

[Uen 96] U. Uenohara and T. Kanade. Geometric invariants for verification in 3-d object tracking. In *Proceedings of the* IEEE/RSJ *International Conference on Intelligent Robots and Systems, Osaka, Japan*, volume II, pages 785–790, November 1996.

[Vi 94] T. Viéville and Q.T. Luong. Computing motion and structure in image sequences without calibration. In *Proceedings of the 12th International Conference on Pattern Recognition, Jerusalem, Israel*, pages 420–425, Jerusalem, October 1994.

[Vi 95a] T. Viéville and O.D. Faugeras. Motion analysis with a camera with unknown, and possibly varying intrinsic parameters. In E. Grimson, editor, *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 750–756. IEEE, IEEE Computer Society Press, June 1995.

[Vi 95b] T. Viéville, C. Zeller, and L. Robert. Using collineations to compute motion and structure in an uncalibrated image sequence. *International Journal of Computer Vision*, 20(3): 213–242, 1995.

[Wal 91] M.W. Walker, L. Shao, and R.A. Volz. Estimating 3D location parameters using dual number quaternions. *Computer Vision, Graphics and Image Processing: Image Understanding*, 54(3): 358–367, November 1991.

[Wei 93] D. Weinshall. Model-based invariants for 3D vision. *International Journal of Computer Vision*, 10(1): 27–42, February 1993.

[Wei 95] D. Weinshall and C. Tomasi. Linear and incremental acquisition of invariant shape models from image sequences. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 17(5): 512–517, May 1995.

[Wen 88] J. Weng, N. Ahuja, and T.S. Huang. Closed-form solution + maximum likelihood: A robust approach to motion and structure estimation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, San Diego, California, USA*, pages 381–386, June 1988.

[Wen 93] J. Weng, N. Ahuja, and T.S. Huang. Optimal motion and structure estimation. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 15(9): 864–884, September 1993.

[Wer 95] M. Werman and A. Shashua. Elimination: An approach to the study of 3D-from-2D. In E. Grimson, editor, *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 473–479. IEEE Computer Society Press, June 1995.

[Wib 76] T. Wiberg. Computation of principal components when data are missing. In J. Gordesch and P. Naeve, editors, *Proc. 2nd Symposium on Computational Statistics, Berlin, Germany*, pages 229–236, 1976.

[Wil 95] C.S. Wiles and M. Brady. Closing the loop on multiple motions. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 308–313. IEEE Computer Society Press, June 1995.

[Wil 96] C. Wiles and M. Brady. On the appropriateness of camera models. In B. Buxton and R. Cipolla, editors, *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, volume 1065 of *Lecture Notes in Computer Science*, pages 228–237. Springer-Verlag, April 1996.

[Wro 92] B.P. Wrobel. Minimum solutions for orientation. In *Proc. of the Workshop on Calibration and Orientation of Cameras in Computer Vision, Washington D.C., USA*. Springer-Verlag, August 1992.

[Yua 89] J.S.C. Yuan. A general phogrammetric solution for the determining object position and orientation. IEEE *Transactions on Robotics and Automation*, 5(2): 129–142, April 1989.

[Zab 94] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondance. In *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 151–158. Springer-Verlag, May 1994.

[Zel 96a] C. Zeller. *Calibration projective, affine et euclidienne en vision par ordinateur et application à la perception tridimensionnelle.* Thèse de doctorat, École Polytechnique, February 1996.

[Zel 96b] C. Zeller and O. Faugeras. Camera self-calibration from video sequences: the kruppa equations revisited. Rapport de recherche 2793, INRIA, February 1996.

[Zha 90a] Z. Zhang. *Analyse du Mouvement d'Une séquence stéréoscopique et ses Applications.* Thèse de doctorat, Université de Paris XI, France, October 1990.

[Zha 90b] Z. Zhang and O. Faugeras. Building a 3D world representation with a mobile robot: 3D line segment representation and integration. In *Proceedings of the 10th International Conference on Pattern Recognition, Atlantic City, New Jersey, USA*, pages 38–42, June 1990.

[Zha 92] Z. Zhang and O. Faugeras. Estimation of displacements from two 3-D frames obtained from stereo. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 14(12): 1141–1156, December 1992.

[Zha 94a] Z. Zhang. Estimating motion and structure from correspondences of line segments between two perspective images. Technical Report 2340, INRIA, September 1994.

[Zha 94b] Z. Zhang. Token tracking in a cluttered scene. *Image and Vision Computing*, 12(2): 110–120, March 1994.

[Zha 94c] Z. Zhang, R. Deriche, O. Faugeras, and Q.T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. Rapport de recherche 2273, INRIA, May 1994.

[Zha 94d] Z. Zhang, Q.T. Luong, and O. Faugeras. Self-calibration of an uncalibrated stereo rig from one unknown motion. In E. Hancock, editor, *Proceedings of the fifth British Machine Vision Conference, York, England*, pages 499–508. British Machine Vision Association, September 1994.

[Zha 95a] Z. Zhang. Estimating motion and stucture from correspondences of line segments between two perspective images. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 257–262. IEEE Computer Society Press, June 1995.

[Zha 95b] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. Technical Report 2676, INRIA, October 1995.

[Zha 95c] Z. Zhang, R. Deriche, O. Faugeras, and Q.T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78: 87–119, 1995.

[Zhu 86] X. Zhuang, T.S. Huang, and R.M. Haralick. Two-view motion analysis: A unified algorithm. *Journal of the Optical Society of America A*, 3(9): 1492–1500, September 1986.

[Zis 95] A. Zisserman, P.A. Beardsley, and I.D. Reid. Metric calibration of a stereo rig. In *Workshop on Representation of Visual Scenes, Cambridge, Massachusetts, USA*, pages 93–100, June 1995.

Localisation et modélisation tridimensionnelles par approximations successives du modèle perspectif de caméra

Résumé

Dans le cadre de cette thèse, nous proposons un algorithme générique permettant de résoudre le problème de calcul de pose et le problème de reconstruction avec un modèle perspectif de caméra.

Étant donnés une image et un modèle 3D de la scène (ou d'un objet) visible dans l'image, le calcul de pose consiste à calculer la position et l'orientation de la caméra par rapport à la scène. Nous étudions successivement le cas de correspondances 2D/3D de points, et le cas de droites. La méthode proposée améliore de manière itérative la pose calculée avec un modèle affine de caméra (orthographique à l'échelle ou paraperspectif) pour converger, à la limite, vers une estimation de la pose calculée avec un modèle perspectif de caméra.

Dans un second temps, nous étendons les algorithmes de calcul de pose précédents au problème de la reconstruction euclidienne avec un modèle perspectif de caméra, à partir d'une séquence d'images. La méthode proposée converge en quelques itérations, est efficace du point de vue calculatoire, et ne souffre pas de la nature non linéaire du problème traité. Nous présentons ensuite une seconde approche du problème de reconstruction euclidienne en considérant un modèle affine de caméra non étalonnée montée sur le bras d'un robot.

Afin de pouvoir utiliser en pratique ces algorithmes de reconstruction, nous présentons une méthode de poursuite de points caractéristiques sur une séquence monoculaire d'images, puis sur une séquence stéréoscopique. Nous proposons également une méthode pour obtenir une précision sous-pixellique des positions des points dans les images.

Mots clefs : vision par ordinateur, reconstruction tridimensionnelle (euclidienne et affine), calcul de pose, mise en correspondance, corrélation, modèle perspectif de caméra, modèle affine de caméra, modèle orthographique à l'échelle, modèle paraperspectif, étalonnage d'une caméra.

Three-dimensional localization et modeling by successive approximations of the perspective camera model

Abstract

In this report, we propose a generic algorithm to compute objet pose and reconstruction, with a perspective camera model.

Given one image and a 3D model of the scene, object pose consists in recovering the position and the orientation of the camera with respect to the camera. We successively study the case of 2D to 3D point correspondences, and the case of line correspondences. The method consists in iteratively improving the pose computed with an affine camera model (weak perspective or paraperspective) to converge, at the limit, to the pose estimation computed with a perspective camera model.

In a second time, we extend the previous object pose algorithms for Euclidean reconstruction from a sequences of images, by using a perspective camera model. The proposed method converges in a few iterations, is computationally efficient, and does not suffer from the non linear nature of the problem. Then, we present a second approach for recovering Euclidean reconstruction, with an uncalibrated affine camera mounted onto a robot arm.

In order to use these algorithms for reconstruction from a practical point of view, we present a method to do the tracking of characteristic points along a sequence of images. Moreover, we also present a method to obtain a subpixel accuracy of the image point coordinates for a low computation cost.

Keywords : computer vision, 3D reconstruction (Euclidean and affine), object pose, tracking, correlation, perspective camera model, affine camera model, weak perspective model, paraperspective model, camera calibration.

Thèse préparée au sein du laboratoire GRAVIR–IMAG dans l'équipe MOVI.
INRIA Rhône-Alpes, 655, Av. de l'Europe, ZIRST, 38330 Montbonnot Saint Martin.