THÈSE

préparée à

L'INRIA Sophia Antipolis

et présentée à

L'ÉCOLE NATIONALE SUPÉRIEURE DES MINES DE PARIS

pour obtenir le grade de

DOCTEUR EN SCIENCES

Spécialité

Informatique Temps Réel, Automatique et Robotique

soutenue publiquement par

Mohamed Selim BEN HIMANE

Sujet de la thèse :

Vers une approche unifiée pour le suivi temps-réel et l'asservissement visuel

le 06 Décembre 2006 devant le jury composé de :

M.	**Yves**	**ROUCHALEAU**	Président
MM.	**François**	**CHAUMETTE**	Rapporteurs
	Michel	**DHOME**	
MM.	**Nassir**	**NAVAB**	Examinateurs
	Ezio	**MALIS**	
	Patrick	**RIVES**	

À ma maman Raoudha, à mon papa Béchir.

Table des matières

Table des figures

Première partie

Introduction, notations et modélisations

Chapitre 1

Introduction

L'accomplissement d'une tâche robotique référencée vision (telle que le suivi, le déplacement ou la manipulation) est basée sur deux types d'informations sensorielles différents. D'une part, il y a une information sur la situation du système asservi (bras manipulateur, robot mobile, etc...) par rapport à son environnement qui est donnée par des capteurs extéroceptifs, c'est-à-dire des capteurs visuels dans notre cas. D'autre part, il y a une information sur l'état interne du système qui est fournie par des capteurs proprioceptifs. Entre deux phases d'acquisition de données extéroceptives, le système robotique évolue en boucle ouverte sur les données visuelles. Par conséquent, la commande ne peut être bouclée que sur l'état interne du système, et elle ne repose pas sur la connaissance précise de la situation du robot.

En utilisant des capteurs visuel plus performants, tels que les caméras, il est possible de spécifier la tâche du robot en terme de commande directement dans le repère de la caméra. Plus particulièrement, utiliser des informations délivrées par des caméras présente l'intérêt de pouvoir positionner plus librement et plus facilement le robot tout en autorisant la localisation ou la reconnaissance d'objets. Cette approche, appelée "commande référencée vision" ou "asservissement visuel", a été initiée il y a un peu plus d'une vingtaine d'années. Au cours des dix dernières années, la commande référencée vision est devenue une technique bien établie pour intégrer les données visuelles acquises par des caméras dans la boucle de commande.

La conjonction entre, d'une part, la disponibilité de systèmes d'acquisition et de traitement d'images rapides et abordables et, d'autre part, le développement théorique dans le domaine de la vision par ordinateur et dans la théorie de la commande a abouti à un nombre d'applications très important. Mis à part les domaines traditionnels de la manipulation et de la saisie robotisées, la commande référencée vision offre aujourd'hui des applications dans des contextes très variés qui englobent non seulement la vision par ordinateur mais aussi l'automatique et la théorie de la commande. Parmi ces applications nous trouvons : la conduite automatique, l'exploration sur de longues distances, l'observation et la surveillance par des robots aériens, la robotique médicale...

La réalisation de ce genre d'applications complexes nécessite l'intégration de divers domaines de recherche en vision par ordinateur et en théorie de la commande : la mise en correspondance entre deux ou plusieurs images, le suivi visuel temps réel et l'asservissement visuel. Afin de concevoir des méthodes de commande référencée vision, il est possible d'utiliser des techniques de vision et de commande qui ont été conçues séparément. Avec de telles approches, l'intégration "vision-commande" peut s'avérer très difficile vu le grand nombre de méthodes de suivi visuel et de commande pour lesquels la compatibilité n'est pas garantie.

Au lieu de considérer séparément les techniques de vision et de commande, dans cette thèse nous essaierons de les intégrer dans une approche unifiée. Notre objectif est de concevoir un système générique, flexible et robuste qui peut être utilisé pour une grande variété d'applications robotiques. Au cours de cette thèse, nous avons contribué dans divers domaines allant dans le sens de la conception d'un système complet :

1. Dans (Benhimane and Malis, 2004a), nous avons proposé une méthode de mise en correspondance d'images à différentes résolutions à l'aide d'invariants aux paramètres intrinsèques.
2. Dans (Benhimane and Malis, 2004c), nous avons proposé une méthode d'auto-calibration des paramètres de distorsion d'une caméra munie d'un zoom motorisé en utilisant une méthode itérative de mise en correspondance d'images à différentes résolutions.
3. Les invariants des deux dernières méthodes ont été utilisés dans (Benhimane and Malis, 2003) pour concevoir une commande d'un robot muni d'une caméra avec zoom motorisé par rapport à des objets plans et non plans.
4. Dans (Benhimane and Malis, 2004b), nous avons proposé une approche de suivi visuel d'objets plans grâce à leurs textures dans l'image en utilisant une minimisation efficace au second-ordre appelé la méthode ESM. Le suivi visuel obtenu a des propriétés meilleures que les méthodes de suivi visuel existantes. En effet, le domaine, le taux et la fréquence de convergence sont plus importants que dans les méthodes existantes.
5. Ce suivi visuel a été couplé avec la commande par asservissement visuel 2D 1/2 dans (Malis and Benhimane, 2004) et (Malis and Benhimane, 2005) pour une ébauche de l'intégration "vision-commande".
6. Dans (Benhimane et al., 2005), le suivi visuel ESM a été utilisé pour effectuer un application d'accrochage immatériel entre deux véhicules pour assurer une mission de convoi de véhicules.
7. La minimisation au second ordre a été utilisée dans (Benhimane and Malis, 2006b) pour intégrer des contraintes Euclidiennes dans le suivi visuel d'objets planaires par morceaux.
8. Cette même approche a été généralisée dans (Mei et al., 2006a; Mei et al., 2006b) pour des caméras omnidirectionnelles.

9. Une nouvelle commande par asservissement visuel 2D a été introduite dans (Benhimane and Malis, 2006d; Benhimane and Malis, 2006a). Cette commande est stable localement et, contrairement à toutes les méthodes existantes, elle ne nécessite pas une connaissance sur les paramètres du modèle de l'objet par rapport auquel la commande par asservissement visuel est effectuée.
10. Cette commande a été généralisée dans (Benhimane and Malis, 2006c) pour la commande de robots munis de caméras omnidirectionnelles.
11. Finalement, le suivi visuel ESM couplé avec la commande par asservissement visuel 2D direct a été publié dans (Benhimane and Malis, 2007).

Ces contributions vont dans le même sens : celui de la conception d'un système de commande par vision le plus général possible. Cependant, pour des raisons de clarté, dans ce manuscrit, nous avons choisi de décrire en détail les contributions 4, 5, 6, 7, 9 et 11 puisqu'elles constituent un ensemble cohérent. Pour avoir plus de détails concernant les autres contributions le lecteur peut se référer aux publications correspondantes.

La suite de cette partie introduit quelques conventions mathématiques, quelques notions de géométrie projective nécessaires à la compréhension de la suite du manuscrit. Nous effectuons également la modélisation de la caméra et des images acquises.

La deuxième partie s'intéresse au suivi visuel temps-réel. Après un bref état de l'art, nous présentons le suivi visuel dans l'image de surfaces planaires avec la méthode ESM. Ensuite, nous décrivons le suivi visuel dans l'espace Cartésien en utilisant cette même technique d'optimisation. Ces deux chapitres suivront la même logique : étude théorique, simulations et expériences.

La troisième partie concerne l'asservissement visuel. Après avoir effectué une classification des différentes méthodes existantes de la commande par asservissement visuel, nous présentons notre nouvelle approche. Le reste de cette partie est consacré à quelques applications développées au cours de cette thèse et faisant appel au suivi et à l'asservissement visuels. Ces applications sont des systèmes complets où la vision et la commande interagissent d'une manière unifiée.

La quatrième et dernière partie conclut ce manuscrit et présente les perspectives de recherche qui peuvent être envisagées à la suite de ce travail de thèse.

Chapitre 2

Notations et modélisations

2.1 Quelques conventions mathématiques

2.1.1 Généralisation de l'opérateur gradient aux matrices

Nous souhaitons généraliser l'opérateur gradient dans le cas des matrices. Soit $\mathbf{A}$ une matrice de dimensions $(r_a \times c_a)$. Soit $\mathbf{a}$ le vecteur de dimensions $(r_a c_a \times 1)$ obtenu en réarrangeant les éléments de la matrice $\mathbf{A}$ ligne par ligne de la manière suivante :

$$\mathbf{a} = [a_{11}, ..., a_{1c_a}, a_{21}, ..., a_{2c_a}, ..., a_{r_a 1}, ..., a_{r_a c_a}]^\top$$

Nous utiliserons la notation $[.]_v$ qui correspond à réarranger les éléments de la matrice $\mathbf{A}$ ligne par ligne en un vecteur. Par conséquent, le vecteur $\mathbf{a}$ peut être écrit :

$$\mathbf{a} = [\mathbf{A}]_v \tag{2.1}$$

Soit $\mathbf{B}$ une matrice de dimensions $(r_b \times c_b)$. Soit $\mathbf{b}$ le vecteur de dimensions $(r_b c_b \times 1)$ obtenu en réarrangeant les éléments de la matrice $\mathbf{B}$ comme suit :

$$\mathbf{b} = [\mathbf{B}]_v$$

Nous généralisons l'opérateur gradient ∇ au cas des matrices. Nous écrivons $\nabla_{\mathbf{B}}(\mathbf{A})$ de la manière suivante :

$$\nabla_{\mathbf{B}}(\mathbf{A}) = \nabla_{\mathbf{b}}(\mathbf{a})$$

où la matrice $\nabla_{\mathbf{B}}(\mathbf{A})$ est de dimensions $(r_a c_a \times r_b c_b)$. Évidemment, si $\mathbf{A}$ et $\mathbf{B}$ sont des vecteurs ou des scalaires cette définition du gradient coïncide avec la définition standard.

2.1.2 Définitions du Jacobien et des Hessiennes

Le gradient d'une fonction vecteur $\mathbf{a}$ de dimensions $(n_a \times 1)$ qui dépend de la variable vecteur $\mathbf{b}$ de dimensions $(n_b \times 1)$ est la matrice Jacobienne $\mathbf{J}(\mathbf{b})$. Il s'agit d'une matrice qui représente la variation du vecteur $\mathbf{a}(\mathbf{b})$ en fonction des paramètres $\mathbf{b}$. Étant la dérivée première du vecteur $\mathbf{a}(\mathbf{b})$ par rapport à $\mathbf{b}$, cette matrice s'écrit sous la forme suivante :

$$\mathbf{J}(\mathbf{b}) = \nabla_{\mathbf{b}}\mathbf{a}(\mathbf{b}) = \begin{bmatrix} \frac{\partial a_1(\mathbf{b})}{\partial b_1} & \frac{\partial a_1(\mathbf{b})}{\partial b_2} & \dots & \frac{\partial a_1(\mathbf{b})}{\partial b_{n_b}} \\ \frac{\partial a_2(\mathbf{b})}{\partial b_1} & \frac{\partial a_2(\mathbf{b})}{\partial b_2} & \dots & \frac{\partial a_2(\mathbf{b})}{\partial b_{n_b}} \\ \dots & \dots & \dots & \dots \\ \frac{\partial a_{n_a}(\mathbf{b})}{\partial b_1} & \frac{\partial a_{n_a}(\mathbf{b})}{\partial b_2} & \dots & \frac{\partial a_{n_a}(\mathbf{b})}{\partial b_{n_b}} \end{bmatrix} \tag{2.2}$$

Nous définissons la matrice $\mathbf{M}(\mathbf{b}_1, \mathbf{b}_2)$ de dimensions $(n_a \times n_b)$ qui s'écrit $\forall(\mathbf{b}_1, \mathbf{b}_2) \in \mathbb{R}^{n_b} \times \mathbb{R}^{n_b}$:

$$\mathbf{M}(\mathbf{b}_1, \mathbf{b}_2) = \nabla_{\mathbf{b}_1}\left(\mathbf{J}(\mathbf{b}_1)\mathbf{b}_2\right) = \begin{bmatrix} \frac{\partial^2 a_1(\mathbf{b}_1)}{\partial \mathbf{b}_1^2}\mathbf{b}_2 & \frac{\partial^2 a_2(\mathbf{b}_1)}{\partial \mathbf{b}_1^2}\mathbf{b}_2 & \dots & \frac{\partial^2 a_{n_a}(\mathbf{b}_1)}{\partial \mathbf{b}_1^2}\mathbf{b}_2 \end{bmatrix}^\top \tag{2.3}$$

où chaque matrice Hessienne $\frac{\partial^2 a_i(\mathbf{b})}{\partial \mathbf{b}^2}$ est de dimensions $(n_b \times n_b)$ et s'écrit :

$$\frac{\partial^2 a_i(\mathbf{b})}{\partial \mathbf{b}^2} = \begin{bmatrix} \frac{\partial^2 a_i(\mathbf{b})}{\partial b_1 \partial b_1} & \frac{\partial^2 a_i(\mathbf{b})}{\partial b_1 \partial b_2} & \dots & \frac{\partial^2 a_i(\mathbf{b})}{\partial b_1 \partial b_{n_b}} \\ \frac{\partial^2 a_i(\mathbf{b})}{\partial b_1 \partial b_2} & \frac{\partial^2 a_i(\mathbf{b})}{\partial b_1 \partial b_2} & \dots & \frac{\partial^2 a_i(\mathbf{b})}{\partial b_2 \partial b_{n_b}} \\ \dots & \dots & \dots & \dots \\ \frac{\partial^2 a_i(\mathbf{b})}{\partial b_1 \partial b_{n_b}} & \frac{\partial^2 a_i(\mathbf{b})}{\partial b_2 \partial b_{n_b}} & \dots & \frac{\partial^2 a_i(\mathbf{b})}{\partial b_{n_b} \partial b_{n_b}} \end{bmatrix} \tag{2.4}$$

2.2 Quelques notions de géométrie projective

2.2.1 Changement de repère et projection perspective

Nous considérons l'espace 3D comme étant Cartésien, c'est-à-dire, un espace Euclidien à trois dimensions muni d'un repère orthonormé. Soit $\mathcal{P}$ un point de l'espace 3D et soit $\mathcal{F}^* = (\mathcal{O}^*, \vec{i}^*, \vec{j}^*, \vec{k}^*)$ le "repère de référence" de l'espace Cartésien. Dans le repère $\mathcal{F}^*$, le point $\mathcal{P}$ a les coordonnées $\boldsymbol{\mathcal{X}}^* = [X^* \ Y^* \ Z^*]^\top \in \mathbb{R}^3$. Soient $\mathbf{R}$ la matrice de rotation et $\mathbf{t}$ le vecteur la translation entre le repère de référence $\mathcal{F}^*$ et un deuxième repère noté $\mathcal{F} = (\mathcal{O}, \vec{i}, \vec{j}, \vec{k})$ et que nous appellerons "le repère courant". La matrice de rotation $\mathbf{R}$ est de dimensions (3×3) et appartient au groupe Spécial Orthogonal de dimension 3 noté $\mathbb{SO}(3)$. Le vecteur $\mathbf{t}$ appartient à $\mathbb{R}^3$. Dans le repère $\mathcal{F}$, le point $\mathcal{P}$ a pour coordonnées $\boldsymbol{\mathcal{X}} = [X \ Y \ Z]^\top \in \mathbb{R}^3$ vérifiant :

$$\boldsymbol{\mathcal{X}} = \mathbf{R}\boldsymbol{\mathcal{X}}^* + \mathbf{t} \tag{2.5}$$

Le déplacement entre le repère de référence $\mathcal{F}^*$ et le repère courant $\mathcal{F}$ peut être représenté à l'aide d'une "matrice de transformation homogène", notée $\mathbf{T}$, de dimensions (4×4) appartenant au groupe Spécial Euclidien $\mathbb{SE}(3)$:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{2.6}$$

Ici et dans le reste du manuscrit, le symbole $\mathbf{0}$ (zéro en gras) correspond à une matrice nulle de dimensions appropriées. Ici, il s'agit d'une matrice nulle de dimensions (1×3).

Si nous utilisons les coordonnées homogènes $\widetilde{\boldsymbol{\mathcal{X}}}^* = \begin{bmatrix} \boldsymbol{\mathcal{X}}^{*\top} & 1 \end{bmatrix}^\top$ et $\widetilde{\boldsymbol{\mathcal{X}}} = \begin{bmatrix} \boldsymbol{\mathcal{X}}^{\top} & 1 \end{bmatrix}^\top$, l'équation (2.5) se réécrit à l'aide de la matrice de transformation $\mathbf{T}$ de la manière suivante :

$$\widetilde{\boldsymbol{\mathcal{X}}} = \mathbf{T}\widetilde{\boldsymbol{\mathcal{X}}}^* \tag{2.7}$$

Nous appelons "image métrique de référence", noté $\mathcal{I}_m^*$, le plan perpendiculaire à l'axe $\mathbf{k}^*$ et situé à un mètre du centre de projection $\mathcal{O}^*$. Le point $\mathcal{P}$ se projette suivant le modèle perspectif sur le plan $\mathcal{I}_m^*$ en un point de l'espace projectif $\mathbf{m}^* \in \mathbb{P}^2$ de coordonnées métriques homogènes $\mathbf{m}^* = [x^*\ y^*\ 1]^\top$ où :

$$\mathbf{m}^* = Z^{*-1}\boldsymbol{\mathcal{X}}^* \tag{2.8}$$

D'une manière analogue, nous appellerons "image métrique courante", noté $\mathcal{I}_m$, le plan perpendiculaire à l'axe $\mathbf{k}$ et situé à un mètre du centre de projection $\mathcal{O}$. Le point $\mathcal{P}$ se projette suivant le modèle perspectif sur le plan $\mathcal{I}_m$ en un point de l'espace projectif $\mathbf{m} \in \mathbb{P}^2$ de coordonnées métriques homogènes $\mathbf{m} = [x\ y\ 1]^\top$ où :

$$\mathbf{m} = Z^{-1}\boldsymbol{\mathcal{X}} \tag{2.9}$$

Nous appelons "point principal", le point $\mathbf{m}_0^*$ de coordonnées métriques homogènes vérifiant :

$$\mathbf{m}_0^* = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\top \tag{2.10}$$

Les points $\mathbf{m}^*$ et $\mathbf{m}$ sont deux "points correspondants" car ils sont la projection d'un même point $\mathcal{P}$.

2.2.2 Transformation projective planaire

Supposons que le point $\mathcal{P}$ appartienne à un plan π. Soit $\mathbf{n}^*$ le vecteur normal au plan π exprimé dans le repère $\mathcal{F}^*$ et d^* la distance entre le plan π et le centre de projection $\mathcal{O}^*$. Si nous choisissons la norme du vecteur $\mathbf{n}^*$ vérifiant :

$$\|\mathbf{n}^*\| = \sqrt{\mathbf{n}^{*\top}\mathbf{n}^*} = d^{*-1} \tag{2.11}$$

l'équation du plan π peut être écrite comme :

$$\mathbf{n}^{*\top}\boldsymbol{\mathcal{X}}^* = 1 \tag{2.12}$$

En utilisant les équations (2.5), (2.8), (2.9) et (2.12), nous pouvons établir une relation entre les points $\mathbf{m}$ et $\mathbf{m}^*$:

$$ZZ^{*-1}\mathbf{m} = \mathbf{H}_n\mathbf{m}^* \tag{2.13}$$

où la matrice $\mathbf{H}_n$ s'écrit :

$$\mathbf{H}_n = \mathbf{R} + \mathbf{t}\mathbf{n}^{*\top} \tag{2.14}$$

Connaissant $\mathbf{m}^*$ et une matrice $\mathbf{H}$ proportionnelle à $\mathbf{H}_n$ vérifiant :

$$\mathbf{H} = \alpha\mathbf{H}_n \tag{2.15}$$

où $\alpha \in \mathbb{R}^*$, il est possible de calculer $\mathbf{m}$ sans connaître le rapport ZZ^{*-1}. En effet, il suffit de multiplier $\mathbf{m}^*$ par $\mathbf{H}$ puis normaliser la troisième composante de $\mathbf{m}$ à 1. L'équation (2.13) s'écrit alors :

$$\mathbf{m} \;\propto\; \mathbf{H}\mathbf{m}^* \tag{2.16}$$

Si nous notons par $\mathbf{h}_1$, $\mathbf{h}_2$ et $\mathbf{h}_3$ les vecteurs correspondant aux lignes de la matrice $\mathbf{H}$, nous pouvons écrire :

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3 \end{bmatrix}^\top \tag{2.17}$$

La normalisation par la troisième composante du point $\mathbf{m}$ se fait de la manière suivante :

$$\mathbf{m} = \begin{bmatrix} x & y & 1 \end{bmatrix}^\top = \begin{bmatrix} \dfrac{\mathbf{h}_1^\top\mathbf{m}^*}{\mathbf{h}_3^\top\mathbf{m}^*} & \dfrac{\mathbf{h}_2^\top\mathbf{m}^*}{\mathbf{h}_3^\top\mathbf{m}^*} & 1 \end{bmatrix}^\top \tag{2.18}$$

La matrice $\mathbf{H}$ est appelée "matrice d'homographie". Cette matrice permet de mettre en correspondance les points du plan π entre les deux images métriques $\mathcal{I}_m^*$ et $\mathcal{I}_m$ (voir la Figure 2.1).

Étant donné un ensemble de N couples de points non colinéaires mis en correspondance $(\mathbf{m}_i^*, \mathbf{m}_i)$ où $i \in \{1, 2, ..., N\}$ et où $N \geq 4$, il est possible de calculer la matrice d'homographie $\mathbf{H}$ (voir (Faugeras, 1993) et (Hartley and Zisserman, 2000) pour plus de détails). Cependant, la matrice $\mathbf{H}$ est généralement calculée à un facteur d'échelle près. En effet, la matrice $\mathbf{H}$ a 9 éléments (car c'est une matrice de dimension (3×3)) mais elle a seulement 8 degrés de liberté : 3 pour la rotation, 3 pour la translation et 2 pour le vecteur normal au plan. Le facteur d'échelle de la matrice $\mathbf{H}$ peut être choisi de plusieurs manières. Nous choisissons de fixer le facteur d'échelle de façon à avoir le déterminant de $\mathbf{H}$ égal à 1. Nous avons donc $\mathbf{H}$ appartient au groupe Spécial Linéaire de dimension 3 noté $\mathbb{SL}(3)$. Ce choix se justifie par le fait que $\det(\mathbf{H}) \leq 0$ n'arrive que si le centre de projection $\mathcal{O}$ traverse le plan π (nous avons $\det(\mathbf{H}) = 0$ quand le centre du repère

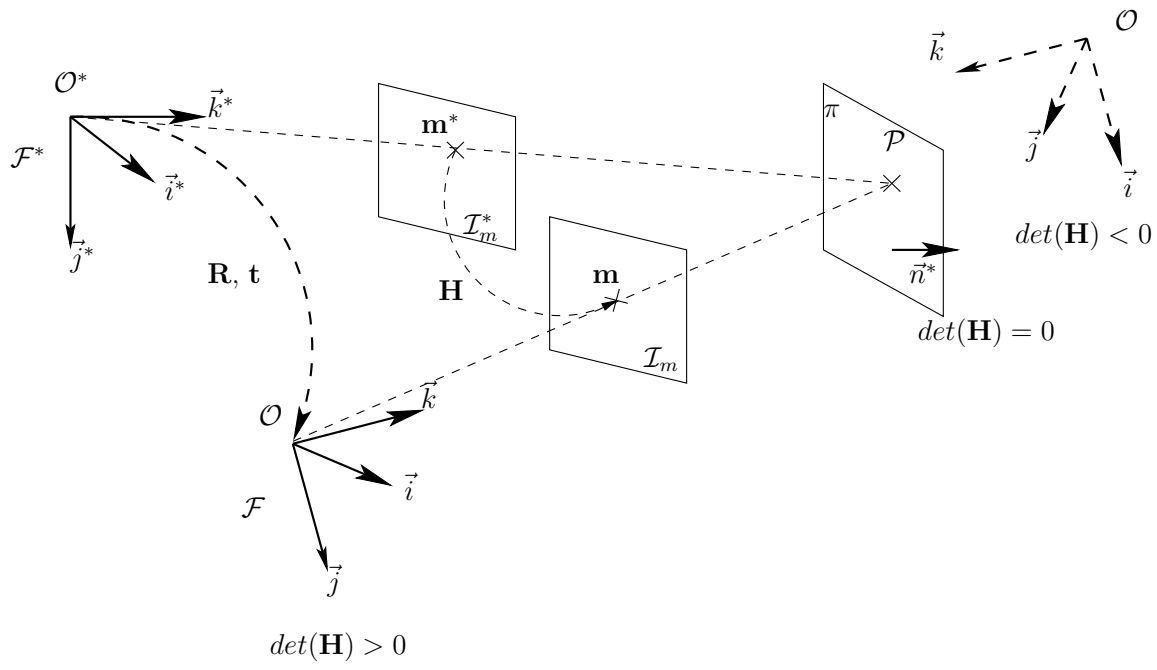


Figure 2.1 – Transformation projective planaire et signe du déterminant de l'homographie

$\mathcal{F}$ est sur le plan π), or nous supposons que les centres de projection $\mathcal{O}^*$ et $\mathcal{O}$ restent toujours du même côté du plan π (voir la Figure 2.1).

2.3 Modélisation de la caméra et des images

2.3.1 Matrice des paramètres intrinsèques

Dans le modèle sténopé (Faugeras, 1993; Horaud and Monga, 1993), la caméra réalise une projection perspective. Si le repère de la caméra coïncide avec le repère de référence $\mathcal{F}^*$, l'image obtenue par la caméra sera appelée "image de référence", notée $\mathcal{I}^*$, et nous dirons que la caméra est à "la position de référence". Si le repère de la caméra coïncide avec le repère courant $\mathcal{F}$, l'image obtenue par la caméra sera appelée "image courante", notée $\mathcal{I}$, et nous dirons que la caméra est à "la position courante". Dans l'image de référence $\mathcal{I}^*$, le point $\mathcal{P}$ se projette en un point $\mathbf{p}^* \in \mathbb{P}^2$ de coordonnées $\mathbf{p}^* = [u^*\ v^*\ 1]^\top$ vérifiant :

$$\mathbf{p}^* = \mathbf{K}\mathbf{m}^* \tag{2.19}$$

et dans l'image courante $\mathcal{I}$, le point $\mathcal{P}$ se projette en un point $\mathbf{p} \in \mathbb{P}^2$ de coordonnées $\mathbf{p} = [u\ v\ 1]^\top$ vérifiant :

$$\mathbf{p} = \mathbf{K}\mathbf{m} \tag{2.20}$$

La matrice $\mathbf{K}$ est une matrice triangulaire supérieure de dimensions (3×3) dépendant des paramètres intrinsèques de la caméra :

$$\mathbf{K} = \begin{bmatrix} f & fs & u_0 \\ 0 & fr & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.21}$$

où f est la focale exprimée en pixels, s modélise l'effet de cisaillement dû au défaut d'orthogonalité des axes du repère image et r est le rapport des dimensions d'un pixel, $[u_0\ v_0\ 1]$ sont les coordonnées du point principal en pixels.

2.3.2 Modèle de l'image

Une image $\mathcal{I}^*$ de dimensions $(n \times m)$ peut être considérée comme une matrice de dimensions $(n \times m)$ contenant les intensités des pixels. L'élément $\mathcal{I}^*(u^*, v^*)$ de la matrice correspond à la valeur d'intensité du pixel de la ligne u^* et à la colonne v^*. Nous supposons qu'il existe une fonction I^* régulière vérifiant :

$$\begin{array}{rccc} I^* : & \mathbb{P}^2 & \rightarrow & \mathbb{R} \\ & \mathbf{p}^* = [u^*\ v^*\ 1]^\top & \mapsto & I^*(u^*, v^*) \end{array} \tag{2.22}$$

Cette fonction associe à tout point $\mathbf{p}^*$ de coordonnées $\mathbf{p}^* = [u^*\ v^*\ 1]^\top$ vérifiant $(u^*, v^*) \in \{1, 2, ..., n\} \times \{1, 2, ..., m\}$ la valeur $I^*(\mathbf{p}^*) = \mathcal{I}^*(u^*, v^*)$. Pour les valeurs non entières de (u^*, v^*), la valeur de la fonction $I^*(\mathbf{p}^*)$ est obtenue par une interpolation des valeurs entières $\mathcal{I}^*(u^*, v^*)$. Dans la suite, nous utiliserons la notation $\mathcal{I}^*$ (en calligraphique) à la fois pour l'image $\mathcal{I}^*$ et pour la fonction I^*, c'est-à-dire, nous considérons l'image définie sur $\mathbb{P}^2$ mais observable seulement en $\{1, 2, ..., n\} \times \{1, 2, ..., m\}$.

Nous supposons également que le pixel correspondant à la projection d'un même point physique $\mathcal{P}$ a la même intensité dans deux images différentes. En d'autres termes, si $\mathbf{p}^*$ est la projection de $\mathcal{P}$ dans l'image de référence $\mathcal{I}^*$ et $\mathbf{p}$ sa projection dans l'image courante $\mathcal{I}$, nous avons :

$$\mathcal{I}^*(\mathbf{p}^*) = \mathcal{I}(\mathbf{p}) \tag{2.23}$$

Il s'agit d'une hypothèse classique dans le domaine de la vision par ordinateur (appelée "image constancy assumption" en anglais). Cette hypothèse implique que les changements d'intensité sont dûs seulement au mouvement du capteur et/ou de la cible. Par conséquent, nous considérons que les changements d'illumination sont des perturbations de notre modèle.

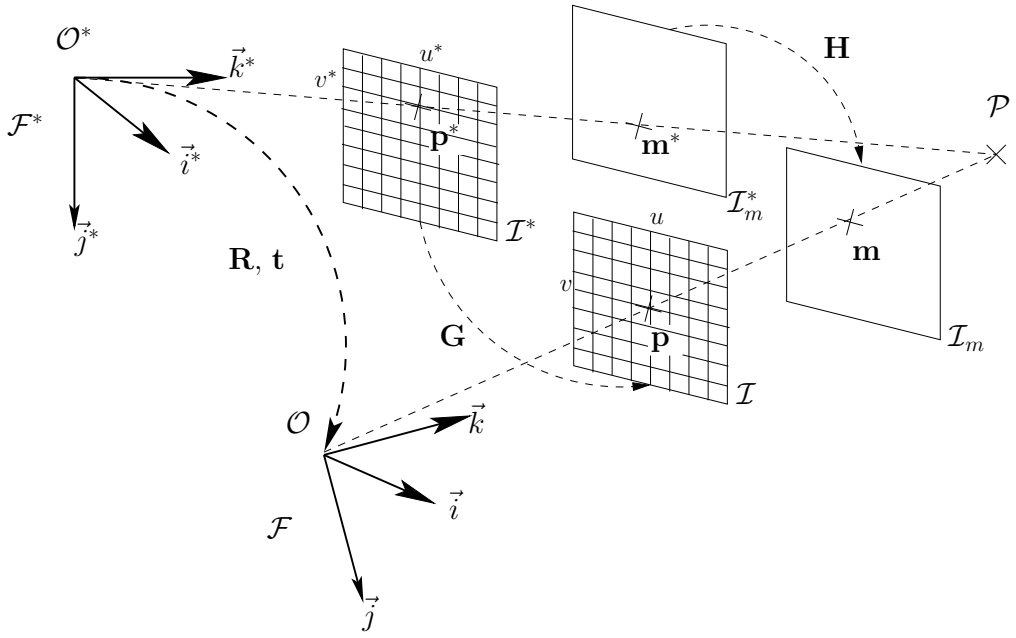


Figure 2.2 – Transformation projective planaire dans l'image

2.3.3 Transformation projective planaire dans l'image

Si le point $\mathcal{P}$ appartient au plan π, en utilisant les équations (2.16), (2.19) et (2.20), les deux points images $\mathbf{p}$ et $\mathbf{p}^*$ vérifient :

$$\mathbf{p} \propto \mathbf{G}\mathbf{p}^* \tag{2.24}$$

où la matrice $\mathbf{G}$ s'écrit de la manière suivante :

$$\mathbf{G} = \mathbf{K}\mathbf{H}\mathbf{K}^{-1} \tag{2.25}$$

Étant donné un ensemble de N couples de points non colinéaires mis en correspondance $(\mathbf{p}_i^*, \mathbf{p}_i)$ où $i \in \{1, 2, ..., N\}$ et où $N \geq 4$, il est possible de calculer la matrice $\mathbf{G}$. D'une manière analogue, la matrice $\mathbf{G}$ est généralement calculée à un facteur d'échelle près. Nous choisissons de fixer le facteur d'échelle de façon à avoir $\mathbf{G} \in \mathbb{SL}(3)$. Il est clair que ce choix est homogène avec $\mathbf{H} \in \mathbb{SL}(3)$ car nous avons : $\mathbf{G} \in \mathbb{SL}(3)$ si et seulement si $\mathbf{H} \in \mathbb{SL}(3)$. Nous pouvons donc dire que la matrice $\mathbf{G}$ a 8 degrés de liberté et peut être définie à l'aide d'un vecteur $\mathbf{x}$ de dimensions (8×1).

La matrice $\mathbf{G}$ définit également une transformation projective planaire dans l'image (voir la Figure 2.2). Il est possible de définir une action de groupe de $\mathbb{SL}(3)$ sur $\mathbb{P}^2$, notée $\mathbf{w}$, comme suit :

$$\mathbf{w} : \quad \mathbb{SL}(3) \times \mathbb{P}^2 \quad \rightarrow \quad \mathbb{P}^2 \tag{2.26}$$

où pour $\mathbf{G} \in \mathbb{SL}(3)$, nous avons $\mathbf{w}(\mathbf{G})$ un automorphisme défini de la manière suivante :

$$\begin{array}{rccl} \mathbf{w}(\mathbf{G}): & \mathbb{P}^2 & \rightarrow & \mathbb{P}^2 \\ & \mathbf{p}^* & \mapsto & \mathbf{p} \quad = \mathbf{w}(\mathbf{G})(\mathbf{p}^*) \propto \mathbf{G}\mathbf{p}^* \end{array} \tag{2.27}$$

Soit $\mathbf{I}$ la matrice identité. La matrice $\mathbf{I}$ correspond à l'élément neutre du groupe de transformation $\mathbb{SL}(3)$. Nous avons les propriétés suivantes :

- $\mathbf{w}(\mathbf{I})$ est l'application identité, c'est-à-dire, $\forall \mathbf{p} \in \mathbb{P}^2$:

$$\mathbf{w}(\mathbf{I})(\mathbf{p}) = \mathbf{p} \tag{2.28}$$

- la composition de deux actions correspond à l'action de la composition, c'est-à-dire, $\forall \mathbf{p}^* \in \mathbb{P}^2$ et $\forall \mathbf{G}_1,\ \mathbf{G}_2 \in \mathbb{SL}(3)$:

$$\mathbf{w}(\mathbf{G}_1)(\mathbf{w}(\mathbf{G}_2)(\mathbf{p}^*)) = \mathbf{w}(\mathbf{G}_1) \circ \mathbf{w}(\mathbf{G}_2)(\mathbf{p}^*) = \mathbf{w}(\mathbf{G}_1\mathbf{G}_2)(\mathbf{p}^*) \tag{2.29}$$

- l'inverse d'une action correspond à action de l'inverse, c'est-à-dire, $\forall \mathbf{G} \in \mathbb{SL}3$:

$$(\mathbf{w}(\mathbf{G}))^{-1} = \mathbf{w}(\mathbf{G}^{-1}) \tag{2.30}$$

Si le point $\mathcal{P}$ appartient à un plan, le principe de l'"image constancy assumption" (décrit dans l'équation (2.23) du paragraphe 2.3.2) peut être écrit :

$$\mathcal{I}\left(\mathbf{w}\left(\mathbf{G}\right)\left(\mathbf{p}^*\right)\right) = \mathcal{I}^*(\mathbf{p}^*) \tag{2.31}$$

La figure 2.3 illustre un exemple du principe de l'"image constancy assumption" dans le cas d'un point appartenant à une surface planaire.

2.3.4 Transformation de l'image : "warping"

Afin d'approcher la valeur de $\mathcal{I}^*(\mathbf{p})$, où le point $\mathbf{p} = \mathbf{w}(\mathbf{G})(\mathbf{p}^*) = [u\ v\ 1]^\top$ ne correspond pas à un pixel (c'est-à-dire, $(u, v) \notin \mathbb{N}^2$), nous effectuons une interpolation à l'aide des valeurs des pixels voisins. Plusieurs interpolations sont possibles parmi lesquelles nous pouvons citer :

- L'interpolation au plus proche voisin "Nearest neighbor" : en notant par $E()$ la fonction partie entière (associant à tout réel sa partie entière), nous calculons $\mathcal{I}^*(\mathbf{p})$ par la formule suivante :

$$\mathcal{I}^*(\mathbf{p}) = \mathcal{I}^*\left(E(u + 0.5), E(v + 0.5)\right) \tag{2.32}$$

  Cette interpolation est simple et rapide. Cependant, l'image obtenue n'est pas de bonne qualité (apparition de discontinuités).

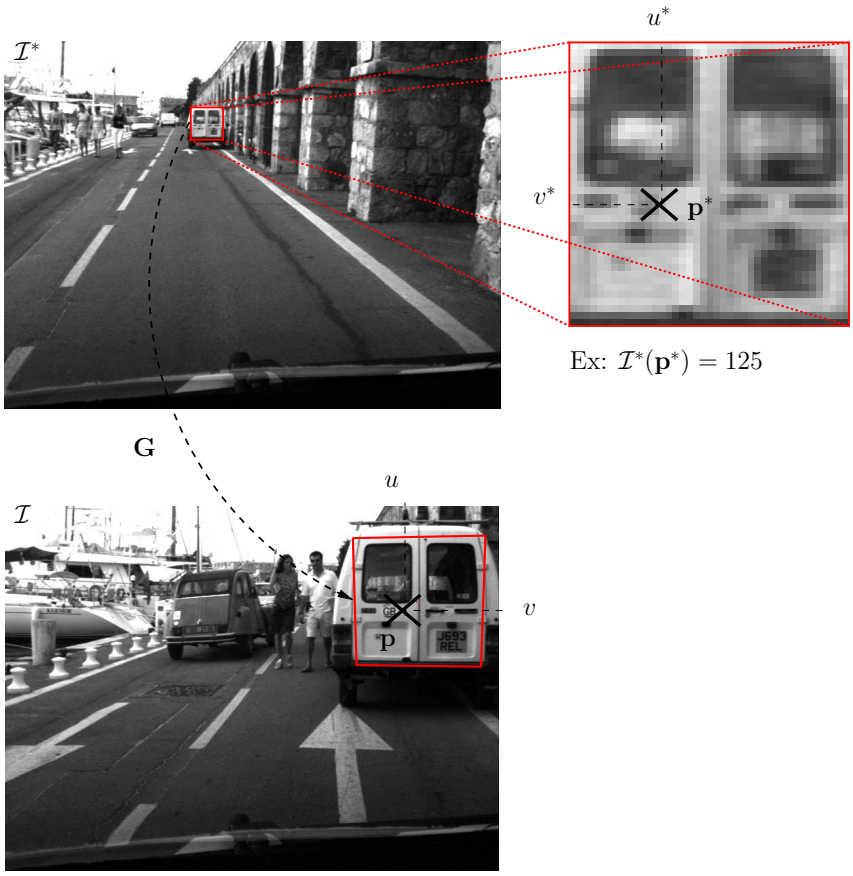


Figure 2.3 – Cas idéal de deux points correspondants ayant la même intensité

– L'interpolation bilinéaire : en notant par $\delta_u = u - E(u)$, $\delta_v = v - E(v)$, et par $\mathcal{I}^*_{ij}(\mathbf{p}) = \mathcal{I}^*\left(E(u)+i, E(v)+j\right)$, nous calculons $\mathcal{I}^*(\mathbf{p})$ par la formule suivante :

$$\mathcal{I}^*(\mathbf{p}) = \begin{bmatrix} 1-\delta_v \\ \delta_v \end{bmatrix}^\top \begin{bmatrix} \mathcal{I}^*_{0,0}(\mathbf{p}) & \mathcal{I}^*_{1,0}(\mathbf{p}) \\ \mathcal{I}^*_{0,1}(\mathbf{p}) & \mathcal{I}^*_{1,1}(\mathbf{p}) \end{bmatrix} \begin{bmatrix} 1-\delta_u \\ \delta_u \end{bmatrix} \tag{2.33}$$

Cette interpolation utilise les 4 voisins les plus proches de $\mathbf{p}$. Elle permet d'avoir une image plus lisse que l'interpolation au plus proche voisin. Toutefois, les dérivées de l'image obtenue restent discontinues.

– L'interpolation bicubique : nous calculons $\mathcal{I}^*(\mathbf{p})$ par la formule suivante :

$$\mathcal{I}^*(\mathbf{p}) = \begin{bmatrix} f(1+\delta_v) \\ f(\delta_v) \\ f(1-\delta_v) \\ f(2-\delta_v) \end{bmatrix}^\top \begin{bmatrix} \mathcal{I}^*_{-1,-1}(\mathbf{p}) & \mathcal{I}^*_{-1,0}(\mathbf{p}) & \mathcal{I}^*_{-1,1}(\mathbf{p}) & \mathcal{I}^*_{-1,2}(\mathbf{p}) \\ \mathcal{I}^*_{0,-1}(\mathbf{p}) & \mathcal{I}^*_{0,0}(\mathbf{p}) & \mathcal{I}^*_{0,1}(\mathbf{p}) & \mathcal{I}^*_{0,2}(\mathbf{p}) \\ \mathcal{I}^*_{1,-1}(\mathbf{p}) & \mathcal{I}^*_{1,0}(\mathbf{p}) & \mathcal{I}^*_{1,1}(\mathbf{p}) & \mathcal{I}^*_{0,2}(\mathbf{p}) \\ \mathcal{I}^*_{2,-1}(\mathbf{p}) & \mathcal{I}^*_{2,0}(\mathbf{p}) & \mathcal{I}^*_{2,1}(\mathbf{p}) & \mathcal{I}^*_{0,2}(\mathbf{p}) \end{bmatrix} \begin{bmatrix} f(1+\delta_u) \\ f(\delta_u) \\ f(1-\delta_u) \\ f(2-\delta_u) \end{bmatrix} \tag{2.34}$$

où la fonction f est définie de la manière suivante :

$$f(t) = \mathrm{sinc}(\pi t) = \begin{cases} 0 & \text{si} \quad t = 0 \\ \frac{\sin(\pi t)}{\pi t} & \text{si} \quad t \neq 0 \end{cases} \tag{2.35}$$

Afin de gagner du temps de calcul (notamment lors du calcul du sinus), cette fonction peut être approchée de la manière suivante :

$$f(t) \approx \begin{cases} 1 - 2|t|^2 + |t|^3 & \text{si} \quad 0 \leq |t| < 1 \\ 4 - 8|t| + 5|t|^2 - |t|^3 & \text{si} \quad 1 \leq |t| < 2 \\ 0 & \text{si} \quad 2 \leq |t| \end{cases} \tag{2.36}$$

Cette interpolation utilise les 16 voisins les plus proches de $\mathbf{p}$. L'image obtenue est plus lisse que celle de l'interpolation au plus proche voisin et les contours sont plus accentués que ceux de l'interpolation bilinéaire. Cependant, malgré l'approximation de l'équation (2.36), le temps de calcul est beaucoup plus important car la formule est plus complexe.

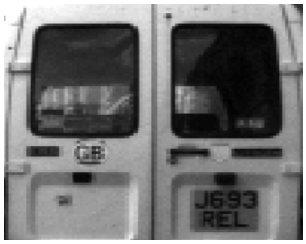
Dans la Figure 2.4, nous comparons les 3 méthodes d'interpolation. Nous utilisons une image extraite d'une séquence "Port d'Antibes" (voir la deuxième image de la Figure 2.3) que nous agrandissons d'un facteur 2 à l'aide d'une transformation projective. Nous pouvons observer la qualité très moyenne de l'interpolation du plus proche voisin dans la Figure 2.4(a). Dans la Figure 2.4(d), nous montrons un détail de l'image où nous voyons clairement les discontinuités obtenues avec une telle interpolation. Dans la Figure 2.4(b), nous pouvons voir le résultat de

l'interpolation bilinéaire. L'image est plus lisse que celle obtenue avec l'interpolation du plus proche voisin (voir le détail de la Figure 2.4(e)). L'interpolation bicubique comme l'interpolation bilinéaire donne une image lissée mais avec des contours plus nets (voir la Figure 2.4(c) et la Figure 2.4(f)).

(a) Nearest neighbor

(b) Bilinéaire

(c) Bicubique



(d) Nearest neighbor (détail)

(e) Bilinéaire (détail)

(f) Bicubique (détail)

Figure 2.4 – Comparaison des différentes méthodes d'interpolation

Deuxième partie

Suivi visuel temps-réel

Résumé

Le suivi visuel (noté SV) d'objets dans une séquence d'images est une étape fondamentale dans de nombreuses applications de vision par ordinateur. Certaines applications (les applications de vidéo-surveillance, par exemple) ne nécessitent qu'un suivi grossier des objets considérés tandis que pour d'autres, notamment celles concernant la commande de robots par vision, l'objectif final est non seulement la localisation précise des objets dans l'image, mais aussi la régulation du déplacement relatif dans l'espace Cartésien entre les objets et la caméra.

Étant donné que nous nous intéressons principalement aux méthodes de SV pouvant être utilisées pour des applications robotiques temps-réel, telles que des applications d'asservissement visuel, il est nécessaire d'avoir un SV efficace (c'est-à-dire, peu coûteux en temps de calcul), précis (permettant d'avoir un positionnement précis) et robuste (peu sensible aux différentes perturbations, bruits ou imperfections dans l'image). Nous souhaitons également que le SV ne nécessite aucune étape d'apprentissage et qu'il soit indépendant de l'existence ou non de certaines primitives visuelles locales dans l'image de la cible suivie. Or, la majorité des méthodes de SV existantes dans la littérature est soit conçue pour des applications purement de vision (et généralement ne sont pas soumises à la contrainte du temps-réel), soit basée sur le suivi d'amers visuels particuliers, soit nécessite une étape d'apprentissage ou encore présente des problèmes de vitesse de convergence lente et/ou de manque de précision.

Dans le premier chapitre de cette partie, nous présentons un bref état de l'art des différentes catégories des méthodes existantes de SV temps-réel et nous expliquerons les raisons qui nous ont conduit à choisir une certaine catégorie de méthodes. Nous présentons également une sorte d'historique des méthodes de cette catégorie et nous argumentons les motivations qui nous ont conduits à proposer une nouvelle alternative.

Dans le deuxième chapitre, nous présentons une nouvelle méthode de SV basée sur un algorithme de minimisation baptisé ESM [1] permettant une estimation précise et efficace de la transformation projective dans l'image d'une surface plane. Cette méthode ne nécessite pas d'étape d'apprentissage. Nous montrons que cet algorithme est très bien adapté à notre problème et qu'il permet d'avoir de très bonnes propriétés permettant de pallier les faiblesses des différentes approches existantes dans sa catégorie.

Dans le troisième chapitre, nous nous intéressons au SV de cibles rigides planaires par morceaux. Le SV permet non seulement l'estimation de la transformation projective dans l'image d'une ou de plusieurs surfaces planes appartenant au même objet rigide, mais aussi d'estimer le mouvement de cet objet dans l'espace Cartésien. Contrairement au premier chapitre, nous supposons avoir le modèle 3D de la cible. Nous supposons avoir les vecteurs normaux aux plans constituant la cible et la distance de ces plans à la caméra (à un facteur d'échelle près), le tout à la position de référence. Nous utilisons l'algorithme de minimisation ESM (présenté dans le premier chapitre) pour résoudre le problème d'optimisation que constitue le SV dans ce

1. Efficient Second-order Minimization

cas particulier. La méthode proposée permet d'utiliser le modèle de la cible quand celui-ci est disponible et d'intégrer d'éventuelles contraintes sur le mouvement relatif entre la caméra et la cible afin de réduire les paramètres à estimer. Cette méthode de SV résout à la fois le problème de suivi dans l'image et l'estimation du mouvement, le tout dans un cadre unifié.

Chapitre 3

État de l'art

3.1 Introduction

D'une manière générale, un algorithme de SV peut être considéré comme étant un problème d'optimisation. Il est possible de classer les différentes approches suivant la fonction à optimiser et suivant la manière de l'optimiser. Il est également possible de les classer suivant les informations disponibles sur les objets suivis telles que, par exemple, si le modèle CAO [1] est fourni ou non, ou bien si la texture des objets suivis a été préalablement apprise ou non.

Nous considérons deux groupes. Le premier groupe concerne les méthodes de suivi des primitives et le deuxième groupe s'intéresse au suivi de régions. Les approches du deuxième groupe peuvent être classées en deux sous-groupes : suivi de régions avec apprentissage hors ligne et suivi de régions sans étape d'apprentissage.

3.2 Méthodes de suivi de primitives

Dans le premier groupe, nous trouvons des méthodes qui cherchent à suivre, entre deux acquisitions successives, des primitives visuelles locales telles des points d'intérêt, des segments, des droites ou encore des contours (Hager and Toyama, 1998; Marchand, 1999). Ces approches nécessitent généralement une extraction préalable des amers visuels grâce à des filtres dédiés tels que (Canny, 1986; Deriche, 1987) pour les contours, ou bien (Harris and Stephens, 1988; Schmid et al., 2000; Lowe, 2004) pour les points d'intérêt, ou encore (Forstner, 1994; Smith and Brady, 1997) pour les contours et les points à la fois. Certaines méthodes de SV nécessitent le modèle 3D (le modèle CAO) de la cible à suivre (Lowe, 1992; Kollnig and Nagel, 1997; Marchand et al., 2001a; Drummond and Cipolla, 2002; Comport et al., 2004). Ces méthodes estiment la position relative de la caméra et de la cible en minimisant l'erreur entre la projection du modèle 3D de la cible et les contours extraits dans l'image.

1. Conception Assistée par Ordinateur

L'avantage de ces méthodes est qu'elles sont robustes aux occultations partielles de la cible et permettent de mesurer le déplacement en imposant des contraintes connues sur le modèle de mouvement de la caméra et/ou sur le modèle 3D de la scène observée. Cependant, elles sont sensibles à la précision de l'estimation du modèle 3D de la cible. Ces méthodes sont difficilement applicables dans le cas de cibles texturées car un fort gradient dans l'image peut être attribué, soit à un contour physique du modèle 3D, soit à de la texture. Par conséquent, le SV dépend grandement de la qualité de la détection des primitives visuelles. Par exemple, dans une image bruitée, un détecteur de points d'intérêt aura tendance à extraire des points correspondant au bruit ; et dans une image présentant une distorsion radiale mal corrigée, il est difficile de faire la différence entre un contour courbé et une ligne droite. Par ailleurs, le SV basé sur des primitives extraites de l'image est inutilisable dans le cas où l'image de l'objet à suivre ne contient pas de primitives visuelles locales particulières.

Certaines méthodes de SV que nous appellerons "hybrides" essaient de résoudre les problèmes du suivi d'un certains type de primitives en le combinant avec le suivi d'autres types. Ou encore, il existe des méthodes hybrides utilisant les primitives et la valeur brute des pixels dans l'image d'une manière séquentielle (Bascle et al., 1994; Chiba and Kanade, 1998; Marchand et al., 2001b) ou bien, plus récemment, d'une manière simultanée (Masson et al., 2004a; Masson et al., 2004b; Vacchetti et al., 2004; Pressigout and Marchand, 2005).

3.3 Méthodes de suivi de régions

Le deuxième groupe contient des méthodes qui utilisent directement les intensités lumineuses, c'est-à-dire, le signal brut de l'image. Ces méthodes estiment les paramètres du modèle de mouvement, de la déformation ou de l'illumination d'une "imagette" de référence entre deux images acquises successivement. L'estimation se fait en minimisant une mesure d'erreur basée sur les intensités lumineuses de l'image que nous appellerons "fonction de coût". La fonction de coût peut être définie de plusieurs manières, mais la plus utilisée reste la somme des carrés des différences (SSD). Plusieurs approches ont été proposées pour effectuer la minimisation de la fonction de coût. D'une manière générale, ces approches se basent sur la linéarisation de la fonction d'erreur par rapport aux paramètres de la transformation dans l'image, puis la minimisation se fait d'une manière itérative. Ces approches peuvent être classées en deux sous-groupes.

3.3.1 Méthodes avec apprentissage hors ligne

Les méthodes du premier sous-groupe effectuent une phase d'apprentissage hors ligne de la variation de la fonction de coût en fonction des paramètres du modèle de transformation dans l'image. La phase d'apprentissage se fait à l'aide d'algorithmes numériques, telle que l'approche de décomposition des différences ("Difference Decomposition approach") introduite dans (Gleicher, 1997) et appliquée dans divers travaux (Sclaroff and Isidoro, 1998; Cootes et al., 1998;

Black and Jepson, 1998; La Cascia et al., 2000; Jurie and Dhome, 2002; Bayro-Corrochano and Ortegon-Aguilar, 2004). Ces méthodes permettent de résoudre le problème du suivi. Cependant, étant donné que la phase d'apprentissage est généralement très coûteuse en temps de calcul, il est difficile d'utiliser ce type de méthode dans certaines applications robotiques où la cible à suivre a un mouvement rapide. En effet, la cible peut sortir du champ de vue de la caméra durant la phase d'apprentissage. Par conséquent, la commande du robot ne peut pas se faire par rapport à cette cible. Dans certaines applications, l'utilisateur sélectionne une zone d'intérêt sur une cible mobile que le robot doit suivre. Le départ du suivi doit être immédiat. Donc, une phase d'apprentissage lourde représentera un handicap majeur pour ces applications. Par ailleurs, ces approches ont un inconvénient de taille : quand une partie (même très petite) de la cible à suivre sort du champ de vue de la caméra (c'est-à-dire sort de l'image) ou bien est occultée, la phase apprentissage doit être refaite parce que l'imagette de référence change. Évidemment, il est difficilement envisageable pour une application robotique d'effectuer l'apprentissage préalable de tous les changements possibles de la cible de référence.

3.3.2 Méthodes sans apprentissage

Les approches du deuxième sous-groupe ne nécessitent pas de phase d'apprentissage. La relation entre l'erreur mesurée et la variation des paramètres de transformation est estimée en ligne. Ces algorithmes permettent d'estimer les paramètres de divers modèles de transformation dans l'image : une simple translation, une transformation affine ou une transformation homographique... Étant donné que l'objectif final est non seulement la localisation précise des objets dans l'image, mais aussi l'estimation du déplacement relatif entre deux positions de la caméra, deux alternatives sont possibles.

La première alternative consiste à estimer la transformation projective (la translation, la transformation affine ou la transformation homographique) de l'objet entre les images successives puis, en utilisant le modèle 3D de l'objet suivi et les paramètres intrinsèques de la caméra, déterminer le déplacement relatif entre l'objet et la caméra. Il est possible d'utiliser, par exemple, dans le cas d'une transformation homographique, l'algorithme proposé dans (Faugeras and Lustman, 1988) pour extraire le déplacement relatif recherché (la rotation et translation Cartésiennes une fois que la transformation projective de l'objet dans l'image est estimée). Pour cette alternative, nous nous focalisons particulièrement sur le problème d'optimisation "bas-niveau" de la fonction de coût permettant de suivre "dans l'image" l'objet considéré. Plusieurs algorithmes de minimisation peuvent être utilisés pour estimer ces paramètres de transformation : la descente de gradient, la minimisation de Gauss-Newton, la minimisation de Levenberg-Marquardt ou encore la minimisation de Newton.

Cette approche a été introduite par (Lucas and Kanade, 1981). Les auteurs proposent d'estimer le déplacement en translation de certaines régions d'intérêt dans l'image en utilisant la

minimisation de Gauss-Newton. Il supposent avoir une estimation préalable du déplacement et déterminent d'une manière itérative les incréments nécessaires pour minimiser la fonction de coût. Cette méthode est basée sur l'approximation au premier-ordre de la variation du signal image par rapport aux incréments du déplacement. Elle est caractérisée par le fait que la direction de descente est estimée par un Jacobien calculé à l'état courant de la minimisation et par le fait que l'incrémentation du déplacement se fait d'une manière additive. D'où son appellation "forward additional approach" par (Baker and Matthews, 2001). Cette approche a été généralisée aux transformations affines par (Shi and Tomasi, 1994), qui par la même occasion propose une manière de sélectionner les "bonnes" régions d'intérêt à suivre. La méthode résultante est connue sous le nom KLT [2]. Récemment dans (Bouguet, 1999), une approche pyramidale de cette méthode a été proposée pour l'implémentation de cet algorithme. Elle permet d'accélérer l'algorithme et d'avoir une zone de convergence plus grande. C'est cette même méthode qui est actuellement implémentée dans la très populaire librairie de traitement d'images et de vision par ordinateur "OpenCV".

Dans (Hager and Belhumeur, 1998), les auteurs proposent une méthode équivalente à cette approche au premier-ordre permettant de pré-calculer, une fois pour toutes, une partie du Jacobien utilisé dans la minimisation de Gauss-Newton afin d'estimer la transformation affine. L'incrémentation du déplacement est également effectuée d'une manière additive sauf que la direction de descente est calculée à l'état de référence. D'où son appellation "inverse additional approach" par (Baker and Matthews, 2001). Ceci permet d'avoir une boucle de minimisation plus rapide et donc permet d'avoir un nombre d'itérations possibles de l'algorithme plus important entre deux images acquises successivement. Cependant, la région de convergence devient moins large. Cette méthode initialement proposée pour des transformations affines a été étendue aux transformations homographiques par (Buenaposada and Baumela, 2002).

Dans (Shum and Szeliski, 2000), les auteurs proposent une approche, appelée "forward compositional" par (Baker and Matthews, 2001), basée également sur la minimisation de Gauss-Newton. Celle-ci est caractérisée par le fait que la direction de descente est estimée par un Jacobien calculé à l'état courant de la minimisation et par le fait que l'incrémentation (ou la mise à jour) du déplacement se fait en utilisant la loi de composition matricielle (la multiplication de matrices). Ceci permet d'avoir une amélioration de l'efficacité de l'algorithme de (Lucas and Kanade, 1981; Shi and Tomasi, 1994), sans avoir recours à des approximations. En effet, une partie du Jacobien peut être pré-calculée une fois pour toutes et il est possible d'estimer des transformations homographiques des objets suivis sans l'image. La mise à jour par la composition a d'autres propriétés intéressantes. Par exemple, elle permet de fixer le groupe auquel appartient la transformation projective estimée, ce qui n'est pas garanti par la mise à jour additive.

Dans (Baker and Matthews, 2001; Baker and Matthews, 2004), les auteurs proposent une

2. Kanade, Lucas, Tomasi

approche qu'ils appellent "inverse compositional". Cette méthode est caractérisée par le fait que la direction de descente est estimée par un Jacobien calculé à l'état de référence de la minimisation et par le fait que la mise à jour du déplacement se fait en utilisant la loi de composition. Ils montrent également que toutes les approches basées sur une approximation au premier-ordre (et donc pour de faibles déplacements) sont plus ou moins équivalentes au niveau du taux et de la fréquence de convergence, mais elles diffèrent par leur efficacité. En effet, étant donné que la totalité du Jacobien est pré-calculée une fois pour toutes (elle n'est pas mise à jour au cours des itérations), l'approche qu'ils proposent est plus rapide au niveau du temps de calcul grâce aux approximations effectuées. Ce qui permet de pouvoir faire plus d'itérations, à des fréquences d'acquisition égales. Deux inconvénients de taille pour cette approche apparaissent : quand une partie (même très faible) de la cible sort du champ de vue de la caméra (de l'image) et quand la cible suivie est partiellement occultée. Dans ces deux cas, l'utilisation du Jacobien constant (correspondant à l'imagette de référence) donne de très mauvais résultats.

La deuxième alternative pour une estimation du déplacement relatif entre deux positions de la caméra consiste à suivre une cible rigide et planaire par morceaux dans une séquence d'images, et en même temps, d'estimer, d'une manière précise, le déplacement relatif entre l'objet et la caméra. Dans (Chiuso et al., 2002; Jin et al., 2003), les auteurs proposent de suivre des zones d'intérêt dans l'image, de déterminer le déplacement et d'estimer le modèle 3D de la scène observée. Ces algorithmes reposent sur une comparaison des intensités lumineuses brutes des pixels de l'image intégrée dans un filtre de Kalman. Dans (Jin et al., 2003), pour chaque région suivie, une mise en correspondance est effectuée ; elle est basée sur une recherche exhaustive du maximum du score du produit de corrélation normalisé autour de la position prédite par le filtre. Le filtrage de Kalman correspond à une mise à jour du premier ordre. En effet, il s'agit d'une minimisation de type Gauss-Newton parce qu'ils utilisent la pseudo-inverse du Jacobien dans la matrice de gain du filtre de Kalman. Dans (Chiuso et al., 2002), chaque région est suivie par l'algorithme proposé dans (Lucas and Kanade, 1981). Les méthodes proposées permettent de résoudre le problème du suivi dans l'image. Cependant, vu le nombre de paramètres à estimer et vue leur complexité algorithmique, elles ne peuvent être appliquées en temps-réel que si le déplacement (et surtout les rotations) entre deux images successives sont très faibles (car les méthodes de suivi proposées sont très sensibles aux rotations).

D'autres méthodes temps-réel permettent d'intégrer le modèle 3D de l'objet suivi et les paramètres intrinsèques de la caméra directement dans l'algorithme de suivi au lieu de les imposer a posteriori. Récemment, plusieurs approches ont été utilisées afin de mesurer le déplacement en imposant des contraintes connues sur le modèle de mouvement de la caméra et/ou sur le modèle 3D de la scène observée (voir par exemple (Sepp and Hirzinger, 2003; Cobzas and Jagersand, 2004)). Ces contraintes sont généralement exprimées dans l'espace Euclidien. Par exemple, quand le déplacement d'une caméra montée sur un robot mobile évoluant sur un sol

plan est estimé par rapport à un objet immobile, seuls trois paramètres de mouvement sont à déterminer (une rotation et deux translations car le mouvement s'effectue sur un plan). Par ailleurs, les différentes régions d'un même objet rigide effectuent le même déplacement dans l'espace Euclidien (dans l'espace 3D), mais leurs transformations dans l'espace projectif (dans l'image) sont différentes. Par exemple, quand la scène est planaire par morceaux, plusieurs imagettes planes peuvent être sélectionnées et suivies indépendamment. Si les imagettes sont sur des plans différents, les homographies correspondantes estimées seront différentes. Dans le cas idéal, toutes les homographies donnent un déplacement cohérent de la caméra (si nous utilisons, par exemple, l'algorithme présenté dans (Faugeras and Lustman, 1988) pour estimer le déplacement à partir des homographies). En pratique, cela n'arrive que très rarement, surtout si nous ne contraignons pas explicitement que les plans sont rigidement attachés les uns aux autres. En effet, il arrive que, dans le cas où les plans n'ont pas une texture riche [3], deux imagettes planaires donnent des déplacements contradictoires. Il est généralement difficile de traduire les informations Euclidiennes additionnelles (concernant le modèle 3D de la cible et concernant le mouvement relatif entre la caméra et la cible suivie) en des contraintes de transformations perspectives simples et exprimées directement dans l'image.

Plusieurs travaux récents se sont intéressés au problème de SV de régions sans apprentissage et en imposant des contraintes Euclidiennes soit concernant le modèle 3D de la scène soit concernant le mouvement relatif entre la caméra et la cible suivie. Dans (Sepp and Hirzinger, 2003), les auteurs s'inspirent du travail de (Buenaposada and Baumela, 2002) pour développer l'approche proposée dans (Hager and Belhumeur, 1998) en rajoutant la contrainte que l'ensemble des points de la surface 3D suivie sont soumis au même déplacement. L'algorithme aboutit à des résultats intéressants, mais reste au premier-ordre, c'est-à-dire, la zone de convergence n'est pas très grande. Dans (Cobzas and Jagersand, 2004), les auteurs évitent le calcul explicite du Jacobien reliant la variation des paramètres de la pose 3D relative entre la caméra et la scène et la variation de l'apparence de la scène dans l'image en utilisant une dérivation implicite de fonctions. La minimisation de l'erreur dans l'image est effectuée d'une manière équivalente à l'algorithme proposé dans (Baker and Matthews, 2004). Dans ce cas particulier où les paramètres à estimer ne sont pas les paramètres de la transformation homographique dans l'image mais les paramètres de déplacement (la translation et la rotation 3D), cet algorithme repose sur une approximation grossière. En effet, la convergence est très locale, c'est-à-dire, pour de faibles déplacements le long de la séquence (voir le Chapitre 5 ou bien (Baker et al., 2004) pour plus de détails). Dans (Sepp, 2005), l'auteur résout le problème des faibles déplacements dû à l'utilisation de l'"inverse compositional" en mettant à jour l'imagette de référence en utilisant l'approche proposée par (Matthews et al., 2003). Cependant, la mise à jour risque de rendre le suivi non fiable parce que l'algorithme n'est plus à l'abri des dérives, c'est-à-dire, si la minimisation n'est

3. Une texture riche est une texture qui impose des contraintes suffisantes pour déterminer la correspondance d'au moins 4 points dans une configuration générale, c'est-à-dire, dans une configuration où parmi les 4 points il n'y a pas 3 points qui sont colinéaires.

pas achevée ou bien si elle aboutit à un minimum local, il est possible que l'algorithme se mette à suivre une cible différente de l'objet initial.

3.4 Conclusion

Étant donné que nous nous intéressons principalement aux méthodes de SV pouvant être utilisées pour des applications robotiques temps-réel, telles que des applications d'asservissement visuel, l'image des objets par rapport auxquels nous souhaitons positionner le robot n'est pas sensée contenir des primitives visuelles locales particulières. La catégorie qui nous intéresse concerne les méthodes où le SV s'effectue grâce aux valeurs des pixels des objets dans l'image. En plus, étant donné que nous voulons pouvoir commencer la commande du robot dès que les objets qui nous intéressent sont dans le champ de vue de la caméra, nous ne pouvons donc pas utiliser les méthodes où une étape d'apprentissage préalable est nécessaire. Nous utiliserons donc une méthode de suivi de régions basée sur la minimisation d'une fonction de coût mesurée à partir des intensités lumineuses de l'image et où la relation entre l'erreur mesurée et la variation des paramètres de transformation est estimée en ligne.

Les différentes méthodes proposées dans la littérature (dont certaines sont citées plus haut et bien d'autres (Romdhani and Vetter, 2003; Xiao et al., 2004)) supposent que l'erreur des intensités lumineuses mesurées varient linéairement par rapport aux paramètres de transformation estimés. Le domaine de convergence de la minimisation est alors très local et dès que le déplacement des objets suivis entre deux images successives devient important, ces méthodes ne réussissent pas à estimer les bons paramètres de déplacement. Si l'on suppose maintenant que l'erreur des intensités lumineuses mesurées varie d'une manière quadratique par rapport aux paramètres de transformation, la résolution du problème d'optimisation avec les approches classiques devient très coûteuse en temps de calcul car elle nécessite l'estimation de matrices Hessiennes représentant la variation quadratique de l'erreur mesurée par rapport aux paramètres de transformation. Ces approches ne peuvent pas résoudre le problème en temps-réel et présentent des problèmes de convergence dans certaines configurations des matrices Hessiennes.

Dans les deux prochains chapitres, nous proposons une méthode de résolution du problème d'optimisation qui suppose que les paramètres de transformation varient d'une manière quadratique par rapport à l'erreur des intensités lumineuses mesurées. Nous estimons dans un premier temps les paramètres de transformation dans l'image. Dans un deuxième temps, nous estimons les paramètres de transformation dans l'espace Cartésien. À chaque fois, une paramétrisation adéquate, basée sur l'algèbre de Lie, est utilisée. La méthode proposée ne nécessite pas le calcul des différentes matrices Hessiennes. Par conséquent, elle peut être utilisée dans des applications temps-réel et elle admet, grâce au fait d'être au second-ordre, une zone de convergence plus grande que les approches existantes.

Chapitre 4

Suivi visuel dans l'image de surfaces planaires avec la méthode ESM

4.1 Introduction

Dans ce chapitre, nous nous intéressons particulièrement au problème d'optimisation bas-niveau permettant de suivre "dans l'image" une certaine cible. Nous supposons que nous n'avons pas de mesures du modèle de la cible à suivre (c'est-à-dire, dans le cas de la cible planaire, le vecteur normal au plan de la cible n'est pas connue). Nous supposons également que le mouvement de la cible est complètement aléatoire. Nous proposons une approche de SV, basée sur un nouvel algorithme de minimisation appelé "ESM". Nous montrons que cet algorithme est très bien adapté au problème de SV et qu'il permet d'avoir des propriétés meilleures que celles des approches existantes.

Nous commençons le chapitre par une étude théorique. Puis, nous effectuons des simulations numériques sur des images synthétiques et réelles. Enfin, nous réalisons des tests sur des séquences vidéo réelles.

4.2 Le suivi visuel dans l'image de surfaces planaires

4.2.1 Modélisation du problème

Supposons vouloir suivre une cible rigide de la scène et planaire par morceaux. Bien que la cible soit rigide, dans un premier temps, nous supposerons vouloir suivre chaque plan de la cible d'une manière indépendante. Considérons, pour des raisons de simplicité, le suivi d'un seul plan. Soit q le nombre total de pixels d'une certaine région d'image de forme quelconque et correspondant à la projection d'une partie planaire de la cible dans l'image de référence $\mathcal{I}^*$. Cette région de l'image sera appelée "imagette de référence". Nous utiliserons essentiellement les notations introduites dans les paragraphes 2.3.2 et 2.3.3 du Chapitre 2.

Suivre visuellement cette imagette de référence dans l'image courante $\mathcal{I}$ consiste à trouver le vecteur $\overline{\mathbf{x}}$ de dimensions (8×1) contenant les paramètres réels de la transformation projective $\mathbf{G}(\overline{\mathbf{x}})$ (dans le cas d'une paramétrisation globale définie partout), ou bien trouver directement la matrice de la transformation projective $\overline{\mathbf{G}}$ (dans le cas d'une paramétrisation locale définie autour de l'identité), qui transforme chaque pixel $\mathbf{p}_i^*$ de l'imagette en son correspondant dans l'image courante $\mathcal{I}$, c'est-à-dire, trouver l'homographie $\overline{\mathbf{G}}$ telle que $\forall i \in \{1,\ 2,\ ...,\ q\}$:

$$\mathcal{I}\left(\mathbf{w}\left(\overline{\mathbf{G}}\right)(\mathbf{p}_i^*)\right) = \mathcal{I}^*(\mathbf{p}_i^*) \tag{4.1}$$

Supposons que nous avons une approximation $\widehat{\mathbf{x}}$ de $\overline{\mathbf{x}}$, ou bien une approximation $\widehat{\mathbf{G}}$ de $\overline{\mathbf{G}}$, le problème consiste à trouver une transformation incrémentale $\mathbf{G}(\mathbf{x})$ telle que la différence entre la région de l'image $\mathcal{I}$ transformée avec la composition $\mathbf{w}(\widehat{\mathbf{G}})\circ\mathbf{w}(\mathbf{G}(\mathbf{x}))$ et la région correspondante dans l'image $\mathcal{I}^*$ est nulle. Il s'agit de trouver le vecteur $\mathbf{x}$ tel que $\forall i \in \{1,\ 2,\ ...,\ q\}$, nous avons :

$$y_i(\mathbf{x}) = \mathcal{I}\left(\mathbf{w}(\widehat{\mathbf{G}}) \circ \mathbf{w}(\mathbf{G}(\mathbf{x}))(\mathbf{p}_i^*)\right) - \mathcal{I}^*(\mathbf{p}_i^*) = 0 \tag{4.2}$$

Remarque 1 La loi de composition $\circ$ peut être définie de plusieurs manières. Dans (Lucas and Kanade, 1981; Shi and Tomasi, 1994; Hager and Belhumeur, 1998), étant donné que la paramétrisation choisie de l'homographie est globale, cette loi a été définie par :

$$\mathbf{w}(\widehat{\mathbf{G}}) \circ \mathbf{w}(\mathbf{G}(\mathbf{x})) = \mathbf{w}(\mathbf{G}(\widehat{\mathbf{x}})) \circ \mathbf{w}(\mathbf{G}(\mathbf{x})) = \mathbf{w}(\mathbf{G}(\widehat{\mathbf{x}} + \mathbf{x})) \tag{4.3}$$

Or, une telle loi de composition n'a pas de bonnes propriétés. Par exemple, elle ne garantit en rien le fait que $det(\mathbf{G}) > 0$ (c'est-à-dire le plan est toujours visible dans l'image). C'est pour cette raison que nous préférons utiliser la loi de composition définie dans le paragraphe 2.3.3 du Chapitre 2 car elle nous permet d'avoir des propriétés plus intéressantes (voir (Baker and Matthews, 2004) pour plus de détails). Dans tout ce qui suit, nous utiliserons la loi de composition définie dans le paragraphe 2.3.3 du Chapitre 2 faisant appel au produit matriciel des matrices de transformation projective et nous comparerons des algorithmes utilisant cette même loi de composition.

4.2.2 Définition vectorielle du système d'équations

Si nous notons par $\mathbf{y}(\mathbf{x})$ le vecteur de dimensions $(q\times 1)$ contenant les différences d'image :

$$\mathbf{y}(\mathbf{x}) = \left[\begin{array}{cccc} y_1(\mathbf{x}) & y_2(\mathbf{x}) & ... & y_q(\mathbf{x}) \end{array}\right]^\top \tag{4.4}$$

le problème revient à trouver le vecteur $\mathbf{x} = \widetilde{\mathbf{x}}$ vérifiant le système d'équations :

$$\mathbf{y}(\widetilde{\mathbf{x}}) = \mathbf{0} \tag{4.5}$$

Grâce aux équations (2.28) et (4.38), il est évident que la solution du système (4.5) vérifie :

$$\mathbf{G}(\widetilde{\mathbf{x}}) = \widehat{\mathbf{G}}^{-1}\overline{\mathbf{G}} \tag{4.6}$$

Le système d'équations (4.5) est généralement non linéaire et plusieurs méthodes de résolution peuvent être envisagées. Cependant, s'agissant d'un problème devant être résolu en temps réel, généralement à la cadence vidéo, et devant être utilisé pour des applications d'asservissement visuel, il est important que la méthode choisie soit efficace, c'est-à-dire avec un faible temps de calcul. C'est pour cette raison que, dans la majeure partie des cas, la résolution du système se fait d'une manière itérative après linéarisation du signal image par rapport aux paramètres de la transformation.

4.2.3 Approximation du système d'équations

Tout d'abord, nous définissons la matrice Jacobienne de l'erreur $\mathbf{J}(\mathbf{x})$. Il s'agit d'une matrice de dimensions $(q \times 8)$ qui représente la variation du vecteur $\mathbf{y}(\mathbf{x})$ en fonction des paramètres $\mathbf{x}$ de la transformation projective. Comme définie dans le paragraphe 2.1.2, cette matrice s'écrit sous la forme suivante :

$$\mathbf{J}(\mathbf{x}) = \nabla_{\mathbf{x}}\mathbf{y}(\mathbf{x}) \tag{4.7}$$

Nous définissons la matrice $\mathbf{M}(\mathbf{x}_1, \mathbf{x}_2)$ de dimensions $(q \times 8)$ qui s'écrit $\forall(\mathbf{x}_1, \mathbf{x}_2) \in \mathbb{R}^8 \times \mathbb{R}^8$:

$$\mathbf{M}(\mathbf{x}_1, \mathbf{x}_2) = \nabla_{\mathbf{x}_1}\left(\mathbf{J}(\mathbf{x}_1)\mathbf{x}_2\right) = \left[\begin{array}{cccc} \dfrac{\partial^2 y_1(\mathbf{x}_1)}{\partial \mathbf{x}_1^2}\mathbf{x}_2 & \dfrac{\partial^2 y_2(\mathbf{x}_1)}{\partial \mathbf{x}_1^2}\mathbf{x}_2 & \ldots & \dfrac{\partial^2 y_q(\mathbf{x}_1)}{\partial \mathbf{x}_1^2}\mathbf{x}_2 \end{array}\right]^\top \tag{4.8}$$

où chaque matrice Hessienne $\frac{\partial^2 y_i(\mathbf{x})}{\partial \mathbf{x}^2}$ est de dimensions (8×8) et s'écrit comme défini dans le paragraphe 2.1.2.

S'agissant d'un problème de SV en temps réel, le déplacement de la cible entre deux images successives est faible, c'est-à-dire, $\mathbf{x} \approx \mathbf{0}$. Il est possible de linéariser le vecteur $\mathbf{y}(\mathbf{x})$ en effectuant un développement en série de Taylor au deuxième ordre au voisinage de $\mathbf{x} = \mathbf{0}$:

$$\mathbf{y}(\mathbf{x}) = \mathbf{y}(\mathbf{0}) + \mathbf{J}(\mathbf{0})\,\mathbf{x} + \frac{1}{2}\mathbf{M}(\mathbf{0}, \mathbf{x})\,\mathbf{x} + \mathbf{O}(\|\mathbf{x}\|^3) \tag{4.9}$$

où $\mathbf{O}(\|\mathbf{x}\|^i)$ représente et représentera un reste d'ordre i. En $\mathbf{x} = \widetilde{\mathbf{x}}$, le système d'équations (4.5) peut être écrit :

$$\mathbf{y}(\widetilde{\mathbf{x}}) \approx \mathbf{y}(\mathbf{0}) + \left(\mathbf{J}(\mathbf{0}) + \frac{1}{2}\mathbf{M}(\mathbf{0}, \widetilde{\mathbf{x}})\right)\widetilde{\mathbf{x}} = \mathbf{0} \tag{4.10}$$

4.2.4 Minimisation itérative du système d'équations

Généralement, la méthode retenue pour résoudre le système d'équations (4.10) est la méthode des moindres carrés. Il s'agit de minimiser d'une manière itérative la fonction de coût suivante :

$$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{y}(\mathbf{0}) + \mathbf{J}(\mathbf{0})\mathbf{x} + \frac{1}{2}\mathbf{M}(\mathbf{0},\mathbf{x})\mathbf{x}\|^2 \tag{4.11}$$

Une condition nécessaire pour qu'un vecteur $\mathbf{x} = \widetilde{\mathbf{x}}$ soit un minimum local ou global de la fonction de coût f est que la dérivée de la fonction de coût soit nulle, c'est-à-dire :

$$\nabla_{\mathbf{x}} f(\mathbf{x})|_{\mathbf{x}=\widetilde{\mathbf{x}}} = \mathbf{0} \tag{4.12}$$

La dérivée de la fonction de coût peut être écrite sous la forme suivante :

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = (\mathbf{J}(\mathbf{0}) + \mathbf{M}(\mathbf{0},\mathbf{x})))^{\top} \left(\mathbf{y}(\mathbf{0}) + \mathbf{J}(\mathbf{0})\mathbf{x} + \mathbf{O}(\|\mathbf{x}\|^2)\right) \tag{4.13}$$

La méthode de minimisation standard de Newton résout le système (4.12) d'une manière itérative. À chaque itération, une solution incrémentale $\widetilde{\mathbf{x}}$ est estimée comme suit :

$$\widetilde{\mathbf{x}} = -\mathbf{S}^{-1}\mathbf{J}(\mathbf{0})^{\top}\mathbf{y}(\mathbf{0}) \tag{4.14}$$

où la matrice $\mathbf{S}$ de dimensions (8×8) s'écrit :

$$\mathbf{S} = \mathbf{J}(\mathbf{0})^{\top}\mathbf{J}(\mathbf{0}) + \sum_{i=0}^{q} \left.\frac{\partial^2 y_i(\mathbf{x})}{\partial \mathbf{x}^2}\right|_{\mathbf{x}=\mathbf{0}} y_i(\mathbf{0}) \tag{4.15}$$

Une fois le vecteur incrémental $\widetilde{\mathbf{x}}$ estimé, nous mettons à jour l'approximation $\widehat{\mathbf{G}}$ de la manière suivante :

$$\widehat{\mathbf{G}} \longleftarrow \widehat{\mathbf{G}}\,\mathbf{G}(\widetilde{\mathbf{x}}) \tag{4.16}$$

À chaque mise à jour de $\widehat{\mathbf{G}}$, nous estimons $\mathbf{y}(\mathbf{0})$ et $\mathbf{J}(\mathbf{0})$ et nous calculons à nouveau $\widetilde{\mathbf{x}}$. Nous arrêtons l'algorithme quand la valeur de $\widetilde{\mathbf{x}}$ calculée devient très faible.

La méthode de Newton a une convergence quadratique dans le voisinage de $\mathbf{x} = \mathbf{0}$. De plus, si la fonction $f(\mathbf{x})$ est convexe quadratique, le minimum global de la fonction peut être trouvé en une seule itération. Cependant, dans le cas où la fonction $f(\mathbf{x})$ n'est pas convexe quadratique, des problèmes de convergence peuvent apparaître si la matrice $\mathbf{S}$ n'est pas définie positive. Par ailleurs, la méthode de Newton nécessite le calcul des q matrices Hessiennes, ce qui coûte très cher en temps de calcul. Plusieurs méthodes de minimisation proposent d'approcher la matrice $\mathbf{S}$ avec une matrice $\widehat{\mathbf{S}}$ définie positive, ce qui revient à utiliser des approximations au premier-ordre dans l'équation (4.9). Parmi ces méthodes, nous pouvons citer :

– La descente de gradient :

$$\mathbf{S} \approx \widehat{\mathbf{S}} = \alpha\, \mathbf{I} \text{ où } \alpha > 0 \tag{4.17}$$

– La méthode de Gauss-Newton :

$$\mathbf{S} \approx \widehat{\mathbf{S}} = \mathbf{J}(\mathbf{0})^\top \mathbf{J}(\mathbf{0}) \tag{4.18}$$

– La méthode de Levenberg-Marquardt :

$$\mathbf{S} \approx \widehat{\mathbf{S}} = \mathbf{J}(\mathbf{0})^\top \mathbf{J}(\mathbf{0}) + \alpha\, \mathbf{I} \text{ où } \alpha > 0 \tag{4.19}$$

Dans la littérature, plusieurs algorithmes de SV de régions sans apprentissage utilisent de telles approximations. Par exemple, dans (Shum and Szeliski, 2000), les auteurs utilisent l'approximation Gauss-Newton avec la mise à jour de la transformation définie par (4.16). Dans (Lucas and Kanade, 1981; Shi and Tomasi, 1994), les auteurs utilisent également l'approximation Gauss-Newton mais la mise à jour des paramètres de la transformation $\widehat{\mathbf{G}}$ est faite d'une manière additive (voir l'équation (4.3)).

Il existe également des algorithmes qui approchent le Jacobien courant $\mathbf{J}(\mathbf{0})$, variable d'une itération à l'autre, par un Jacobien constant permettant ainsi d'accélérer l'algorithme au prix de la réduction de la zone de convergence.

$$\mathbf{J}(\mathbf{0}) \approx \widehat{\mathbf{J}} \tag{4.20}$$

De telles approches permettent de calculer la matrice $\mathbf{J}(\mathbf{0})$ et d'inverser la matrice $\widehat{\mathbf{S}}$ une fois pour toutes. Par exemple, dans (Hager and Belhumeur, 1998) (resp. (Baker and Matthews, 2001)), les auteurs proposent une approche équivalente au premier-ordre à celle de (Shi and Tomasi, 1994) (resp. (Shum and Szeliski, 2000)) mais plus rapide.

Pour conclure, l'approximation au deuxième ordre de la fonction de coût reste très peu utilisée parce qu'elle nécessite le calcul des Hessiens et connaît certains problèmes de convergence quand la matrice $\mathbf{S}$ n'est pas définie positive. C'est pour cette raison que les approches les plus populaires dans le SV des régions sans étape d'apprentissage utilisent essentiellement des approximations au premier-ordre de la fonction de coût.

Nous proposons dans le paragraphe suivant, un algorithme efficace de résolution du système d'équations (4.5) au second-ordre, qu'on appellera "la méthode ESM", et qui ne nécessite pas le calcul coûteux des matrices Hessiennes.

4.3 Le suivi visuel avec la méthode ESM

4.3.1 L'algèbre de Lie $\mathfrak{sl}(3)$

Comme expliqué dans le paragraphe 2.3.3, nous choisissons la matrice de la transformation projective planaire dans l'image $\mathbf{G}(\mathbf{x})$ appartenant au groupe $\mathbb{SL}(3)$. Ce groupe est un groupe de Lie et nous notons par $\mathfrak{sl}(3)$ l'algèbre de Lie associée à ce groupe. Les éléments de cette algèbre sont des matrices de dimensions (3×3) de trace nulle. L'application exponentielle matricielle est un homéomorphisme entre un voisinage de la matrice identité $\mathbf{I} \in \mathbb{SL}(3)$ et un voisinage de la matrice nulle $\mathbf{0}$ de l'algèbre de Lie $\mathfrak{sl}(3)$. Soit $\{\mathbf{A}_1,\ \mathbf{A}_2, ...,\ \mathbf{A}_8\}$ une base de l'algèbre de Lie $\mathfrak{sl}(3)$. Une matrice de $\mathbf{A}(\mathbf{x}) \in \mathfrak{sl}(3)$ peut s'écrire :

$$\mathbf{A}(\mathbf{x}) = \sum_{i=1}^{8} x_i \mathbf{A}_i \tag{4.21}$$

Une transformation projective planaire dans l'image $\mathbf{G}(\mathbf{x}) \in \mathbb{SL}(3)$ dans un voisinage de la matrice identité $\mathbf{I}$ peut donc être paramétrée de la manière suivante :

$$\mathbf{G}(\mathbf{x}) = \exp\left(\mathbf{A}(\mathbf{x})\right) = \sum_{i=0}^{\infty} \frac{1}{i!} \left(\mathbf{A}(\mathbf{x})\right)^i \tag{4.22}$$

Il existe, au moins, deux formes analytiques de cette matrice. La première est un polynôme de deuxième de degré de la matrice $\mathbf{A}(\mathbf{x})$. Il s'agit du reste de la division Euclidienne du polynôme infini de l'exponentiel par le polynôme caractéristique de la matrice $\mathbf{A}(\mathbf{x})$ puis appliqué à celle-ci. La deuxième, plus classique, est obtenue après diagonalisation de la matrice $\mathbf{A}(\mathbf{x})$. L'exponentiel s'applique alors seulement aux éléments de la matrice diagonale obtenue. La matrice $\mathbf{G}(\mathbf{x})$ est ensuite reconstruite à l'aide de la base diagonalisante.

Il est à noter que seules les transformations projectives planaires incrémentales (dans le voisinage de la matrice identité $\mathbf{I}$) peuvent être paramétrées de cette manière.

Cette paramétrisation nous permet d'avoir les deux propriétés suivantes :

1. Soit $\mathbf{x}_1$ et $\mathbf{x}_2$ deux vecteurs de $\mathbb{R}^8$ dans le voisinage du vecteur nul $\mathbf{0} \in \mathbb{R}^8$. Nous avons l'équivalence suivante :
$$\mathbf{G}(\mathbf{x}_1) = \mathbf{G}(\mathbf{x}_2) \iff \mathbf{x}_1 = \mathbf{x}_2 \tag{4.23}$$

2. Soit $\mathbf{x}_1$ et $\mathbf{x}_2$ deux vecteurs de $\mathbb{R}^8$. Nous avons l'équivalence suivante :
$$\mathbf{G}(\mathbf{x}_1) = \mathbf{G}(\mathbf{x}_2)^{-1} \iff \mathbf{x}_1 = -\mathbf{x}_2 \tag{4.24}$$

Ces deux propriétés émanent directement de la définition de la paramétrisation des transformation projectives et de l'homéomorphisme local de l'application exponentielle matricielle.

Nous utilisons la base de l'algèbre de Lie $\mathfrak{sl}(3)$ composée des matrices suivantes :

$$\begin{array}{llll} \mathbf{A}_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, & \mathbf{A}_3 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, & \mathbf{A}_5 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, & \mathbf{A}_7 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \\ \mathbf{A}_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, & \mathbf{A}_4 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, & \mathbf{A}_6 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, & \mathbf{A}_8 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \end{array} \tag{4.25}$$

4.3.2 Calcul des Jacobiens

Nous souhaitons calculer la matrice Jacobienne $\mathbf{J}(\mathbf{x})$ correspondant à la dérivée du vecteur $\mathbf{y}(\mathbf{x})$ défini par les équations (4.4) et (4.38) calculée aux points $\mathbf{x} = \mathbf{0}$ et $\mathbf{x}$ vérifiant la contrainte (4.6). Nous verrons, par la suite, l'intérêt de ce calcul et les propriétés des ces matrices Jacobiennes.

4.3.2.1 Le Jacobien courant

Nous appelons Jacobien courant la matrice Jacobienne $\mathbf{J}(\mathbf{x})$ correspondant à la dérivée du vecteur $\mathbf{y}(\mathbf{x})$ calculée au point $\mathbf{x} = \mathbf{0}$. En effet, ce Jacobien est relatif à l'état courant de la minimisation, c'est-à-dire avant la composition par l'incrément estimé. Cette appellation vient également du fait, comme nous le verrons par la suite, que ce Jacobien fait appel à l'image courante ayant subie la transformation projective planaire $\mathbf{w}(\widehat{\mathbf{G}})$. La ligne i du Jacobien $\mathbf{J}(\mathbf{0})$ s'écrit sous la forme suivante :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\mathbf{0}} = \nabla_{\mathbf{x}} \mathcal{I}\left(\mathbf{w}(\widehat{\mathbf{G}}\mathbf{G}(\mathbf{x}))(\mathbf{p}_i^*)\right)\Big|_{\mathbf{x}=\mathbf{0}} \tag{4.26}$$

Grâce à l'équation (2.29), nous pouvons écrire :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\mathbf{0}} = \nabla_{\mathbf{x}} \mathcal{I}\left(\mathbf{w}(\widehat{\mathbf{G}})(\mathbf{w}(\mathbf{G}(\mathbf{x}))(\mathbf{p}_i^*))\right)\Big|_{\mathbf{x}=\mathbf{0}} \tag{4.27}$$

La ligne i de ce Jacobien peut s'écrire sous la forme d'un produit de 3 Jacobiens :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\mathbf{0}} = \mathbf{J}_{\mathcal{I}}\, \mathbf{J}_{\mathbf{w}}\, \mathbf{J}_{\mathbf{G}} \tag{4.28}$$

1. Le Jacobien $\mathbf{J}_{\mathcal{I}}$ est une matrice de dimensions (1×3) qui correspond à la dérivée spatiale de l'image courante ayant subie la transformation projective planaire $\mathbf{w}(\widehat{\mathbf{G}})$. En effet, nous avons :

$$\mathbf{J}_{\mathcal{I}} = \nabla_{\mathbf{z}} \mathcal{I}\left(\mathbf{w}(\widehat{\mathbf{G}})(\mathbf{z})\right)\Big|_{\mathbf{z}=\mathbf{w}(\mathbf{G}(\mathbf{0}))(\mathbf{p}_i^*)} \tag{4.29}$$

et en utilisant l'équation (2.28), nous pouvons écrire :

$$\mathbf{J}_{\mathcal{I}} = \nabla_{\mathbf{z}} \mathcal{I} \left(\mathbf{w}(\widehat{\mathbf{G}})(\mathbf{z}) \right) \Big|_{\mathbf{z}=\mathbf{p}_i^*} \tag{4.30}$$

Le point $\mathbf{p}_i^* \in \mathbb{P}^2$ est de dimensions (3×1) avec la troisième composante constante et égale à 1. Nous ne considérons ici que les dérivées suivant les deux premières composantes et la troisième composante du vecteur ligne $\mathbf{J}_{\mathcal{I}}$ sera égale à zéro.

2. En utilisant la généralisation de l'opérateur gradient définie dans le paragraphe 2.1.1 du Chapitre 2, le Jacobien $\mathbf{J}_{\mathbf{w}}$ est une matrice de dimensions (3×9) qui correspond à la dérivée du vecteur $\mathbf{w}(\mathbf{Z})(\mathbf{p}_i^*)$ par rapport aux éléments de la matrice $\mathbf{Z}$:

$$\mathbf{J}_{\mathbf{w}} = \nabla_{\mathbf{Z}} \mathbf{w}(\mathbf{Z})(\mathbf{p}_i^*)|_{\mathbf{Z}=\mathbf{G}(\mathbf{0})=\mathbf{I}} \tag{4.31}$$

En posant $\mathbf{p}_i^* = [u_i^* \; v_i^* \; 1]$, l'expression explicite de ce Jacobien s'écrit :

$$\mathbf{J}_{\mathbf{w}} = \begin{bmatrix} \mathbf{p}_i^{*\top} & \mathbf{0} & -u_i^* \, \mathbf{p}_i^{*\top} \\ \mathbf{0} & \mathbf{p}_i^{*\top} & -v_i^* \, \mathbf{p}_i^{*\top} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{4.32}$$

3. Le troisième et dernier Jacobien $\mathbf{J}_{\mathbf{G}}$ est une matrice de dimensions (9×8) qui s'écrit :

$$\mathbf{J}_{\mathbf{G}} = \nabla_{\mathbf{x}} \mathbf{G}(\mathbf{x})|_{\mathbf{x}=\mathbf{0}} \tag{4.33}$$

Si nous utilisons l'équation (4.21) et la définition de l'application exponentielle matricielle de l'équation (4.22), l'expression explicite de ce Jacobien s'écrit :

$$\mathbf{J}_{\mathbf{G}} = \begin{bmatrix} [\mathbf{A}_1]_v & [\mathbf{A}_2]_v & \ldots & [\mathbf{A}_8]_v \end{bmatrix} \tag{4.34}$$

où la notation $[\mathbf{A}_k]_v$ correspond au vecteur obtenu en réarrangeant les éléments de la matrice $\mathbf{A}_k$ ligne par ligne.

Nous remarquons que les deux Jacobiens $\mathbf{J}_{\mathbf{w}}$ et $\mathbf{J}_{\mathbf{G}}$ sont constants. Donc, ils peuvent être calculés une fois pour toutes. Cependant, le Jacobien $\mathbf{J}_{\mathcal{I}}$ doit être calculé à chaque itération de l'algorithme de minimisation car il dépend de la transformation $\mathbf{w}(\widehat{\mathbf{G}})$.

4.3.2.2 Le Jacobien de référence

Étant donné que l'application exponentielle matricielle est un homéomorphisme entre un voisinage de la matrice identité $\mathbf{I} \in \mathbb{SL}(3)$ et un voisinage de la matrice nulle $\mathbf{0}$ de l'algèbre de Lie $\mathfrak{sl}(3)$, il existe un vecteur $\widetilde{\mathbf{x}}$ vérifiant :

$$\overline{\mathbf{G}}^{-1} \widehat{\mathbf{G}} = \mathbf{G}(-\widetilde{\mathbf{x}}) = \mathbf{G}(\widetilde{\mathbf{x}})^{-1} \tag{4.35}$$

Pour $\mathbf{x}$ vérifiant la contrainte (4.6), nous avons :

$$\overline{\mathbf{G}}^{-1}\widehat{\mathbf{G}}\mathbf{G}(\mathbf{x}) = \mathbf{G}(\widetilde{\mathbf{x}})^{-1}\mathbf{G}(\mathbf{x}) = \mathbf{I} \tag{4.36}$$

Par conséquent, pour $\mathbf{x}$ vérifiant la contrainte (4.6), à l'aide des équations (4.21) et (4.36), nous pouvons écrire :

$$\mathbf{x} = \widetilde{\mathbf{x}} \tag{4.37}$$

Grâce aux équations (2.29), (2.30) et à l'équation (4.1), $y_i(\mathbf{x})$ peut s'écrire sous la forme suivante :

$$y_i(\mathbf{x}) = \mathcal{I}^*\left(\mathbf{w}(\overline{\mathbf{G}}^{-1}\widehat{\mathbf{G}}\mathbf{G}(\mathbf{x}))(\mathbf{p}_i^*)\right) - \mathcal{I}^*(\mathbf{p}_i^*) \tag{4.38}$$

Nous appelons Jacobien de référence la matrice Jacobienne $\mathbf{J}(\mathbf{x})$ correspondant à la dérivée du vecteur $\mathbf{y}(\mathbf{x})$ calculée au point $\mathbf{x}$ vérifiant la contrainte (4.6). En effet, ce Jacobien est relatif à l'état de référence de la minimisation, c'est-à-dire l'état de la minimisation que nous désirons atteindre. Cette appellation vient également du fait que ce Jacobien fait appel à l'image de référence. À l'aide de l'équation (4.35), la ligne i de ce Jacobien peut s'écrire sous la forme suivante :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\widetilde{\mathbf{x}}} = \nabla_{\mathbf{x}}\mathcal{I}^*\left(\mathbf{w}(\mathbf{G}(\widetilde{\mathbf{x}})^{-1}\mathbf{G}(\mathbf{x}))(\mathbf{p}_i^*)\right)\big|_{\mathbf{x}=\widetilde{\mathbf{x}}} \tag{4.39}$$

La ligne i de ce Jacobien peut s'écrire sous la forme d'un produit de 3 Jacobiens :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\widetilde{\mathbf{x}}} = \mathbf{J}_{\mathcal{I}^*}\,\mathbf{J}_{\mathbf{w}_0}\,\mathbf{J}_{\widetilde{\mathbf{G}}} \tag{4.40}$$

1. Le Jacobien $\mathbf{J}_{\mathcal{I}^*}$ est une matrice de dimensions (1×3) qui correspond à la dérivée spatiale de l'image de référence. En effet, nous avons :

   $$\mathbf{J}_{\mathcal{I}^*} = \nabla_{\mathbf{z}}\mathcal{I}^*(\mathbf{z})|_{\mathbf{z}=\mathbf{w}(\mathbf{G}(\widetilde{\mathbf{x}})^{-1}\mathbf{G}(\widetilde{\mathbf{x}}))(\mathbf{p}_i^*)} \tag{4.41}$$

   et en utilisant l'équation (2.28), nous pouvons écrire :

   $$\mathbf{J}_{\mathcal{I}^*} = \nabla_{\mathbf{z}}\mathcal{I}^*(\mathbf{z})|_{\mathbf{z}=\mathbf{p}_i^*} \tag{4.42}$$

   D'une manière analogue à $\mathbf{J}_{\mathcal{I}}$, nous ne considérons ici que les dérivées suivant les deux premières composantes et la troisième composante du vecteur ligne $\mathbf{J}_{\mathcal{I}^*}$ sera égale à zéro.

2. Le Jacobien $\mathbf{J}_{\mathbf{w}_0}$ est une matrice de dimensions (3×9) qui s'écrit sous la forme :

   $$\mathbf{J}_{\mathbf{w}_0} = \nabla_{\mathbf{Z}}\mathbf{w}(\mathbf{Z})(\mathbf{p}_i^*)|_{\mathbf{Z}=\mathbf{G}(\widetilde{\mathbf{x}})^{-1}\mathbf{G}(\widetilde{\mathbf{x}})=\mathbf{I}} \tag{4.43}$$

   Nous remarquons que nous avons :

   $$\mathbf{J}_{\mathbf{w}_0} = \mathbf{J}_{\mathbf{w}} \tag{4.44}$$

3. Le troisième Jacobien $\mathbf{J}_{\widetilde{\mathbf{G}}}$ est une matrice de dimensions (9×8) qui s'écrit :

$$\mathbf{J}_{\widetilde{\mathbf{G}}} = \nabla_{\mathbf{x}} \mathbf{G}(\widetilde{\mathbf{x}})^{-1} \mathbf{G}(\mathbf{x})\big|_{\mathbf{x}=\widetilde{\mathbf{x}}} \tag{4.45}$$

Nous remarquons que les deux Jacobiens $\mathbf{J}_{\mathcal{I}^*}$ et $\mathbf{J}_{\mathbf{w}_0}$ sont constants. Donc, ils peuvent être calculés une fois pour toute. Cependant, l'expression explicite du Jacobien $\mathbf{J}_{\widetilde{\mathbf{G}}}$ n'est pas simple et dépend de l'inconnue $\widetilde{\mathbf{x}}$. Toutefois, en utilisant les propriétés des groupes de Lie nous avons démontré la propriété remarquable suivante :

$$\mathbf{J}_{\widetilde{\mathbf{G}}}\, \widetilde{\mathbf{x}} = \mathbf{J}_{\mathbf{G}}\, \widetilde{\mathbf{x}} \tag{4.46}$$

Cette formule nous permet de calculer l'approximation au deuxième ordre de manière analytique sans connaître $\widetilde{\mathbf{x}}$. La démonstration de cette formule est dans l'Annexe B.

4.3.3 Minimisation itérative du système d'équations

Dans l'approche que nous proposons, nous choisissons comme paramétrisation de la matrice de transformation projective planaire dans l'image $\mathbf{G}$ celle définie dans le paragraphe 4.3.1.

Dans l'approximation du développement en série de Taylor au second-ordre de la fonction vecteur $\mathbf{y}(\mathbf{x})$ au voisinage de $\mathbf{x} = \mathbf{0}$ de l'équation (4.9), nous rappelons que le calcul exact de la matrice $\mathbf{M}(\mathbf{0}, \mathbf{x})$ nécessite le calcul des q matrices Hessiennes de la fonction vecteur $\mathbf{y}(\mathbf{x})$. Or, pour avoir une approximation du développement en série de Taylor au second-ordre de la fonction vecteur $\mathbf{y}(\mathbf{x})$ au voisinage de $\mathbf{x} = \mathbf{0}$, il suffit d'avoir une approximation du développement en série de Taylor au premier-ordre de la matrice $\mathbf{M}(\mathbf{0}, \mathbf{x})$ au voisinage de $\mathbf{x} = \mathbf{0}$. Si nous utilisons l'approximation du développement en série de Taylor au premier-ordre du Jacobien $\mathbf{J}(\mathbf{x})$ au voisinage de $\mathbf{x} = \mathbf{0}$, nous pouvons écrire :

$$\mathbf{M}(\mathbf{0}, \mathbf{x}) = \mathbf{J}(\mathbf{x}) - \mathbf{J}(\mathbf{0}) + \mathbf{O}(\|\mathbf{x}\|^2) \tag{4.47}$$

L'équation (4.9) peut donc s'écrire sans avoir à calculer les matrices Hessiennes de $\mathbf{y}(\mathbf{x})$:

$$\mathbf{y}(\mathbf{x}) = \mathbf{y}(\mathbf{0}) + \frac{1}{2}\left(\mathbf{J}(\mathbf{0}) + \mathbf{J}(\mathbf{x})\right)\mathbf{x} + \mathbf{O}(\|\mathbf{x}\|^3) \tag{4.48}$$

Il s'agit d'une approximation au second-ordre de la fonction $\mathbf{y}(\mathbf{x})$ dans le voisinage de $\mathbf{x} = \mathbf{0}$. Quand nous avons $\mathbf{x} = \widetilde{\mathbf{x}}$, nous avons :

$$\mathbf{y}(\widetilde{\mathbf{x}}) \approx \mathbf{y}(\mathbf{0}) + \frac{1}{2}\left(\mathbf{J}(\mathbf{0}) + \mathbf{J}(\widetilde{\mathbf{x}})\right)\widetilde{\mathbf{x}} \tag{4.49}$$

Si nous utilisons l'expression de $\mathbf{J}(\mathbf{0})$ et de $\mathbf{J}(\widetilde{\mathbf{x}})$ du paragraphe 4.3.2, nous pouvons écrire :

$$\mathbf{J}(\mathbf{0}) + \mathbf{J}(\widetilde{\mathbf{x}}) = \mathbf{J}_{\mathcal{I}}\mathbf{J}_{\mathbf{w}}\mathbf{J}_{\mathbf{G}} + \mathbf{J}_{\mathcal{I}^*}\mathbf{J}_{\mathbf{w}}\mathbf{J}_{\widetilde{\mathbf{G}}} \tag{4.50}$$

Grâce à la formule de l'équation (4.46), l'équation (4.49) peut s'écrire :

$$\mathbf{y}(\widetilde{\mathbf{x}}) \approx \mathbf{y}(\mathbf{0}) + \frac{1}{2}\left(\mathbf{J}_{\mathcal{I}} + \mathbf{J}_{\mathcal{I}^*}\right)\ \mathbf{J}_{\mathbf{w}}\ \mathbf{J}_{\mathbf{G}}\ \widetilde{\mathbf{x}} \tag{4.51}$$

D'une manière analogue à celle du paragraphe 4.2, la solution du système (4.5) par la méthode des moindres carrés peut être obtenue d'une manière itérative. Nous définissons la matrice $\mathbf{J}_{esm}$ (mais s'il ne s'agit pas d'une matrice Jacobienne) :

$$\mathbf{J}_{esm} = \frac{1}{2}\left(\mathbf{J}_{\mathcal{I}} + \mathbf{J}_{\mathcal{I}^*}\right)\ \mathbf{J}_{\mathbf{w}}\ \mathbf{J}_{\mathbf{G}} \tag{4.52}$$

Il s'agit de minimiser d'une manière itérative la fonction de coût suivante :

$$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{y}(\mathbf{0}) + \mathbf{J}_{esm}\mathbf{x}\|^2 \tag{4.53}$$

La condition nécessaire (4.12) pour qu'un vecteur $\mathbf{x} = \widetilde{\mathbf{x}}$ soit un minimum local ou global de la fonction de coût f est que la dérivée de la fonction de coût soit nulle. Nous avons alors :

$$\nabla_{\mathbf{x}} f(\mathbf{x})|_{\mathbf{x}=\widetilde{\mathbf{x}}} = \mathbf{J}_{esm}^{\top}\left(\mathbf{y}(\mathbf{0}) + \mathbf{J}_{esm}\widetilde{\mathbf{x}}\right) = \mathbf{0} \tag{4.54}$$

Comme dans le paragraphe 4.2, nous procédons d'une manière itérative. D'abord nous calculons :

$$\widetilde{\mathbf{x}} = \left(\mathbf{J}_{esm}^{\top}\mathbf{J}_{esm}\right)^{-1}\mathbf{J}_{esm}^{\top}\mathbf{y}(\mathbf{0}) \tag{4.55}$$

ou plus génériquement :

$$\widetilde{\mathbf{x}} = \mathbf{J}_{esm}^{+}\mathbf{y}(\mathbf{0}) \tag{4.56}$$

Ensuite, nous mettons à jour l'approximation $\widehat{\mathbf{G}}$ comme défini dans l'équation (4.16). À chaque mise à jour de $\widehat{\mathbf{G}}$, nous estimons $\mathbf{y}(\mathbf{0})$ et $\mathbf{J}_{\mathcal{I}}$ et nous calculons à nouveau $\widetilde{\mathbf{x}}$. Nous arrêtons l'algorithme quand la valeur de $\widetilde{\mathbf{x}}$ calculée devient très faible.

4.3.4 Simulations avec des images synthétiques

L'avantage principal d'avoir une approximation de l'erreur au second-ordre par rapport aux paramètres de transformation est le taux de convergence élevé. Un autre avantage est l'évitement des minima locaux proches du minimum global, c'est-à-dire, quand l'approximation au second-ordre est quasiment vérifiée. Nous allons présenter ces deux avantages à l'aide de deux exemples. Nous comparons 6 méthodes différentes de minimisation itérative :

- la méthode de descente de gradient variable (équations (4.14), (4.17)) ;
- la méthode de descente de gradient constant (équations (4.14), (4.17) et (4.20)) ;
- la minimisation Gauss-Newton avec Jacobien variable (équations (4.14), (4.18)) ;
- la minimisation Gauss-Newton avec Jacobien constant (équations (4.14), (4.18) et (4.20)) ;

– la minimisation de Newton (équations (4.14), (4.15)) ;
– la méthode ESM (équation (4.55)).

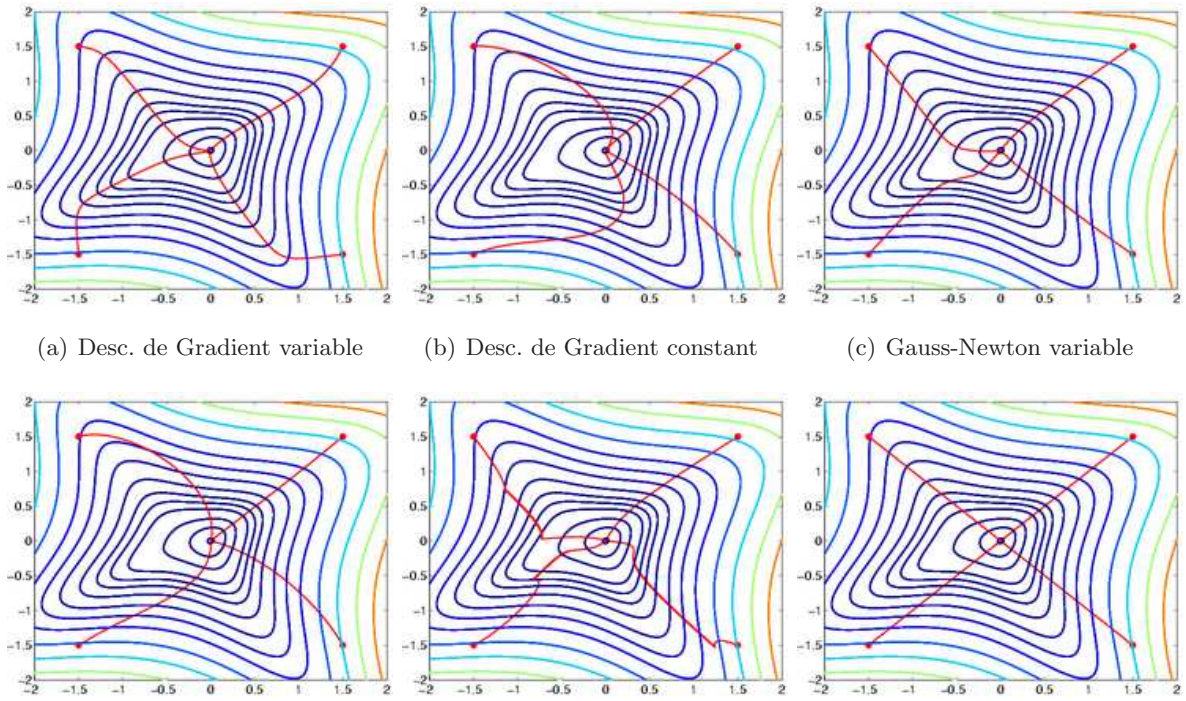
Dans les deux exemples, nous considérons un vecteur $\mathbf{y}(\mathbf{x})$ de dimensions (4×1) quadratique en fonction du vecteur $\mathbf{x}$ de dimensions (2×1) contenant les paramètres de la transformation. Pour chaque exemple et pour chaque méthode de minimisation, la simulation est effectuée 4 fois avec des points de départ différents : $\mathbf{x} \in \{(\pm 1.5, \pm 1.5)\}$. Nous traçons la valeur de la fonction sous forme de contours représentant des lignes de niveaux où la fonction de coût a la même valeur en chaque point. Nous représentons par des lignes rouges les trajectoires de la minimisation à partir de chaque point de départ. Évidemment, la trajectoire optimale recherchée (c'est-à-dire, la plus courte) est une ligne droite du point de départ vers le minimum de la fonction de coût.

4.3.4.1 Simulation sans minima locaux

Dans le premier exemple, la fonction quadratique $\mathbf{y}(\mathbf{x})$ n'a pas de minimum local proche du minimum global. Le résultat des six méthodes de minimisation est donné dans la Figure 4.1. La Figure 4.1(a) montre que, en tout point, les trajectoires de la minimisation avec la méthode de descente de gradient variable sont toujours perpendiculaires aux lignes de niveaux. C'est pour cette raison que la convergence est lente et que le minimum ne peut être atteint en une ligne droite. Les trajectoires de la minimisation avec la méthode de descente de gradient constant sont encore plus longues (voir les trajectoires de la Figure 4.1(b)). La minimisation de Gauss-Newton avec Jacobien constant (Figure 4.1(d)) et variable (Figure 4.1(c)) ont respectivement de meilleurs résultats que ceux de la descente de gradient constant et variable. En effet, dans la minimisation de Gauss-Newton constante et variable une meilleurs approximation est utilisée. En ce qui concerne la méthode de Newton, nous remarquons qu'un mauvais conditionnement des matrices Hessiennes est la cause d'oscillations de la trajectoire de la minimisation (voir Figure 4.1(e)). Finalement, la méthode ESM donne le meilleur résultat car les trajectoires de la Figure 4.1(f) sont des lignes droites. En effet, quand la fonction $\mathbf{y}(\mathbf{x})$ est réellement quadratique, il est possible d'estimer la vraie transformation en une seule itération et par conséquent la bonne direction de descente, quelle que soit la forme des lignes de niveaux.

4.3.4.2 Simulation avec un minimum local

Dans le deuxième exemple, la fonction quadratique $\mathbf{y}(\mathbf{x})$ a un minimum local proche du minimum global. La méthode de Newton et les minimisations avec des Jacobiens variables convergent vers le minimum local quand le point de départ de la minimisation est proche de ce minimum local (voir les Figures 4.2(a), 4.2(c) et 4.2(e)). Dans ces mêmes cas, les méthodes avec des Jacobiens peuvent même diverger (voir les Figures 4.2(b) et 4.2(d)). En effet, l'approximation du Jacobien constant n'est vérifiée que dans un voisinage proche du minimum global. Cependant, la méthode ESM suit la trajectoire la plus courte (voir la Figure 4.2(f)). En effet, si $\mathbf{y}(\mathbf{x})$ est localement quadratique la méthode ESM est capable d'éviter les minima locaux. Bien



(a) Desc. de Gradient variable (b) Desc. de Gradient constant (c) Gauss-Newton variable

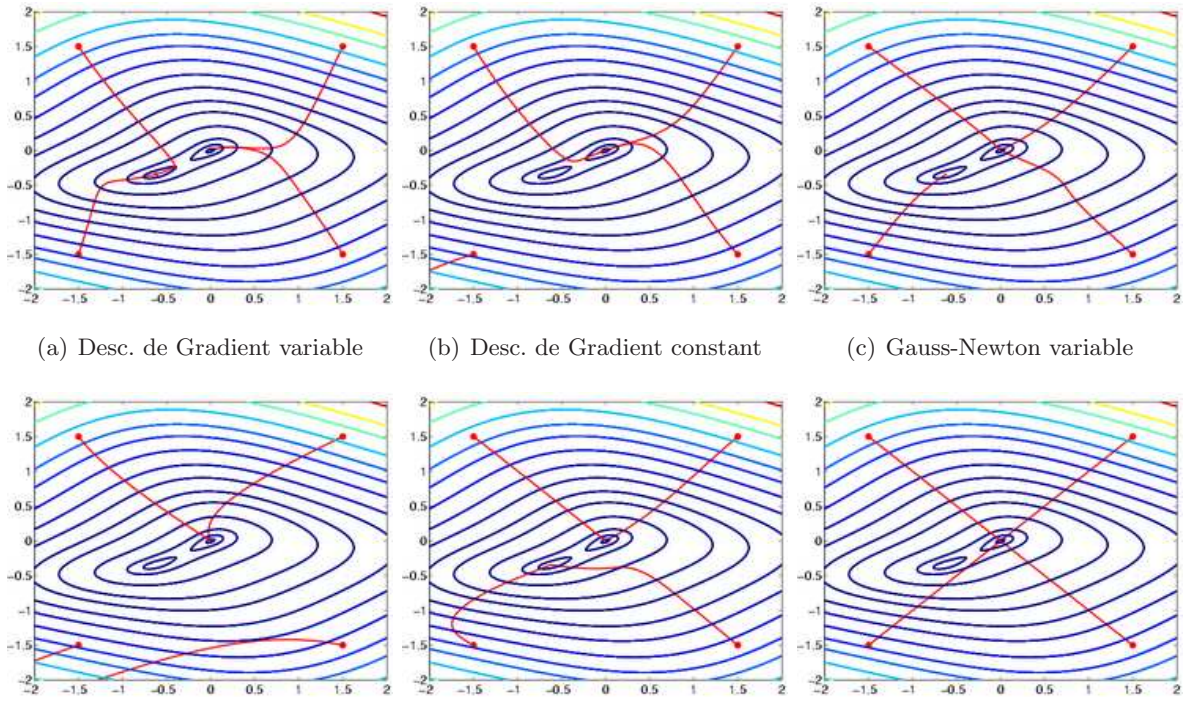
(d) Gauss-Newton constant (e) Newton (f) ESM

Figure 4.1 – Comparaison de 6 méthodes de minimisation sans minimum local

évidemment, si le minimum local est loin du minimum global, l'approximation au second-ordre n'est plus vérifiée (sauf si $\mathbf{y}(\mathbf{x})$ est globalement quadratique).

4.3.5 Simulations avec des images réelles

Afin d'avoir une preuve par "vérité terrain", le résultat de la méthode ESM est comparé, en simulations Matlab, à celui de deux autres méthodes sur une image statique ayant subie des transformations projectives planaires connues. Nous comparons la méthode ESM avec la minimisation de Gauss-Newton avec Jacobien constant (CGN) proposé dans (Baker and Matthews, 2001) et avec la minimisation de Gauss-Newton avec Jacobien variable (VGN) proposée dans (Shum and Szeliski, 2000). Afin d'avoir la même plate-forme de simulation, nous utilisons la librairie Matlab disponible sur la page Web de Simon Baker au laboratoire Robotics Institute de Carnegie Mellon University (http ://www.ri.cmu.edu/people/baker_simon.html). Les simulations ont été effectuées sur une image de la séquence du port d'Antibes (voir Figure 4.3(a)). Nous sélectionnons une région au centre de l'image de dimensions (124×124) que nous supposons vouloir suivre (voir Figure 4.3(b)). Nous appliquons à l'imagette différentes transformations projectives planaires aléatoires. D'une manière similaire à (Baker and Matthews, 2001), chaque transformation projective est calculée en rajoutant un bruit Gaussien aux coordonnées des 4 coins de l'imagette. Le quadrilatère obtenu sert à calculer la transformation projective planaire



(a) Desc. de Gradient variable (b) Desc. de Gradient constant (c) Gauss-Newton variable

(d) Gauss-Newton constant (e) Newton (f) ESM

Figure 4.2 – Comparaison de 6 méthodes de minimisation avec minimum local

initiale. L'écart type σ du bruit Gaussien varie de 1 à 12 avec des pas de 0.5. Il représente en quelque sorte l'amplitude du déplacement. Pour chaque valeur de σ, nous appliquons à l'imagette 1000 transformations projectives planaires aléatoires. Dans la Figure 4.3(c), nous pouvons voir un exemple de position initiale avec un écart type $\sigma = 12$ et dans la Figure 4.3(d), nous pouvons voir l'imagette initiale correspondante.

4.3.5.1 Comparaison des fréquences de convergence

Pour chaque algorithme testé, nous effectuons 15 itérations. Nous considérons un algorithme comme ayant convergé, si la moyenne de la norme de l'erreur des coordonnées des 4 coins à l'issue des 15 itérations est inférieure à 1 pixel. En d'autres termes, si nous notons par $\boldsymbol{\varepsilon}_i = [\varepsilon_{ui},\ \varepsilon_{vi}]^\top$ la différence des coordonnées (en pixels) du coin i entre sa position après application de la vraie transformation et sa position après application de la transformation estimée, nous considérons un algorithme comme ayant convergé si nous avons :

$$\frac{1}{8}\sum_{i=1}^{4} \varepsilon_{xi}^2 + \varepsilon_{yi}^2 < 1$$

(a) Position de référence de l'imagette (b) L'imagette de référence

(c) Exemple de position initiale avec $\sigma = 12$ (d) Exemple d'imagette initiale avec $\sigma = 12$

Figure 4.3 – L'imagette de référence

Dans la Figure 4.4, nous traçons les fréquences de convergence (% sur 1000 tests) en fonction de l'écart type σ. Pour de faibles déplacements (jusqu'à $\sigma = 4$), nous pouvons voir que les 3 algorithmes ont des fréquences de convergence équivalentes. En effet, les 3 algorithmes convergent quasiment tout le temps. Ceci n'est pas surprenant car pour de faibles déplacements, l'approximation au premier-ordre est vérifiée. Par conséquent, l'algorithme ESM proposé est équivalent à un algorithme du premier-ordre. Le Jacobien courant et le Jacobien de référence sont très proches et nous avons :

$$\mathbf{J}_{esm} = \frac{1}{2}(\mathbf{J}(\mathbf{0}) + \mathbf{J}(\widetilde{\mathbf{x}})) \approx \mathbf{J}(\mathbf{0}) \approx \mathbf{J}(\widetilde{\mathbf{x}})$$

Mais plus σ est grand, plus la transformation entre l'imagette de référence et l'imagette initiale est importante et plus il est difficile de minimiser la norme de l'erreur entre les deux imagettes. C'est pour cette raison que les fréquences de convergence des 3 algorithmes diminuent quand σ augmente. Cependant, nous remarquons que l'approximation de second-ordre obtenue avec la méthode ESM permet à la fréquence de convergence de cet algorithme de décroître moins vite que celle des deux algorithmes CGN et VGN.

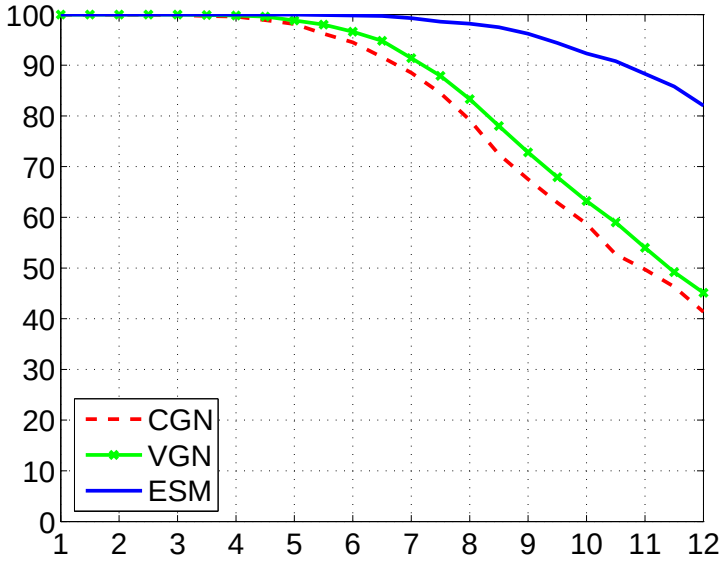


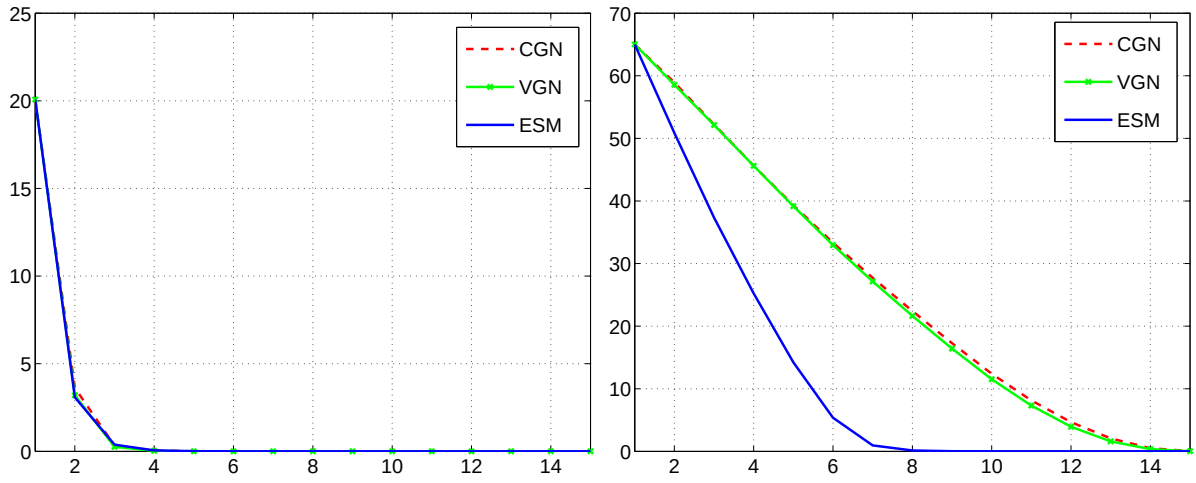
Figure 4.4 – Comparaison des fréquences de convergence

En effet, plus σ augmente, moins l'approximation au premier-ordre est vérifiée. Tandis que l'approximation au second-ordre reste valable pour de grands σ (évidemment jusqu'à un certain point, puis elle devient non valable). Pour $\sigma = 10$, alors que la fréquence de convergence de l'algorithme ESM est plus de 90% celle des deux autres algorithmes est déjà à 60%. Pour $\sigma = 12$, les fréquences de convergence des deux algorithmes CGN et VGN sont du même ordre autour de 45% (avec une légère supériorité du VGN), tandis que la fréquence de convergence de l'algorithme ESM dépasse les 80%. Nous pouvons donc conclure que pour de grands déplacements l'algorithme ESM converge 2 fois plus souvent que les deux autres algorithmes.

4.3.5.2 Comparaison des taux de convergence

Nous comparons le taux de convergence moyen (sur 1000 tests) des différents algorithmes. Nous ne considérons que les tests où les 3 algorithmes ont convergé afin de ne pas biaiser le résultat par des cas où un ou plusieurs algorithmes ont divergé. Cette comparaison nous permet de voir le nombre d'itérations nécessaire à chacun des algorithmes pour converger et ce pour différentes amplitudes de déplacement. Elle nous permet également d'observer la décroissance de la norme de l'erreur entre l'imagette de référence et l'imagette courante.

Dans la Figure 4.5, nous traçons la norme de l'erreur entre l'imagette de référence et l'imagette courante. Partant d'une norme d'erreur de même valeur, nous observons la décroissance de cette norme pour différentes amplitudes de déplacement. Nous remarquons que pour de faibles déplacements (par exemple, pour $\sigma = 1$), les 3 algorithmes sont équivalents. En moyenne, seulement 4 itérations sont nécessaires pour que les 3 algorithmes convergent. Plus σ augmente, plus l'erreur initiale augmente et plus le nombre d'itérations nécessaire à la convergence est important. Cependant, pour de grands σ, nous remarquons que la décroissance de la norme de l'erreur de la méthode ESM est plus rapide que les deux algorithmes CGN et VGN. En effet, pour $\sigma = 12$, quand, pour converger, la méthode ESM nécessite 8 itérations, les deux autres méthodes nécessitent plus de 14 itérations.



(a) $\sigma = 1$ (b) $\sigma = 12$

Figure 4.5 – Comparaison des taux de convergence

4.3.5.3 Comparaison des résidus moyens

Quand le SV est effectué en temps réel, le nombre d'itérations est fixé par la fréquence d'acquisition. Il peut arriver que la minimisation effectuée lors de l'estimation du déplacement ne soit pas achevée et que le minimum global de la norme de l'erreur entre la l'imagette de

référence et l'imagette courante ne soit pas atteint. Dans ce genre de cas, l'erreur résiduelle perturbe le SV et nous observons alors que les erreurs d'estimation s'additionnent et qu'une erreur de traînage apparaît lors du SV. Dans la Figure 4.6, nous traçons, pour différentes valeurs de σ, les résidus moyens dans le cas où les algorithmes n'ont pas divergé (nous considérons qu'un algorithme diverge quand l'erreur finale est plus grande que l'erreur initiale). Nous remarquons que pour les trois algorithmes, le résidu moyen augmente quand l'amplitude du déplacement initial grandit. Pour toutes les valeur de σ testées, le résidu moyen de la méthode ESM reste inférieur à celui de la méthode CGN et à celui de la méthode VGN. Il est même deux fois moins important dans le cas de grands déplacements initiaux ($\sigma = 12$).

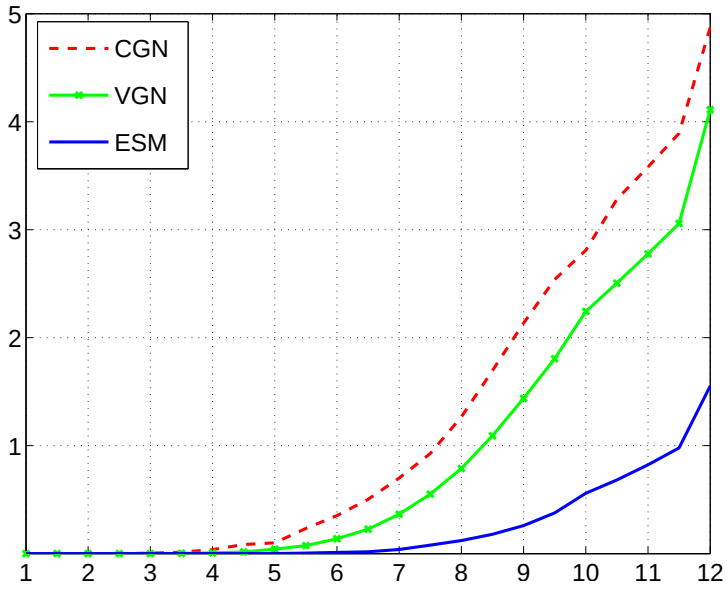


Figure 4.6 – Comparaison des résidus moyens

4.3.5.4 Comparaison de la robustesse au sous échantillonnage

Le temps de calcul de la méthode ESM est équivalent à celui de la méthode VGN et il est supérieur à celui de la méthode CGN. Afin d'avoir le même temps d'exécution par itération, nous pouvons effectuer un sous échantillonnage, c'est-à-dire, au lieu d'utiliser tous les pixels de l'imagette pour estimer le mouvement, n'en utiliser qu'une partie. Le sous échantillonnage réduit la performance des algorithmes : généralement plus nous sous échantillonnons, moins les algorithmes de SV convergent. Dans la Figure 4.7, nous traçons la fréquence de convergence des 3 algorithmes, dans le cas de grands déplacements ($\sigma = 12$), en fonction du taux d'échantillonnage r_e, par exemple, quand $r_e = 1/1$, toute l'imagette est utilisée et quand $r_e = 1/10$, nous utilisons 1% des pixels de l'imagette (1 pixel sur 10 par ligne et par colonne). Nous remarquons que, parmi les 3 algorithmes testés, la méthode ESM est la plus robuste au sous échantillonnage. Pour $r_e = 1/7$, sa fréquence de convergence est meilleure que les deux autres algorithmes sans

échantillonnage. Par conséquent, nous pouvons obtenir de meilleurs résultats avec un temps de calcul plus faible.

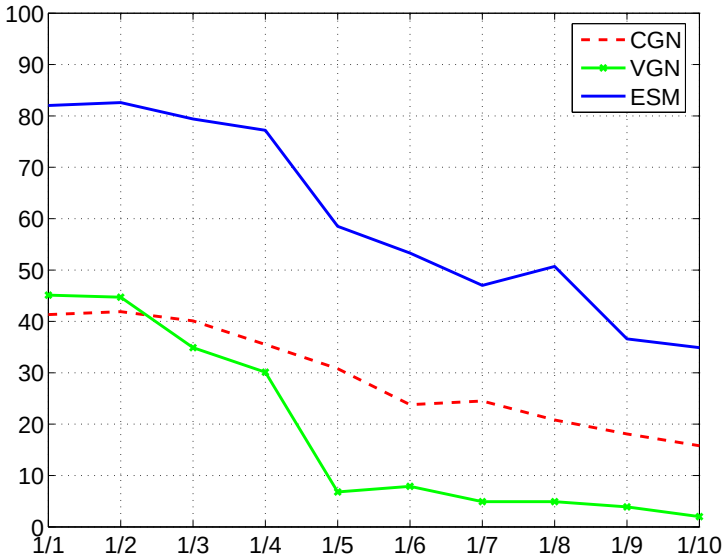
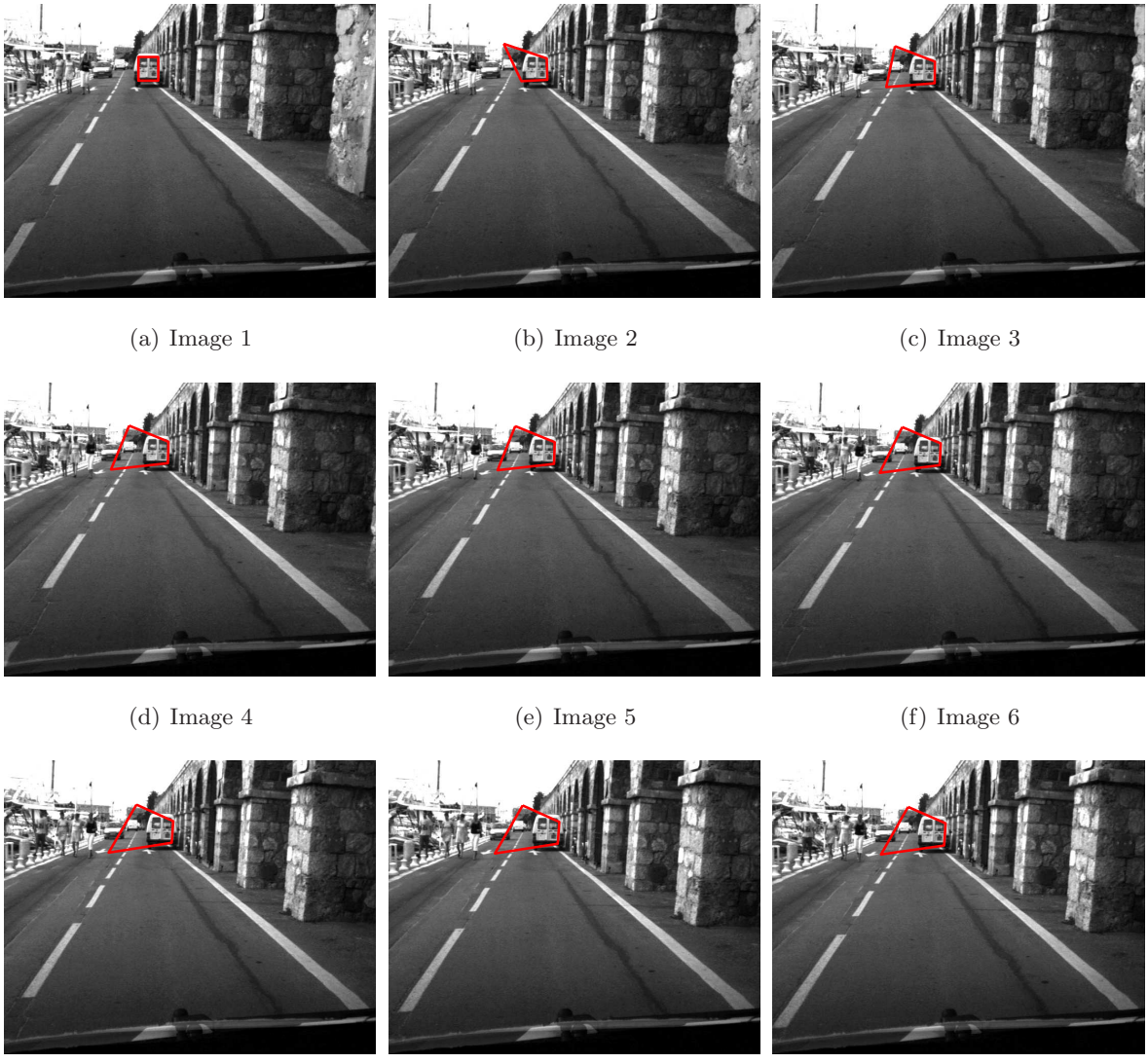


Figure 4.7 – Comparaison de la robustesse au sous-échantillonnage

4.3.5.5 Comparaison sur un extrait d'une séquence réelle

Nous comparons les trois algorithmes sur 9 images de la séquence du port d'Antibes. Nous suivons l'arrière d'un véhicule stationné sur le bord de la route avec une caméra montée sur un deuxième véhicule en mouvement. Nous appliquons les 3 algorithmes CGN, VGN et ESM. Nous sélectionnons une petite imagette de dimensions (40×44) dans la première image. L'algorithme CGN est rapide parce qu'il utilise un Jacobien constant. Donc, pour être équitable, nous lui permettons d'effectuer 50 itérations à chaque nouvelle image. Malgré cela, cet algorithme est incapable d'estimer correctement le déplacement de la cible dans l'image le long de la séquence (voir la Figure 4.8). Dès la deuxième image de la séquence, cet algorithme perd la cible et la reprojection de l'imagette courante est différente de la première imagette : l'imagette de référence sélectionnée (voir la Figure 4.9). Le déplacement entre la première et la deuxième image ne semble pas très important, mais il est assez conséquent car l'imagette sélectionnée est de petite taille. Donc, l'imagette de référence et l'imagette courante ne se superposent pas assez et l'approximation au premier ordre n'est plus vérifiée. L'algorithme VGN étant moins rapide car le Jacobien doit être calculé et inversé à chaque itération. Donc, pour le suivi, nous ne lui permettons d'effectuer que 7 itérations à chaque nouvelle image. Mais, cela s'avère insuffisant pour estimer correctement le déplacement de la cible dans l'image le long de la séquence (voir la Figure 4.10). Dès la deuxième ou la troisième image de la séquence, cet algorithme perd également la cible et la reprojection de l'imagette courante est différente de l'imagette de réfé-

rence (voir la Figure 4.11). Grâce au fait d'être au second-ordre, et malgré le fait d'avoir une complexité équivalente au VGN, l'algorithme ESM ne nécessite que 5 itérations pour effectuer correctement le suivi le long de la séquence parce que cet algorithme a une zone de convergence plus grande que les deux autres approches (voir la Figure 4.12 pour les images et la position de la cible). La Figure 4.13 permet de vérifier que le suivi visuel s'est bien déroulé le long de la séquence puisque la reprojection des imagettes courantes est identique à celle de l'imagette de référence.



(a) Image 1 (b) Image 2 (c) Image 3

(d) Image 4 (e) Image 5 (f) Image 6

(g) Image 7 (h) Image 8 (i) Image 9

Figure 4.8 – SV avec CGN

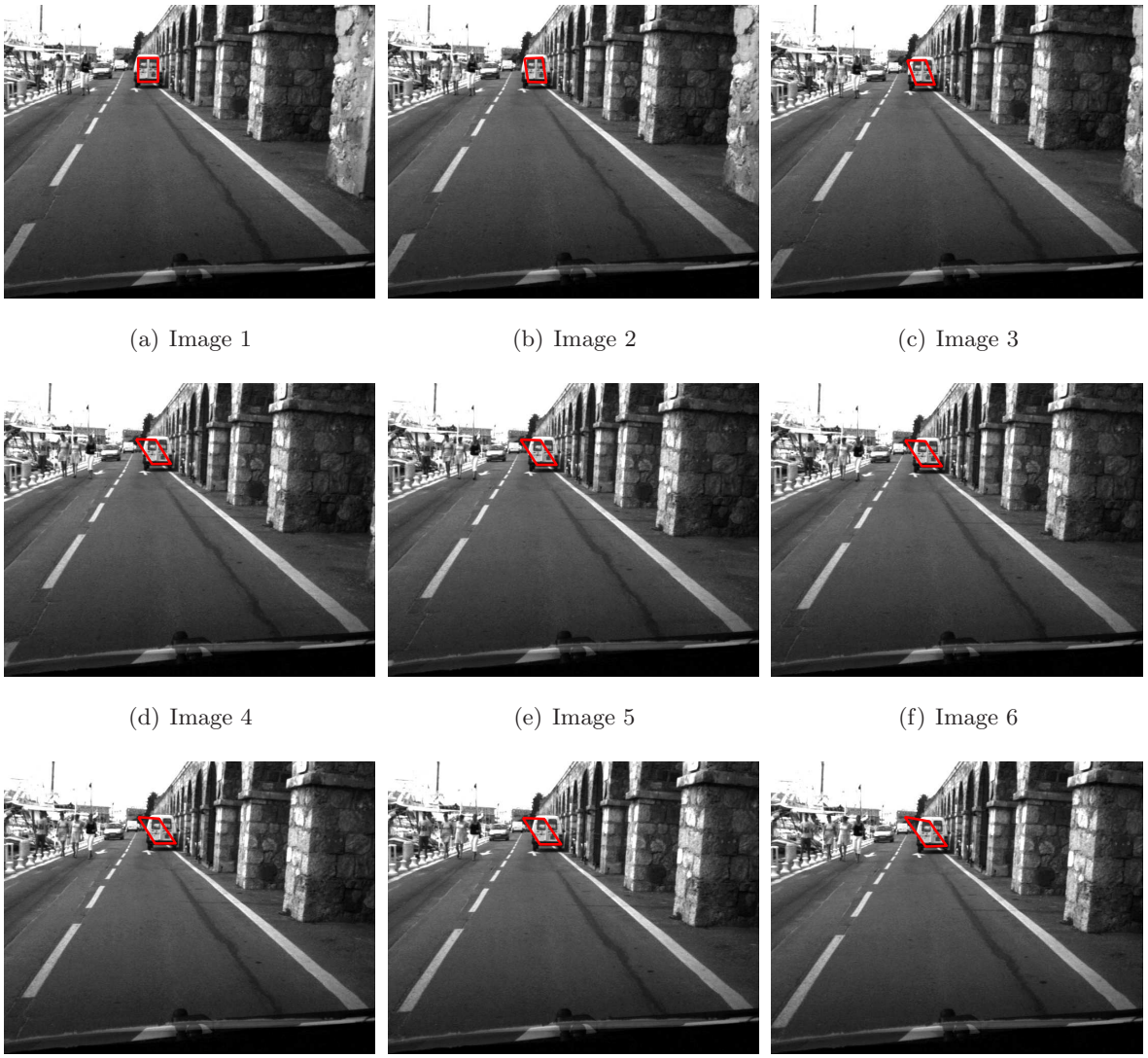
Les résultats obtenus dans cette section ont été confirmé par des dizaines d'autres exemples d'images et de séquences et les tests effectués ont abouti à des résultats très similaires.

(a) Imagette 1 (b) Imagette 2 (c) Imagette 3

(d) Imagette 4 (e) Imagette 5 (f) Imagette 6

(g) Imagette 7 (h) Imagette 8 (i) Imagette 9

Figure 4.9 – SV avec CGN : la reprojection des imagettes

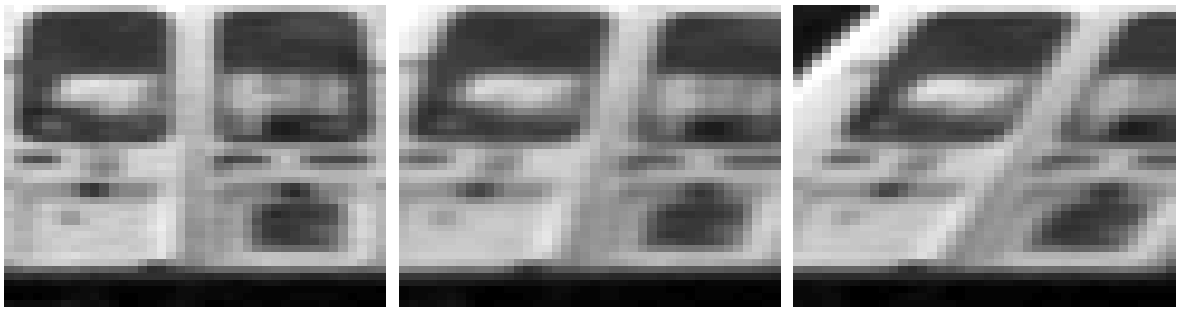


(a) Image 1 (b) Image 2 (c) Image 3

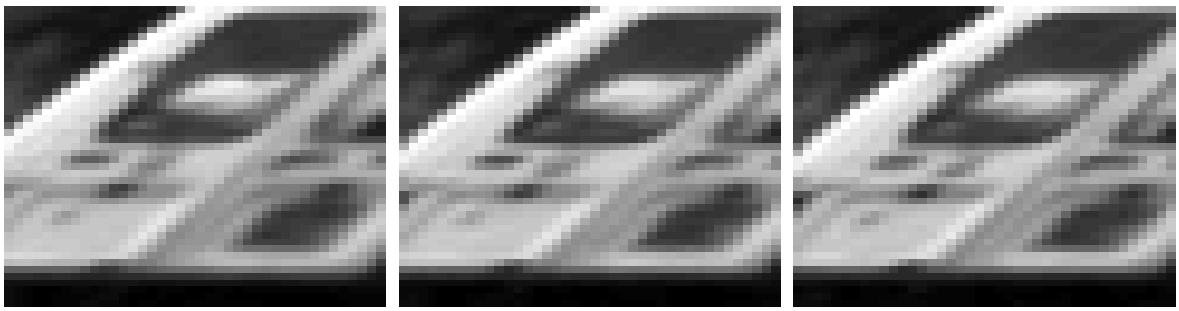
(d) Image 4 (e) Image 5 (f) Image 6

(g) Image 7 (h) Image 8 (i) Image 9

Figure 4.10 – SV avec VGN



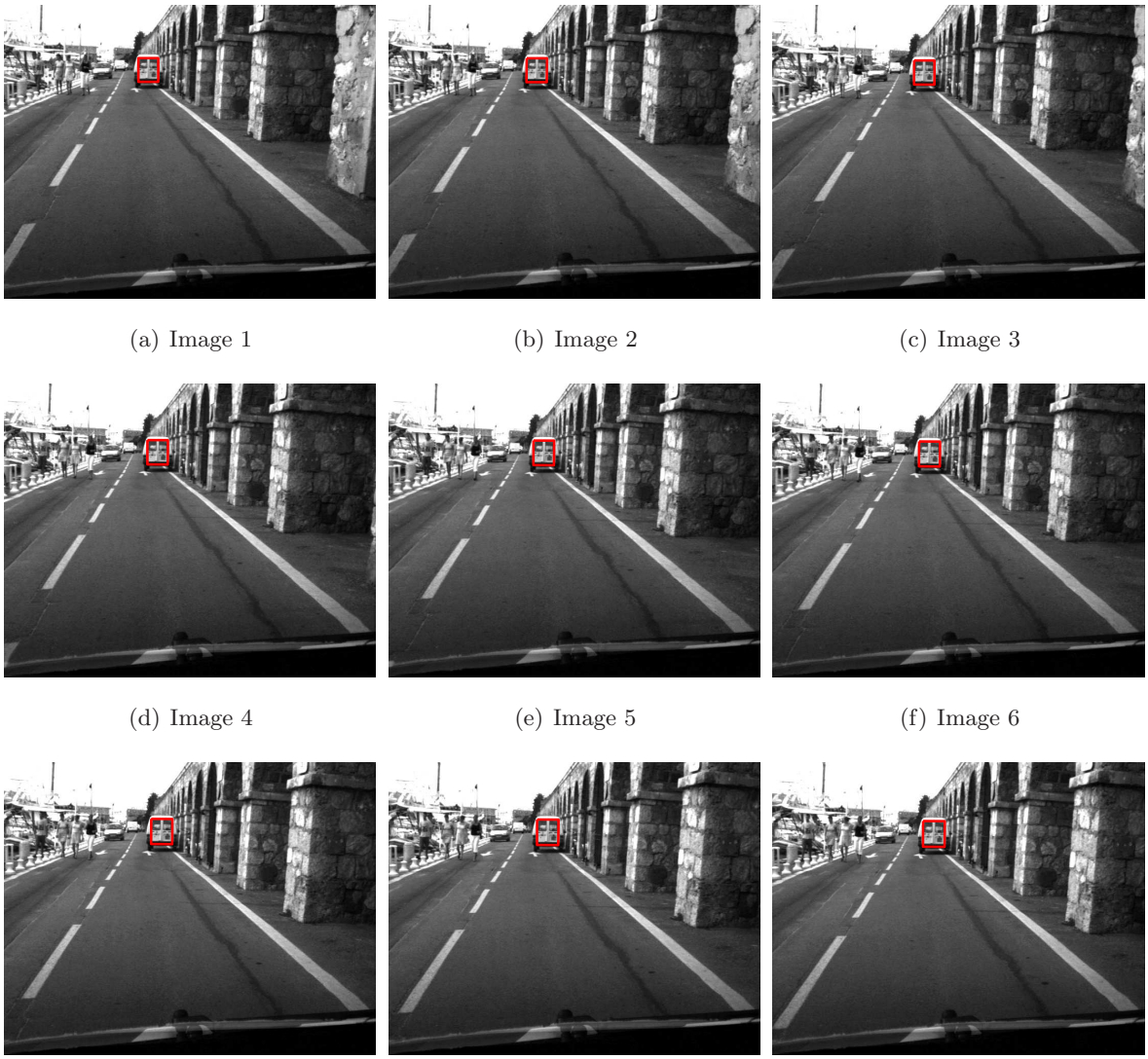
(a) Imagette 1 (b) Imagette 2 (c) Imagette 3



(d) Imagette 4 (e) Imagette 5 (f) Imagette 6

(g) Imagette 7 (h) Imagette 8 (i) Imagette 9

Figure 4.11 – SV avec VGN : la reprojection des imagettes

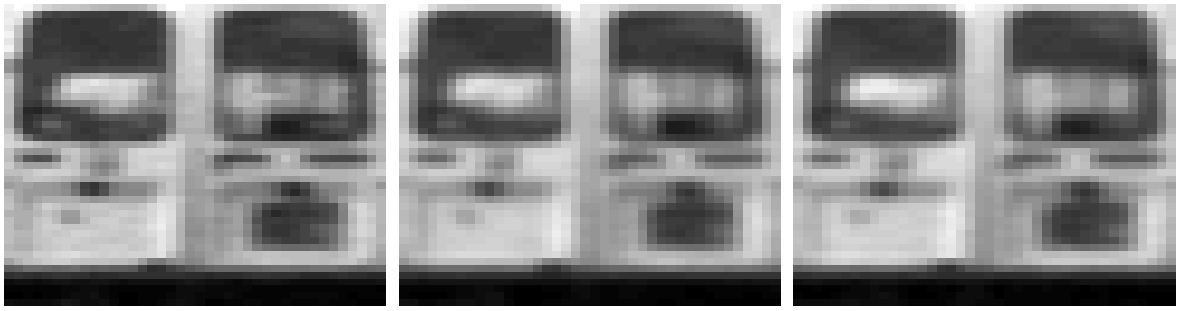


(a) Image 1 (b) Image 2 (c) Image 3

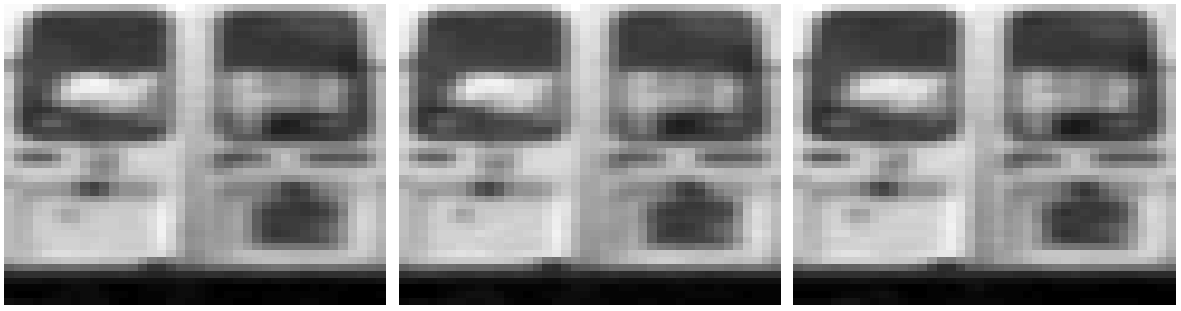
(d) Image 4 (e) Image 5 (f) Image 6

(g) Image 7 (h) Image 8 (i) Image 9

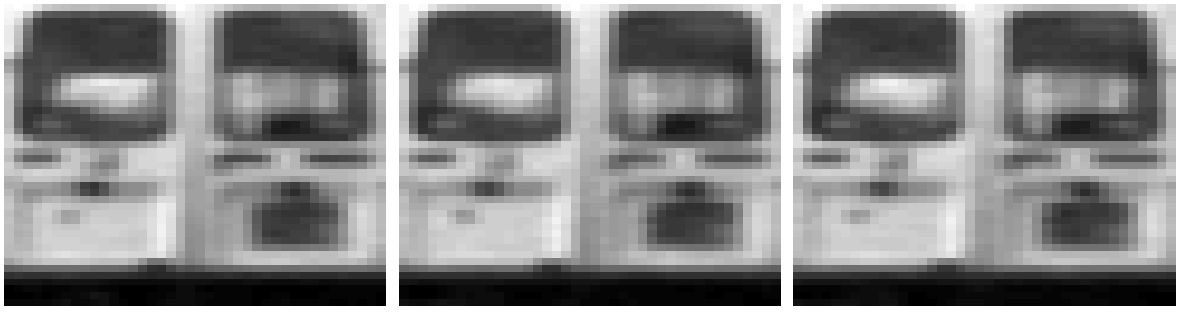
Figure 4.12 – SV avec ESM



(a) Imagette 1 (b) Imagette 2 (c) Imagette 3



(d) Imagette 4 (e) Imagette 5 (f) Imagette 6



(g) Imagette 7 (h) Imagette 8 (i) Imagette 9

Figure 4.13 – SV avec ESM : la reprojection des imagettes

4.3.6 Améliorations de l'algorithme

Nous avons présenté dans le paragraphe 4.3.3, l'algorithme ESM "brut", c'est-à-dire, la résolution du système d'équations (4.5) est effectué, en utilisant toutes les équations sans les manipuler. L'estimation de la solution se fait sans aucune prédiction. D'ores et déjà, les simulations présentées dans les paragraphes 4.3.4 et 4.3.5, nous permettent de conclure que, nous obtenons de meilleurs résultats avec l'algorithme ESM "brut" comparé à des méthodes classiques. Néanmoins, il est possible d'améliorer encore, non seulement, les résultats du SV, mais aussi, le temps de calcul de l'algorithme. Dans ce paragraphe, nous présentons quelques améliorations possibles de la méthode permettant de réaliser un SV rapide et efficace dans des applications temps-réel sans avoir recours à du matériel dédié (processeurs puissants, module hardware de traitement d'images bas niveau...).

4.3.6.1 Utilisation de la pyramide d'images

Une réduction du nombre de pixels utilisés dans l'image pour effectuer la minimisation lors du SV permet de réduire le temps de calcul de l'algorithme. Comme nous l'avons vu dans le paragraphe 4.3.5.4, la méthode ESM est robuste au sous échantillonnage. En effet, avec un sous échantillonnage à 6.25% (1 pixel sur 4 par ligne et par colonne), la fréquence de convergence du SV reste élevée de l'ordre de 65% pour les grands déplacements (sachant que sans sous échantillonnage la fréquence de convergence du SV est de l'ordre de 72% pour le même type de déplacements). Nous proposons d'utiliser une méthode inspirée de (Bouguet, 1999), où le SV s'effectue d'une manière pyramidale, c'est-à-dire, de la résolution la plus basse (où le nombre d'échantillons est faible) à la résolution la plus élevée (où tous les pixels de l'imagette sont utilisés). Cette stratégie permet d'avoir une estimation de la transformation projective de moins en moins grossière (ou de plus en plus fine) car plus nous sous échantillonnons, moins l'estimation est précise. Elle est connue sous le nom de "coarse-to-fine strategy". Cependant, contrairement à (Bouguet, 1999), au lieu de construire la pyramide d'images à l'aide d'un sous échantillonnage suivi d'un lissage par un filtre Gaussien, nous proposons d'appliquer à l'imagette des transformations projectives planaires suivie d'une interpolation bilinéaire (voir le paragraphe 2.3.4).

Un changement de résolution de l'image peut être obtenu à l'aide d'une transformation affine (qui n'est autre qu'un cas particulier de la transformation projective planaire). Soit ${}^{i}\mathbf{S}_{j}$ la transformation projective permettant de passer de la résolution j à la résolution i. Quand l'imagette de référence est sélectionnée dans l'image de référence en pleine résolution, nous lui appliquons n transformations projectives ${}^{1}\mathbf{S}_{0}$, ${}^{2}\mathbf{S}_{0}$, ..., ${}^{n}\mathbf{S}_{0}$ jusqu'à atteindre une résolution minimale. Par exemple, la résolution minimale (la résolution n) peut être la plus petite résolution où l'imagette a au moins 500 pixels. Nous obtenons, une fois pour toutes, une pyramide de $n+1$ imagettes de référence. Si la transformation projective permettant de passer de l'imagette de référence à l'imagette courante à la résolution j est ${}^{j}\mathbf{G}$, alors la transformation projective

correspondante à la résolution i s'écrit :

$$^{i}\mathbf{G} = {}^{i}\mathbf{S}_{j}\ {}^{j}\mathbf{G} \tag{4.57}$$

Nous avons illustré l'approche multirésolution de la méthode ESM dans la Figure 4.14. Le SV commence à l'échelle n en utilisant une estimation initiale de la matrice de transformation projective $^{n}\mathbf{G}$ (s'il n'y a pas de prédiction, nous utilisons comme estimation la matrice $^{n}\mathbf{G}$ égale à la matrice $^{n}\mathbf{S}_{0}$). Une fois la matrice $^{n}\mathbf{G}$ mise à jour par la méthode ESM, nous obtenons la transformation projective $^{n-1}\mathbf{G}$ en changeant d'échelle :

$$^{n-1}\mathbf{G} = {}^{n-1}\mathbf{S}_{n}\ {}^{n}\mathbf{G} \tag{4.58}$$

La minimisation est effectuée avec un certain nombre d'itérations et à l'échelle 0, nous obtenons la matrice de transformation projective $^{0}\mathbf{G}$. La boucle se répète après avoir calculé la transformation projective de l'échelle la plus élevée n :

$$^{n}\mathbf{G} = {}^{n}\mathbf{S}_{0}\ {}^{0}\mathbf{G} \tag{4.59}$$

Nous pouvons également arrêter l'algorithme simplement à l'échelle $k > 0$ de la pyramide (par exemple, si le temps de calcul est limité) en calculant la matrice de transformation projective à l'échelle la plus basse (correspondant à la résolution originale) :

$$^{0}\mathbf{G} = {}^{0}\mathbf{S}_{k}\ {}^{k}\mathbf{G} \tag{4.60}$$

4.3.6.2 Utilisation de la sélection de pixels

Afin d'accélérer l'algorithme de SV, au lieu d'utiliser tous les pixels de l'imagette, nous n'en utilisons qu'un sous-ensemble car en réduisant la taille des imagettes suivies, nous réduisons le coût en temps de calcul de chaque itération. Ce sous-ensemble peut être réparti d'une manière uniforme dans l'imagette. Par exemple, avec l'approche pyramidale, nous pouvons arrêter l'algorithme ESM à une échelle $k > 0$. Le choix peut être effectué d'une manière non uniforme en utilisant la norme des gradients des pixels. En effet, les pixels ayant un faible gradient apportent moins d'information que les pixels ayant un fort gradient. En effet, la ligne des Jacobiens correspondant à des pixels appartenant à une zone uniforme (où le gradient est nul) est nulle. Il est plus intéressant de ne considérer que les lignes non nulles des Jacobiens car, ainsi, nous gagnons du temps dans le calcul des Jacobiens et dans le calcul de la pseudo-inverse.

La manière la plus simple de sélectionner les pixels est d'utiliser la norme des gradients dans l'imagette de référence. Nous ne choisissons pas des pixels correspondant à des primitives visuelles particulières (par exemple, à l'aide d'extracteurs de points d'intérêt (Harris and Ste-

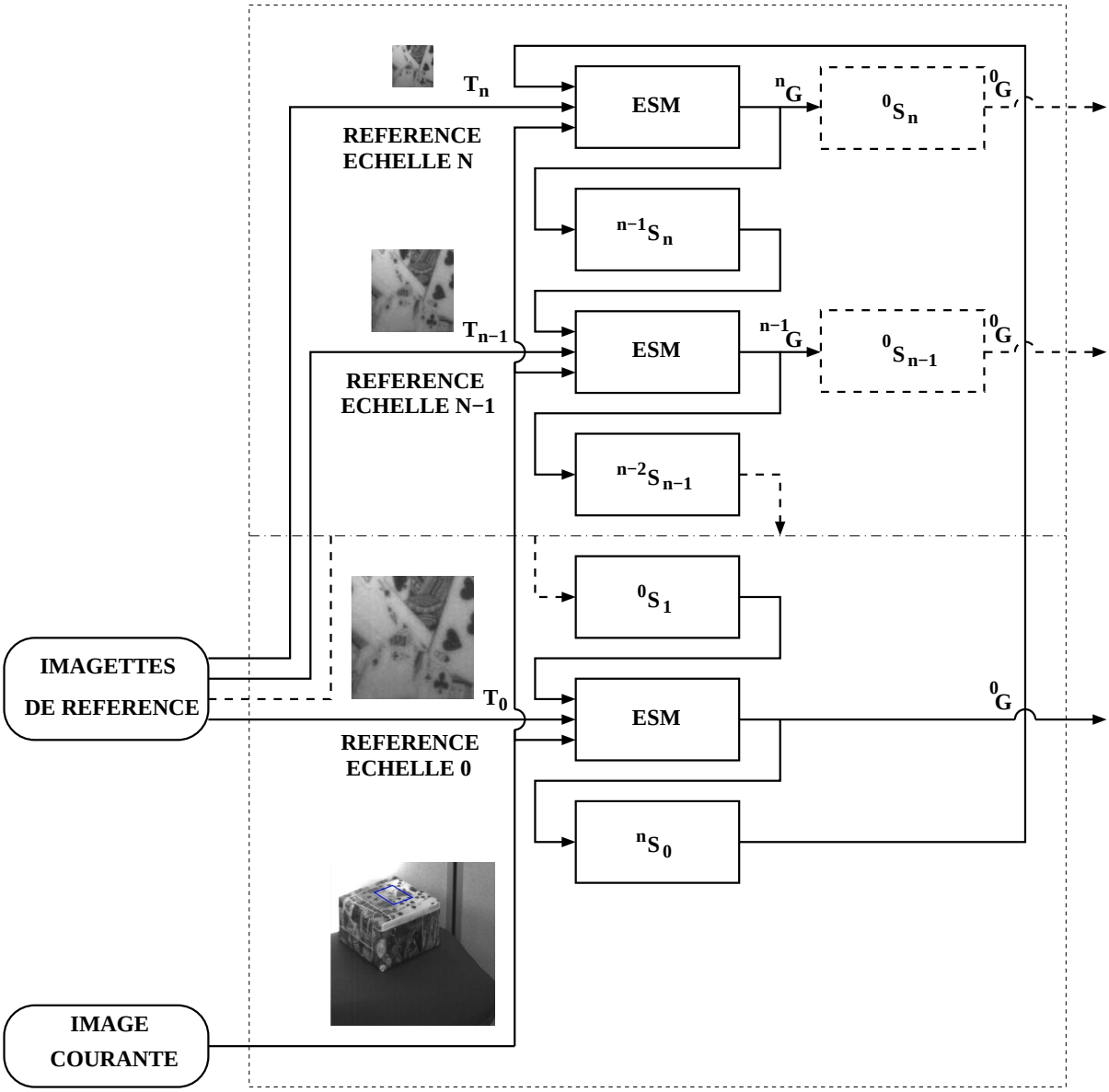


Figure 4.14 – Suivi visuel multirésolution avec la méthode ESM

phens, 1988) ou de contours (Canny, 1986)). Simplement, nous utilisons les points $\mathbf{p}_i^*$ ayant une norme de gradients supérieure à une certaine valeur (un seuil pré-défini) :

$$\|\nabla_{\mathbf{z}} \mathcal{I}^*(\mathbf{z})|_{\mathbf{z}=\mathbf{p}_i^*}\| \geq seuil \quad \text{où} \quad seuil > 0 \tag{4.61}$$

La Figure 4.15 montre un exemple de la sélection de pixels. L'imagette de référence est de dimensions (300×300) pixels. Choisir seulement les pixels ayant un fort gradient permet d'accélérer l'estimation de la transformation projective. Les points blancs dans la Figure 4.15 sont les pixels sélectionnés après seuillage et représentent seulement 10% de l'imagette de référence. Nous pouvons voir que les pixels appartenant à des zones uniformes de l'imagette n'ont pas été retenus. Une autre manière de sélectionner les pixels est d'utiliser non seulement les gradients de l'imagette de référence mais aussi ceux de l'imagette courante. Cette méthode est mieux adaptée à notre algorithme de minimisation. En effet, dans l'algorithme ESM, nous calculons le Jacobien avec la moyenne des gradients de l'imagette de référence et ceux de l'imagette courante. Nous utilisons les points $\mathbf{p}_i^*$ ayant une norme de gradients supérieure à une certaine valeur (un seuil pré-défini) :

$$\frac{1}{2}\|\nabla_{\mathbf{z}} \mathcal{I}^*(\mathbf{z})|_{\mathbf{z}=\mathbf{p}_i^*} + \nabla_{\mathbf{z}} \mathcal{I}(\mathbf{z})|_{\mathbf{z}=\mathbf{p}_i^*}\| \geq seuil \quad \text{où} \quad seuil > 0 \tag{4.62}$$

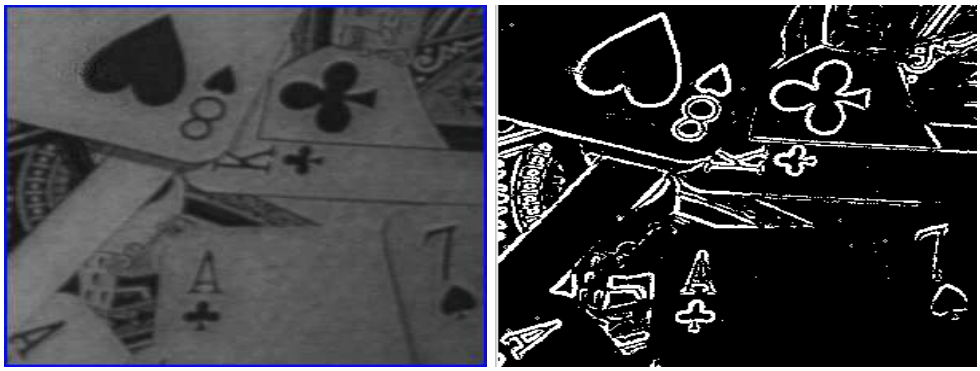


Figure 4.15 – Sélection de pixels à fort gradient

4.3.6.3 Prédiction de l'homographie par corrélation

Quand les déplacements de l'objet suivi entre deux images acquises successives est "raisonnable" (quand chaque coin de l'imagette se déplace de moins de 10% de la taille de l'imagette), la fréquence de convergence du SV avec la méthode ESM est élevée même si la transformation projective estimée initiale (au tout début de la minimisation) est égale à l'identité $\mathbf{I}$. Dans certains cas, pour des déplacements plus grands, même avec l'utilisation de la pyramide d'images, la minimisation par la méthode ESM peut tomber dans des minima locaux. Pour éviter les mi-

nima locaux, nous pouvons effectuer une mise en correspondance sur une fenêtre de recherche locale. La manière la plus simple pour le faire est d'utiliser le produit de corrélation normalisé. Le produit de corrélation normalisé permet de donner une mesure de ressemblance entre l'imagette de référence $\mathcal{I}^*$ et l'imagette courante $\mathcal{I}$ ayant effectué une translation $(u_c,\ v_c)$:

$$\mathcal{C}(u_c, v_c) = \frac{\sum_{(u,v)\in W} \left(\mathcal{I}(u-u_c, v-v_c) - \overline{\mathcal{I}}(u_c, v_c)\right)\left(\mathcal{I}^*(u,v) - \overline{\mathcal{I}^*}\right)}{\sqrt{\sum_{(u,v)\in W} \left(\mathcal{I}(u-u_c, v-v_c) - \overline{\mathcal{I}}(u_c, v_c)\right)^2}\sqrt{\sum_{(u,v)\in W} \left(\mathcal{I}^*(u,v) - \overline{\mathcal{I}^*}\right)^2}} \tag{4.63}$$

où $W = \{1, 2, ..., n\} \times \{1, 2, ..., m\}$ est la fenêtre dépendant de la taille des imagettes, $q = n\ m$ est le nombre total des pixels des imagettes, $\overline{\mathcal{I}}(u_c, v_c) = \frac{1}{q}\sum_{(u,v)\in W} \mathcal{I}(u-u_c, v-v_c)$ est la moyenne des valeurs des pixels de l'imagette courante $\mathcal{I}$ ayant effectué une translation $(u_c,\ v_c)$ et $\overline{\mathcal{I}^*} = \frac{1}{q}\sum_{(u,v)\in W} \mathcal{I}^*(u,v)$ est la moyenne des valeurs des pixels de l'imagette de référence $\mathcal{I}$. Le coefficient de corrélation normalisé $\mathcal{C}(u_c, v_c)$ est compris entre -1 et 1. Le calcul des coefficients de corrélation est coûteux en temps. C'est pour cette raison que mieux vaut l'effectuer sur l'étage de la pyramide le plus haut. Pour que cette méthode soit efficace, le facteur de changement d'échelle, la rotation et l'effet perspectif entre les imagettes dans les deux images doivent être faibles car le produit de corrélation n'est significatif que pour des translations planaires dans l'image. En effet, le calcul du produit de corrélation ne donne qu'une prédiction de 2 degrés de liberté (la translation planaire dans l'image). Cependant, dans la majorité des cas expérimentaux, nous remarquons que cette prédiction permet de trouver une transformation projective planaire $\mathbf{G}$ telle que l'approximation au deuxième ordre de l'erreur est valable.

Exemple 1 Supposons vouloir suivre une imagette (voir Figure 4.16(b)) de dimensions (110×95) encadrée en bleu dans l'image de référence 4.16(a) de dimensions (512×512). La transformation projective planaire à cette position est égale à :

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 206 \\ 0 & 1 & 206 \\ 0 & 0 & 1 \end{bmatrix}$$

Supposons que pour une certaine raison (fréquence d'acquisition faible, optique de caméra avec une grande focale, distance faible entre la caméra et la cible, mouvement rapide de la cible), il y a un grand mouvement de la cible entre l'image de référence 4.16(a) et l'image courante 4.16(c). Afin de faciliter la minimisation de la méthode ESM, nous effectuons une prédiction à l'aide de la corrélation. Sur une fenêtre de recherche (voir la zone encadrée en rouge de la Figure 4.16(c)), nous calculons les coefficients de corrélation normalisés (donnés par la formule (4.63)). Si nous construisons une image où un faible coefficient de corrélation (proche de -1) est représenté par un niveau de gris foncé (noir si il est égal à -1) et où un fort coefficient de corrélation (proche de 1) est représenté par un niveau de gris clair (blanc si il est égal à 1), nous obtenons l'image de la Figure 4.16(d). Nous pouvons déterminer, grâce aux coefficients

de corrélation normalisés, la position approximative (ou la position prédite) de l'imagette de référence dans l'image courante (voir le rectangle vert de la Figure 4.16(e)). La transformation projective planaire à cette position est égale à :

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 149 \\ 0 & 1 & 195 \\ 0 & 0 & 1 \end{bmatrix}$$

À partir de cette position, nous effectuons la minimisation avec la méthode ESM et nous obtenons le résultat de la Figure 4.16(f). La transformation projective planaire à cette position est égale à :

$$\mathbf{G} = \begin{bmatrix} 0,996 & -0,074 & 153,358 \\ 0,055 & 0,988 & 191,624 \\ -0,000 & -0,000 & 1,000 \end{bmatrix}$$

Dans cet exemple, il est vrai que le mouvement relatif de la caméra est quasiment une translation fronto-parallèle au plan de la cible, cas très favorable à la corrélation. Mais, il est possible de se ramener à ce cas, si nous transformons l'image courante par l'homographie estimée dans l'image précédente. Par conséquent, les effets de rotations et de changements d'échelle seront moins importants et l'utilisation de la corrélation est bien justifiée.

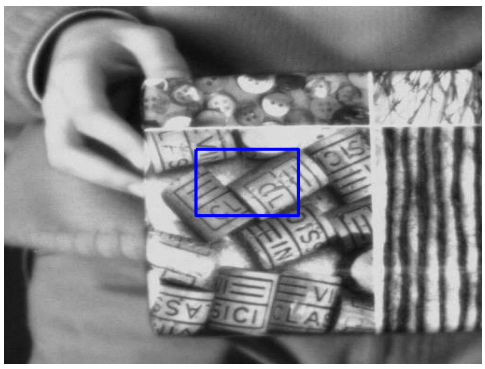
Remarque 2 D'autres améliorations sont également envisageables. La plupart concernent le traitement préalable des images avant de commencer la boucle du suivi. Par exemple, il est possible d'effectuer une égalisation et/ou un élargissement de l'histogramme de la zone suivie afin d'avoir une meilleure distribution des niveaux de gris dans l'image. Ou encore, pour réduire le bruit dans l'image (susceptible de réduire la précision du suivi de l'imagette), il est possible d'effectuer un filtrage des images par filtre moyenneur ou bien un filtre Gaussien.

4.4 Résultats expérimentaux

L'algorithme proposé a été testé sur un très grand nombre de séquences : des séquences d'applications robotiques terrestres, aériennes, sous-marines, ou encore des applications de vidéo-surveillance ou médicales. Nous présentons ici quelques résultats.

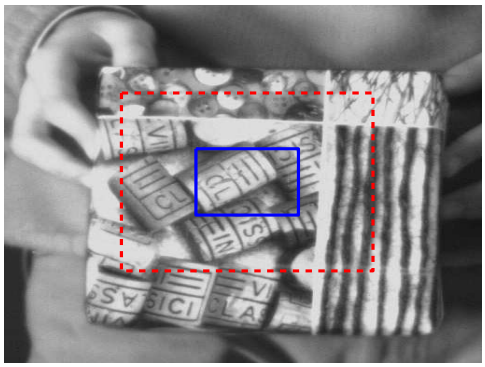
4.4.1 Suivi visuel d'un objet plan

La première séquence est une séquence d'intérieur d'environ 200 images de dimensions (512 $\times$ 512) acquise avec une caméra fixe et l'objet à suivre est un parallélépipède en mouvement. Dans la première image de la séquence (voir Figure 4.17(a)), nous sélectionnons l'imagette de référence : une zone texturée de dimensions (150 $\times$ 150) sur la face visible (voir Figure 4.17(f)). Nous désignons par un cadre rouge la position de la zone suivie. Nous observons que dans

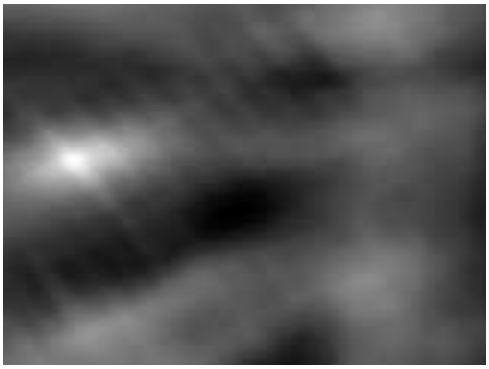


(a) Image de référence

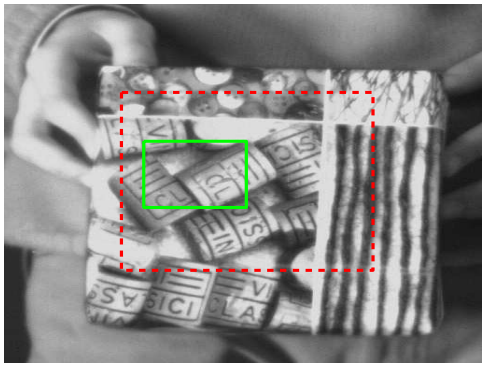
(b) Imagette de référence



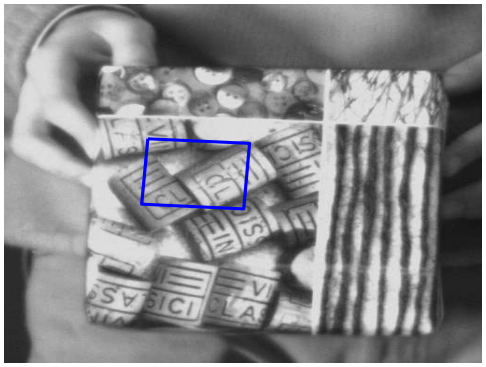
(c) Image courante et fenêtre de recherche



(d) Image des coefficients de corrélation



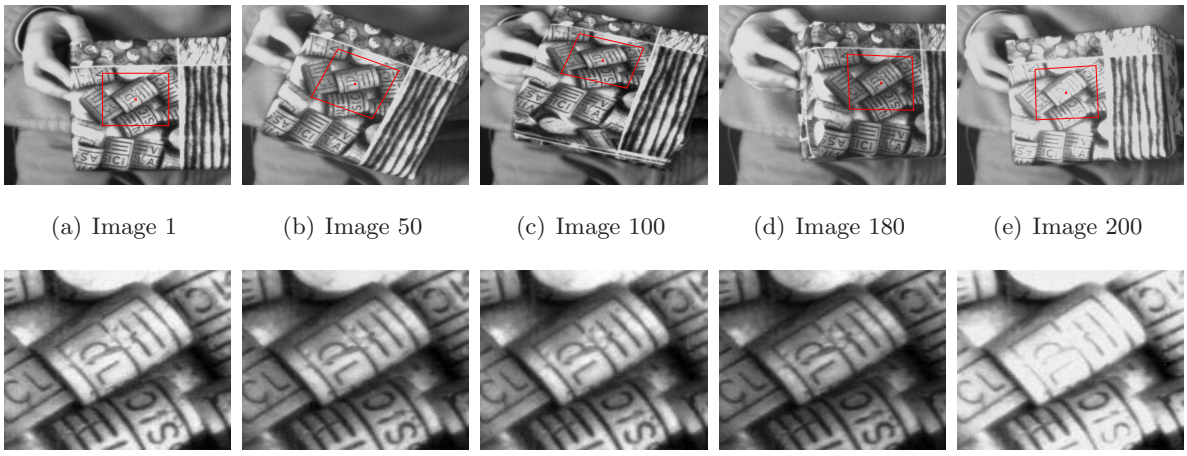
(e) Position du maximum de corrélation



(f) Résultat du SV avec la méthode ESM

Figure 4.16 – Utilisation de la prédiction par corrélation

les différentes images extraites de la séquence (voir les images de la première ligne de la Figure 4.17), la méthode ESM permet de retrouver la position de l'objet d'une manière précise. En effet, grâce à transformation projective obtenue avec la méthode ESM, il possible d'appliquer à la zone suivie la transformation projective inverse et d'obtenir des imagettes courantes très proches de l'imagette de référence (voir les images de la deuxième ligne de la Figure 4.17). Durant la séquence, l'objet effectue un mouvement projectif générique. Par exemple, dans les Figures 4.17(b) et 4.17(c), nous observons respectivement une translation et une rotation autour des axes $\vec{z}$ et $\vec{x}$. Une variation des conditions d'éclairage de la zone suivie est également observable dans la séquence. Par exemple, dans les Figures 4.17(d) et 4.17(e), au cours d'une rotation autour de l'axe $\vec{y}$, malgré le fait que l'image passe du foncé au clair, le SV reste très précis.



(a) Image 1 (b) Image 50 (c) Image 100 (d) Image 180 (e) Image 200

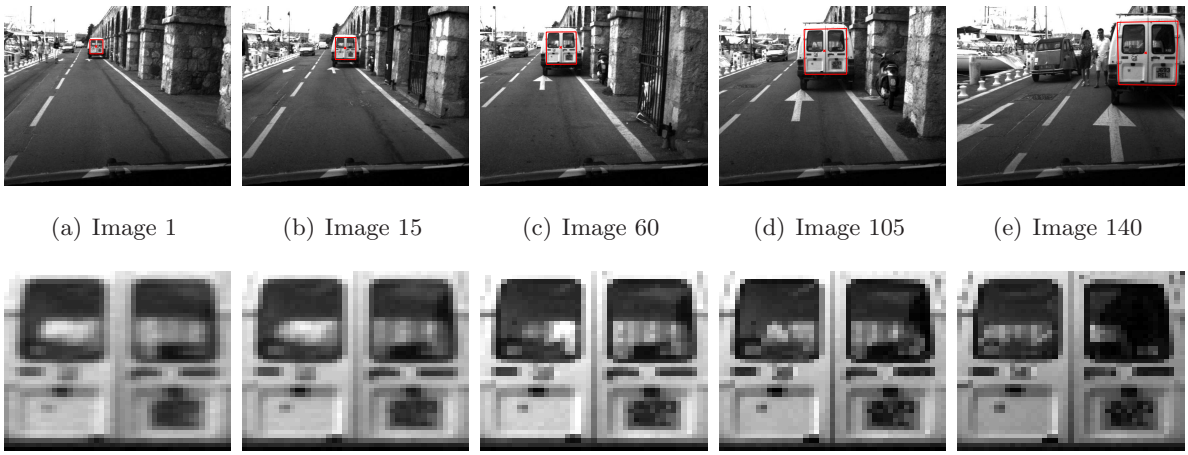
(f) Imagette 1 (g) Imagette 50 (h) Imagette 100 (i) Imagette 180 (j) Imagette 200

Figure 4.17 – SV d'un objet plan

4.4.2 Suivi visuel de l'arrière d'une voiture

La deuxième séquence est une séquence d'extérieur d'environ 150 images de dimensions (728 × 532) acquise par une caméra embarquée sur une voiture en train de rouler en ville. L'objet à suivre est une voiture stationnée sur le bord de la route. Dans la première image de la séquence (voir Figure 4.18(a)), nous sélectionnons l'imagette de référence de dimensions (43 × 43) correspondant à l'arrière de la voiture stationnée (voir Figure 4.18(f)). S'agissant d'une voiture de type utilitaire, l'arrière est approximativement plat et la méthode ESM peut donc être utilisée. Malgré le fait que le changement d'échelle entre la première image de la séquence et la dernière image est de plus de 4, la méthode ESM permet d'assurer le SV de la zone sélectionnée d'une manière précise. En effet, dans la première ligne de la Figure 4.18, dans différentes images extraites de la séquence, il est possible de retrouver la position du véhicule et d'estimer la

transformation projective de l'imagette de référence. La deuxième ligne de la Figure 4.17 montre que les imagettes courantes sont très proches de l'imagette de référence. Vu le grand changement de résolution, ces imagettes sont de moins en moins floues. Nous remarquons également que le mouvement des piétons devant le véhicule suivi, observable à travers les pare-brises, ne perturbe pas le SV. Ces mouvement, bien qu'ils modifient l'apparence de la cible suivie, ne perturbent pas le suivi visuel proposé. Ceci traduit la robustesse intrinsèque de l'algorithme aux faibles mesures aberrantes.



(a) Image 1 (b) Image 15 (c) Image 60 (d) Image 105 (e) Image 140

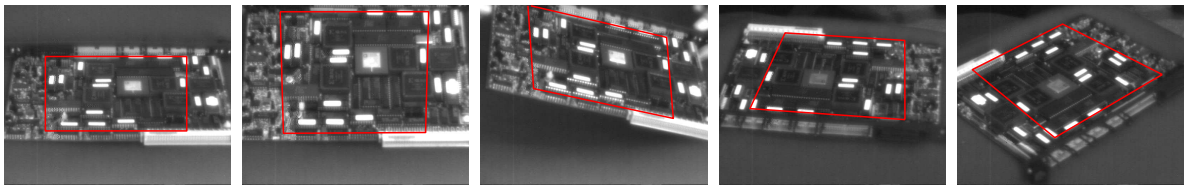
(f) Imagette 1 (g) Imagette 15 (h) Imagette 60 (i) Imagette 105 (j) Imagette 140

Figure 4.18 – Suivi visuel de l'arrière d'une voiture

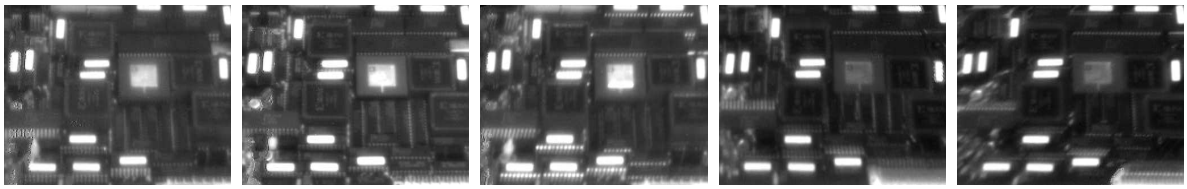
4.4.3 Suivi visuel d'une carte électronique

La dernière séquence est une séquence d'intérieur d'environ 1600 images de dimensions (640 × 480) acquise par une caméra à 100Hz (100 images/seconde). L' expérience a été effectuée en temps-réel. Seule une image sur quatre a été enregistrée (pour ne pas perturber le suivi visuel). Nous suivons en temps-réel une carte électronique qui n'est pas parfaitement plane. Dans la première image de la séquence (voir Figure 4.19(a)), nous sélectionnons l'imagette de référence de dimensions (400 × 200) (voir Figure 4.19(f)). Malgré le fait que la zone sélectionnée contient des composants non-plans et certains sont constitués de matériaux réfléchissants (la réponse à l'éclairement varie selon l'orientation de la carte), malgré les auto-occultations et les sorties partielles de la cible de l'image, la méthode ESM permet d'assurer le SV de la zone sélectionnée d'une manière précise. En effet, dans la Figure 4.19, dans différentes images extraites de la séquence, il est possible d'estimer la transformation projective de l'imagette de référence. Là-encore, nous pouvons appliquer la transformation projective inverse et obtenir des imagettes courantes quasi-identiques à l'imagette de référence (voir les images de la deuxième et la quatrième ligne de la Figure 4.19). Nous pouvons voir que l'approche proposée gère bien

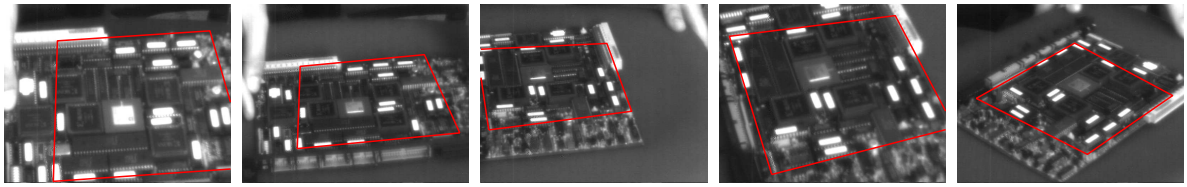
les occultations partielles dues à la non-planarité de la zone suivie.



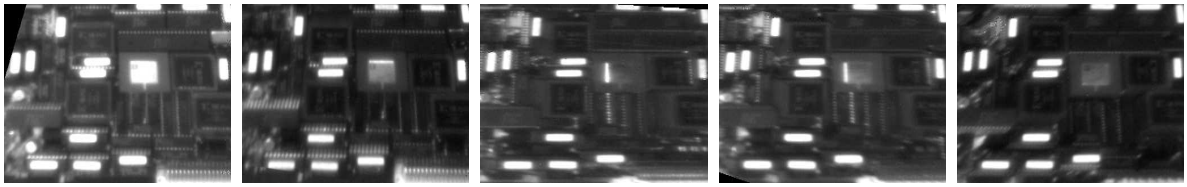
(a) Image 1 (b) Image 41 (c) Image 81 (d) Image 121 (e) Image 161



(f) Imagette 1 (g) Imagette 41 (h) Imagette 81 (i) Imagette 121 (j) Imagette 161



(k) Image 201 (l) Image 241 (m) Image 281 (n) Image 321 (o) Image 361



(p) Imagette 201 (q) Imagette 241 (r) Imagette 281 (s) Imagette 321 (t) Imagette 361

Figure 4.19 – Suivi visuel d'une carte électronique

4.5 Conclusion

Dans ce chapitre, nous avons présenté une méthode de SV permettant la résolution du problème d'optimisation en supposant que les paramètres de transformation varient, localement, d'une manière quadratique par rapport à l'erreur des intensités lumineuses mesurées. Les paramètres de transformation estimés sont les paramètres de transformation projective dans l'image. Nous avons prouvé théoriquement, puis à l'aide de simulations que cette approche est meilleure que les approches existantes en terme d'efficacité, de fréquence et de taux de convergence. Nous avons testé l'algorithme sur divers types de séquences et là-encore, l'estimation du déplacement dans l'image a donné des résultats satisfaisants. Les limitations classiques des approches de

suivi visuel standards, telles que le changement d'éclairement, la non-planarité de la cible, les occultations partielles, les sorties de la cible du champ de vue de la caméra, ont été considérées. Par conséquent, il est possible d'estimer, a posteriori, le mouvement relatif entre la caméra et la scène en utilisant les paramètres intrinsèques de la caméra et une approximation des vecteurs normaux aux plans suivis (exprimée dans la position de référence).

Dans le chapitre suivant, nous présentons un suivi visuel très similaire, mais permettant d'estimer le mouvement relatif entre la caméra et la scène (dans l'espace Cartésien) et d'intégrer les informations sur le modèle de la caméra et de la scène a priori, c'est-à-dire, seuls six degrés de libertés seront estimés (trois pour la rotation et trois pour la translation).

Chapitre 5

Suivi visuel dans l'espace Cartésien de surfaces planaires avec la méthode ESM

5.1 Introduction

Dans ce chapitre, nous proposons une extension de la méthode ESM permettant l'estimation du déplacement relatif dans l'espace Cartésien entre une cible planaire par morceaux et la caméra. La méthode proposée permet d'utiliser le modèle de la cible quand celui-ci est disponible et d'intégrer d'éventuelles contraintes sur le mouvement relatif entre la caméra et la cible afin de réduire les paramètres à estimer. Cette méthode de suivi visuel résout à la fois le problème de suivi dans l'image et l'estimation du mouvement, le tout dans un cadre unifié. En effet, cette approche permet, dans le cas du suivi visuel d'une cible rigide et planaire par morceaux, d'imposer que tous les plans constituant la cible effectuent la même rotation et la même translation. Par conséquent, seulement six paramètres sont à estimer (beaucoup moins que si nous calculons une homographie pour chaque plan suivi). De plus, si les imagettes sont sur des plans différents, les homographies correspondantes estimées seront différentes. Dans le cas idéal, toutes les homographies donnent un déplacement cohérent de la caméra (si nous utilisons, par exemple, l'algorithme présenté dans (Faugeras and Lustman, 1988) pour estimer le déplacement à partir des homographies). En pratique, cela n'arrive que très rarement, surtout si nous ne contraignons pas explicitement que les plans sont rigidement attachés les uns aux autres. En effet, il arrive que, dans le cas où les plans n'ont pas une texture riche, deux imagettes planaires donnent des déplacements contradictoires.

Pour estimer le déplacement, nous effectuons le calcul explicite de la dépendance entre l'apparence de l'objet dans l'image (qui dépend des paramètres de la transformation homographique, dans le cas plan) et les paramètres du déplacement 3D de la caméra (la translation

et la rotation Cartésiennes). Ensuite, nous appliquons l'algorithme de minimisation ESM (proposé dans le chapitre précédent) à ce problème. Nous verrons comment cet algorithme permet, encore une fois, d'améliorer le domaine et le taux de convergence des algorithmes classiques (au premier-ordre) tout en ayant une complexité équivalente (nombre d'opérations nécessaires équivalent).

Nous commençons par une étude théorique. Puis, nous effectuons des simulations numériques sur des images synthétiques et réelles. Enfin, nous réalisons des tests sur des séquences vidéo réelles acquises par un robot mobile muni de capteurs odométriques précis qui nous permettront d'évaluer la qualité de nos résultats.

5.2 Le suivi visuel dans l'espace Cartésien de surfaces planaires

Supposons que la caméra observe une cible planaire définie par un plan π à une distance d^* de l'origine du repère $\mathcal{F}^*$ et ayant un vecteur normal $\mathbf{n}^*$ (exprimé dans le repère de référence $\mathcal{F}^*$) vérifiant l'équation (2.11). Grâce à l'équation (2.25), la matrice de transformation projective planaire $\mathbf{G}$ reliant les projections $\mathbf{p}^*$ et $\mathbf{p}$ d'un même point $\mathcal{P} \in \pi$ respectivement dans l'image $\mathcal{I}^*$ (acquise à la position de référence $\mathcal{F}^*$) et l'image courante $\mathcal{I}$ (acquise par la même caméra à la position courante $\mathcal{F}$) peut s'écrire à l'aide de la matrice de rotation $\mathbf{R}$, du vecteur de translation $\mathbf{t}$ (entre $\mathcal{F}^*$ et $\mathcal{F}$) et de la matrice $\mathbf{K}$ des paramètres intrinsèques de la caméra :

$$\mathbf{G} \propto \mathbf{K} \left(\mathbf{R} + \mathbf{t}\mathbf{n}^{*\top} \right) \mathbf{K}^{-1} \tag{5.1}$$

Par la suite, nous noterons $\mathbf{G}(\mathbf{T})$ la matrice de transformation projective planaire dans l'image, où $\mathbf{T}(\mathbf{R}, \mathbf{t}) \in \mathbb{SE}(3)$, définie par l'équation (2.5) est le déplacement entre deux repères $\mathcal{F}^*$ et $\mathcal{F}$. Par ailleurs, étant donné que la matrice $\mathbf{G}$ est définie à un facteur d'échelle près, nous pouvons imposer à $\mathbf{G}(\mathbf{T})$ d'être dans $\mathbb{SL}(3)$ en divisant cette matrice par la racine cubique de son déterminant :

$$\mathbf{G}(\mathbf{T}) = \mathbf{K} \left(\frac{\mathbf{R} + \mathbf{t}\mathbf{n}^{*\top}}{\sqrt[3]{1 + \mathbf{n}^{*\top}\mathbf{R}^\top\mathbf{t}}} \right) \mathbf{K}^{-1} \tag{5.2}$$

L'application $\mathbf{G}$ définit une application entre $\mathbb{SE}(3)$ et $\mathbb{SL}(3)$. Mais cette application n'est pas un homomorphisme de groupe. En effet, en général, pour $(\mathbf{T}_1, \mathbf{T}_2) \in \mathbb{SE}(3) \times \mathbb{SE}(3)$, nous avons :

$$\mathbf{G}(\mathbf{T}_1)\mathbf{G}(\mathbf{T}_2) \neq \mathbf{G}(\mathbf{T}_1\mathbf{T}_2) \tag{5.3}$$

Par conséquent, l'application de transformation de l'image (ou "warping") notée $\mathbf{w}(\mathbf{G})$:

$$\mathbf{w}(\mathbf{G}) : \quad \mathbb{SE}(3) \times \mathbb{P}^2 \quad \rightarrow \quad \mathbb{P}^2 \tag{5.4}$$

n'est pas une action de groupe de $\mathbb{SE}(3)$ vers $\mathbb{P}^2$ et nous avons :

$$\mathbf{w}(\mathbf{G}(\mathbf{T}_1))(\mathbf{w}(\mathbf{G}(\mathbf{T}_2)(\mathbf{p})) \neq \mathbf{w}(\mathbf{G}(\mathbf{T}_1\mathbf{T}_2))(\mathbf{p}) \tag{5.5}$$

Dans certains cas particuliers, quand $\mathbf{T}$ appartient à un certain sous-groupe de $\mathbb{SE}(3)$, l'application $\mathbf{w}(\mathbf{G})$ est une action de groupe. Par exemple, si $\mathbf{t} = \mathbf{0}$ alors $\mathbf{G}$ dépend seulement de la rotation $\mathbf{R} \in \mathbb{SO}(3)$ et l'application $\mathbf{w}(\mathbf{G})$ devient une action de $\mathbb{SO}(3)$ sur $\mathbb{P}^2$.

5.2.1 Modélisation du problème

Supposons vouloir suivre une cible rigide de la scène et planaire par morceaux. La contrainte de rigidité impose que tous les plans sont soumis au même mouvement dans l'espace Cartésien. Nous cherchons à estimer la position de la cible dans l'image et en même temps son mouvement dans l'espace 3D. Considérons, pour des raisons de simplicité, le suivi d'un seul plan. Soit q le nombre total de pixels d'une certaine région d'image de forme quelconque et correspondant à la projection d'une partie planaire de la cible dans l'image de référence $\mathcal{I}^*$. Cette région de l'image sera appelée "imagette de référence".

Suivre visuellement cette imagette de référence dans l'image courante $\mathcal{I}$ consiste donc à trouver la matrice de transformation $\overline{\mathbf{T}} \in \mathbb{SE}(3)$ qui transforme chaque pixel $\mathbf{p}_i^*$ de l'imagette de référence en son correspondant dans l'image courante $\mathcal{I}$, c'est-à-dire, trouver $\overline{\mathbf{T}}$ telle que $\forall i \in \{1,\ 2,\ ...,\ q\}$:

$$\mathcal{I}\left(\mathbf{w}\left(\mathbf{G}(\overline{\mathbf{T}})\right)(\mathbf{p}_i)\right) = \mathcal{I}^*(\mathbf{p}_i) \tag{5.6}$$

Supposons que nous avons une approximation $\widehat{\mathbf{T}}$ de la transformation $\overline{\mathbf{T}}$, le problème revient à trouver une transformation incrémentale $\mathbf{T}(\mathbf{x})$ telle que la différence entre la région de l'image $\mathcal{I}$ transformée avec la composition $\mathbf{w}(\mathbf{G}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})))$ et la région correspondante dans l'image $\mathcal{I}^*$ est nulle. Par conséquent, il s'agit de trouver le vecteur $\mathbf{x}$ tel que $\forall i \in \{1,\ 2,\ ...,\ q\}$, nous avons :

$$y_i(\mathbf{x}) = \mathcal{I}\left(\mathbf{w}(\mathbf{G}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})))(\mathbf{p}_i)\right) - \mathcal{I}^*(\mathbf{p}_i) = 0 \tag{5.7}$$

où $\mathbf{x} = [x_1,\ x_2,\ ...,\ x_6]^\top$ est un vecteur de dimensions (6×1) contenant les paramètres de la matrice de transformation incrémentale.

5.2.2 Définition vectorielle du système d'équations

Considérons le vecteur $\mathbf{y}(\mathbf{x})$ de dimensions $(q \times 1)$ contenant les différences d'image :

$$\mathbf{y}(\mathbf{x}) = \begin{bmatrix} y_1(\mathbf{x}) & y_2(\mathbf{x}) & ... & y_q(\mathbf{x}) \end{bmatrix}^\top \tag{5.8}$$

le problème revient à trouver le vecteur $\mathbf{x} = \widetilde{\mathbf{x}}$ vérifiant le système d'équations :

$$\mathbf{y}(\widetilde{\mathbf{x}}) = \mathbf{0} \tag{5.9}$$

Il est évident que la solution du système (5.9) vérifie :

$$\mathbf{T}(\widetilde{\mathbf{x}}) = \widehat{\mathbf{T}}^{-1}\overline{\mathbf{T}} \tag{5.10}$$

Le système d'équations (5.9) est généralement non-linéaire, il est donc difficile de le résoudre en une seule itération. Un algorithme de minimisation itérative est alors utilisé, une fois que le système a été linéarisé.

5.2.3 Approximation du système d'équations

Dans ce paragraphe, nous suivons la même méthode utilisée dans le paragraphe 4.2.3. Mais les paramètres de la transformation correspondent au mouvement 3D de la cible dans l'espace (au lieu des paramètres de la transformation dans l'image).

D'une manière analogue, nous définissons, la matrice Jacobienne $\mathbf{J}(\mathbf{x})$ qui est une matrice de dimensions $(q\times 6)$ qui représente la variation du vecteur d'erreur $\mathbf{y}(\mathbf{x})$ en fonction des paramètres $\mathbf{x}$ du mouvement Cartésien. Nous définissons la matrice $\mathbf{M}(\mathbf{x}_1, \mathbf{x}_2)$ de dimensions $(q \times 6)$ qui s'écrit $\forall(\mathbf{x}_1, \mathbf{x}_2) \in \mathbb{R}^6 \times \mathbb{R}^6$:

$$\mathbf{M}(\mathbf{x}_1, \mathbf{x}_2) = \nabla_{\mathbf{x}_1} \left(\mathbf{J}(\mathbf{x}_1)\mathbf{x}_2\right) \tag{5.11}$$

Encore une fois, s'agissant d'un problème de SV en temps réel, le déplacement de la cible entre deux images successives est faible, c'est-à-dire, les paramètres de la matrice de transformation incrémentale sont faibles $\mathbf{x} \approx \mathbf{0}$. Il est possible de linéariser le vecteur $\mathbf{y}(\mathbf{x})$ en effectuant un développement en série de Taylor au deuxième ordre au voisinage de $\mathbf{x} = \mathbf{0}$:

$$\mathbf{y}(\mathbf{x}) = \mathbf{y}(\mathbf{0}) + \mathbf{J}(\mathbf{0})\ \mathbf{x} + \frac{1}{2}\mathbf{M}(\mathbf{0}, \mathbf{x})\ \mathbf{x} + \mathbf{O}(\|\mathbf{x}\|^3) \tag{5.12}$$

En $\mathbf{x} = \widetilde{\mathbf{x}}$, le système d'équations (5.9) peut être écrit :

$$\mathbf{y}(\widetilde{\mathbf{x}}) \approx \mathbf{y}(\mathbf{0}) + \left(\mathbf{J}(\mathbf{0}) + \frac{1}{2}\mathbf{M}(\mathbf{0}, \widetilde{\mathbf{x}})\right)\widetilde{\mathbf{x}} = \mathbf{0} \tag{5.13}$$

5.2.4 Minimisation itérative du système d'équations

Généralement, la méthode retenue pour résoudre le système d'équations (5.9) est la méthode des moindres carrés. Il s'agit de minimiser d'une manière itérative la fonction de coût suivante :

$$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{y}(\mathbf{0}) + \mathbf{J}(\mathbf{0})\mathbf{x} + \frac{1}{2}\mathbf{M}(\mathbf{0}, \mathbf{x})\mathbf{x}\|^2 \tag{5.14}$$

Une condition nécessaire pour qu'un vecteur $\mathbf{x} = \widetilde{\mathbf{x}}$ soit un minimum local ou global de la fonction de coût f est que la dérivée de la fonction de coût soit nulle, c'est-à-dire :

$$\nabla_{\mathbf{x}} f(\mathbf{x})|_{\mathbf{x}=\widetilde{\mathbf{x}}} = \mathbf{0} \tag{5.15}$$

La dérivée de la fonction de coût peut être écrite sous la forme suivante :

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = \big(\mathbf{J}(\mathbf{0}) + \mathbf{M}(\mathbf{0},\mathbf{x}) + \mathbf{O}(\|\mathbf{x}\|^2)\big)^\top \big(\mathbf{y}(\mathbf{0}) + \mathbf{J}(\mathbf{0})\mathbf{x} + \mathbf{O}(\|\mathbf{x}\|^2)\big) \tag{5.16}$$

Comme nous l'avons déjà vu dans le chapitre précédent, la méthode de minimisation standard de Newton résout le système (5.15) d'une manière itérative. À chaque itération, une solution incrémentale $\widetilde{\mathbf{x}}$ est estimée comme suit :

$$\widetilde{\mathbf{x}} = -\mathbf{S}^{-1}\mathbf{J}(\mathbf{0})^\top \mathbf{y}(\mathbf{0}) \tag{5.17}$$

où la matrice $\mathbf{S}$ de dimensions (6×6) s'écrit :

$$\mathbf{S} = \mathbf{J}(\mathbf{0})^\top \mathbf{J}(\mathbf{0}) + \sum_{i=0}^{q} \left.\frac{\partial^2 y_i(\mathbf{x})}{\partial \mathbf{x}^2}\right|_{\mathbf{x}=\mathbf{0}} y_i(\mathbf{0}) \tag{5.18}$$

Une fois la transformation incrémentale $\mathbf{T}(\widetilde{\mathbf{x}})$ calculée, nous mettons à jour l'approximation de la matrice de transformation $\widehat{\mathbf{T}}$ de la manière suivante :

$$\widehat{\mathbf{T}} \longleftarrow \widehat{\mathbf{T}}\mathbf{T}(\widetilde{\mathbf{x}}) \tag{5.19}$$

D'une manière similaire au chapitre précédent, à chaque mise à jour de $\widehat{\mathbf{T}}$, nous estimons $\mathbf{y}(\mathbf{0})$ et $\mathbf{J}(\mathbf{0})$ et nous calculons à nouveau $\widetilde{\mathbf{x}}$. Nous arrêtons l'algorithme quand la valeur de $\widetilde{\mathbf{x}}$ calculée devient très faible.

La méthode de Newton nécessite le calcul des q matrices Hessiennes et a une convergence quadratique dans le voisinage de $\mathbf{x} = \mathbf{0}$. Plusieurs méthodes de minimisation proposent d'approcher la matrice $\mathbf{S}$ avec une matrice $\widehat{\mathbf{S}}$ définie positive, ce qui revient à utiliser des approximations au premier-ordre dans l'équation (5.12). Il existe également des algorithmes qui approchent le Jacobien courant $\mathbf{J}(\mathbf{0})$, variable d'une itération à l'autre, par un Jacobien constant permettant ainsi d'accélérer l'algorithme au prix de la réduction de la zone de convergence. Ces approches permettent de calculer la matrice $\mathbf{J}(\mathbf{0})$ et d'inverser la matrice $\widehat{\mathbf{S}}$ une fois pour toutes (Sepp and Hirzinger, 2003). Mais, dans ce cas (où les paramètres sont dans $\mathbb{SE}(3)$), ces approches restent très locales. En effet, l'algorithme "inverse compositional" proposé par (Baker et al., 2004) ne peut pas être appliqué pour ce groupe (voir (Baker et al., 2004) pour les détails).

Pour conclure, l'approximation au deuxième ordre de la fonction de coût reste très peu utilisée parce qu'elle nécessite le calcul des Hessiens et connaît certains problèmes de convergence quand la matrice $\mathbf{S}$ n'est pas définie positive. C'est pour cette raison que les approches les

plus populaires dans le SV des régions sans étape d'apprentissage utilisent essentiellement des approximations au premier-ordre de la fonction de coût.

Pour résoudre le système d'équations (5.9), dans le paragraphe suivant, nous proposons d'adapter la méthode ESM au suivi visuel dans l'espace Cartésien.

5.3 Adaptation du suivi visuel avec la méthode ESM

5.3.1 L'algèbre de Lie $\mathfrak{se}(3)$

Soit $\{\mathbf{A}_i, i \in \{1, 2, ..., 6\}\}$, un ensemble de matrice de dimensions (4×4) représentant une base de l'algèbre de Lie $\mathfrak{se}(3)$ associée au groupe de Lie, le groupe Spécial Euclidien $\mathbb{SE}(3)$. Toute matrice $\mathbf{A} \in \mathfrak{se}(3)$ peut être écrite comme une combinaison linéaire des matrices $\mathbf{A}_i$ formant cette base :

$$\mathbf{A}(\mathbf{x}) = \sum_{i=1}^{6} x_i \mathbf{A}_i \tag{5.20}$$

où $\mathbf{x} = [x_1,\ x_2,\ ...,\ x_6]^\top$ est un vecteur de dimensions (6×1) et où x_i est la i-ème coordonnée. Si nous notons par $\mathcal{F}^* = (\mathcal{O}^*, \vec{i}^*, \vec{j}^*, \vec{k}^*)$ le "repère de référence" de l'espace Cartésien, les matrices génératrices des translations peuvent être écrites :

$$\mathbf{A}_1 = \begin{bmatrix} \mathbf{0} & \mathbf{i}^* \\ \mathbf{0} & 0 \end{bmatrix}, \mathbf{A}_2 = \begin{bmatrix} \mathbf{0} & \mathbf{j}^* \\ \mathbf{0} & 0 \end{bmatrix}, \mathbf{A}_3 = \begin{bmatrix} \mathbf{0} & \mathbf{k}^* \\ \mathbf{0} & 0 \end{bmatrix}$$

et les matrices génératrices des rotations peuvent être écrites :

$$\mathbf{A}_4 = \begin{bmatrix} [\mathbf{i}^*]_\times & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}, \mathbf{A}_5 = \begin{bmatrix} [\mathbf{j}^*]_\times & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}, \mathbf{A}_6 = \begin{bmatrix} [\mathbf{k}^*]_\times & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}$$

où $[\mathbf{i}^*]_\times$, $[\mathbf{j}^*]_\times$ et $[\mathbf{k}^*]_\times$ sont respectivement les matrices antisymétriques associées aux vecteurs $\mathbf{i}^*$, $\mathbf{j}^*$ et $\mathbf{k}^*$ (eux-mêmes associés aux entités géométriques $\vec{i}^*$, $\vec{j}^*$ et $\vec{k}^*$).

Dans un voisinage de la matrice identité $\mathbf{I}$, le groupe $\mathbb{SE}(3)$ peut être paramétré à l'aide de l'algèbre de Lie associée $\mathfrak{se}(3)$ via l'application exponentiel matricielle :

$$\exp : \mathfrak{se}(3) \rightarrow \mathbb{SE}(3)$$

Il existe un ouvert autour de $\mathbf{0}$ dans $\mathfrak{se}(3)$ et un voisinage de la matrice identité $\mathbf{I}$ dans $\mathbb{SE}(3)$ tels que l'application exponentiel est $\mathcal{C}^\infty$ et admet une application inverse $\mathcal{C}^\infty$. Le voisinage de la matrice identité $\mathbf{I}$ dans $\mathbb{SE}(3)$ est large (l'angle de la rotation doit être inférieur à 2π). Une matrice de transformation $\mathbf{T}(\mathbf{x})$ appartenant au groupe $\mathbb{SE}(3)$ et étant dans ce voisinage peut

donc être paramétrée à l'aide du vecteur $\mathbf{x}$ de la manière suivante :

$$\mathbf{T}(\mathbf{x}) = \exp\left(\mathbf{A}(\mathbf{x})\right) = \sum_{i=0}^{\infty} \frac{1}{i!} \left(\mathbf{A}(\mathbf{x})\right)^i \tag{5.21}$$

où $\mathbf{A}(\mathbf{x})$ est une matrice qui s'écrit sous la forme (5.20). Si nous notons par $\boldsymbol{\beta}(\mathbf{x}) = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^\top$ et par $\mathbf{u}(\mathbf{x})\theta(\mathbf{x}) = \begin{bmatrix} x_4 & x_5 & x_6 \end{bmatrix}^\top$ (où $\mathbf{u}(\mathbf{x})$ est un vecteur unitaire correspondant à l'axe de la rotation et où $\theta(\mathbf{x}) = \sqrt{x_4^2 + x_5^2 + x_6^2}$ correspond à l'angle de la rotation), la matrice de transformation $\mathbf{T}(\mathbf{x})$ peut être écrite de la manière suivante :

$$\mathbf{T}(\mathbf{x}) = \exp\left(\begin{bmatrix} [\mathbf{u}(\mathbf{x})\theta(\mathbf{x})]_\times & \boldsymbol{\beta}(\mathbf{x}) \\ \mathbf{0} & 0 \end{bmatrix}\right) \tag{5.22}$$

Si nous utilisons l'expression de la matrice $\mathbf{T}(\mathbf{x}) \in \mathbb{SE}(3)$ (voir l'équation 2.5), nous pouvons écrire :

$$\mathbf{T}(\mathbf{x}) = \begin{bmatrix} \mathbf{R}(\mathbf{x}) & \mathbf{t}(\mathbf{x}) \\ \mathbf{0} & 1 \end{bmatrix} \tag{5.23}$$

où la matrice de rotation peut être écrite à l'aide de la formule de Rodriguez :

$$\mathbf{R}(\mathbf{x}) = \exp\left([\mathbf{u}(\mathbf{x})\theta(\mathbf{x})]_\times\right) = \mathbf{I} + \sin(\theta(\mathbf{x}))\,[\mathbf{u}(\mathbf{x})]_\times + (1 - \cos(\theta(\mathbf{x})))\,[\mathbf{u}(\mathbf{x})]_\times^2 \tag{5.24}$$

et le vecteur de translation peut être écrit à l'aide de l'équation suivante :

$$\mathbf{t}(\mathbf{x}) = \sum_{k=0}^{\infty} \frac{1}{(k+1)!} [\mathbf{u}(\mathbf{x})\theta(\mathbf{x})]_\times^k \boldsymbol{\beta}(\mathbf{x}) \tag{5.25}$$

et étant donné que $[\mathbf{u}(\mathbf{x})]_\times^3 = -[\mathbf{u}(\mathbf{x})]_\times$, nous pouvons écrire que :

$$\begin{aligned} \mathbf{t}(\mathbf{x}) &= \left(\mathbf{I} - \sum_{k=1}^{\infty} \frac{(-1)^k}{2k!}\theta^{2k-1}[\mathbf{u}(\mathbf{x})]_\times - \sum_{k=1}^{\infty} \frac{(-1)^k}{(2k+1)!}\theta^{2k}[\mathbf{u}(\mathbf{x})]_\times^2\right)\boldsymbol{\beta}(\mathbf{x}) \\ &= \left(\mathbf{I} + \frac{1-\cos(\theta(\mathbf{x}))}{\theta(\mathbf{x})}[\mathbf{u}(\mathbf{x})]_\times + \left(1 - \frac{sin(\theta(\mathbf{x}))}{\theta(\mathbf{x})}\right)[\mathbf{u}(\mathbf{x})]_\times^2\right)\boldsymbol{\beta}(\mathbf{x}) \end{aligned} \tag{5.26}$$

Le vecteur $\mathbf{x}$ peut être considéré comme la paramétrisation d'un mouvement 3D local de la caméra. Donc, d'une manière analogue, et en utilisant l'équation (5.1), une matrice de transformation projective planaire dans l'image peut être paramétrée à l'aide du vecteur $\mathbf{x}$ de la manière suivante :

$$\mathbf{G}(\mathbf{T}(\mathbf{x})) \propto \mathbf{K}\left(\mathbf{R}(\mathbf{x}) + \mathbf{t}(\mathbf{x})\mathbf{n}^{*\top}\right)\mathbf{K}^{-1} \tag{5.27}$$

où $\mathbf{K}$ est la matrice des paramètres intrinsèques de la caméra et $\mathbf{n}^*$ est le vecteur normal au plan π considéré.

5.3.2 Calcul des Jacobiens

Nous souhaitons calculer la matrice Jacobienne $\mathbf{J}(\mathbf{x})$ correspondant à la dérivée du vecteur $\mathbf{y}(\mathbf{x})$ aux points $\mathbf{x} = \mathbf{0}$ et $\mathbf{x} = \widetilde{\mathbf{x}}$ vérifiant la contrainte (5.10). Nous verrons, par la suite, l'intérêt de ce calcul et les propriétés des ces matrices Jacobiennes. Nous suivrons la même démarche que celle présentée dans le chapitre précédent.

5.3.2.1 Le Jacobian courant

La ligne i du premier Jacobien $\mathbf{J}(\mathbf{0})$, que nous appellerons Jacobien courant, s'écrit sous la forme suivante :

$$\begin{aligned}\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\mathbf{0}} &= \nabla_{\mathbf{x}} \mathcal{I}\left(\mathbf{w}\left(\mathbf{G}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x}))\right)(\mathbf{p}_i^*)\right)\Big|_{\mathbf{x}=\mathbf{0}} \\ &= \nabla_{\mathbf{x}} \mathcal{I}\left(\mathbf{w}\left(\mathbf{G}(\widehat{\mathbf{T}})\mathbf{G}(\widehat{\mathbf{T}})^{-1}\mathbf{G}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x}))\right)(\mathbf{p}_i^*)\right)\Big|_{\mathbf{x}=\mathbf{0}}\end{aligned}$$

Grâce à l'équation (2.29), nous pouvons écrire :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\mathbf{0}} = \nabla_{\mathbf{x}} \mathcal{I}\left(\mathbf{w}\left(\mathbf{G}(\widehat{\mathbf{T}})\right)\left(\mathbf{w}\left(\mathbf{G}(\widehat{\mathbf{T}})^{-1}\mathbf{G}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x}))\right)(\mathbf{p}_i^*)\right)\right)\Big|_{\mathbf{x}=\mathbf{0}} \tag{5.28}$$

La ligne i du Jacobien courant peut être écrite comme le produit de 5 Jacobiens différents :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\mathbf{0}} = \mathbf{J}_{\mathcal{I}}\, \mathbf{J}_{\mathbf{w}}\, \mathbf{J}_K\, \mathbf{J}_{\widehat{\mathbf{T}}}\, \mathbf{J}_{\mathbf{X}}(\mathbf{0}) \tag{5.29}$$

1. Le Jacobien $\mathbf{J}_{\mathcal{I}}$ est une matrice de dimensions (1×3) qui correspond à la dérivée spatiale de l'image courante ayant subie la transformation projective planaire $\mathbf{w}(\mathbf{G}(\widehat{\mathbf{T}}))$. En effet, nous avons :
$$\mathbf{J}_{\mathcal{I}} = \nabla_{\mathbf{z}} \mathcal{I}\left(\mathbf{w}(\mathbf{G}(\widehat{\mathbf{T}}))(\mathbf{z})\right)\Big|_{\mathbf{z}=\mathbf{w}(\mathbf{I})(\mathbf{p}_i^*)} \tag{5.30}$$
et en utilisant l'équation (2.28), nous pouvons écrire :
$$\mathbf{J}_{\mathcal{I}} = \nabla_{\mathbf{z}} \mathcal{I}\left(\mathbf{w}(\mathbf{G}(\widehat{\mathbf{T}}))(\mathbf{z})\right)\Big|_{\mathbf{z}=\mathbf{p}_i^*} \tag{5.31}$$
Le point $\mathbf{p}_i^* \in \mathbb{P}^2$ est de dimensions (3×1) avec la troisième composante constante et égale à 1. Nous ne considérons ici que les dérivées suivant les deux premières composantes et la troisième composante du vecteur ligne $\mathbf{J}_{\mathcal{I}}$ sera égale à zéro.

2. Le Jacobien $\mathbf{J}_{\mathbf{w}}$ est une matrice de dimensions (3×9) qui correspond à la dérivée du vecteur $\mathbf{w}(\mathbf{Z})(\mathbf{p}_i^*)$ par rapport aux éléments de la matrice $\mathbf{Z}$:
$$\mathbf{J}_{\mathbf{w}} = \nabla_{\mathbf{Z}} \mathbf{w}(\mathbf{Z})(\mathbf{p}_i^*)|_{\mathbf{Z}=\mathbf{G}(\widehat{\mathbf{T}})^{-1}\mathbf{G}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{0}))=\mathbf{I}} \tag{5.32}$$

En posant $\mathbf{p}_i^* = [u_i^*\ v_i^*\ 1]^\top$, l'expression explicite de ce Jacobien s'écrit :

$$\mathbf{J_w} = \begin{bmatrix} \mathbf{p}_i^{*\top} & \mathbf{0} & -u_i^*\,\mathbf{p}_i^{*\top} \\ \mathbf{0} & \mathbf{p}_i^{*\top} & -v_i^*\,\mathbf{p}_i^{*\top} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{5.33}$$

3. En utilisant l'équation (2.25), nous pouvons écrire :

$$\mathbf{G}(\widehat{\mathbf{T}})^{-1}\mathbf{G}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x})) = \mathbf{K}\mathbf{H}(\widehat{\mathbf{T}})^{-1}\mathbf{H}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x}))\mathbf{K}^{-1}$$

Par conséquent, le troisième Jacobien $\mathbf{J_K}$ est calculé en utilisant l'équation (2.25) comme suit :

$$\mathbf{J_K} = \nabla_{\mathbf{Z}}\mathbf{K}\mathbf{Z}\mathbf{K}^{-1}\big|_{\mathbf{Z}=\mathbf{H}(\widehat{\mathbf{T}})^{-1}\mathbf{H}(\widehat{\mathbf{T}})=\mathbf{I}} \tag{5.34}$$

Ce Jacobien ne dépend que de la matrice $\mathbf{K}$ des paramètres intrinsèques de la caméra. Il s'agit d'une matrice de dimensions (9×9) :

$$\mathbf{J_K} = \mathbf{K} \otimes \mathbf{K}^{-\top} = \begin{bmatrix} f\mathbf{K}^{-\top} & fs\mathbf{K}^{-\top} & u_0\mathbf{K}^{-\top} \\ \mathbf{0} & fr\mathbf{K}^{-\top} & v_0\mathbf{K}^{-\top} \\ \mathbf{0} & \mathbf{0} & \mathbf{K}^{-\top} \end{bmatrix}$$

où $\otimes$ correspond au produit de Kronecker. Les deux matrices Jacobiennes $\mathbf{J_w}$ et $\mathbf{J_K}$ sont constantes. Par conséquent, le produit $\mathbf{J_w}\mathbf{J_K}$ peut être calculé une fois pour toute.

4. La quatrième et la cinquième matrices Jacobiennes sont respectivement $\mathbf{J}_{\widehat{\mathbf{T}}}$:

$$\mathbf{J}_{\widehat{\mathbf{T}}} = \nabla_{\mathbf{Z}}\ \mathbf{H}(\widehat{\mathbf{T}})^{-1}\mathbf{H}(\widehat{\mathbf{T}}\mathbf{Z})\Big|_{\mathbf{Z}=\mathbf{I}} \tag{5.35}$$

et $\mathbf{J_X}(\mathbf{0})$:

$$\mathbf{J_X}(\mathbf{0}) = \nabla_{\mathbf{z}}\mathbf{T}(\mathbf{z})|_{\mathbf{z}=\mathbf{0}}$$

La matrice Jacobienne $\mathbf{J_X}(\mathbf{0})$ est constante. Nous pouvons calculer directement le produit $\mathbf{J}_{\widehat{\mathbf{T}}}\mathbf{J_X}(\mathbf{0})$. Si nous écrivons la matrice de transformation $\widehat{\mathbf{T}}$ de la manière suivante :

$$\widehat{\mathbf{T}} = \begin{bmatrix} \widehat{\mathbf{R}} & \widehat{\mathbf{t}} \\ \mathbf{0} & 1 \end{bmatrix}$$

et si nous notons par $\widehat{\boldsymbol{\tau}} = [\widehat{\tau}_x\ \widehat{\tau}_y\ \widehat{\tau}_z]^\top$ le vecteur de dimensions (3×1) suivant :

$$\widehat{\boldsymbol{\tau}} = \frac{-\widehat{\mathbf{R}}^\top\widehat{\mathbf{t}}}{1 + \mathbf{n}^{*\top}\widehat{\mathbf{R}}^\top\widehat{\mathbf{t}}}$$

Le produit $\mathbf{J}_{\widehat{\mathbf{T}}}\mathbf{J}_{\mathbf{X}}(\mathbf{0})$ est une matrices de dimensions (9×6) qui s'écrit :

$$\mathbf{J}_{\widehat{\mathbf{T}}}\mathbf{J}_{\mathbf{X}}(\mathbf{0}) = \begin{bmatrix} \mathbf{n}^* \ (\widehat{\tau}_x\mathbf{n}^* + \mathbf{b}_x)^\top & [\widehat{\tau}_x\mathbf{n}^* + \mathbf{b}_x]_\times \\ \mathbf{n}^* \ (\widehat{\tau}_y\mathbf{n}^* + \mathbf{b}_y)^\top & [\widehat{\tau}_y\mathbf{n}^* + \mathbf{b}_y]_\times \\ \mathbf{n}^* \ (\widehat{\tau}_z\mathbf{n}^* + \mathbf{b}_z)^\top & [\widehat{\tau}_z\mathbf{n}^* + \mathbf{b}_z]_\times \end{bmatrix}$$

Cette matrice dépend de la matrice $\widehat{\mathbf{T}}$. Donc, elle doit être mise à jour à chaque itération de la minimisation.

5.3.2.2 Le Jacobien de référence

Le Jacobien de référence $\mathbf{J}(\widetilde{\mathbf{x}})$ représente la dérivée du vecteur $\mathbf{y}(\mathbf{x})$ calculé au point $\mathbf{x} = \widetilde{\mathbf{x}}$ vérifiant la contrainte (5.10). La ligne i de Jacobien $\mathbf{J}(\widetilde{\mathbf{x}})$ s'écrit sous la forme suivante :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\widetilde{\mathbf{x}}} = \nabla_{\mathbf{x}} \mathcal{I}\left(\mathbf{w}\left(\mathbf{G}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x}))\right)(\mathbf{p}_i)\right)\Big|_{\mathbf{x}=\widetilde{\mathbf{x}}} \tag{5.36}$$

En utilisant l'équation (5.6), nous pouvons écrire :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\widetilde{\mathbf{x}}} \quad = \quad \nabla_{\mathbf{x}} \mathcal{I}^*\left(\mathbf{w}\left(\mathbf{G}(\overline{\mathbf{T}})^{-1}\mathbf{G}(\widehat{\mathbf{T}}\mathbf{T}(\mathbf{x}))\right)(\mathbf{p}_i)\right)\Big|_{\mathbf{x}=\widetilde{\mathbf{x}}}$$

La ligne i du Jacobien de référence peut également être écrite comme le produit de 5 Jacobiens différents :

$$\nabla_{\mathbf{x}} y_i(\mathbf{x})|_{\mathbf{x}=\widetilde{\mathbf{x}}} \quad = \quad \mathbf{J}_{\mathcal{I}^*}\ \mathbf{J}_{\mathbf{w}}\ \mathbf{J}_{\mathbf{K}}\ \mathbf{J}_{\overline{\mathbf{T}}}\ \mathbf{J}_{\mathbf{X}}(\widetilde{\mathbf{x}})$$

1. Le premier Jacobien $\mathbf{J}_{\mathcal{I}^*}$ est une matrice de dimensions (1×3) qui correspond à la dérivée spatiale de l'imagette de référence :

$$\mathbf{J}_{\mathcal{I}^*} = \nabla_{\mathbf{z}} \mathcal{I}^*(\mathbf{z})|_{\mathbf{z}=\mathbf{p}_i^*} \tag{5.37}$$

2. Les matrices Jacobiennnes $\mathbf{J}_{\mathbf{w}}$ et $\mathbf{J}_{\mathbf{K}}$ sont les mêmes matrices constantes définies dans les équations (5.33) et (5.34).

3. Si nous écrivons la matrice de transformation $\overline{\mathbf{T}}$ sous la forme :

$$\overline{\mathbf{T}} = \begin{bmatrix} \overline{\mathbf{R}} & \overline{\mathbf{t}} \\ \mathbf{0} & 1 \end{bmatrix}$$

nous pouvons définir le vecteur $\overline{\boldsymbol{\tau}}$ de dimensions (3×1) comme suit :

$$\overline{\boldsymbol{\tau}} = \frac{-\overline{\mathbf{R}}^\top \overline{\mathbf{t}}}{1 + \mathbf{n}^{*\top}\overline{\mathbf{R}}^\top \overline{\mathbf{t}}}$$

Le Jacobien $\mathbf{J}_{\overline{\mathbf{T}}}$ est similaire au Jacobien $\mathbf{J}_{\mathbf{T}}$ donné dans l'équation (5.35), mais le vecteur $\boldsymbol{\tau}$ est remplacé par $\overline{\boldsymbol{\tau}}$:

$$\mathbf{J}_{\overline{\mathbf{T}}} = \nabla_{\mathbf{Z}}\mathbf{H}(\overline{\mathbf{T}})^{-1}\mathbf{H}(\overline{\mathbf{T}}\mathbf{Z})\big|_{\mathbf{Z}=\mathbf{I}}$$

Si nous avions $\mathbf{H}(\overline{\mathbf{T}})^{-1}\mathbf{H}(\overline{\mathbf{T}}\mathbf{T}) = \mathbf{H}(\mathbf{T})$ alors le Jacobien peut être calculé sans connaître $\overline{\mathbf{T}}$. Mais en général, ce Jacobien ne peut pas être calculé exactement car $\overline{\mathbf{T}}$ est inconnu. Cependant, il peut être approché par $\mathbf{J}_{\overline{\mathbf{T}}} \approx \mathbf{J}_{\widehat{\mathbf{T}}}$ en supposant que $\widehat{\mathbf{T}} \approx \overline{\mathbf{T}}$. Le dernier Jacobien s'écrit :

$$\mathbf{J}_{\mathbf{X}}(\widetilde{\mathbf{x}}) = \nabla_{\mathbf{z}}\overline{\mathbf{T}}^{-1}\widehat{\mathbf{T}}\mathbf{T}(\mathbf{z})\Big|_{\mathbf{z}=\widetilde{\mathbf{x}}} = \nabla_{\mathbf{z}}\mathbf{T}(\widetilde{\mathbf{x}})^{-1}\mathbf{T}(\mathbf{z})\big|_{\mathbf{z}=\widetilde{\mathbf{x}}}$$

Ce Jacobien dépend généralement de l'inconnue $\widetilde{\mathbf{x}}$. Or, grâce à la paramétrisation utilisant l'algèbre de Lie $\mathfrak{se}(3)$, nous pouvons montrer que :

$$\mathbf{J}_{\mathbf{X}}(\widetilde{\mathbf{x}})\widetilde{\mathbf{x}} = \mathbf{J}_{\mathbf{X}}(\mathbf{0})\widetilde{\mathbf{x}} \tag{5.38}$$

La démonstration de cette formule est analogue à celle de l'équation (4.46).

5.3.3 Minimisation itérative du système d'équations

Dans l'approche que nous proposons, nous choisissons comme paramétrisation de la matrice de transformation projective planaire dans l'image $\mathbf{G}$ celle définie dans le paragraphe 5.3.1. L'approche développée dans le chapitre précédent peut être appliquée à la fonction de vecteur $\mathbf{y}(\mathbf{x})$ au voisinage de $\mathbf{x} = \mathbf{0}$. L'équation (5.12) peut donc s'écrire sans avoir à calculer les matrices Hessiennes de $\mathbf{y}(\mathbf{x})$:

$$\mathbf{y}(\mathbf{x}) = \mathbf{y}(\mathbf{0}) + \frac{1}{2}\left(\mathbf{J}(\mathbf{0}) + \mathbf{J}(\mathbf{x})\right)\mathbf{x} + \mathbf{O}(\|\mathbf{x}\|^3) \tag{5.39}$$

Il s'agit d'une approximation au second-ordre de la fonction $\mathbf{y}(\mathbf{x})$ dans le voisinage de $\mathbf{x} = \mathbf{0}$. Quand nous avons $\mathbf{x} = \widetilde{\mathbf{x}}$, nous avons :

$$\mathbf{y}(\widetilde{\mathbf{x}}) \approx \mathbf{y}(\mathbf{0}) + \frac{1}{2}\left(\mathbf{J}(\mathbf{0}) + \mathbf{J}(\widetilde{\mathbf{x}})\right)\widetilde{\mathbf{x}} \tag{5.40}$$

Si nous utilisons l'expression de $\mathbf{J}(\mathbf{0})$ et de $\mathbf{J}(\widetilde{\mathbf{x}})$ du paragraphe 5.3.2, nous pouvons écrire :

$$\mathbf{J}(\mathbf{0}) + \mathbf{J}(\widetilde{\mathbf{x}}) \approx \mathbf{J}_{\mathcal{I}}\mathbf{J}_{\mathbf{w}}\,\mathbf{J}_{\mathbf{K}}\,\mathbf{J}_{\widehat{\mathbf{T}}}\mathbf{J}_{\mathbf{X}}(\mathbf{0}) + \mathbf{J}_{\mathcal{I}^*}\mathbf{J}_{\mathbf{w}}\,\mathbf{J}_{\mathbf{K}}\,\mathbf{J}_{\widehat{\mathbf{T}}}\mathbf{J}_{\mathbf{X}}(\widetilde{\mathbf{x}}) \tag{5.41}$$

Grâce à la formule de l'équation (5.38), l'équation (5.40) peut s'écrire :

$$\mathbf{y}(\widetilde{\mathbf{x}}) \approx \mathbf{y}(\mathbf{0}) + \frac{1}{2}\left(\mathbf{J}_{\mathcal{I}} + \mathbf{J}_{\mathcal{I}^*}\right)\,\mathbf{J}_{\mathbf{w}}\,\mathbf{J}_{\mathbf{K}}\,\mathbf{J}_{\widehat{\mathbf{T}}}\,\mathbf{J}_{\mathbf{X}}(\mathbf{0})\,\widetilde{\mathbf{x}} \tag{5.42}$$

D'une manière analogue à celle du paragraphe 4.2, la solution du système (5.9) par la méthode des moindres carrés peut être obtenue d'une manière itérative. Nous définissons la matrice $\mathbf{J}_{esm}$:

$$\mathbf{J}_{esm} = \frac{1}{2}\left(\mathbf{J}_{\mathcal{I}} + \mathbf{J}_{\mathcal{I}^*}\right)\ \mathbf{J}_{\mathbf{w}}\ \mathbf{J}_{\mathbf{K}}\ \mathbf{J}_{\widehat{\mathbf{T}}}\ \mathbf{J}_{\mathbf{X}}(\mathbf{0}) \tag{5.43}$$

Il s'agit de minimiser d'une manière itérative la fonction de coût suivante :

$$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{y}(\mathbf{0}) + \mathbf{J}_{esm}\mathbf{x}\|^2 \tag{5.44}$$

La condition nécessaire (5.15) pour qu'un vecteur $\mathbf{x} = \widetilde{\mathbf{x}}$ soit un minimum local ou global de la fonction de coût f est que la dérivée de la fonction de coût soit nulle. Nous avons alors :

$$\nabla_{\mathbf{x}} f(\mathbf{x})|_{\mathbf{x}=\widetilde{\mathbf{x}}} = \mathbf{J}_{esm}^{\top}\left(\mathbf{y}(\mathbf{0}) + \mathbf{J}_{esm}\widetilde{\mathbf{x}}\right) = \mathbf{0} \tag{5.45}$$

Comme dans le paragraphe 4.2, nous procédons d'une manière itérative. D'abord nous calculons :

$$\widetilde{\mathbf{x}} = \mathbf{J}_{esm}^{+}\mathbf{y}(\mathbf{0}) \tag{5.46}$$

Ensuite, nous mettons à jour l'approximation $\widehat{\mathbf{T}}$ comme défini dans l'équation (5.19). À chaque mise à jour de $\widehat{\mathbf{T}}$, nous estimons $\mathbf{y}(\mathbf{0})$ et $\mathbf{J}_{\mathcal{I}}$ et nous calculons à nouveau $\widetilde{\mathbf{x}}$. Nous arrêtons l'algorithme quand la valeur de $\widetilde{\mathbf{x}}$ calculée devient très faible.

5.3.4 Simulations

5.3.4.1 Suivi d'une scène planaire par morceaux

Afin d'avoir une vérité terrain, nous avons généré une séquence d'images d'une scène synthétique planaire par morceaux. Les différentes textures planaires de la scène sont extraites d'images réelles afin de rendre la simulation la plus réaliste possible. Dans cette simulation, nous nous proposons de suivre trois régions planaires le long d'une séquence de 100 images. La caméra effectue une trajectoire en boucle fermée tout en gardant l'objet suivi dans le champ de vision.

Nous comparons deux stratégies différentes :

– Dans la première stratégie, nous estimons la transformation projective (une homographie) dans l'image de chaque région planaire d'une manière indépendante (voir la Figure 5.1(a)). En effectuant le SV des plans séparément, nous utilisons la version de la méthode ESM présentée dans le chapitre précédent. Ensuite, connaissant la matrice des paramètres intrinsèques $\mathbf{K}$ de la caméra et le vecteur normal à chaque plan, nous déterminons la rotation et la translation de la caméra en utilisant la méthode proposée dans (Faugeras and Lustman, 1988). Lors de cette stratégie, la contrainte que le déplacement de la caméra est le même pour tous les plans ne peut pas être imposée. Une deuxième étape est

alors nécessaire pour obtenir un déplacement cohérent avec les résultats du SV des plans séparément.

– Dans la deuxième stratégie, nous effectuons le SV des plans d'une manière unifiée (voir la Figure 5.1(b)). Étant donné la matrice des paramètres intrinsèques $\mathbf{K}$ de la caméra et le vecteur normal à chaque plan, grâce à la paramétrisation des déplacements incrémentaux dans le groupe $\mathbb{SE}(3)$ (et son algèbre associée $\mathfrak{se}(3)$), il est possible d'imposer la contrainte Euclidienne que la rotation et la translation de la caméra sont les mêmes pour les trois transformations projectives des plans (les trois homographies). Pour chaque image de la séquence, nous estimons directement la rotation et la translation de la caméra par rapport à un système de coordonnées de référence.

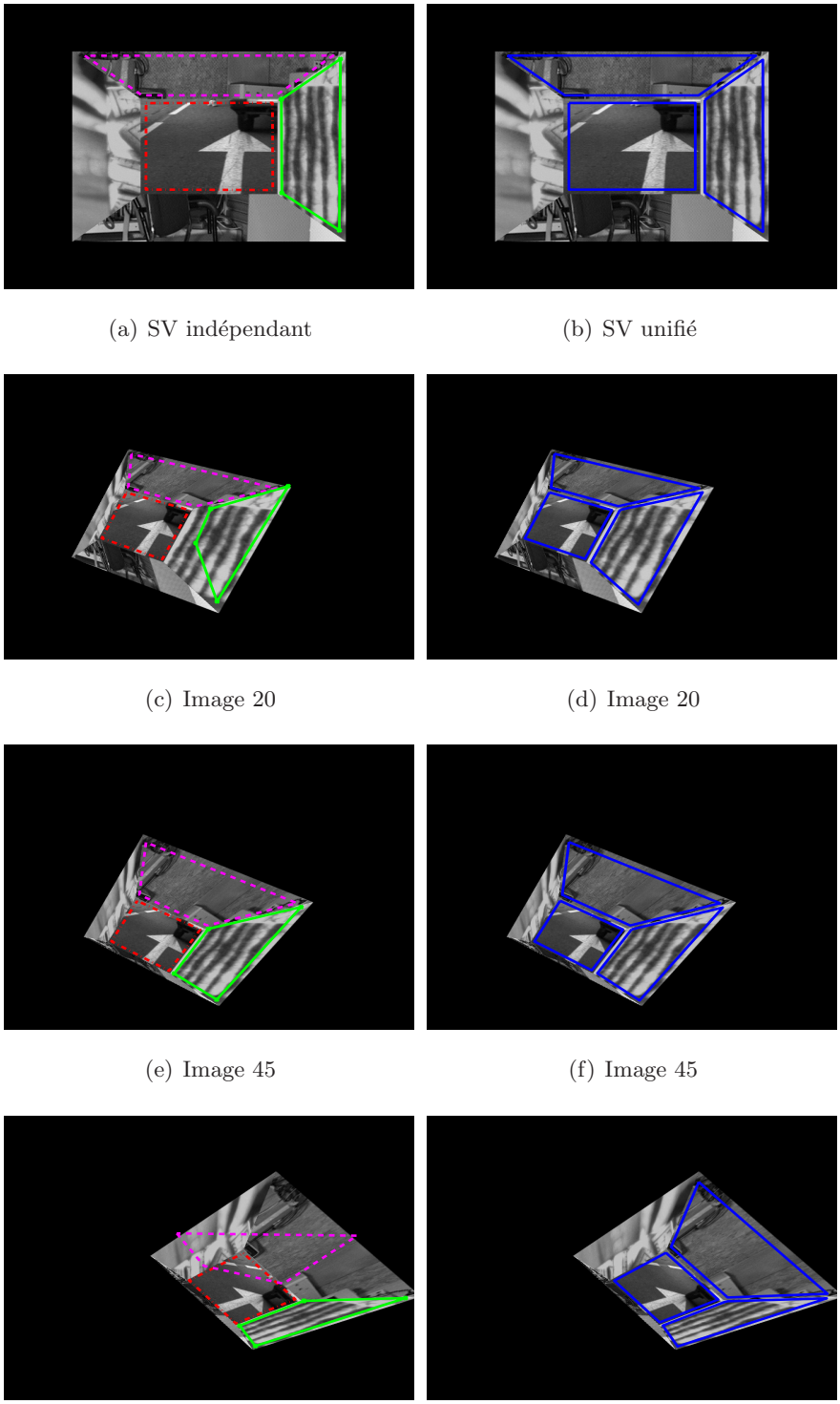
Visuellement, il est possible de voir que le SV des plans d'une manière unifiée donne de meilleurs résultats que le SV des plans d'une manière indépendante (voir quelques images extraites de la séquence vidéo dans la Figure 5.1). Dans la colonne de droite, en bleu, nous avons les zones suivies d'une manière unifiée. L'estimation de la position des zones est très précise. Alors que, dans la colonne de gauche, en rouge, en magenta et en vert, nous avons les même zones suivies d'une manière indépendante. Le suivi est clairement moins précis : par exemple, dans la Figure 5.1(g), nous pouvons voir que l'un des plans suivis a été pratiquement perdu par l'algorithme de SV.

S'agissant d'une simulation, nous connaissons parfaitement le mouvement effectué par l'objet suivi. Dans la Figure 5.2, nous comparons le déplacement simulé avec celui estimé par chacune des deux stratégies utilisées pour le SV. Nous traçons l'erreur entre les vraies rotations et translations et celles qui sont estimées. Dans la colonne de gauche, nous comparons les erreurs de rotation suivant chacun des axes. Dans la colonne de droite, nous comparons les erreurs de translation suivant chacune des directions. Les courbes en magenta, en rouge et en vert représentent les erreurs d'estimation du déplacement du premier, du deuxième et du troisième plan obtenues lors du SV effectué d'une manière indépendante. La courbe bleue correspond à l'erreur d'estimation du déplacement de l'objet en tenant compte des trois plans d'une manière unifiée.

En effectuant le SV d'une manière unifiée, les erreurs commises sont beaucoup moins importantes que lors du SV des plans indépendamment. En effet, dans le cas unifié, l'erreur en rotation ne dépasse pas $3.13\ 10^{-4}$ radians et l'erreur en translation ne dépasse pas $3.32\ 10^{-4}$ mètres. Alors que, dans le cas du SV des plan indépendamment, l'erreur en rotation d'un des plans atteint 2.03 radians et l'erreur en translation atteint 3.29 mètres.

5.3.4.2 Avantages de la minimisation au second-ordre

Nous comparons maintenant la minimisation au second-ordre (notée "SO") et deux algorithmes de minimisation standards au premier-ordre. Le premier algorithme utilise seulement le Jacobien de référence (noté "RJ"), le deuxième algorithme utilise seulement le Jacobien courant (noté "CJ"). Les trois algorithmes utilisent la paramétrisation avec l'algèbre de Lie décrite



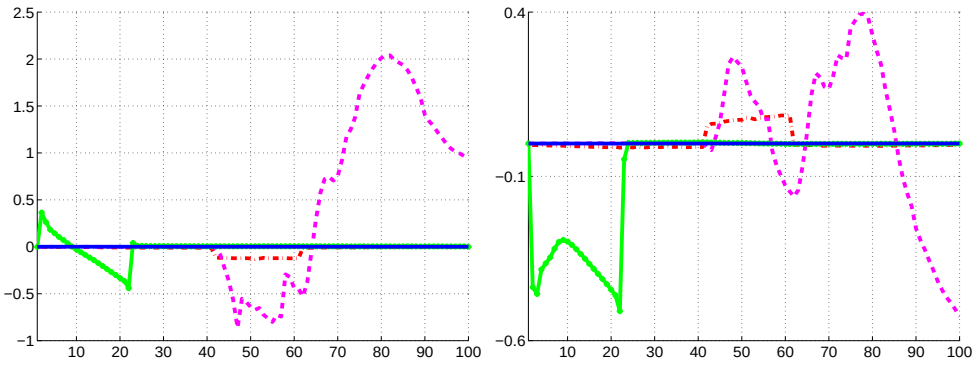
(a) SV indépendant (b) SV unifié

(c) Image 20 (d) Image 20

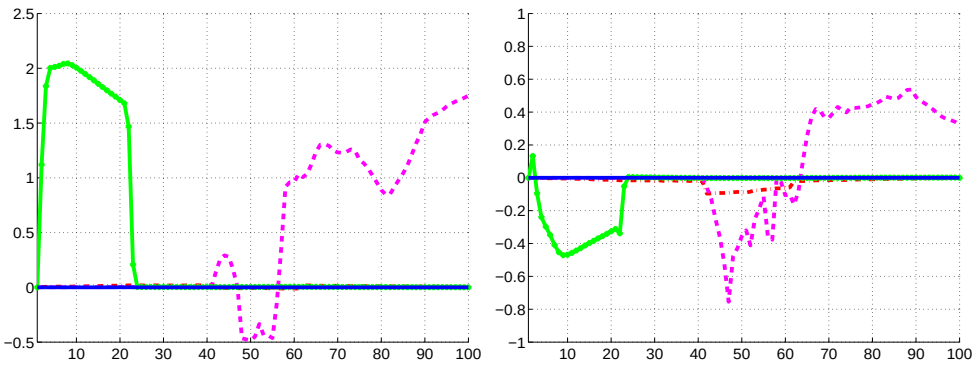
(e) Image 45 (f) Image 45

(g) Image 70 (h) Image 70

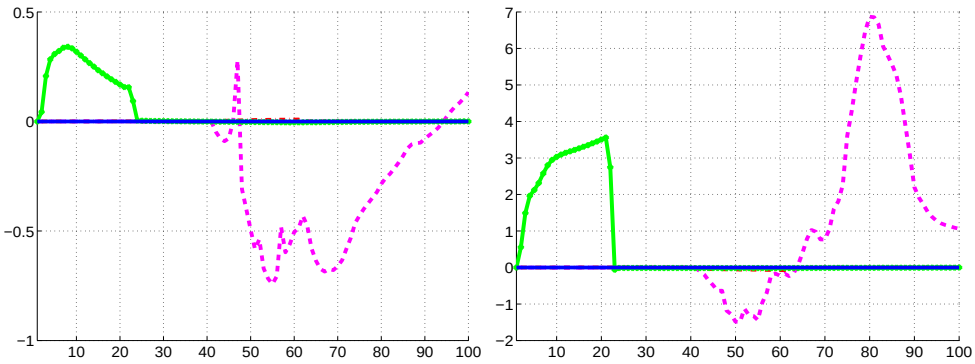
Figure 5.1 – Quelques images du SV de la séquence simulée



(a) Erreur en θu_x (radians) (b) Erreur en t_x (mètres)



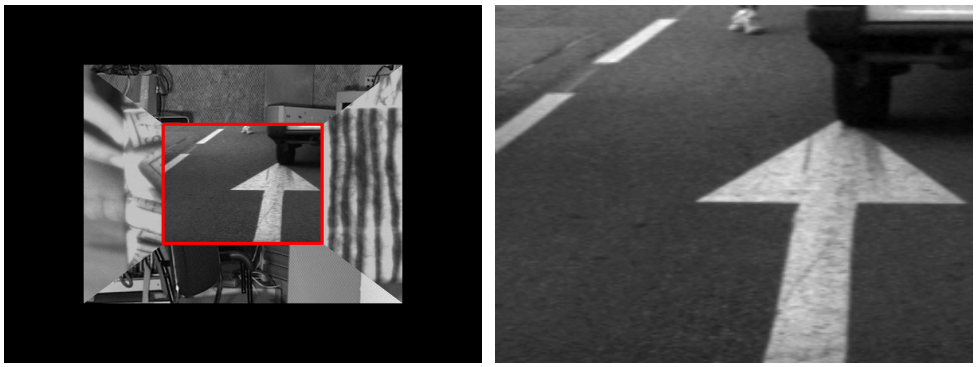
(c) Erreur en θu_y (radians) (d) Erreur en t_y (mètres)



(e) Erreur en θu_z (radians) (f) Erreur en t_z (mètres)

Figure 5.2 – Comparaison des erreurs de déplacement estimé par les deux stratégies

dans le paragraphe 5.3.1. Nous utilisons également la même scène planaire par morceaux que précédemment. L'imagette dans la Figure 5.3(b) a été sélectionnée dans le centre de l'image à la position de référence (voir la Figure 5.3(a)). La taille de l'imagette est de (200×200) pixels. Afin d'avoir la même plate-forme de simulation, nous utilisons la librairie Matlab disponible sur la page Web de Dr. Simon Baker au laboratoire Robotics Institute of the Carnegie Mellon University.



(a) L'image à la position de référence (b) L'imagette de référence

Figure 5.3 – L'image et l'imagette de référence

Nous simulons des mouvements aléatoires de la caméra observant la scène. Les rotations et translations effectuées sont calculées de façons à ce que leur effet dans l'image produit un bruit Gaussien sur les coordonnées des 4 coins de l'imagette de référence. L'écart type σ du bruit Gaussien varie de 1 à 10 avec des pas de 0.5. Il représente en quelque sorte l'amplitude du déplacement de la caméra. Pour chaque valeur de σ, nous appliquons à l'imagette 1000 transformations projectives planaires aléatoires. Pour chaque algorithme testé, nous effectuons 15 itérations. Nous considérons un algorithme comme ayant convergé, si la moyenne de la norme de l'erreur des coordonnées des 4 coins à l'issue des 15 itérations est inférieure à 1 pixel.

Dans la Figure 5.4, nous traçons les fréquences de convergence (% sur 1000 tests) en fonction de l'écart type σ du bruit appliqué. Pour de faibles déplacements (jusqu'à $\sigma = 2$), nous pouvons voir que les 3 algorithmes ont des fréquences de convergence équivalentes et qu'ils ils convergent quasiment tout le temps. Ce qui n'est pas surprenant car pour de faibles déplacements ($\widetilde{\mathbf{x}} \approx \mathbf{0}$), l'approximation au premier-ordre est vérifiée. Le Jacobien courant et le Jacobien de référence sont très proches et nous avons :

$$\mathbf{J}_{esm} = \frac{1}{2}(\mathbf{J}(\mathbf{0}) + \mathbf{J}(\widetilde{\mathbf{x}})) \approx \mathbf{J}(\mathbf{0}) \approx \mathbf{J}(\widetilde{\mathbf{x}})$$

Plus σ est grand, plus la transformation entre l'imagette de référence et l'imagette initiale est importante. Les fréquences de convergence des 3 algorithmes diminuent quand σ augmente. Cependant, nous remarquons que l'approximation de second-ordre "SO" permet à la fréquence

de convergence de cet algorithme de décroître moins vite que celle des deux algorithmes au premier-ordre “RJ” et “CJ”. Pour $\sigma = 8$, alors que la fréquence de convergence de l'algorithme au second-ordre est de 90% celle des deux autres algorithmes est déjà à 70%. Pour $\sigma = 10$, les fréquences de convergence des deux algorithmes au premier-ordre sont autour de 50%, tandis que la fréquence de convergence de l'algorithme au second-ordre dépasse les 80%. Nous pouvons donc conclure que pour de grands déplacements l'algorithme au second-ordre converge plus souvent que les deux algorithmes au premier-ordre.

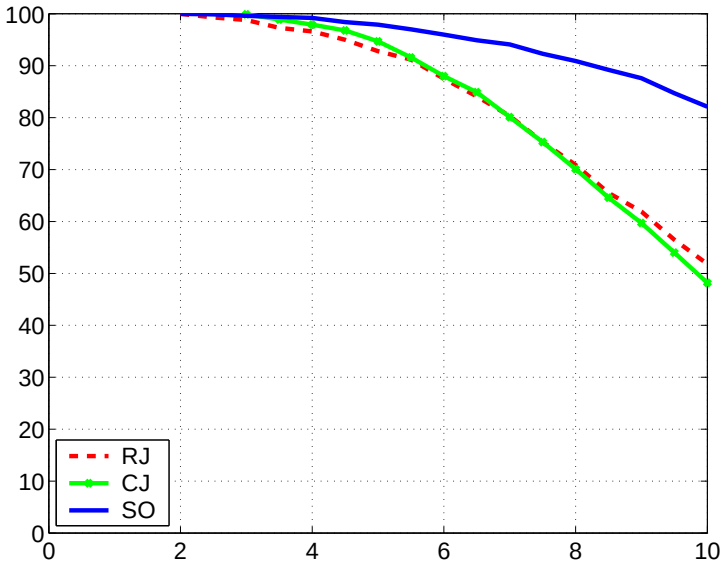


Figure 5.4 – Comparaison de la fréquence de convergence

Maintenant, nous comparons le taux de convergence moyen (sur 1000 tests) des différents algorithmes. Nous ne considérons que les tests où les 3 algorithmes ont convergé afin de ne pas biaiser le résultat par des cas où un ou plusieurs algorithmes ont divergé. Cette comparaison nous permet de voir le nombre d'itérations nécessaire à chacun des algorithmes pour converger et ce pour différentes amplitudes de déplacement. Elle nous permet également d'observer la décroissance de la norme de l'erreur entre l'imagette de référence et l'imagette courante. Dans la Figure 5.5, nous traçons la norme de l'erreur entre l'imagette de référence et l'imagette courante pour de grands déplacements ($\sigma = 10$). Partant d'une norme d'erreur de même valeur, nous remarquons que la décroissance de la norme de l'erreur de la méthode au second-ordre est plus rapide que les deux algorithmes au premier-ordre. En effet, pour converger, la méthode au second-ordre nécessite seulement 7 itérations, les deux autres méthodes nécessitent, en moyenne, plus de 14 itérations. Il est donc possible, avec l'algorithme proposé, d'effectuer un SV en temps réel avec de grandes fréquences d'acquisition car la vitesse de convergence est 2 fois plus rapide que les algorithmes au premier-ordre.

Dans les simulations présentées, les vecteurs normaux aux plans suivis ne contiennent pas

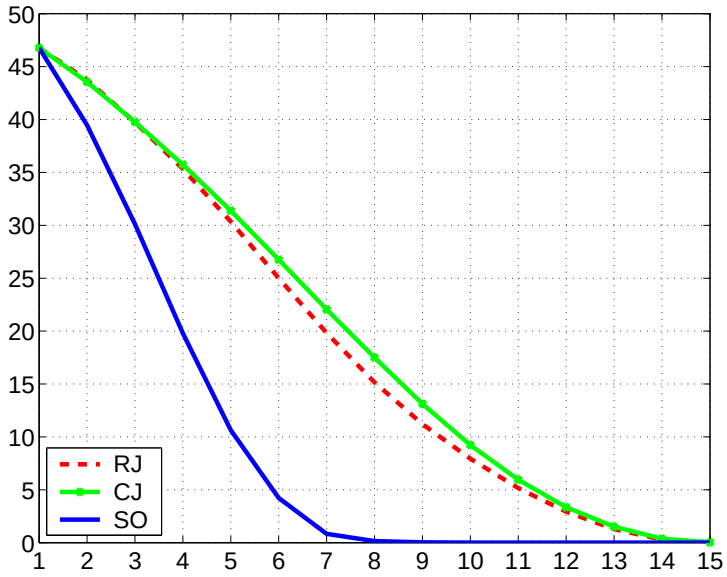


Figure 5.5 – Comparaison du taux de convergence

d'erreurs. Les algorithmes de suivi visuels ont donc été comparés avec des normales correctes et l'algorithme ESM a montré qu'il permet d'améliorer grandement les propriétés de convergences du suivi visuel.

Dans le paragraphe suivant, nous effectuons des expériences où les vecteur normaux aux plans suivis sont grossièrement estimés. Cela nous permettra de justifier l'utilisation de l'algorithme proposé dans le cas pratique des applications robotiques.

5.4 Résultats expérimentaux

Nous avons testé notre algorithme sur une séquence de 1000 images acquises par une caméra montée sur le robot mobile de l'équipe ICARE à l'INRIA Sophia Antipolis (voir la Figure 5.6) le long d'une trajectoire de plus de 6 mètres. Ce robot possède un capteur odométrique très précis. L'estimation du mouvement par ce capteur sera comparée à celle obtenue par le système de vision et le SV de notre algorithme.

Trois régions planaires texturées de la scène observée par la caméra ont été sélectionnées dans la première image acquise. Chaque région appartient à un plan différent. La caméra a été calibrée et les vecteurs normaux aux plans ont été estimés grossièrement. Dans la Figure 5.7, nous pouvons voir quelques images extraites de la séquence. Nous pouvons voir que le déplacement dans l'image des différentes régions suivies est bien estimé tout au long de la séquence (il s'agit des régions encadrées par un rectangle rouge dans les images).

Dans la figure 5.8, nous pouvons voir que la trajectoire estimée par le suivi visuel (en trait continu) et celle obtenue grâce aux capteurs odométriques (en trait pointillé) sont très proches.

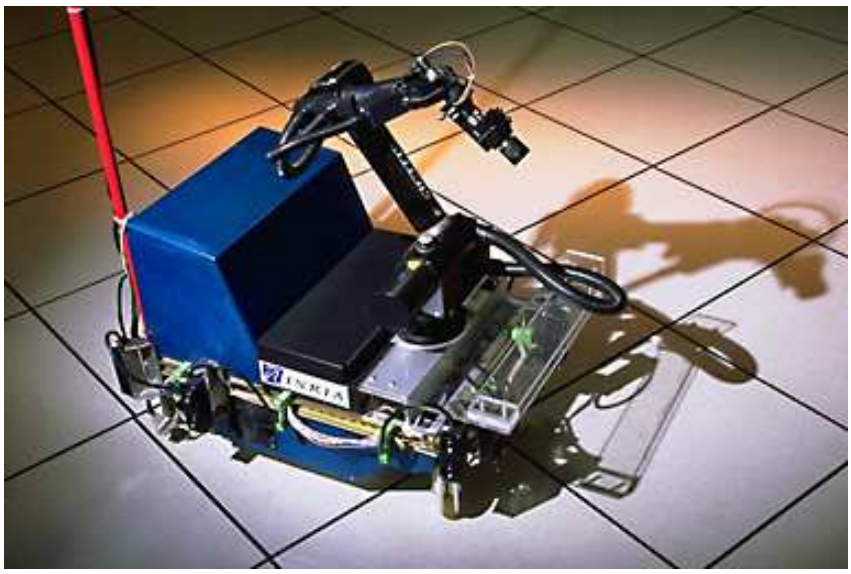


Figure 5.6 – Le robot Anis : une base mobile munie d'un bras manipulateur

Bien qu'il existe quelques faibles erreurs dans l'estimation du mouvement, il n'y a pas de dérive.

Malgré le fort grossissement dû à une translation dans la direction des différents plans et malgré le fait que l'un des plans est partiellement sorti du champ de vue de la caméra, l'estimation du déplacement 3D est, visuellement, très précise.

En effet, dans la Figure 5.9, nous pouvons voir les courbes de la translation 5.9(a) et de la rotation 5.9(b) estimées par le capteur odométrique (en traits pointillés) et estimées par la vision (en traits continus). La valeur absolue de l'erreur entre les deux estimations de la rotation est inférieure à 1.8 degré et l'erreur entre les deux estimations de la translation est inférieure à 12 centimètres (voir les Figures 5.9(c) et 5.9(d)). Sachant que le capteur odométrique utilisé est très précis, nous pouvons conclure que la méthode de SV proposée donne de très bonnes estimations. Ces courbes permettent de confirmer que, malgré quelques faibles erreurs dans l'estimation du mouvement, il n'y a pas de dérive.

5.5 Conclusion

Dans ce chapitre, nous avons présenté un suivi visuel permettant de résoudre à la fois le problème de suivi dans l'image et l'estimation du mouvement relatif entre la caméra et la scène (dans l'espace Cartésien), le tout dans un cadre unifié. Les informations sur le modèle de la caméra et de la scène sont introduites a priori, c'est-à-dire, la rotation et la translation Cartésiennes sont obtenues directement à l'issue de l'algorithme. L'approximation au second-ordre a permis, encore une fois, d'améliorer le domaine et le taux de convergence des algorithmes classiques tout en ayant une complexité équivalente. Les simulations numériques et les résultats expérimentaux ont prouvé l'efficacité de la méthode. Par ailleurs, notre approche a été

Image 0000

Image 0250

Image 0350

Image 0500

Image 0650

Image 0750

Image 0900

Image 1000

Figure 5.7 – Quelques images de la séquence acquise par le robot

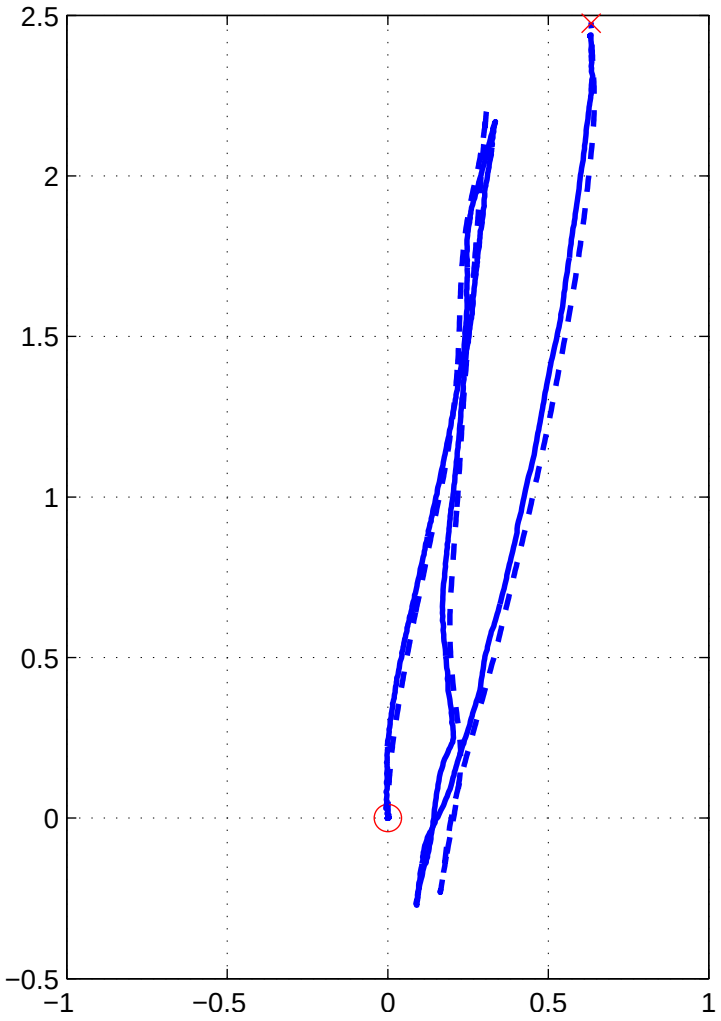
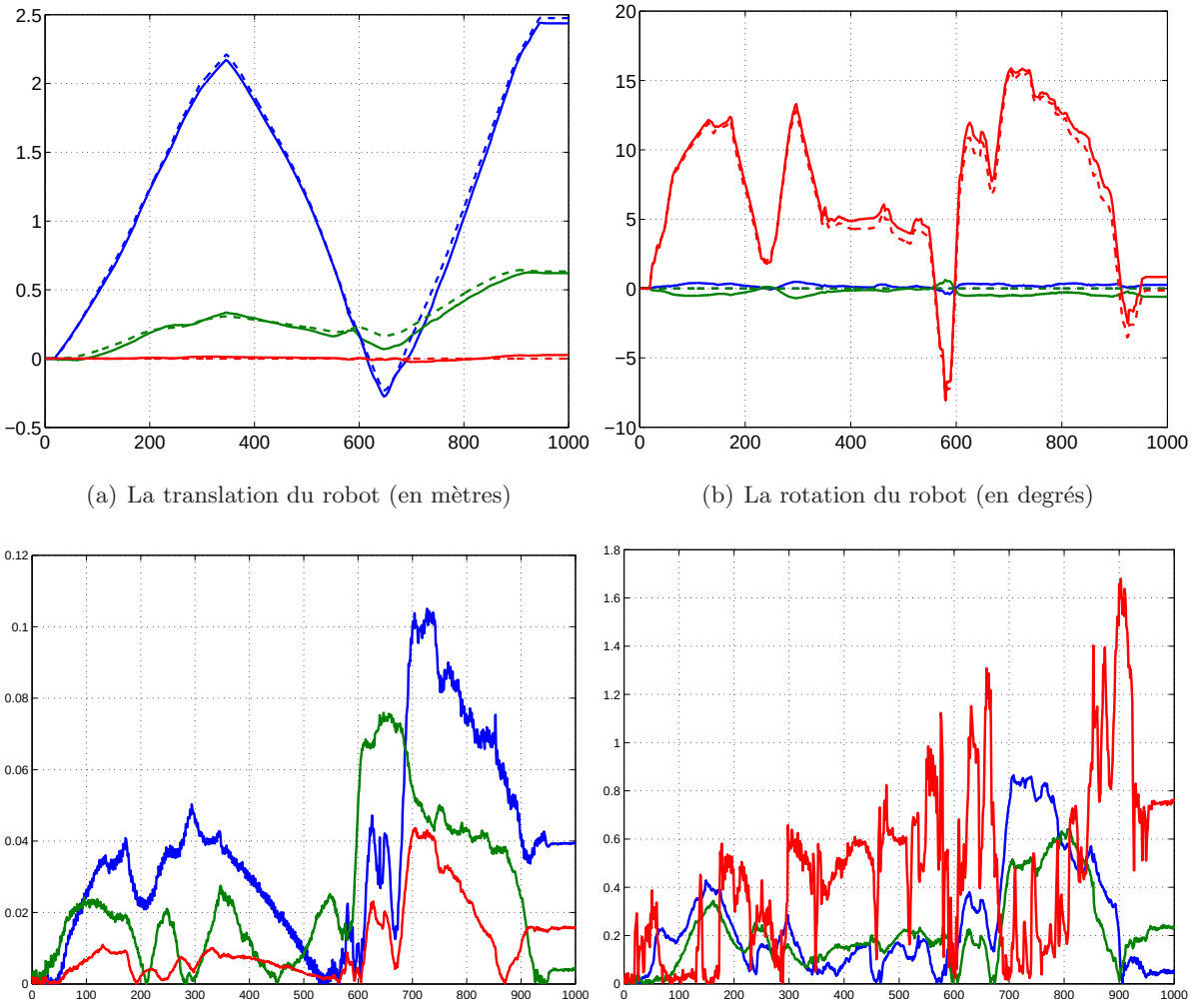


Figure 5.8 – La trajectoire du robot estimée par le SV et par l'odométrie



(a) La translation du robot (en mètres)

(b) La rotation du robot (en degrés)

(c) L'erreur en translation du robot (en mètres)

(d) L'erreur en rotation du robot (en degrés)

Figure 5.9 – Comparaison des déplacements estimés : par le SV et par l'odométrie

étendue aux caméras munies de capteurs omnidirectionnelles. Pour plus de détail, voir les deux publications suivantes : (Mei et al., 2006a; Mei et al., 2006b).

Au moment de l'écriture du manuscrit, nous sommes en train de développer des approches de suivi visuel, utilisant l'algorithme ESM, où les vecteurs normaux aux plans sont estimés en ligne afin de ne plus avoir à fournir cette information. L'apparition de nouvelles facettes susceptibles d'être suivies et leur intégration dans l'algorithme de suivi est également une direction naturelle à suivre pour la complétudes du travail. La robustesse aux mesures aberrantes dues à des occultations partielles et aux changement d'éclairement fait aussi partie des sujets qui nous intéressent désormais. Nous sommes en train d'adapter certaines approches telles que celle proposée dans (Comport et al., 2004) pour tenir compte d'éventuelles erreurs de mesure.

Troisième partie

Asservissement visuel

Résumé

Les différentes approches d'asservissement visuel sont classées selon la manière avec laquelle les informations visuelles extraites des images acquises sont utilisées pour construire la fonction de tâche, c'est-à-dire, selon l'espace dans lequel la fonction de tâche est exprimée. Dans la littérature, nous pouvons distinguer 3 catégories d'asservissement visuel qui ont été proposées : l'asservissement visuel 3D, l'asservissement visuel 2D et l'asservissement visuel 2D 1/2. L'inconvénient commun à toutes ces méthodes est qu'une information 3D du modèle de la cible observée est nécessaire soit pour établir une fonction de tâche localement isomorphe à l'attitude de la caméra soit pour construire une loi de commande stable ou bien garantir sa stabilité. L'objectif de ce chapitre est de proposer une nouvelle méthode d'asservissement visuel stable qui ne nécessite aucune mesure a priori de la structure 3D de la cible observée et par rapport à laquelle l'asservissement est effectué. Seules des informations visuelles issues d'une image capturée à la position de référence et d'une image capturée à la position courante doivent suffire pour :

- calculer une fonction de tâche localement isomorphe à l'attitude de la caméra ;
- calculer la loi de commande stable.

Nous commençons le chapitre par un bref état de l'art des différentes méthodes d'asservissement visuel existantes dans la littérature. Nous effectuons une classification de ces méthodes et nous expliquons les aspects qui ne nous satisfont pas dans ces méthodes et les raisons qui nous ont amenés à proposer une nouvelle approche. Ensuite, nous décrivons la méthode que nous proposons. Nous prouvons de manière théorique l'existence d'une fonction de tâche localement isomorphe à l'attitude de la caméra et la stabilité locale de la loi de commande. Des résultats expérimentaux ont été obtenus avec deux robots différents à 6 degrés de liberté. Nous testons expérimentalement la robustesse de la méthodes aux différentes incertitudes dans les paramètres intrinsèques de la caméra et des mesures dans l'image.

Chapitre 6

État de l'art

6.1 Introduction

L'asservissement visuel (AV) est une tâche robotique qui consiste à commander un robot grâce à des informations visuelles issues d'une ou de plusieurs caméras (Hashimoto, 1993; Hutchinson et al., 1996). Les caméras sont soit montées sur le robot soit en train de l'observer. Cette tâche peut être exprimée comme la régulation d'une fonction de tâche $\mathbf{e}(\mathbf{q}, t)$ qui dépend de la configuration articulaire $\mathbf{q}$ du robot et du temps t (Samson et al., 1991).

Nous nous intéressons aux asservissements visuels dont l'objectif peut être posé en terme de réalisation d'un positionnement désiré d'une caméra montée sur un robot par rapport à une cible de modèle inconnu appartenant à la scène observée. Nous supposons qu'une image de référence $\mathcal{I}^*$ de la scène est capturée dans une certaine position (position de référence) du robot par rapport à laquelle nous voulons le repositionner. Partant d'une autre position permettant de voir la cible sous un angle différent (position initiale), le robot est alors commandé afin qu'il atteigne la position de référence. À chaque image acquise $\mathcal{I}$ à un instant t (image courante), un algorithme de suivi visuel fournit une estimation des informations visuelles qui dépendent de la position relative entre la cible et la caméra et qui permettent de calculer la commande. Nous considérons que la mise en correspondence (l'initialisation du suivi visuel) est effectuée manuellement ou bien par un algorithme adéquat (tel celui proposé par (Lowe, 2004)). À la convergence de l'asservissement visuel, le robot a atteint la position de référence, si les informations visuelles observées coïncident avec celles de l'image de référence.

Les différentes approches d'asservissement visuel sont classées selon la manière avec laquelle les informations visuelles extraites des images acquises sont utilisées pour construire la fonction de tâche $\mathbf{e}(\mathbf{q}, t)$, c'est-à-dire, selon l'espace dans lequel la fonction de tâche $\mathbf{e}(\mathbf{q}, t)$ est exprimée. Dans la littérature, nous pouvons distinguer 3 catégories d'asservissement visuel qui ont été proposées : l'asservissement visuel 3D, l'asservissement visuel 2D et l'asservissement visuel 2D 1/2.

6.2 Méthodes d'asservissement visuel 3D

Les méthodes de la première catégorie sont appelées AV 3D. La fonction de tâche $\mathbf{e}(\mathbf{q}, t)$ s'exprime dans l'espace Cartésien, c'est-à-dire, les informations visuelles issues des deux images (de référence et courante) sont utilisées pour reconstruire explicitement l'erreur d'attitude (translation et rotation) de la caméra (Wilson et al., 1996; Martinet et al., 1997; Basri et al., 1998; Taylor et al., 2000; Malis and Chaumette, 2002).

Il est possible de reconstruire l'erreur d'attitude en connaissant le modèle 3D complet d'un certain nombre de points de la cible (Lowe, 1991; Dementhon and Davis, 1995). Dans ce cas, l'attitude de la caméra dans la position de référence et dans la position courante est d'abord reconstruite, puis, l'erreur d'attitude est calculée. L'estimation de l'erreur en translation (à un facteur d'échelle près) et de l'erreur en rotation de la caméra peuvent être obtenues directement à partir de la matrice essentielle (Longuet-Higgins, 1981; Hartley, 1992; Faugeras, 1993; Svoboda and Pajdla, 2002; Geyer and Daniilidis, 2003). Toutefois, l'estimation de la matrice essentielle n'est pas possible dans certains cas particuliers (Longuet-Higgins, 1984), par exemple quand la cible observée est plane ou bien quand la caméra effectue une rotation pure. Il est donc préférable d'estimer la translation (à un facteur d'échelle près) et la rotation de la caméra à partir d'une matrice d'homographie (Malis et al., 2000; Hadj Abdelkader et al., 2005). Dans le cas d'une cible planaire, la reconstruction à partir de deux images et d'une matrice d'homographie n'est pas unique (2 solutions sont possibles). Pour choisir la bonne solution, il faut soit avoir une approximation du vecteur normal au plan de la cible, soit utiliser plus de deux images.

Grâce à l'estimation explicite des erreurs dans l'espace Cartésien le découplage de la fonction de tâche est possible, c'est-à-dire, la commande en rotation et la commande en translation deviennent indépendantes. Ceci implique qu'une commande générée suite à une erreur de rotation (resp. de translation) n'induit pas un mouvement de translation (resp. de rotation). L'inconvénient de cette approche est que la commande, étant générée avec des erreurs dans l'espace Cartésien, ne tient pas compte de la visibilité de la cible dans l'image. En effet, si le déplacement initial de la caméra est important, il est possible que la cible sorte du champ de vue de la caméra rendant la tâche non observable. Certaines approches ont été proposée pour pallier cet inconvénient. Par exemple, dans (Mezouar and Chaumette, 2002), un asservissement visuel 2D est proposé couplé à une planification de trajectoire en 3D. Cette planification de trajectoire permet de garantir que la visibilité de la cible dans le champ de vue de la caméra. Du coup, le système "asservissement visuel + planification de trajectoire" devient très complexe.

6.3 Méthodes d'asservissement visuel 2D

Les méthodes de la deuxième catégorie sont appelées AV 2D. La fonction de tâche $\mathbf{e}(\mathbf{q}, t)$ s'exprime dans l'image, c'est-à-dire que ces asservissements visuels ne nécessitent pas l'estimation explicite de l'erreur d'attitude dans l'espace Cartésien (Espiau et al., 1992; Barreto et al.,

2003; Chaumette, 2004; Malis, 2004; Mezouar et al., 2004).

Ces méthodes cherchent à construire une fonction de tâche localement isomorphe à l'attitude de la caméra. Pour que la tâche de positionnement puisse être accomplie par la commande, les informations extraites des images acquises doivent être isomorphes à la position Cartésienne du repère de la caméra. À notre connaissance, sauf cas particuliers de cibles construites "ad hoc" (Cowan and Chang, 2002), l'isomorphisme est généralement admis sans aucune preuve formelle. Une fonction de tâche réellement isomorphe à l'attitude de la caméra permettrait d'éviter les situations où la fonction de tâche est nulle mais la caméra n'est pas correctement positionnée (Chaumette, 1998). D'une manière générale, la fonction de tâche est construite à partir de primitives visuelles simples telles que les coordonnées dans l'image des points d'intérêt.

L'avantage de ces méthodes est que la cible a plus de chance de rester visible par la caméra car la régulation a lieu dans l'image, et ce, même si le déplacement initial est important. Cependant, l'inconvénient de telles méthodes d'asservissement visuel est que la trajectoire du robot dans l'espace Cartésien n'est pas optimale car la fonction de tâche n'est pas découplée, c'est-à-dire que la commande en rotation et la commande en translation ne sont pas indépendantes. En d'autres termes, une commande générée suite à une erreur de rotation (resp. de translation) peut induire un mouvement de translation (resp. de rotation). C'est pour cette raison que différentes méthodes ont été proposées afin de découpler la fonction de tâche autant que possible (Corke and Hutchinson, 2001; Tahri and Chaumette, 2005). Une autre alternative pour résoudre le problème de la trajectoire définie dans l'image est d'utiliser une commande au deuxième ordre (Malis, 2004; Lapreste and Mezouar, 2004).

Il existe des cas particuliers de configuration de cible pour lesquels les AV 2D ne sont pas stables ou que l'isomorphisme entre les données visuelles et l'attitude de la caméra n'est pas vérifié (Chaumette, 1998). Afin d'avoir une commande stable, il est nécessaire d'avoir une bonne estimation de la distribution des profondeurs ce qui correspond à avoir la normale au plan de la cible dans le cas d'une cible planaire (Malis and Rives, 2003). Toutefois, l'expérience montre (sans preuve formelle) que la stabilité locale pour des configurations et des cibles "normales" est vérifiée avec une estimation "plus ou moins bonne" de la distribution des profondeurs. Il est à noter que, souvent, la construction de la fonction de tâche dépend également de la distribution des profondeurs.

6.4 Méthodes d'asservissement visuel 2D 1/2

Les méthodes de la troisième catégorie sont appelées AV 2D 1/2. Ce sont des méthodes hybrides où la fonction de tâche $\mathbf{e}(\mathbf{q}, t)$ s'exprime à la fois dans l'espace Cartésien et dans l'image, c'est-à-dire, l'erreur en rotation est reconstruite explicitement et l'erreur en translation est exprimée dans l'image. L'erreur en translation est, par exemple, l'erreur des coordonnées métriques étendues d'un point de l'espace projectif. Il est possible de montrer que la fonction

de tâche ainsi définie est localement isomorphe à l'attitude de la caméra.

Initialement introduite par (Malis et al., 1997), cette catégorie a été utilisée dans divers travaux (Deguchi, 1998; Malis et al., 1999; Corke and Hutchinson, 2001; Hadj Abdelkader et al., 2005), sous d'autres appellations notamment "asservissement visuel hybride" ou bien "asservissement visuel partionné". L'avantage de ce type d'asservissements visuels est qu'il permet de découpler la commande en rotation et la commande en translation. Grâce à un tel asservissement visuel, le contrôle a lieu en partie dans l'image. Par conséquent, nous avons plus de probabilité de garder la cible dans l'image. Un autre avantage de cette approche est qu'il est possible, en supposant être capable de mesurer la matrice d'homographie du plan de l'infini, de démontrer analytiquement la stabilité et la robustesse de la loi de commande (Malis and Chaumette, 2002).

Cependant, de la même manière que pour les AV 3D, la reconstruction à partir d'une matrice d'homographie n'est pas unique (2 solutions sont possibles). Pour choisir la bonne solution, il faut avoir une approximation du vecteur normal au plan de la cible. Si aucune approximation n'est disponible, la stabilité de l'AV 2D 1/2 n'est pas garantie.

Une estimation en ligne du vecteur normal au plan de la cible peut également être envisagée afin d'avoir une meilleure stabilité. Mais, une telle estimation alourdit le système global.

6.5 Conclusion

L'inconvénient commun à toutes ces méthodes est qu'une information 3D du modèle de la cible observée est nécessaire. Dans le cas des AV 2 1/2 D et 3D, la reconstruction n'est pas unique (2 solutions sont possibles). Pour choisir la bonne solution, il faut soit avoir une approximation du vecteur normal au plan, soit utiliser plus de deux images. Récemment, une méthode d'asservissement basée sur une combinaison des deux solutions possibles a été proposée dans (Vargas and Malis, 2005). Toutefois, cette méthode qui nécessite la décomposition de l'homographie n'est pas encore aboutie et une commutation de loi de commande rend difficile la preuve de stabilité. Dans les cas des AV 2D, afin d'avoir une commande stable (Malis and Rives, 2003), il est nécessaire d'avoir une estimation de la distribution des profondeurs (ou bien avoir la normale au plan de la cible).

Dans le chapitre suivant, nous proposons une nouvelle méthode d'AV 2D qui permet de commander un robot en construisant une fonction de tâche localement isomorphe à l'attitude de la caméra dans l'espace Cartésien. Nous proposerons un isomorphisme local entre une fonction de tâche $\mathbf{e}$ mesurée seulement à partir des deux images (référence et courante) et un sous-ensemble de l'attitude de la caméra (c'est-à-dire, la fonction $\mathbf{e}(\mathbf{q}, t)$ est nulle seulement lorsque la caméra occupe l'attitude désirée par rapport à la cible). Contrairement à l'AV 2D classique, nous démontrerons analytiquement que nous n'avons pas besoin d'informations 3D pour garantir la stabilité de la commande. La construction de l'erreur passe par une estimation d'une homographie qui

lie, entre deux images, les points d'un même plan. Une fois l'homographie estimée, le calcul de la fonction de tâche et de la commande est très simple (pas de matrice d'interaction à estimer, et pas de décomposition de la matrice d'homographie) et ne nécessite aucune information 3D de la scène observée.

Malgré la simplicité de la méthode proposée, il s'agit de l'aboutissement de nombreuses recherches dans le domaine de l'asservissement visuel. La force de cette approche réside aussi dans cette simplicité (simple à comprendre, à mettre en œuvre et à implémenter). Grâce à sa généralité, cette méthode peut être non seulement appliquée quand le capteur est une simple caméra perspective mais aussi quand la caméra est omnidirectionnelle.

Par ailleurs, contrairement aux méthodes d'asservissement visuel classiques, la méthode proposée ne sépare pas la partie vision et la partie commande. En effet, généralement, les asservissements visuel standards peuvent être représentés grâce au schéma bloc de la Figure 6.1. Ces approches nécessitent une extraction puis un suivi visuel des primitives. Ce qui permet, grâce à une étape intermédiaire, d'estimer la position courante (ou plus généralement la fonction de tâche). Cette étape intermédiaire est, dans le cas de l'AV 2D 1/2 par exemple, l'estimation et la décomposition de l'homographie.

L'approche que nous proposons ne nécessite pas d'extraction de primitive car le suivi visuel est effectué directement à partir des intensités lumineuses dans l'image. Le résultat du suivi visuel est directement l'homographie. Cette homographie est utilisée telle quelle pour le calcul de la fonction de tâche. Ce qui réduit le schéma bloc de la méthode proposée à la forme de la Figure 6.2.

Cette nouvelle approche permet d'unifier le suivi visuel temps réel et l'asservissement visuel. Le résultat du suivi est l'entrée de la commande de l'asservissement.

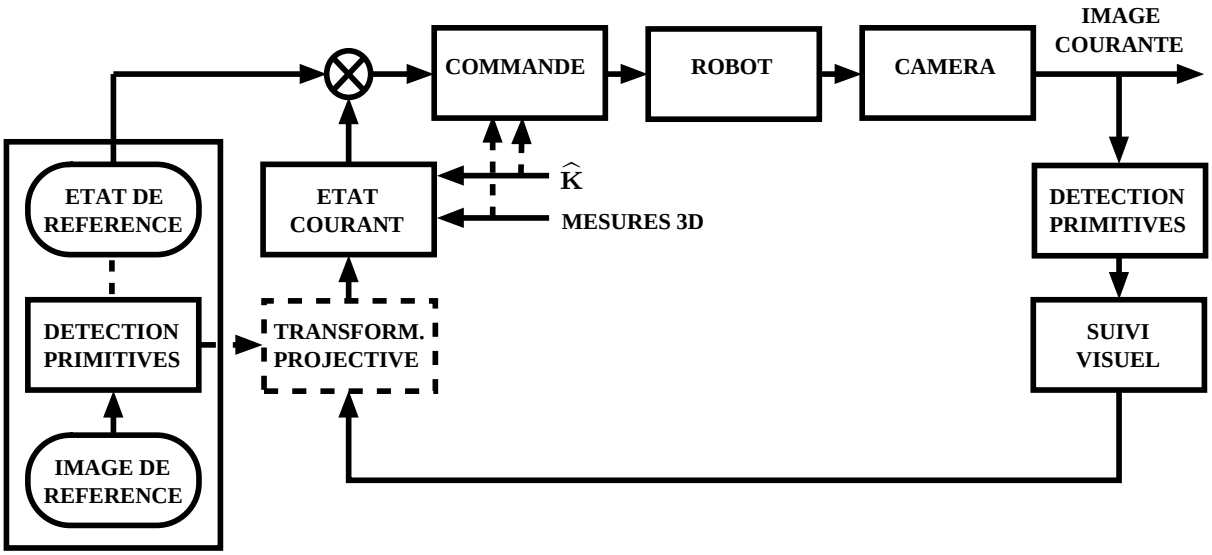


Figure 6.1 – Schéma standard de l'asservissement visuel couplé avec la vision

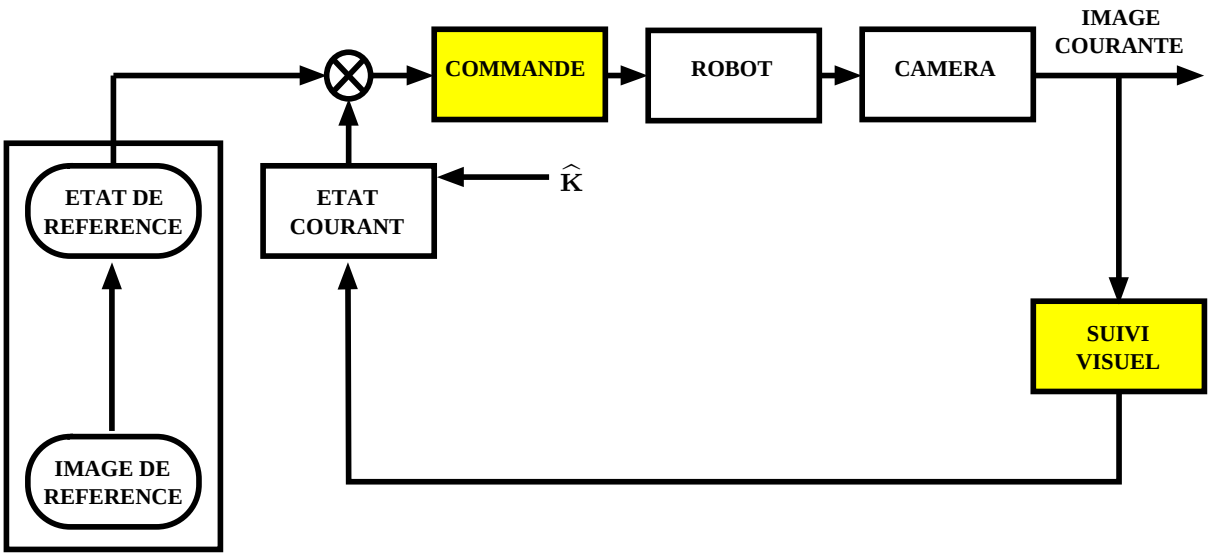


Figure 6.2 – Schéma de l'asservissement visuel direct couplé avec la vision

Chapitre 7

Asservissement visuel 2D direct

7.1 Introduction

L'objectif de ce chapitre est de concevoir une méthode d'asservissement visuel qui ne nécessite pas de mesures de la structure 3D (la normale au plan et la distance par rapport à la caméra) de la cible planaire observée. Pour ce faire, nous devons définir un isomorphisme local entre l'attitude de la caméra et les informations visuelles extraites des images de référence et courante seulement. À partir de cet isomorphisme local, nous calculons une loi de commande stable et indépendante de la configuration 3D de la cible, et ce, également, à l'aide d'informations visuelles uniquement.

Les informations visuelles extraites des images serviront à construire une fonction de tâche $\mathbf{e}(\mathbf{q}, t)$ qui permettra de calculer la loi de commande de la vitesse de la caméra dans l'espace Cartésien. En pratique, une fois cette vitesse calculée, nous appliquons le Jacobien inverse du robot pour déterminer les vitesses $\dot{\mathbf{q}}$ des différentes articulations du robot. Nous nous intéressons dans ce chapitre seulement au calcul de la vitesse dans l'espace Cartésien. Sans perte de généralité, nous considérons le robot à commander comme étant Cartésien. Nous ne traitons que la partie cinématique de la commande, la partie dynamique ne sera pas considérée. C'est pour cette raison que nous noterons la fonction de tâche $\mathbf{e}(\mathbf{q}, t)$ simplement par $\mathbf{e}$.

7.2 Quelques rappels théoriques

Comme nous l'avons dit dans le paragraphe 6.1, nous considérons que l'objectif de l'asservissement visuel est de contrôler un robot sur lequel une caméra est montée de manière à ce que le repère courant $\mathcal{F}$ attaché à la caméra dans sa position actuelle se superpose au repère de référence $\mathcal{F}^*$ attaché à la caméra quand l'image de référence $\mathcal{I}^*$ a été acquise. Nous noterons par $\mathcal{I}$ l'image acquise en temps-réel par la caméra dans sa position actuelle et nous supposons que l'objet par rapport auquel l'asservissement visuel s'effectue peut être vu par la caméra dans les deux positions (la position de référence et la position courante).

Soit $\mathcal{F}^* = (\mathcal{O}^*, \vec{i}^*, \vec{j}^*, \vec{k}^*)$ le repère de référence de l'espace Cartésien. Soient $\mathbf{R}$ la matrice de rotation et $\mathbf{t}$ le vecteur de translation entre le repère de référence $\mathcal{F}^*$ et le repère courant $\mathcal{F} = (\mathcal{O}, \vec{i}, \vec{j}, \vec{k})$. Nous supposons que la caméra observe une cible planaire appartenant à un plan qui se situe à une distance d^* du centre $\mathcal{O}^*$ et ayant un vecteur normal $\mathbf{n}^*$ exprimé dans le repère $\mathcal{F}^*$ où $\mathbf{n}^*$ vérifie : $\|\mathbf{n}^*\| = \sqrt{\mathbf{n}^{*\top}\mathbf{n}^*} = d^{*-1}$.

La commande que nous allons présenter se base sur l'existence d'une matrice d'homographie $\mathbf{H}$ permettant de mettre en correspondance les projections dans deux images acquises dans différentes positions d'un même point 3D appartenant à un certain plan.

Dans le cas d'une caméra perspective standard, nous savons qu'il existe une matrice d'homographie $\mathbf{H}$ qui permet de mettre en correspondance les points du plan π des deux images $\mathcal{I}_m^*$ et $\mathcal{I}_m$ et nous avons : $\mathbf{H} \propto \mathbf{R} + \mathbf{t}\mathbf{n}^{*\top}$.

7.3 Définition de la fonction de tâche

Il est évident que les deux repères $\mathcal{F}$ et $\mathcal{F}^*$ coïncident, si et seulement si la matrice $\mathbf{H}$ est égale à la matrice identité $\mathbf{I}$. Nous construisons, une fonction d'erreur $\mathbf{e} \in \mathbb{R}^6$, la fonction de tâche, de la manière suivante :

$$\mathbf{e} = \begin{bmatrix} \mathbf{e}_\nu \\ \mathbf{e}_\omega \end{bmatrix} = \begin{bmatrix} Z^{*-1}(\mathbf{t} + (\mathbf{R} - \mathbf{I})\boldsymbol{\mathcal{X}}^*) \\ 2\sin(\theta)\mathbf{u} + [\mathbf{n}^*]_\times \mathbf{t} \end{bmatrix} \tag{7.1}$$

où $\boldsymbol{\mathcal{X}}^* = [X^* \ Y^* \ Z^*]^\top$ sont les coordonnées exprimées dans $\mathcal{F}^*$ d'un point de l'espace $\mathcal{P}$ visible dans l'image de référence et dans l'image courante. Ce choix sera justifié dans la suite du développement.

La fonction de tâche ainsi définie peut être complètement calculée à partir des deux images $\mathcal{I}$ et $\mathcal{I}^*$ seulement, c'est-à-dire sans connaître aucune mesure du modèle de la cible ($\mathbf{n}^*$ et Z^*) et sans avoir à reconstruire la pose. Il suffit pour cela d'estimer la matrice d'homographie $\mathbf{H}$. Or, ayant déjà préposé dans la partie précédente un suivi visuel qui permet d'estimer d'une manière efficace et précise l'homographie $\mathbf{H}$, le résultat du suivi visuel peut être directement utilisé dans le calcul de la fonction de tâche. Ce qui fait l'unité de la partie vision et de la partie commande.

7.3.1 Expression de $\mathbf{e}_\nu$ à l'aide de H

D'après l'équation (7.1), le vecteur $\mathbf{e}_\nu$ s'écrit :

$$\mathbf{e}_\nu = Z^{*-1}(\mathbf{t} + (\mathbf{R} - \mathbf{I})\boldsymbol{\mathcal{X}}^*) = Z^{*-1}(\mathbf{R}\boldsymbol{\mathcal{X}}^* + \mathbf{t} - \boldsymbol{\mathcal{X}}^*) \tag{7.2}$$

En utilisant l'équation (2.5), $\mathbf{e}_\nu$ devient :

$$\mathbf{e}_\nu = Z^{*-1}(\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{X}}^*)$$

D'après les équations (2.8) et (2.9), nous pouvons écrire :

$$\mathbf{e}_\nu = Z^{*-1} Z\mathbf{m} - \mathbf{m}^*$$

Enfin, grâce à l'équation (2.13), $\mathbf{e}_\nu$ peut s'écrire en fonction de $\mathbf{H}$ et de $\mathbf{m}^*$ seulement :

$$\mathbf{e}_\nu = \mathbf{H}\mathbf{m}^* - \mathbf{m}^* = (\mathbf{H} - \mathbf{I})\mathbf{m}^* \tag{7.3}$$

Le point $\mathbf{m}^*$ est obtenu directement à partir de l'image de référence grâce à l'équation (2.19) et la matrice d'homographie $\mathbf{H}$ est également calculée à partir des images (soit points mis en correspondance entre l'image de référence et l'image courante soit obtenue par l'algorithme de suivi visuel). Par conséquent, le vecteur $\mathbf{e}_\nu$ peut être calculé sans aucune mesure du modèle de la cible.

7.3.2 Expression de $\mathbf{e}_\omega$ à l'aide de H

Grâce à l'équation (2.14), nous avons :

$$\mathbf{H} - \mathbf{H}^\top = \mathbf{R} + \mathbf{t}\mathbf{n}^{*\top} - \mathbf{R}^\top - \mathbf{n}^*\mathbf{t}^\top$$

Or, en utilisant la formule de Rodriguez, la matrice $\mathbf{R}$ s'écrit :

$$\mathbf{R} = \mathbf{I} + \sin(\theta)\,[\mathbf{u}]_\times + 2\cos^2\left(\frac{\theta}{2}\right)[\mathbf{u}]_\times^2$$

nous pouvons écrire :

$$\mathbf{R} - \mathbf{R}^\top = 2\sin(\theta)\,[\mathbf{u}]_\times$$

Donc, sachant la propriété suivante :

$$\mathbf{t}\mathbf{n}^{*\top} - \mathbf{n}^*\mathbf{t}^\top = \left[[\mathbf{n}^*]_\times\,\mathbf{t}\right]_\times$$

la partie antisymétrique de la matrice $\mathbf{H}$ peut s'écrire :

$$\mathbf{H} - \mathbf{H}^\top = \left[2\sin(\theta)\mathbf{u} + [\mathbf{n}^*]_\times\,\mathbf{t}\right]_\times$$

Or d'après l'équation (7.1), le vecteur $\mathbf{e}_\omega$ s'écrit :

$$\mathbf{e}_\omega = 2\sin(\theta)\mathbf{u} + [\mathbf{n}^*]_\times\,\mathbf{t}$$

Par conséquent, nous avons bien :

$$\mathbf{H} - \mathbf{H}^\top = [\mathbf{e}_\omega]_\times \tag{7.4}$$

Le vecteur $\mathbf{e}_\omega$ peut donc être calculé simplement à partir de la matrice d'homographie sans aucune mesure du modèle de la cible.

7.3.3 Théorème de l'isomorphisme

Nous avons montré que la fonction de tâche $\mathbf{e}$ peut être directement obtenue à partir des images de référence et courante via la matrice d'homographie $\mathbf{H}$. Reste à démontrer qu'elle définit un isomorphisme local entre l'attitude de la caméra et les informations visuelles extraites des images de référence et courante.

Théorème 1
Soient $\mathbf{R}$ *la matrice de rotation et* $\mathbf{t}$ *le vecteur de translation entre* $\mathcal{F}^*$ *et* $\mathcal{F}$*, avec* $\mathbf{R} = exp\left(\theta\,[\mathbf{u}]_\times\right)$*, où* $\theta \in \,]-\pi,\pi[$ *et soit* $\mathcal{P}$ *un point de coordonnées* $\boldsymbol{\mathcal{X}}^* = [X^*\ Y^*\ Z^*]^\top$ *exprimés dans le repère* $\mathcal{F}^*$ *et appartenant à un plan* π *de vecteur normal* $\mathbf{n}^*$ *exprimé dans le repère* $\mathcal{F}^*$*. Nous définissons la fonction de tâche* $\mathbf{e}$ *comme suit :*

$$\mathbf{e} = \begin{bmatrix} \mathbf{e}_\nu \\ \mathbf{e}_\omega \end{bmatrix} = \begin{bmatrix} Z^{*-1}(\mathbf{t} + (\mathbf{R} - \mathbf{I})\boldsymbol{\mathcal{X}}^*) \\ 2\sin(\theta)\mathbf{u} + [\mathbf{n}^*]_\times \mathbf{t} \end{bmatrix} \tag{7.5}$$

La fonction $\mathbf{e}$ *est isomorphe localement à l'attitude de la caméra, c'est-à-dire* $\mathbf{e} = \mathbf{0}$*, si et seulement si,* $\theta = 0$ *et* $\mathbf{t} = \mathbf{0}$*.*

Afin de simplifier la preuve du théorème (1), nous montrons les propositions suivantes.

Proposition 1 *: La matrice* $\mathbf{H}\mathbf{H}^\top$ *a une valeur propre égale à 1. Le vecteur propre correspondant s'écrit :* $\mathbf{v} = [\mathbf{R}\mathbf{n}^*]_\times \mathbf{t}$*.*

Preuve de la proposition 1
En utilisant l'équation (2.14), nous avons :

$$\mathbf{H}\mathbf{H}^\top = (\mathbf{R} + \mathbf{t}\mathbf{n}^{*\top})(\mathbf{R}^\top + \mathbf{n}^*\mathbf{t}^\top)$$

Étant donné que $\mathbf{R} \in \mathbb{SO}(3)$ alors $\mathbf{R}\mathbf{R}^\top = \mathbf{I}$. Donc, nous avons :

$$\mathbf{H}\mathbf{H}^\top = \mathbf{I} + \mathbf{t}(\mathbf{R}\mathbf{n}^*)^\top + (\mathbf{R}\mathbf{n}^* + \|\mathbf{n}^*\|^2\mathbf{t})\mathbf{t}^\top$$

La matrice $\mathbf{H}\mathbf{H}^\top$ est la somme de la matrice identité $\mathbf{I}$ et une matrice de rang 2. Donc $\mathbf{H}\mathbf{H}^\top$ admet une valeur propre égale à 1. Soit $\mathbf{v}$ le vecteur qui s'écrit $\mathbf{v} = [\mathbf{R}\mathbf{n}^*]_\times \mathbf{t}$, nous avons par construction :

$$(\mathbf{R}\mathbf{n}^*)^\top \mathbf{v} = 0$$

$$\mathbf{t}^\top \mathbf{v} = 0$$

D'où $\mathbf{v}$ est la valeur propre de $\mathbf{H}\mathbf{H}^\top$ correspondant à la valeur propre 1 :

$$\mathbf{H}\mathbf{H}^\top \mathbf{v} = \mathbf{v}$$

Proposition 2 *:* Si nous avons $\mathbf{H} = \mathbf{H}^\top$ et $\sin(\theta) \neq 0$, alors nous avons $\mathbf{n}^{*\top}\,\mathbf{u} = 0$, $\mathbf{t}^\top\,\mathbf{u} = 0$ et $\mathbf{n}^{*\top}\,\mathbf{v} = 0$ (où $\mathbf{v} = [\mathbf{R}\mathbf{n}^*]_\times\,\mathbf{t}$).

Preuve de la proposition 2
Si nous avons $\mathbf{H} = \mathbf{H}^\top$, alors :

$$2\sin(\theta)\mathbf{u} + [\mathbf{n}^*]_\times\,\mathbf{t} = \mathbf{0} \tag{7.6}$$

En multipliant par $\mathbf{n}^{*\top}$ de chaque côté l'équation (7.6), nous obtenons :

$$2\sin(\theta)\mathbf{n}^{*\top}\mathbf{u} = \mathbf{0}$$

Étant donné que nous supposons que $\sin(\theta) \neq 0$, nous avons :

$$\mathbf{n}^{*\top}\mathbf{u} = 0$$

D'une manière analogue, en multipliant de chaque côté l'équation (7.6) par $\mathbf{t}^\top$, nous avons :

$$\mathbf{t}^\top\mathbf{u} = 0$$

Finalement, en utilisant la formule de Rodriguez des matrices de rotation, nous avons :

$$\begin{aligned}
\mathbf{R}\mathbf{n}^* &= \left(\mathbf{I} + \sin(\theta)\,[\mathbf{u}]_\times + 2\cos^2\left(\frac{\theta}{2}\right)[\mathbf{u}]_\times^2\right)\mathbf{n}^* \\
&= \mathbf{n}^* + \sin(\theta)\,[\mathbf{u}]_\times\,\mathbf{n}^* + 2\cos^2\left(\frac{\theta}{2}\right)[\mathbf{u}]_\times^2\,\mathbf{n}^* \\
&= \mathbf{n}^* + \sin(\theta)\,[\mathbf{u}]_\times\,\mathbf{n}^* + 2\cos^2\left(\frac{\theta}{2}\right)\left(\mathbf{u}\mathbf{u}^\top - \mathbf{I}\right)\mathbf{n}^*
\end{aligned}$$

Étant donné que $\mathbf{n}^{*\top}\mathbf{u} = 0$, alors nous avons :

$$\mathbf{R}\mathbf{n}^* = \mathbf{n}^* + \sin(\theta)\,[\mathbf{u}]_\times\,\mathbf{n}^* - 2\cos^2\left(\frac{\theta}{2}\right)\mathbf{n}^* \tag{7.7}$$

La matrice antisymétrique associée au vecteur $\mathbf{R}\mathbf{n}^*$ s'écrit :

$$[\mathbf{R}\mathbf{n}^*]_\times = [\mathbf{n}^*]_\times + \sin(\theta)\left[[\mathbf{u}]_\times\,\mathbf{n}^*\right]_\times - 2\cos^2\left(\frac{\theta}{2}\right)[\mathbf{n}^*]_\times$$

et ayant $\left[[\mathbf{u}]_\times \mathbf{n}^*\right]_\times = \mathbf{n}^*\mathbf{u}^\top - \mathbf{u}\mathbf{n}^{*\top}$, nous pouvons écrire :

$$[\mathbf{R}\mathbf{n}^*]_\times = [\mathbf{n}^*]_\times + \sin(\theta)\left(\mathbf{n}^*\mathbf{u}^\top - \mathbf{u}\mathbf{n}^{*\top}\right) - 2\cos^2\left(\frac{\theta}{2}\right)[\mathbf{n}^*]_\times$$

En multipliant l'équation de chaque côté par $\mathbf{n}^{*\top}$, nous obtenons :

$$\mathbf{n}^{*\top}[\mathbf{R}\mathbf{n}^*]_\times = \|\mathbf{n}^*\|^2 \sin(\theta)\mathbf{u}^\top \tag{7.8}$$

En multipliant l'équation de chaque côté par $\mathbf{t}$, nous obtenons :

$$\mathbf{n}^{*\top}[\mathbf{R}\mathbf{n}^*]_\times \mathbf{t} = \|\mathbf{n}^*\|^2 \sin(\theta)\mathbf{u}^\top\mathbf{t}$$

Ayant $\mathbf{u}^\top\mathbf{t} = 0$, nous pouvons conclure que :

$$\mathbf{n}^{*\top}\mathbf{v} = 0$$

Proposition 3 *: Si* $\mathbf{H} = \mathbf{H}^\top$, $\mathbf{v} = [\mathbf{R}\mathbf{n}^*]_\times \mathbf{t} = \mathbf{0}$ *et* $\sin(\theta) \neq 0$*, alors nous avons* $\det(\mathbf{H}) = -1$.

Preuve de la proposition 3
Si $\mathbf{v} = [\mathbf{R}\mathbf{n}^*]_\times \mathbf{t} = \mathbf{0}$, il existe $\alpha > 0$ tel que :

$$\mathbf{t} = \alpha\mathbf{R}\mathbf{n}^*$$

À partir de l'équation (7.8), nous obtenons :

$$[\mathbf{n}^*]_\times \mathbf{R}\mathbf{n}^* = \left(\mathbf{n}^{*\top}[\mathbf{R}\mathbf{n}^*]_\times\right)^\top = \|\mathbf{n}^*\|^2 \sin(\theta)\mathbf{u} \tag{7.9}$$

Donc, à partir de l'équation (7.6) et de l'équation (7.9), nous obtenons :

$$2\sin(\theta)\mathbf{u} = -[\mathbf{n}^*]_\times \mathbf{t} = -\alpha[\mathbf{n}^*]_\times \mathbf{R}\mathbf{n}^* = -\alpha\|\mathbf{n}^*\|^2 \sin(\theta)\mathbf{u}$$

En multipliant l'équation de chaque côté par $\mathbf{u}^\top$, nous obtenons :

$$2\sin(\theta) = -\alpha\sin(\theta)\|\mathbf{n}^*\|^2$$

Étant donné que nous supposons $\sin(\theta) \neq 0$, nous pouvons donc écrire :

$$\alpha = -\frac{2}{\|\mathbf{n}^*\|^2}$$

Le déterminant de $\mathbf{H}$ vérifie alors :

$$\det(\mathbf{H}) = 1 + \mathbf{n}^{*\top}\mathbf{R}^{\top}\mathbf{t} = 1 + \alpha\|\mathbf{n}^*\|^2 = -1$$

Preuve du théorème 1

Il est évident que si nous avons $\theta = 0$ et $\mathbf{t} = \mathbf{0}$, alors $\mathbf{e} = \mathbf{0}$.

Maintenant, nous montrons que si nous avons $\mathbf{e} = \mathbf{0}$, alors $\theta = 0$ et $\mathbf{t} = \mathbf{0}$.

Supposons que $\mathbf{e} = \mathbf{0}$. Il est évident que si $\theta = 0$ alors $\mathbf{t} = \mathbf{0}$, et si $\mathbf{t} = \mathbf{0}$ alors $\theta = 0$.

Supposons que $\mathbf{e} = \mathbf{0}$ et que $\mathbf{t} \neq \mathbf{0}$ et $\theta \neq 0$. Si $\mathbf{e}_\nu = \mathbf{0}$ alors $\mathbf{H}\mathbf{m}^* = \mathbf{m}^*$. Donc, $\mathbf{H}$ a une valeur propre égale à 1 et $\mathbf{m}^*$ est le vecteur propre correspondant à cette valeur propre. Le vecteur $\mathbf{m}^*$ est aussi un vecteur propre de la matrice $\mathbf{H}^2$ correspondant à la valeur propre 1. Étant donné que $\mathbf{e}_\omega = \mathbf{0}$ alors $\mathbf{H} = \mathbf{H}^\top$ et $\mathbf{H}^2 = \mathbf{H}\mathbf{H}^\top$. Étant donné la Proposition 1, $\mathbf{m}^*$ est colinéaire au vecteur $\mathbf{v} = [\mathbf{R}\mathbf{n}^*]_\times \mathbf{t}$. Étant donné que nous avons $\det(\mathbf{H}) > 0$, ce vecteur est différent du vecteur nul $\mathbf{0}$ (voir la Proposition 3). D'un autre côté, la Proposition 2 montre que dans ce cas $\mathbf{n}^{*\top}\mathbf{m}^* = Z^* = 0$. Ce qui est impossible car par définition $Z^* > 0$. Donc, il est impossible que $\mathbf{e} = \mathbf{0}$ ayant $\mathbf{t} \neq \mathbf{0}$ et $\theta \neq 0$.

Par conséquent, contrôler la caméra de façon à avoir $\mathbf{e} = \mathbf{0}$ consiste à faire coïncider les deux projections $\boldsymbol{\mathcal{X}}^*$ et $\boldsymbol{\mathcal{X}}$ du point $\mathcal{P}$ dans les repères $\mathcal{F}^*$ et $\mathcal{F}$ (ce qui correspond à $\mathbf{e}_\nu = \mathbf{0}$) et rendre la matrice $\mathbf{H}$ symétrique (ce qui correspond à $\mathbf{e}_\omega = \mathbf{0}$).

Remarque 3

Il existe toute une classe d'isomorphismes qui peuvent être construits à partir de l'homographie $\mathbf{H}$. Nous donnons ici, à titre indicatif, un choix supplémentaire pour la fonction $\mathbf{e}$. En posant $\overline{\mathbf{m}} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{m}_k$ et $\overline{\mathbf{m}}^* = \frac{1}{n}\sum_{k=1}^{n}\mathbf{m}_k^*$, et où chaque paire $\mathbf{m}_k^*$ et $\mathbf{m}_k$ correspond à deux points mis en correspondance entre l'image de référence et l'image courante, nous pouvons choisir $\mathbf{e}$ tel que :

$$\begin{aligned} \mathbf{e}_\nu &= \frac{\overline{\mathbf{m}}^\top \mathbf{H}\overline{\mathbf{m}}^*}{\overline{\mathbf{m}}^\top\overline{\mathbf{m}}}\overline{\mathbf{m}} - \overline{\mathbf{m}}^* \\ [\mathbf{e}_\omega]_\times &= \mathbf{H} - \mathbf{H}^\top \end{aligned}$$

Il est également possible de montrer que cette fonction est isomorphe localement à l'attitude de la caméra.

7.4 La loi de commande

Nous commençons par calculer la matrice d'interaction. Il s'agit de la matrice qui met en relation la dérivée de la fonction de tâche $\dot{\mathbf{e}}$ et les vitesses de la caméra : la vitesse en translation $\boldsymbol{\nu}$ et la vitesse en rotation $\boldsymbol{\omega}$. À partir de la définition de $\mathbf{e}_\nu$, nous pouvons écrire :

$$\dot{\mathbf{e}}_\nu = Z^{*-1}(\dot{\mathbf{t}} + \dot{\mathbf{R}}\mathbf{X}^*)$$

Sachant que nous avons :

$$\dot{\mathbf{t}} = \boldsymbol{\nu} + [\boldsymbol{\omega}]_\times \mathbf{t} \tag{7.10}$$

$$\dot{\mathbf{R}} = [\boldsymbol{\omega}]_\times \mathbf{R} \tag{7.11}$$

nous pouvons écrire :

$$\dot{\mathbf{e}}_\nu = Z^{*-1}(\boldsymbol{\nu} + [\boldsymbol{\omega}]_\times \mathbf{t} + [\boldsymbol{\omega}]_\times \mathbf{R}\mathbf{X}^*) = Z^{*-1}\boldsymbol{\nu} + Z^{*-1}[\boldsymbol{\omega}]_\times (\mathbf{t} + \mathbf{R}\mathbf{X}^*)$$

En utilisant la définition de $\mathbf{e}_\nu$, nous avons :

$$\dot{\mathbf{e}}_\nu = Z^{*-1}\boldsymbol{\nu} + [\boldsymbol{\omega}]_\times \left(\mathbf{e}_\nu + Z^{*-1}\mathbf{X}^*\right) = Z^{*-1}\boldsymbol{\nu} - [\mathbf{e}_\nu + \mathbf{m}^*]_\times \boldsymbol{\omega} \tag{7.12}$$

Grâce à la définition de $\mathbf{e}_\omega$ dans l'équation (7.1), nous pouvons écrire :

$$\dot{\mathbf{e}}_\omega = 2\frac{d\sin(\theta)\mathbf{u}}{dt} + [\mathbf{n}]_\times \dot{\mathbf{t}}$$

Soit $\mathbf{L}_\omega$ la matrice définie par :

$$\mathbf{L}_\omega = \mathbf{I} - \frac{\sin(\theta)}{2}[\mathbf{u}]_\times - \sin^2\left(\frac{\theta}{2}\right)(2\mathbf{I} + [\mathbf{u}]_\times^2) \tag{7.13}$$

En utilisant le résultat de (Malis and Chaumette, 2002), nous avons :

$$\frac{d\sin(\theta)\mathbf{u}}{dt} = \mathbf{L}_\omega \boldsymbol{\omega}$$

et si nous utilisons l'équation (7.10), nous pouvons écrire :

$$\dot{\mathbf{e}}_\omega = 2\mathbf{L}_\omega + [\mathbf{n}]_\times (\boldsymbol{\nu} + [\boldsymbol{\omega}]_\times \mathbf{t}) = [\mathbf{n}]_\times \boldsymbol{\nu} + (2\mathbf{L}_\omega - [\mathbf{n}]_\times [\mathbf{t}]_\times)\boldsymbol{\omega} \tag{7.14}$$

La relation entre la dérivée de la fonction de tâche $\dot{\mathbf{e}}$ et les vitesses de la caméra en translation $\boldsymbol{\nu}$ et en rotation $\boldsymbol{\omega}$ s'écrit :

$$\dot{\mathbf{e}} = \mathbf{L}\begin{bmatrix} \boldsymbol{\nu} \\ \boldsymbol{\omega} \end{bmatrix} \tag{7.15}$$

où la matrice d'interaction $\mathbf{L}$ de dimensions (6×6) s'écrit sous la forme :

$$\mathbf{L} = \begin{bmatrix} Z^{*-1} & -[\mathbf{e}_\nu + \mathbf{m}^*]_\times \\ [\mathbf{n}^*]_\times & 2\mathbf{L}_\omega - [\mathbf{n}^*]_\times [\mathbf{t}]_\times \end{bmatrix} \tag{7.16}$$

Le calcul ou l'estimation de cette matrice n'est pas nécessaire au calcul d'une commande stable comme le montre le théorème suivant.

Théorème 2 *:*
La loi de commande définie par :

$$\begin{bmatrix} \boldsymbol{\nu} \\ \boldsymbol{\omega} \end{bmatrix} = - \begin{bmatrix} \lambda_\nu \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \lambda_\omega \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{e}_\nu \\ \mathbf{e}_\omega \end{bmatrix} \tag{7.17}$$

où $\lambda_\nu \in \mathbb{R}_+^*$ et $\lambda_\omega \in \mathbb{R}_+^*$ est localement stable.

Preuve du théorème 2
En effet, si nous linéarisons l'équation (7.15) dans le voisinage $\mathbf{e} = \mathbf{0}$ (et par conséquent, dans le voisinage de $\mathbf{t} = \mathbf{0}$ et de $\theta = 0$), nous obtenons le système linéaire suivant :

$$\dot{\mathbf{e}} = - \begin{bmatrix} \lambda_\nu / Z^* & -\lambda_\omega [\mathbf{m}^*]_\times \\ \lambda_\nu [\mathbf{n}^*]_\times & 2\lambda_\omega \mathbf{I} \end{bmatrix} \mathbf{e} = -\mathbf{L}_0 \mathbf{e}$$

La matrice $\mathbf{L}_0$ est constante et admet les valeurs propres suivantes :

$$\frac{\lambda_\omega}{2Z^*} \begin{bmatrix} 2\lambda \\ 4Z^* \\ 2Z^* + \lambda + \sqrt{\lambda^2 + 4Z^{*2}} \\ 2Z^* + \lambda + \sqrt{\lambda^2 + 4Z^{*2}} \\ 2Z^* + \lambda - \sqrt{\lambda^2 + 4Z^{*2}} \\ 2Z^* + \lambda - \sqrt{\lambda^2 + 4Z^{*2}} \end{bmatrix}$$

où $\lambda = \lambda_\nu / \lambda_\omega$. Étant donné que $\lambda > 0$ et $Z^* > 0$, les valeurs propres de la matrice $\mathbf{L}_0$ sont toujours positives. Par conséquent, la loi de commande définie dans l'équation équation (7.17) est toujours localement stable quelque soit $\mathbf{n}^*$ et quelque soit $\mathbf{m}^*$.

Cette loi de commande, ne dépendant que de la fonction de tâche $\mathbf{e}$, peut être complètement calculée à partir des deux images $\mathcal{I}$ et $\mathcal{I}^*$ seulement. Le calcul de la matrice d'interaction $\mathbf{L}$ (ou même d'une approximation de cette matrice) est donc totalement inutile. Cette matrice n'est

utilisée que pour prouver d'une manière analytique la stabilité locale de la loi de commande. La loi de commande présentée est simple et cherche à faire converger exponentiellement le vecteur $\mathbf{e}$ vers zéro. La force de cette approche réside en partie en cette grande simplicité. Sans avoir à connaître le vecteur $\mathbf{n}^*$ et quelque soit le point $\mathcal{P}$ choisi dans le plan, la stabilité locale de la commande est garantie. Cette loi de commande permet également de faire converger $\mathbf{e}_\nu$ et $\mathbf{e}_\omega$ à des vitesses différentes, et ce, en jouant sur le choix des deux gains $\lambda_\nu \in \mathbb{R}_+^*$ et $\lambda_\omega \in \mathbb{R}_+^*$.

Cette approche résout le problème de dépendance de la mesure du modèle de la cible (que posent les approches classiques d'asservissement visuel) pour calculer la fonction de tâche et garantir la stabilité locale de la commande. L'approche que nous proposons ne nécessite pas d'extraction de primitive car le suivi visuel est effectué directement à partir des intensités lumineuses dans l'image. Le résultat du suivi visuel est directement l'homographie. Cette homographie est utilisée telle quelle pour le calcul de la fonction de tâche ce qui permet d'unifier le suivi visuel temps réel et l'asservissement visuel. Le résultat du suivi est l'entrée de la commande de l'asservissement.

Cependant, un inconvénient réside dans le fait que la commande est couplée, c'est-à-dire, étant donné que la fonction de tâche n'est pas découplée, la rotation et la translation ne peuvent pas être commandées séparément (une commande générée suite à une erreur de rotation (resp. de translation) peut induire un mouvement de translation (resp. de rotation)).

7.5 Généralisation de l'approche aux caméras omnidirectionnelles

La commande présentée se base sur l'existence d'une matrice d'homographie $\mathbf{H}$ permettant de mettre en correspondance les projections dans deux images acquises dans différentes positions d'un même point 3D appartenant à un certain plan. Par conséquent, nous généralisons, ici, la notion d'homographie pour des capteurs omnidirectionnels

7.5.1 Généralisation du modèle de projection

Soit $\mathcal{P}$ un point 3D avec des coordonnées Cartésiennes $\boldsymbol{\mathcal{X}} = [X\ Y\ Z]^\top$ dans le repère courant $\mathcal{F}$. Ce point peut être projeté sur la sphère unité $\mathcal{S}$ en un point 3D ayant les coordonnées $\boldsymbol{\mathcal{X}}_s = [X_s\ Y_s\ Z_s]^\top$:

$$\boldsymbol{\mathcal{X}}_s = \frac{1}{\rho}\boldsymbol{\mathcal{X}} \tag{7.18}$$

où $\rho = \|\boldsymbol{\mathcal{X}}\|$ (voir Figure 7.1). Si nous utilisons le modèle de projection des caméras omnidirectionnelles proposé par (Baker and Nayar, 1999; Geyer and Daniilidis, 2000; Barreto and Araujo, 2002), nous pouvons projeter le point $\boldsymbol{\mathcal{X}}_s$ en utilisant le centre de projection $\mathcal{C}_p$ en un point

$\mathbf{m} = [x\ y\ 1]^\top$ sur le plan normalisé :

$$\mathbf{m} = \hbar(\boldsymbol{\mathcal{X}}_s) = \begin{bmatrix} \frac{X_s}{Z_s - \xi} & \frac{Y_s}{Z_s - \xi} & 1 \end{bmatrix}^\top \tag{7.19}$$

où ξ est un scalaire positif ($0 \leq \xi \leq 1$) qui dépend de la géométrie du miroir omnidirectionnel (voir le Tableau 7.1). Le point $\mathbf{m}$ est projeté dans le plan image en un point $\mathbf{p} = [u\ v\ 1]^\top$:

$$\mathbf{p} = \mathbf{K}\boldsymbol{\eta}\mathbf{m} \tag{7.20}$$

où $\mathbf{K}$ est la matrice des paramètres intrinsèques de la caméra définie dans (2.21) et $\boldsymbol{\eta}$ est une matrice diagonale contenant un autre paramètre η qui dépend de la géométrie du miroir (voir le Tableau 7.1) :

$$\boldsymbol{\eta} = \begin{bmatrix} \eta & 0 & 0 \\ 0 & \eta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7.21}$$

La transformation inverse est définie comme suit : d'un point image $\mathbf{p}$ mesuré dans l'image, nous pouvons calculer le point

$$\mathbf{m} = \boldsymbol{\eta}^{-1}\mathbf{K}^{-1}\mathbf{p} \tag{7.22}$$

La projection sur la sphère unité du point $\mathbf{m}$ s'écrit :

$$\boldsymbol{\mathcal{X}}_s = \hbar^{-1}(\mathbf{m}) = [\gamma x, \gamma y, \gamma + \xi]^\top \tag{7.23}$$

où le paramètre γ s'écrit :

$$\gamma = -\frac{\xi + \sqrt{\xi^2 + (1 - \xi^2)\|\mathbf{m}\|^2}}{\|\mathbf{m}\|^2} = -\frac{\xi + \sqrt{1 + (1 - \xi^2)(x^2 + y^2)}}{x^2 + y^2 + 1} \tag{7.24}$$

	Équation	ξ	η
Miroir parabolique	$\rho = Z + 2p$	1	$-2p$
Miroir hyperbolique	$\frac{(Z+\frac{d}{2})^2}{a^2} - \frac{X^2}{b^2} - \frac{Y^2}{b^2} = 1$	$\frac{d}{\sqrt{d^2+4p^2}}$	$\frac{-2p}{\sqrt{d^2+4p^2}}$
Miroir elliptique	$\frac{(Z+\frac{d}{2})^2}{a^2} + \frac{X^2}{b^2} + \frac{Y^2}{b^2} = 1$	$\frac{d}{\sqrt{d^2+4p^2}}$	$\frac{2p}{\sqrt{d^2+4p^2}}$
Miroir plan	$Z = -\frac{d}{2}$	0	-1
Caméra perspective standard	Aucune	0	1

p et d sont des paramètres du miroir

$a = 1/2(\sqrt{d^2 + 4p^2} \pm 2p)$, '$-$' pour un miroir hyperbolique, '$+$' pour un miroir elliptique

$b = \sqrt{p(\sqrt{d^2 + 4p^2} \pm 2p)}$, '$-$' pour un miroir hyperbolique, '$+$' pour un miroir elliptique

TAB. 7.1 – Les paramètres des différents types de miroirs

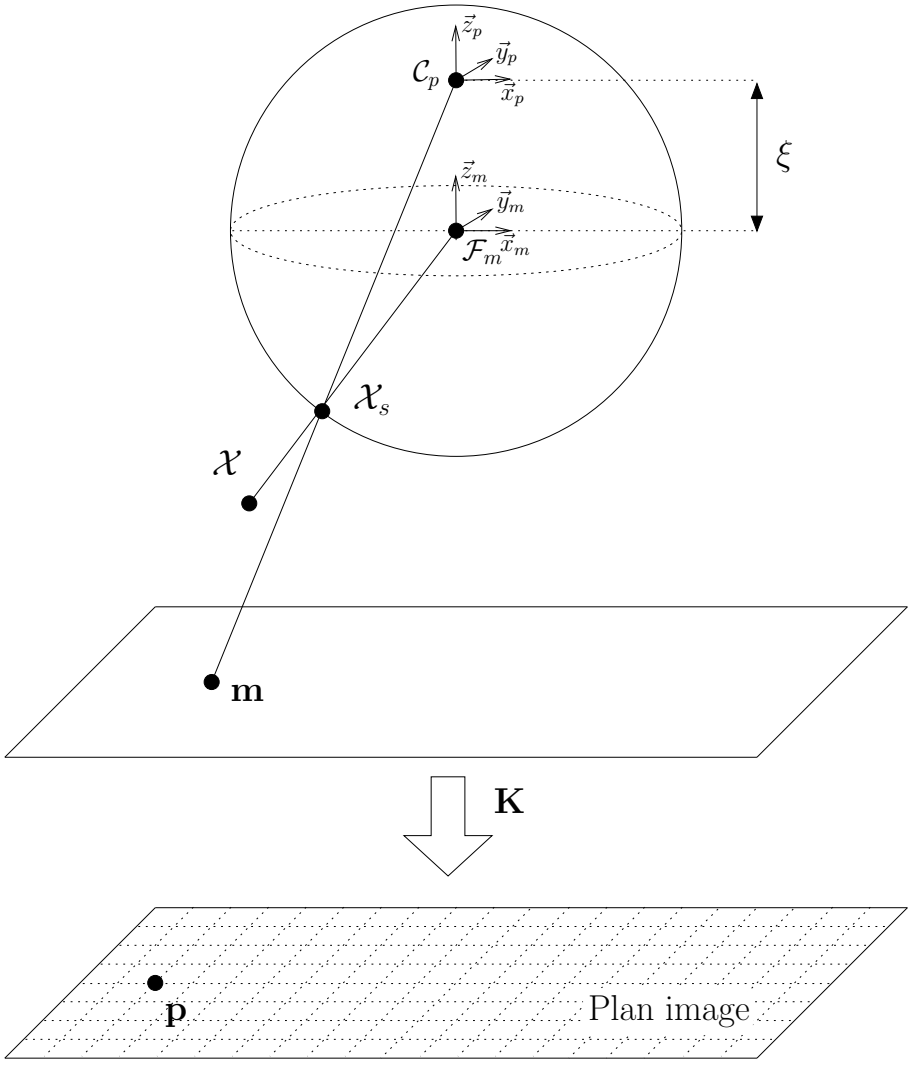


Figure 7.1 – Modèle de projection généralisé

7.5.2 Généralisation de l'homographie entre deux projections sphériques

Dans la position de référence le point $\mathcal{P}$ se projette sur une sphère unité de référence $\mathcal{S}^*$ en un point $\boldsymbol{\mathcal{X}}_s^* = [X_s^*\ Y_s^*\ Z_s^*]^\top$ où :

$$\boldsymbol{\mathcal{X}}_s^* = \frac{1}{\rho^*}\boldsymbol{\mathcal{X}}^* \tag{7.25}$$

et $\rho^* = \|\boldsymbol{\mathcal{X}}^*\|$. Si nous supposons que $\mathcal{P}$ appartient à un plan π de vecteur normal $\mathbf{n}^*$ où $\|\mathbf{n}^*\| = \sqrt{\mathbf{n}^{*\top}\mathbf{n}^*} = d^{*-1}$, où d^* est la distance entre le plan π et le centre de la sphère. En utilisant les équations (7.18), (7.25), (2.5) et (2.12), nous pouvons définir une homographie entre $\boldsymbol{\mathcal{X}}_s$ et $\boldsymbol{\mathcal{X}}_s^*$:

$$\frac{\rho}{\rho^*}\boldsymbol{\mathcal{X}}_s = \mathbf{H}_n\boldsymbol{\mathcal{X}}_s^* \tag{7.26}$$

où la matrice $\mathbf{H}$ s'écrit d'une manière similaire à l'équation (2.14) : $\mathbf{H}_n = \mathbf{R}+\mathbf{t}\mathbf{n}^{*\top}$. Connaissant $\boldsymbol{\mathcal{X}}_s^*$ et une matrice $\mathbf{H}$ proportionnelle à $\mathbf{H}_n$ vérifiant : $\mathbf{H} = \alpha\mathbf{H}_n$, où $\alpha \in \mathbb{R}^*$, il est possible de calculer $\boldsymbol{\mathcal{X}}_s$ sans connaître le rapport $\frac{\rho}{\rho^*}$. En effet, il suffit de multiplier $\boldsymbol{\mathcal{X}}_s^*$ par $\mathbf{H}$, puis de normaliser la troisième composante de $\mathbf{m}$ à 1. Nous pouvons donc écrire : $\mathbf{m} \propto \mathbf{H}\mathbf{m}^*$. Voir la Figure 7.2 pour une illustration.

Par conséquent, il est possible de définir une matrice d'homographie entre deux points, projections dans deux images d'un même point 3D appartenant à un plan, même dans le cas d'une projection omnidirectionnelle. Par conséquent, la généralisation de la commande dans le cas des caméras omnidirectionnelles est immédiate (Benhimane and Malis, 2006c). La fonction de tâche s'écrit d'une manière similaire aux équations (7.3) et (7.4) :

$$\mathbf{e} = \begin{bmatrix} \mathbf{e}_\nu \\ \mathbf{e}_\omega \end{bmatrix} \text{ avec } \begin{cases} \mathbf{e}_\nu = (\mathbf{H}-\mathbf{I})\boldsymbol{\mathcal{X}}_s^* \\ [\mathbf{e}_\omega]_\times = \mathbf{H}-\mathbf{H}^\top \end{cases} \tag{7.27}$$

7.6 Expériences

7.6.1 Expériences avec le robot Anis de l'équipe ICARE

Nous avons testé l'asservissement visuel proposé sur le manipulateur mobile Anis disponible dans l'équipe ICARE (voir figure 5.6). Seul le bras manipulateur a été utilisé pour la validation expérimentale. La caméra, montée sur l'organe terminal du robot, a été grossièrement calibrée. Il s'agit d'une caméra munie d'un capteur CMOS et permettant d'acquérir 30 images par seconde.

La validation expérimentale se fait de la manière suivante. Une image de référence de la scène est capturée dans une certaine position (position de référence) du robot par rapport à laquelle nous voulons le repositionner. La tâche de positionnement se fait par rapport à une zone plane, la zone délimitée par le carré rouge dans la figure 7.3(a). La position de la cible par rapport au repère de caméra est complètement inconnue.

Partant d'une autre position permettant de voir le même objet sous un angle différent (posi-

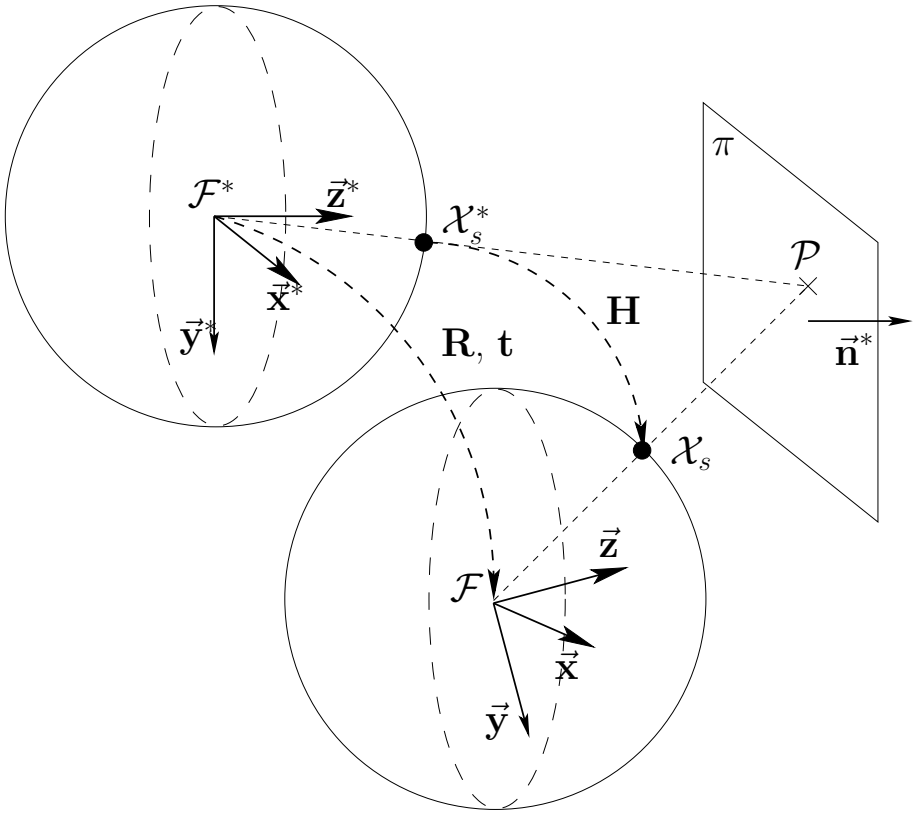
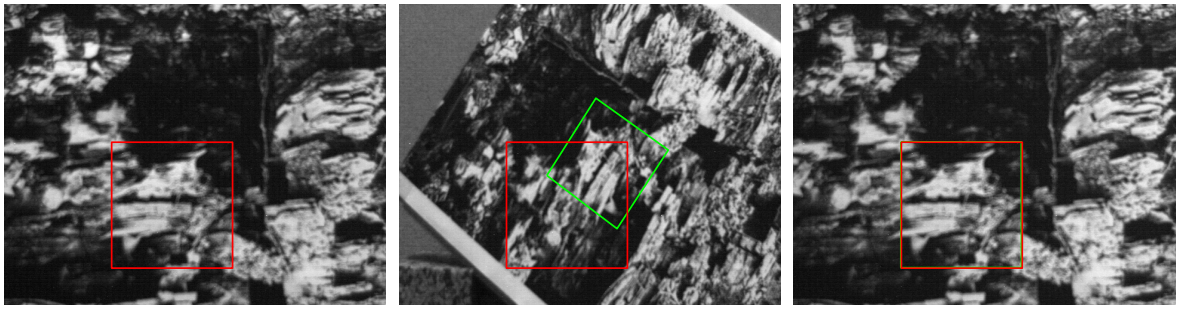


Figure 7.2 – Homographie entre deux projections sphériques

tion initiale), le robot est alors commandé, avec la loi de commande décrite par l'équation (7.17) avec $\lambda_\nu = \lambda_\omega = 0.025$, afin qu'il atteigne la position de référence. À la position initiale, la même zone d'intérêt est délimitée par le carré vert dans la figure 7.3(b). Nous utilisons l'algorithme de suivi de cible décrit dans le Chapitre 4 qui assure le suivi d'une cible plane en temps-réel pendant l'asservissement visuel et fournit à chaque image acquise la matrice de transformation projective planaire dans l'image $\mathbf{G}$. En utilisant une approximation des paramètres intrinsèques, la matrice d'homographie $\mathbf{H}$ permettant de calculer la commande est obtenue à l'aide de la formule suivante : $\mathbf{H} = \mathbf{K}^{-1}\mathbf{G}\mathbf{K}$. Un tel suivi permet d'utiliser n'importe quelle cible ayant un minimum de texture et permet d'éviter d'extraire des amers visuel particuliers (points d'intérêts, droites, segments, arcs, ...) puis de les suivre. Nous utilisons comme point de contrôle ($\mathbf{m}^*$ dans l'équation (7.3)) le centre de gravité de la zone sélectionnée. À la convergence de l'asservissement visuel, le robot a atteint sa position de référence et les informations visuelles observées coïncident avec celles de l'image de référence (voir figure 7.3(c)).



(a) Image de référence (b) Image initiale (c) Image finale

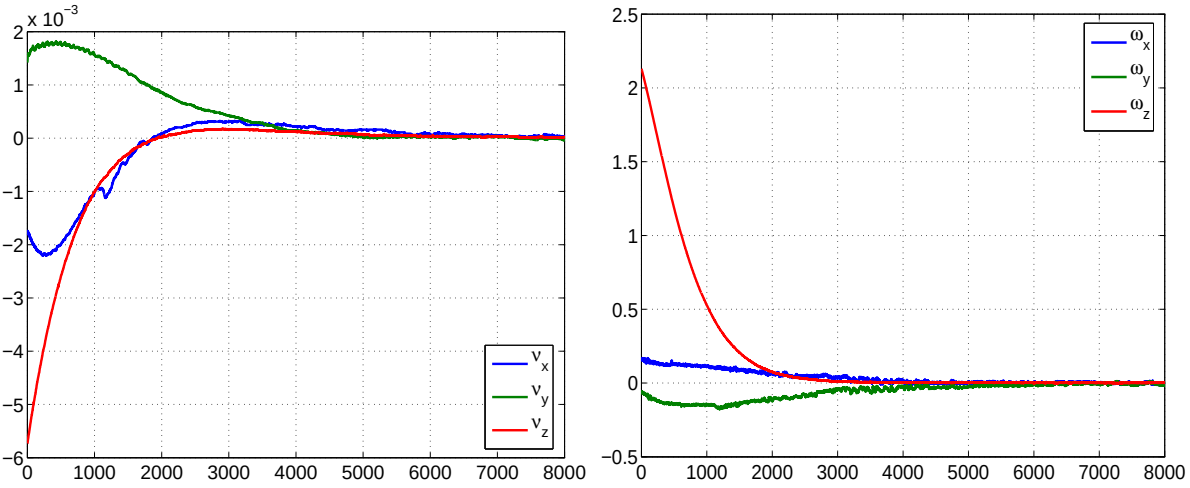
Figure 7.3 – Expérience 1 : Images de référence, initiale et finale

Toutes les composantes de la fonction de tâche convergent vers 0. Malgré la calibration grossière des paramètres intrinsèques de la caméra, les figures 7.4(a) et 7.4(b) montrent que la commande est stable. En effet, la vitesse en translation et la vitesse en rotation convergent vers zéro.

Une deuxième expérience a été effectuée dans les même conditions (calibration grossière du robot et de la caméra, vecteur normal au plan de la cible inconnu, distance à la cible inconnue...). Le positionnement se fait par rapport à une autre cible (voir figure 7.5(a)) et un mouvement différent a été effectué entre la position de référence et la position initiale (voir figure 7.5(b)).

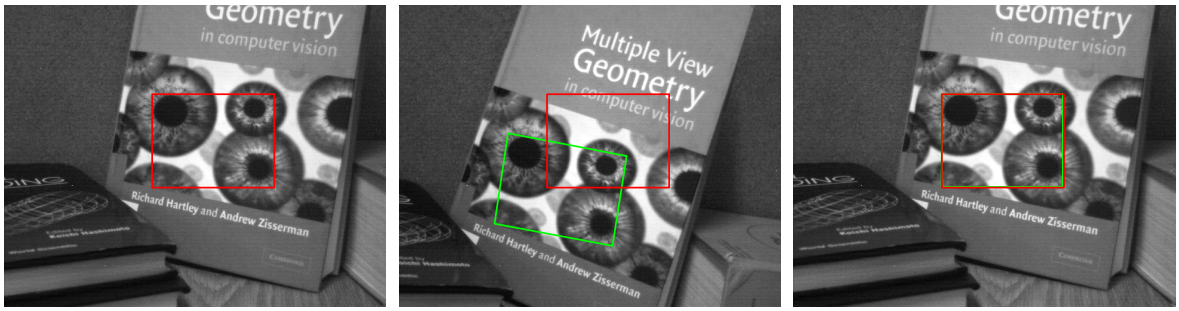
Cette fois encore, toutes les composantes de la fonction de tâche convergent vers zéro et la commande est stable (voir figures 7.6(a) et 7.6(b)). À la convergence, les informations visuelles observées coïncident avec celles de l'image de référence (voir figure 7.5(c)) et le robot a atteint sa position de référence.

Ces deux expériences ont été rapidement implémentées pour tester la loi de commande sur



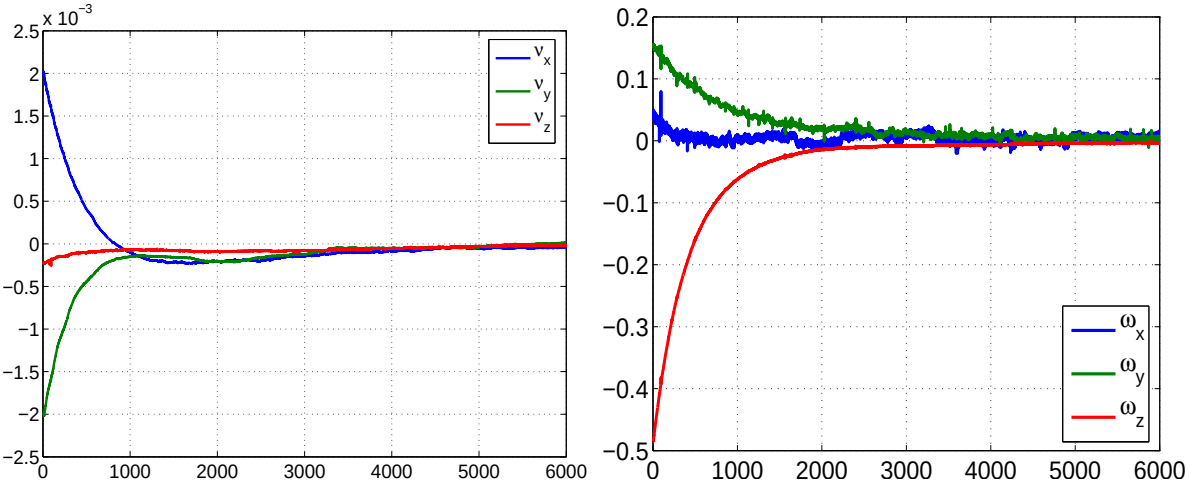
(a) Vitesse de translation (b) Vitesse de rotation

Figure 7.4 – Expérience 1 : Positionnement d'une caméra par rapport à un objet plan



(a) Image de référence (b) Image initiale (c) Image finale

Figure 7.5 – Expérience 2 : Images de référence, initiale et finale



(a) Vitesse de translation (b) Vitesse de rotation

Figure 7.6 – Expérience 2 : Positionnement d'une caméra par rapport à un objet plan

le robot et ne représentent qu'une sorte de preuve de concept. Le robot utilisé ne fournit pas l'erreur en position Cartésienne entre la position finale et la position de référence, sa calibration est grossière et sa géométrie entraine un certain nombre de singularités. L'espace de travail du robot est également restreint et ne permet que des mouvements de faibles amplitudes. Des expériences sur un robot qui permet de fournir les erreurs en rotation et translation Cartésiennes ont permis de valider quantitativement les performance de l'algorithme.

7.6.2 Expériences avec le robot Cartésien de l'équipe LAGADIC

Nous avons décidé d'effectuer des tests supplémentaires de la loi de commande proposée dans ce chapitre sur un deuxième robot. Il s'agit du robot Cartésien à 6 degrés de liberté utilisé par l'équipe LAGADIC de l'IRISA (INRIA Rennes). La calibration de ce robot est très précise et son espace de travail est grand permettant des mouvements d'amplitude plus importante que celle des mouvements permis par le robot Anis.

La caméra utilisée acquiert des images de dimensions (384×288). Elle est calibrée et la matrice des paramètres intrinsèques s'écrit :

$$\mathbf{K} = \begin{bmatrix} 592.15 & 0 & 198.36 \\ 0 & 569.78 & 139.62 \\ 0 & 0 & 1 \end{bmatrix} \tag{7.28}$$

Les gains de la loi de commande décrite par l'équation (7.17) utilisés sont : $\lambda_\nu = \lambda_\omega = 0.1$ et la dimension des zones d'intérêt sélectionnées est (100×100).

Figure 7.7 – Le robot Cartésien à 6 degrés de liberté de l'équipe LAGADIC de l'IRISA

7.6.2.1 AV par rapport à une cible plane avec une caméra calibrée

Une image de référence de la scène est capturée à la position de référence du robot par rapport à laquelle nous voulons le repositionner. La tâche de positionnement se fait par rapport à une zone plane, la zone délimitée par le carré rouge dans la figure 7.8(a). La position de la cible par rapport au repère de caméra est complètement inconnue.

Partant d'une position initiale, le robot est alors commandé, avec la loi de commande décrite par l'équation (7.17), afin qu'il atteigne la position de référence. À la position initiale, la même zone d'intérêt est délimitée par le carré vert dans la figure 7.8(b). La translation effectuée dans l'espace Cartésien entre la position de référence et la position initiale est $[-0.62,\ -0.14,\ -0.22]$ en mètres, la rotation est de 95.93 degrés suivant l'axe $[0.17,\ 0.40,\ 0.90]$.

(a) Image de référence (b) Image initiale (c) Image finale

Figure 7.8 – Expérience 1 : AV par rapport à une cible plane avec une caméra calibrée avec un grand déplacement initial

Nous utilisons l'algorithme de suivi de cible décrit dans le Chapitre 4 qui fournit à chaque image acquise la transformation projective planaire $\mathbf{G}$ dans l'image. Dans cette expérience, nous utilisons les paramètres intrinsèques issus de la calibration (7.28) pour déterminer l'homographie $\mathbf{H}$ (dans l'espace normalisé) à partir de $\mathbf{G}$. Nous utilisons comme point de contrôle ($\mathbf{m}^*$ dans l'équation (7.3)) le point correspondant dans l'image normalisée au centre de gravité $\mathbf{p}^*$ de la zone sélectionnée ($\mathbf{m}^* = \mathbf{K}^{-1}\mathbf{p}^*$).

À la convergence de l'asservissement visuel, le robot a atteint sa position de référence et les informations visuelles observées coïncident avec celles de l'image de référence (voir figure 7.8(c)). L'erreur en translation suivant les trois axes est très faible : $[-0.004,\ -0.001,\ -0.000]$ en mètre (voir Figure 7.9(c)) et l'erreur en rotation est également faible de 0.29 degré (voir Figure 7.9(d)). Les figures 7.9(a) et 7.9(b) montrent que la commande est stable. En effet, la vitesse en translation et la vitesse en rotation convergent toutes les deux vers zéro.

Nous effectuons une deuxième expérience identique à la précédente, c'est-à-dire, l'objet par rapport auquel le robot se positionne est plan et nous utilisons les paramètres intrinsèques de la caméra issus de la calibration. La transformation entre la position de référence et la position à laquelle l'image initiale a été capturée est également la même que précédemment (voir Figure 7.10(b)), sauf qu'ici la zone considérée est différente de la première expérience (voir Figure 7.10(a)).

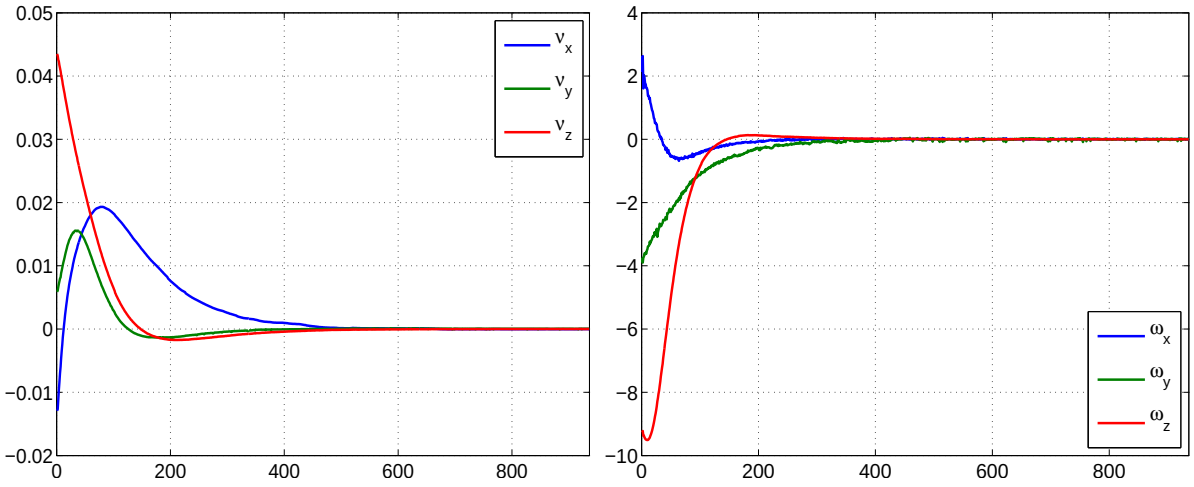
L'erreur en translation suivant les trois axes est faible : $[-0.011,\ 0.000,\ 0.000]$ en mètre (voir Figure 7.11(c)) et l'erreur en rotation est également faible de 0.81 degré (voir Figure 7.11(d)). Les figures 7.11(a) et 7.11(b) montrent que la commande est stable car la vitesse en translation et la vitesse en rotation convergent toutes les deux vers zéro.

Bien que les textures des zones suivies soient différentes, nous remarquons que les lois de commande et la décroissance des erreurs en rotation et en translation ont les mêmes allures pour les deux expériences. Ce qui nous amène à dire que l'expérience est répétable quelle que soit la cible observée (pourvu qu'il y ait un minimum de texture pour bien calculer la transformation projective planaire dans l'image).

7.6.2.2 AV par rapport à une cible plane avec une caméra non calibrée

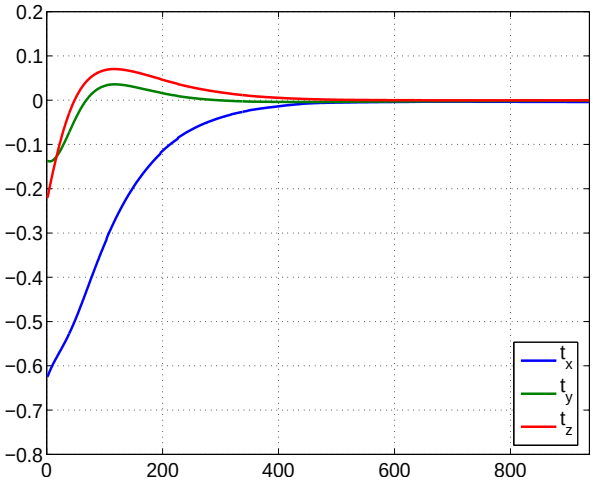
Nous nous proposons, ici, de refaire une expérience identique à la première, c'est-à-dire, l'objet par rapport auquel le robot se positionne est plan, la texture est la même et la transformation entre la position de référence et la position à laquelle l'image initiale a été capturée sont les mêmes (voir Figures 7.12(b) et 7.12(b)).

Seulement, ici, afin de tester expérimentalement la robustesse de la commande par rapport aux erreurs des paramètres internes de la caméra, nous utilisons volontairement des paramètres intrinsèques différents de ceux obtenus à l'issue de la calibration. En effet, les paramètres $\widehat{\mathbf{K}}$ utilisés sont complètement différents de $\mathbf{K}$. Les coordonnées du point principal sont décentrées de $[-100,\ +60]$ pixels et la focale suivant $\vec{x}$ est modifiée de $+30\%$ et la focale suivant $\vec{y}$ est

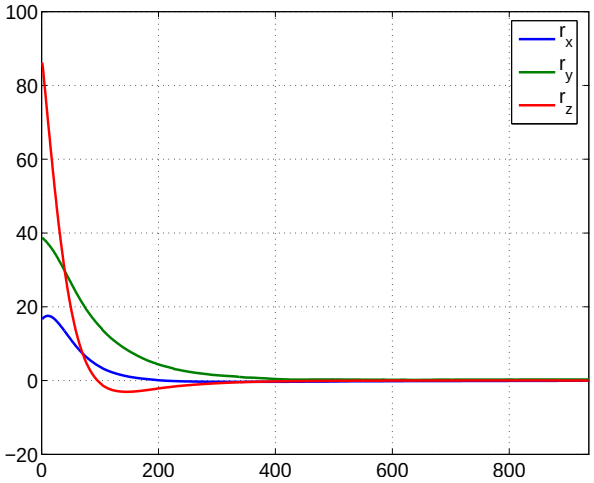


(a) Vitesse en translation m/s

(b) Vitesse en rotation deg/s

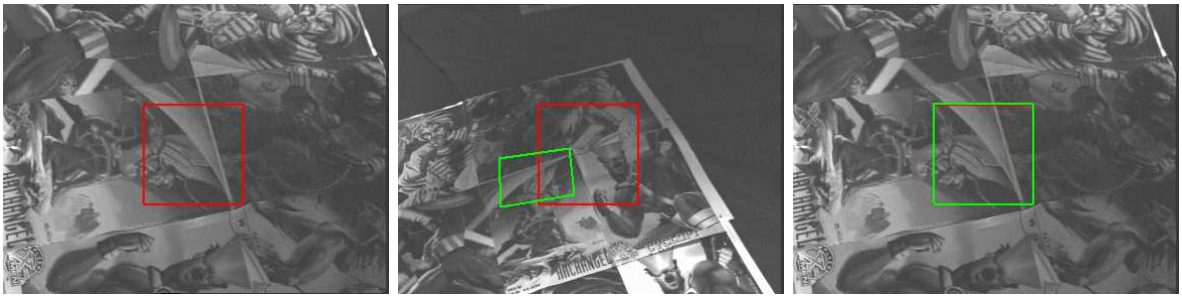


(c) Erreur en translation m



(d) Erreur en rotation deg

Figure 7.9 – Expérience 1 : Courbes des vitesses de rotations et de translation et courbes des erreurs de translation et de rotation



(a) Image de référence (b) Image initiale (c) Image finale

Figure 7.10 – Expérience 2 : AV par rapport à une cible plane avec une caméra calibrée avec un grand déplacement initial

modifiée de -30% :

$$\widehat{\mathbf{K}} = \begin{bmatrix} 800 & 0 & 100 \\ 0 & 400 & 200 \\ 0 & 0 & 1 \end{bmatrix} \tag{7.29}$$

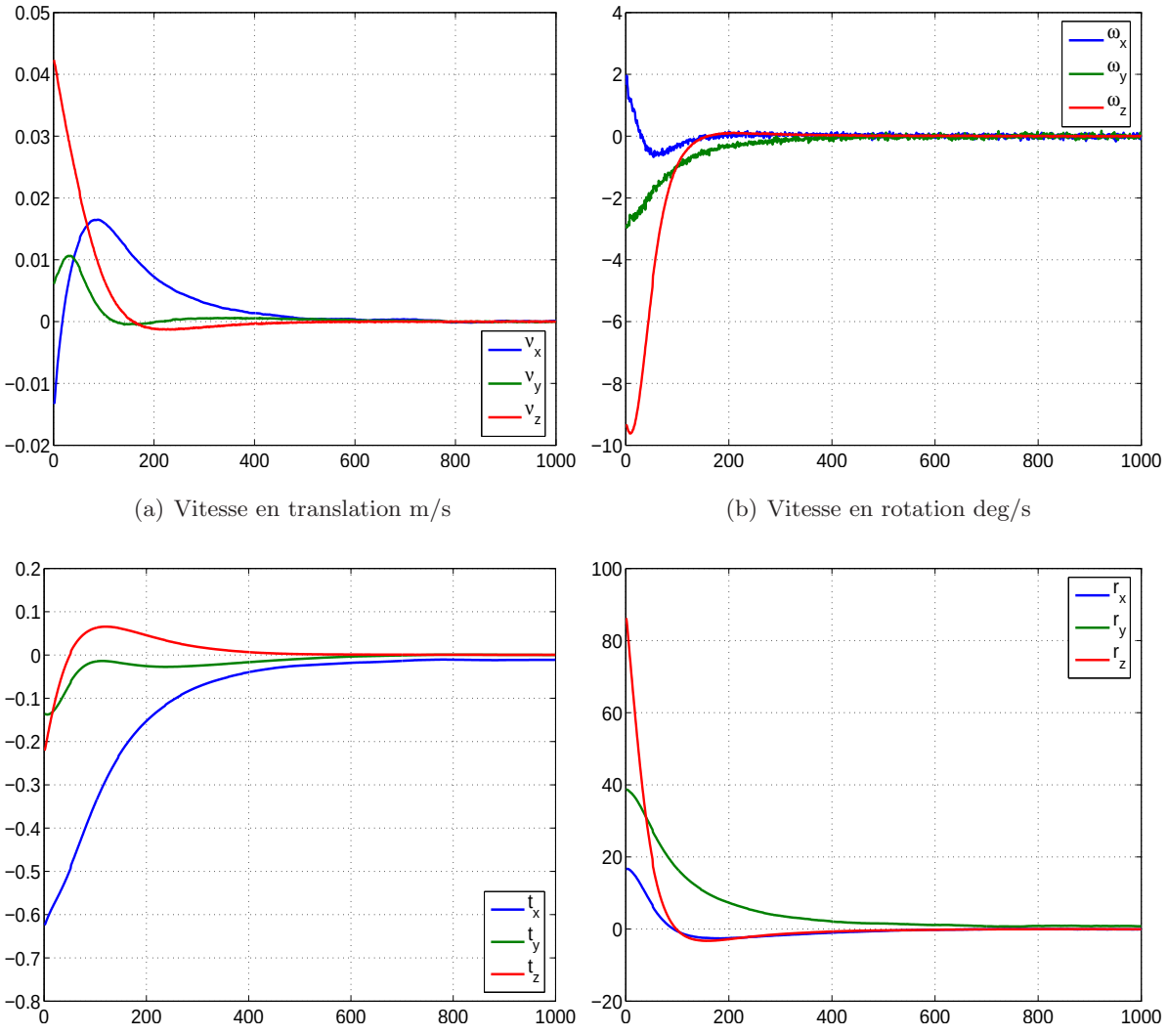
Malgré le fait que les paramètres intrinsèques sont faux, nous remarquons que la loi de commande reste stable (voir Figures 7.13(a) et 7.13(b)). Le positionnement reste précis puisque l'image acquise par la caméra à l'issue de l'asservissement visuel coïncide avec l'image de référence (voir Figure 7.12(c)), l'erreur en translation est de $[0.002,\ 0.000,\ -0.001]$ en mètre (voir Figure 7.13(c)) et l'erreur en rotation est de 0.22 degré (voir Figure 7.13(d)). L'allure de la décroissance des erreurs et celles des vitesses sont identiques à celles de la première expérience, ce qui nous amène à dire que la loi de commande admet une certaine robustesse aux erreurs de calibration des paramètres intrinsèques de la caméra utilisée.

7.6.2.3 AV par rapport à une cible non plane avec une caméra calibrée

Ici, nous réalisons la commande par rapport à un objet non plan : un ballon (voir Figure 7.14(a)). L'objectif est de tester la robustesse de la commande proposée aux erreurs de l'estimation de la transformation projective planaire dans l'image $\mathbf{G}$. En effet, la matrice $\mathbf{G}$ estimée est forcément erronée car l'objet suivi n'est pas plan. Donc, il n'existe pas d'homographie entre deux images acquises à des positions différentes. L'homographie $\widehat{\mathbf{G}}$ fournie par l'algorithme de suivi est une approximation de la transformation projective planaire dans l'image correspondant au plan moyen tangent à la zone sélectionnée.

Nous commençons par un déplacement faible entre la position de référence et la position initiale (voir Figure 7.14(b)). Nous effectuons une translation de $[-0.01,\ -0.08,\ -0.10]$ en mètre et une rotation suivant l'axe $[-0.1897,\ 0.1473,\ 0.9707]$ avec un angle de 34.56 degrés.

Au début de l'asservissement visuel, la commande a un léger dépassement dû au fait que l'objet par rapport auquel la commande est effectuée n'est pas plan. Cependant, la commande



(a) Vitesse en translation m/s

(b) Vitesse en rotation deg/s

(c) Erreur en translation m

(d) Erreur en rotation deg

Figure 7.11 – Expérience 2 : Courbes des vitesses de rotations et de translation et courbes des erreurs de translation et de rotation

(a) Image de référence (b) Image initiale (c) Image finale

Figure 7.12 – AV par rapport à une cible plane avec une caméra non calibrée avec un grand déplacement initial

est restée stable (voir Figures 7.15(a) et 7.15(b)) et le positionnement final est précis. En effet, l'image finale coïncide avec l'image de référence (voir Figure 7.14(c)). L'erreur en translation est très faible de l'ordre du millimètre : $[0.002,\ -0.001,\ -0.001]$ (voir Figure 7.15(c)) et l'erreur en rotation est de 0.21 degré (voir Figure 7.15(d)).

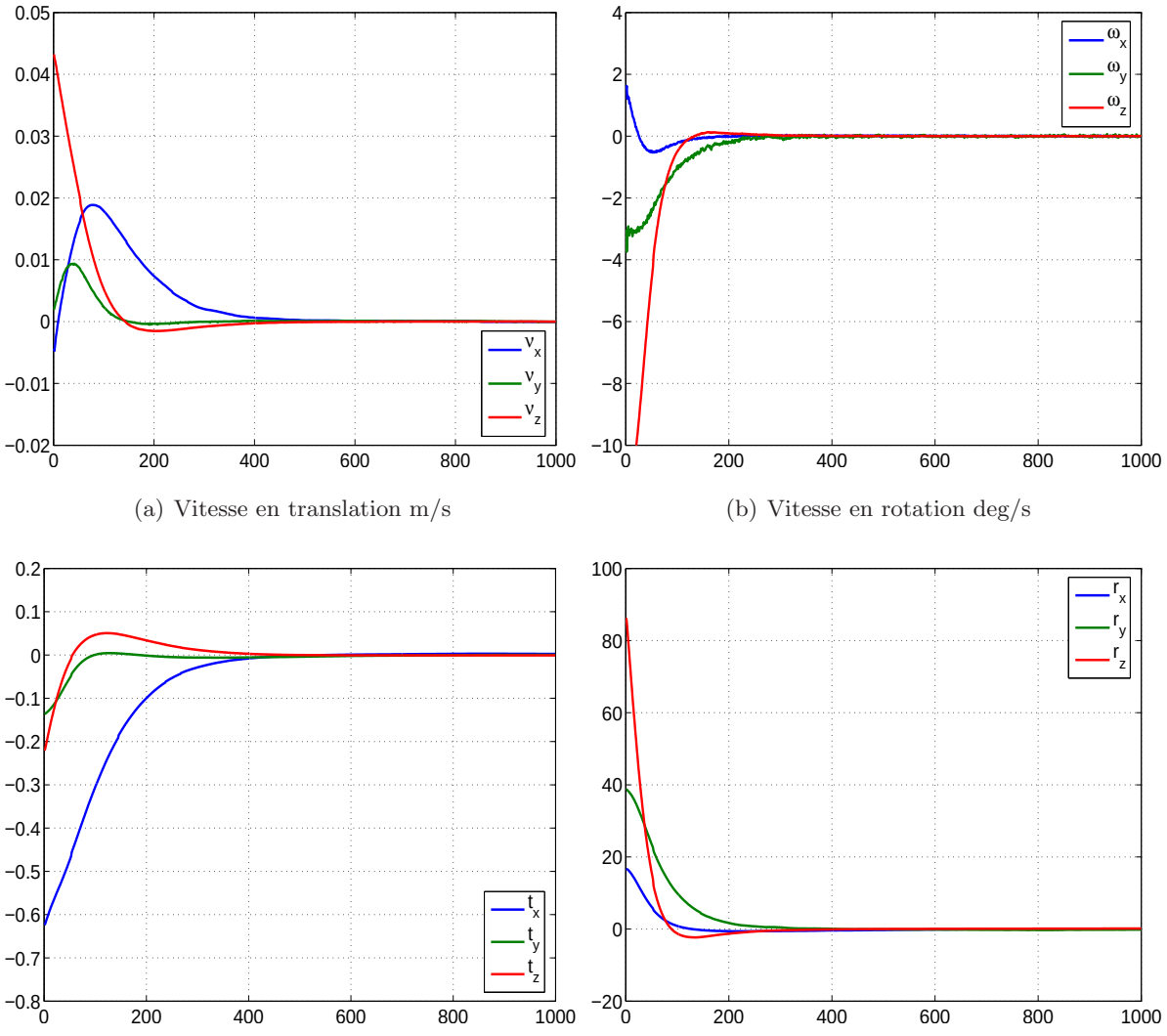
Dans la deuxième expérience, le positionnement se fait par rapport au même objet non plan. Cependant, le déplacement initial effectué est plus important que précédemment. En effet, la translation initiale est de $[0.01,\ 0.06,\ -0.22]$ et la rotation est suivant l'axe $[-0.15,\ 0.09,\ 0.98]$ et avec un angle de 56.42 degrés.

Durant l'asservissement, la commande est stable (voir Figures 7.17(a) et 7.17(b)) et le positionnement est précis. À l'issue de l'asservissement visuel l'erreur en translation est de $[0.005,\ -0.003,\ -0.000]$ (voir Figure 7.17(c)) et l'erreur en rotation est inférieure à 0.6 degrés (voir Figure 7.17(d)).

Là encore, nous sommes amenés à dire, que vu les résultats obtenus lors des expériences réalisées sur une cible non plane, la commande proposée admet une certaine robustesse aux erreurs de mesures.

7.6.3 Résultats de simulations dans le cas de capteurs omnidirectionnels

Dans ce paragraphe, nous présentons brièvement quelques résultats de simulation de la loi de commande dans le cas omnidirectionnel. Nous avons effectué des simulations Matlab d'une caméra montée sur un robot munie d'un capteur parabolique et hyperbolique. Afin de calculer l'homographie, nous utilisons cinq points sur le plan de la cible. Nous supposons être capable d'effectuer la mise en correspondance entre les images lors de l'asservissement visuel. Dans les deux simulations, nous appliquons la loi de commande que nous avons présentée dans ce chapitre avec comme paramètres de l'équation (7.17) : $\lambda_\nu = \lambda_\omega = 1$. Étant donné que l'homographie est calculée à partir de la projection des points sur la sphère unité, la même loi de commande peut être appliquée quelle que soit la géométrie du miroir utilisé. Dans les deux simulations, nous



(a) Vitesse en translation m/s

(b) Vitesse en rotation deg/s

(c) Erreur en translation m

(d) Erreur en rotation deg

Figure 7.13 – Courbes des vitesses de rotations et de translation et courbes des erreurs de translation et de rotation

(a) Image de référence (b) Image initiale (c) Image finale

Figure 7.14 – AV par rapport à une cible non plane avec une caméra calibrée avec un déplacement initial moyen

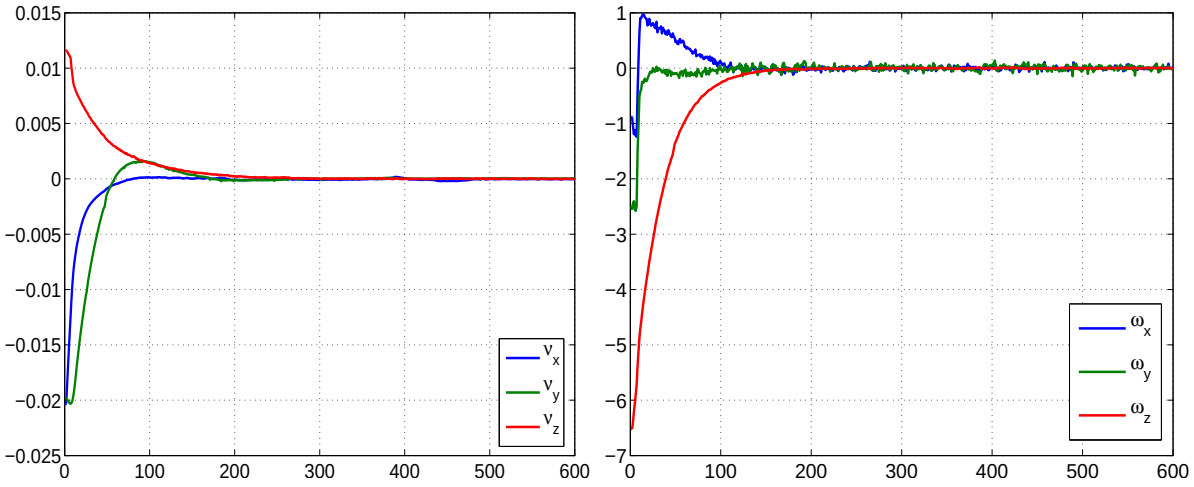
n'avons pas utilisé l'équation du plan lors de la commande.

7.6.3.1 Cas d'un mirror parabolique

Dans la première simulation, le déplacement initial du robot en translation est égal à $\mathbf{t} = [-0.5,\ 1.0,\ -1.0]$ (en mètres). La rotation initiale effectuée a un axe égal à $[0.41,\ 0.41,\ -0.82]$ et un angle égal à 36.74 degrés. La Figure 7.18 montre le résultat d'une tâche de positionnement. Dans la Figure 7.18(a), nous pouvons voir la trajectoire des points dans l'image. Ces points convergent vers leurs coordonnées de référence. La loi de commande est stable. En effet, dans la Figure 7.18(b), nous pouvons voir que les vitesses en rotation et en translation convergent vers zéro et que le robot a atteint sa position de référence (voir la Figure 7.18(c) pour l'erreur en translation et la Figure 7.18(d) pour l'erreur en rotation).

7.6.3.2 Cas d'un mirror hyperbolique

Dans la deuxième simulation, le miroir est hyperbolique et le déplacement initial est plus grand que celui de la simulation précédente : la translation initiale est égale à $\mathbf{t} = [-1.0,\ 2.0,\ -2.0]$ (en mètre) et l'axe de la rotation initiale est $[0.47,\ -0.40,\ 0.79]$ et son angle est égal à 76.13 degrés. Malgré le fait que nous n'avons démontré que la stabilité locale de la loi de commande, c'est-à-dire, pour de faibles déplacements, nous pouvons voir que la loi de commande est également stable pour de grands déplacements (voir la Figure 7.19(b)). La translation et la rotation de la caméra convergent tous les deux vers zéro (voir Figures 7.19(c) et Figures 7.19(d)), ce qui veut dire que le robot a atteint sa position de référence à l'issue de l'asservissement visuel. Nous pouvons voir à partir de la trajectoire des points dans la Figure 7.19(a), qu'à la position finale, les points courants coïncident avec les points de référence.



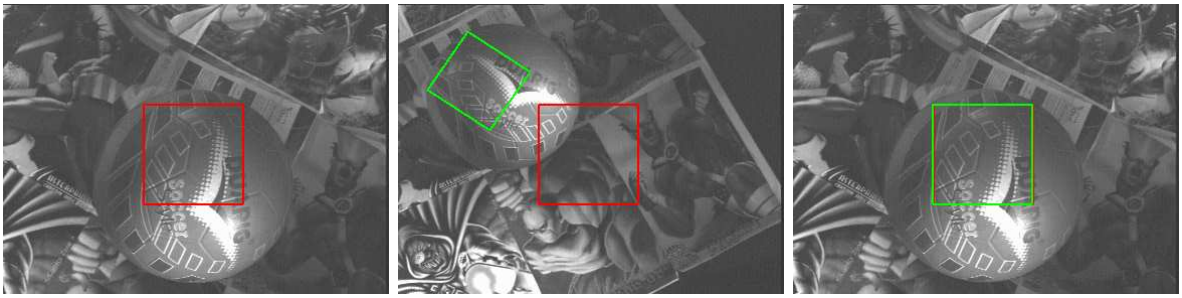
(a) Vitesse en translation m/s

(b) Vitesse en rotation deg/s

(c) Erreur en translation m

(d) Erreur en rotation deg

Figure 7.15 – Courbes des vitesses de rotations et de translation et courbes des erreurs de translation et de rotation



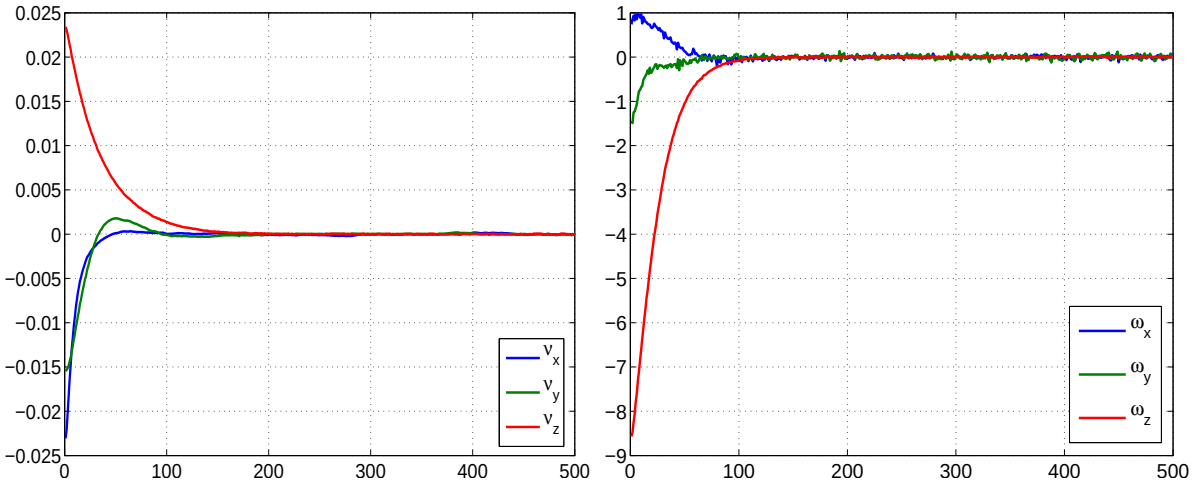
(a) Image de référence (b) Image initiale (c) Image finale

Figure 7.16 – AV par rapport à une cible non plane avec une caméra calibrée avec un grand déplacement initial

7.7 Conclusion

Dans ce chapitre, nous avons proposé pour la première fois une nouvelle approche directe d'asservissement visuel qui ne nécessite aucune connaissance sur la mesure du modèle 3D de la cible observée. Nous avons proposé une loi de commande simple et nous avons prouvé de manière analytique sa stabilité locale. Nous pensons que cette approche directe ouvre une nouvelle direction de recherche dans le domaine de l'asservissement visuel. En effet, à notre connaissance aucune méthode existante ne permet de repositionner d'une manière stable et précise un robot par rapport à une image de référence d'une cible plane sans connaissance a priori (ni approximation, ni estimation en ligne) de la mesure de la structure 3D de la cible.

Plusieurs améliorations de l'approche proposée sont envisageables. Par exemple, une bonne robustesse aux erreurs de calibration de la caméra a été observée dans les simulations numériques et dans les expériences mais n'a pas encore été démontrée analytiquement. Par ailleurs, une grande zone de stabilité a été observée dans les simulations et dans les expériences. Il serait intéressant de connaître les limites de cette zone d'une manière analytique. En attendant, une plus grande région de stabilité peut être garantie avec une planification de trajectoire (telle que celle proposée dans (Mezouar and Chaumette, 2002)). Ceci permet de prendre en compte également les contraintes de visibilité de la cible afin d'assurer la faisabilité de la tâche. En utilisant la méthode que nous proposons pour effectuer l'asservissement visuel, il est possible d'effectuer une planification de trajectoire pour les grands déplacements sans avoir des mesures sur le modèle 3D de la cible, contrairement à la méthode de (Mezouar and Chaumette, 2002) où le modèle 3D est nécessaire.



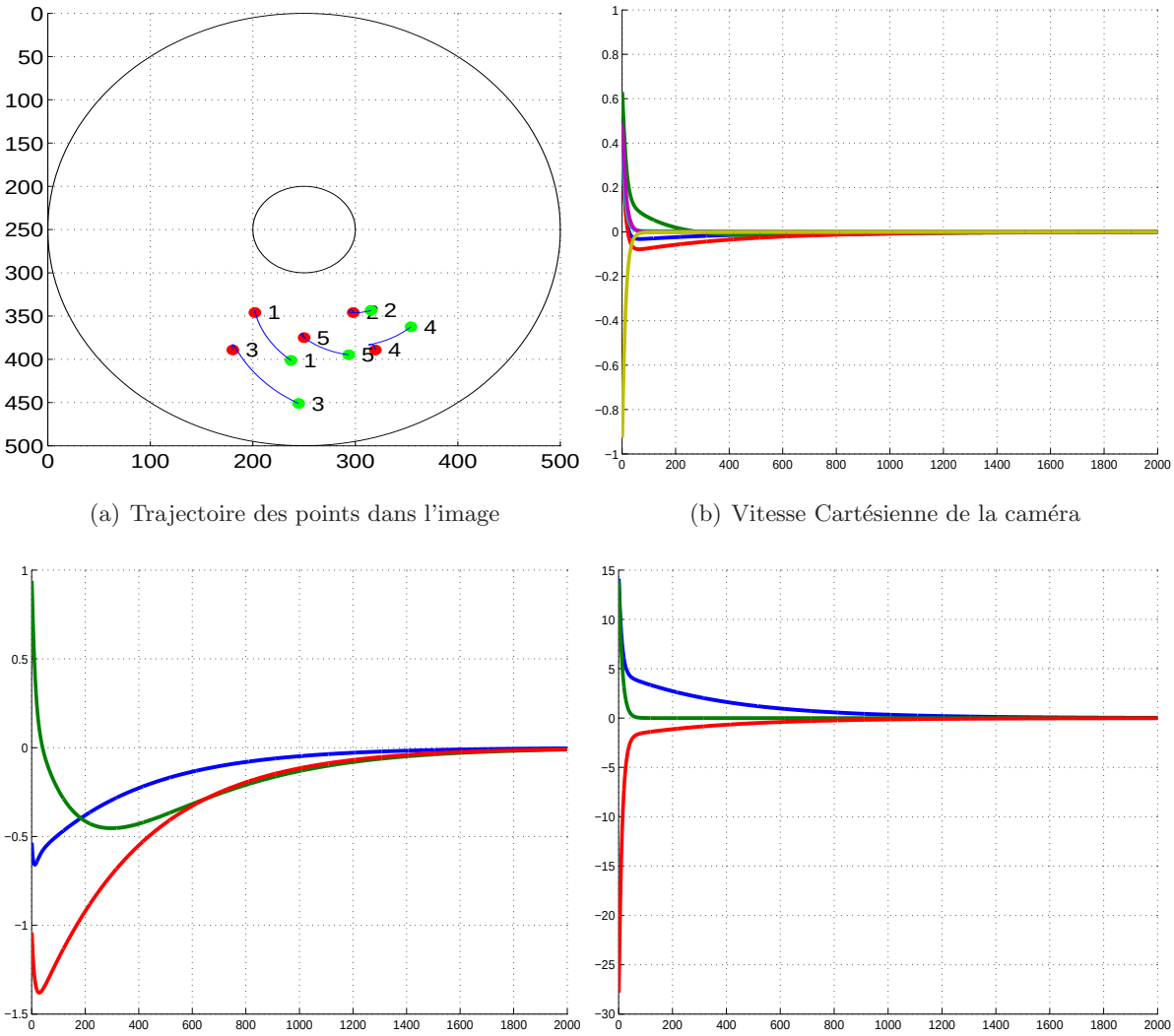
(a) Vitesse en translation m/s

(b) Vitesse en rotation deg/s

(c) Erreur en translation m

(d) Erreur en rotation deg

Figure 7.17 – Courbes des vitesses de rotations et de translation et courbes des erreurs de translation et de rotation



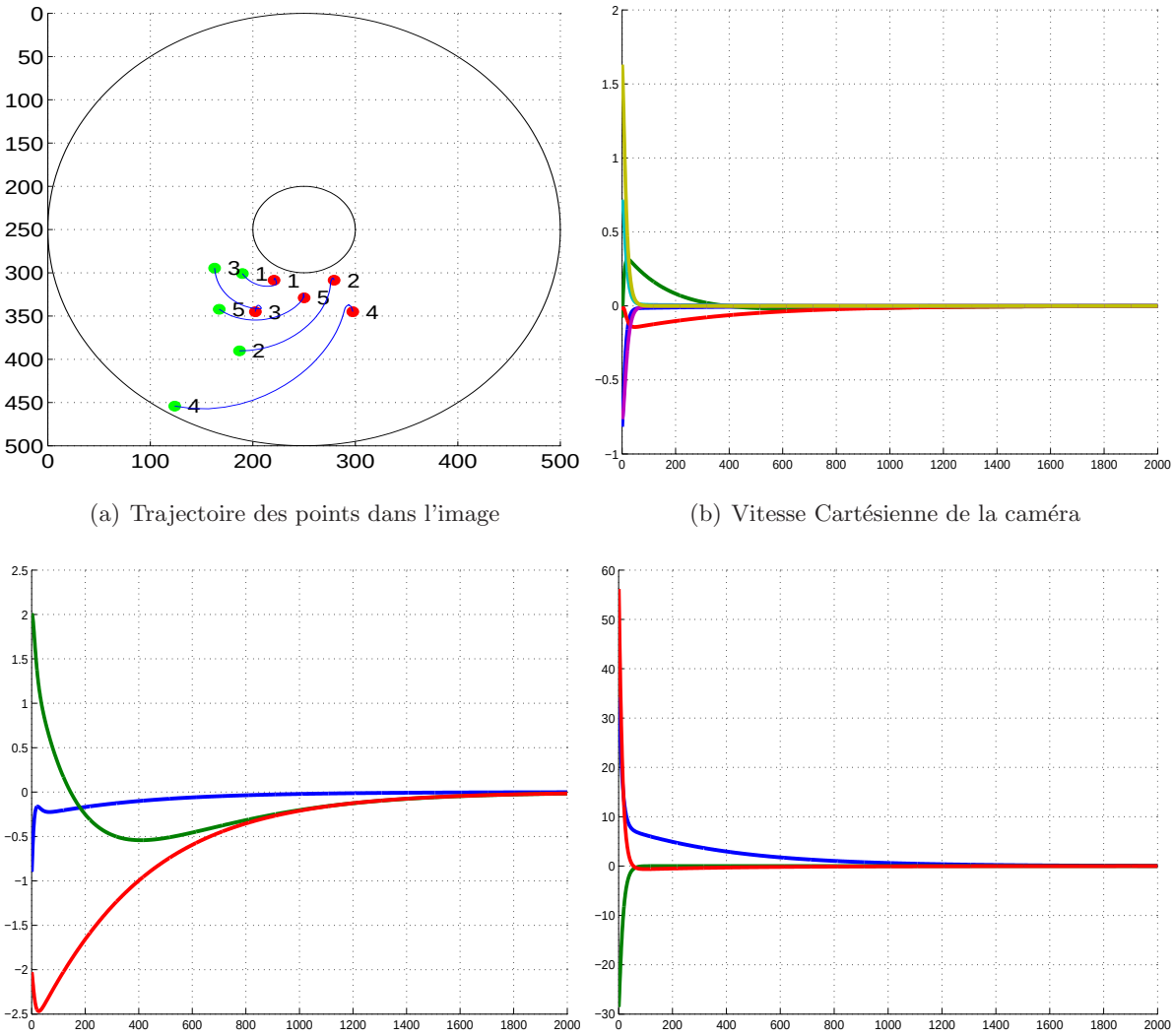
(a) Trajectoire des points dans l'image

(b) Vitesse Cartésienne de la caméra

(c) Erreur en translation

(d) Erreur en rotation

Figure 7.18 – Simulation 1 : AV avec une caméra omnidirectionnelle (miroir parabolique)



(a) Trajectoire des points dans l'image

(b) Vitesse Cartésienne de la caméra

(c) Erreur en translation

(d) Erreur en rotation

Figure 7.19 – Simulation 2 : AV avec une caméra omnidirectionnelle (miroir hyperbolique)

Chapitre 8

Suivi visuel et positionnement par rapport à une cible mobile

8.1 Présentation du système

Nous présentons dans ce chapitre une application de l'algorithme de suivi visuel avec la méthode ESM couplé avec l'asservissement visuel 2D direct que nous venons de présenter. Nous utilisons le robot Cartésien à 6 degrés de liberté de l'équipe LAGADIC de l'IRISA (voir la Figure 7.7). L'objectif et de suivre dans l'image d'une cible plane puis de contrôler le robot de façon à ce que la transformation entre la caméra montée sur l'organe terminal et la cible reste constante. Contrairement aux expériences présentées dans le chapitre 7, ici, la cible effectue un mouvement dans l'espace 3D. Nous effectuons deux expériences différentes.

- Dans la première, le mouvement de la cible est connu. En effet, il est périodique et l'objectif est de vérifier que le suivi visuel présenté dans le chapitre 4 et que l'asservissement visuel présenté dans le chapitre 7 sont bien coordonnés et permettent de garantir un positionnement correct du robot.
- Dans la deuxième expérience, le mouvement de la cible est aléatoire. La vérification sera uniquement visuelle, c'est-à-dire, nous observerons le comportement du suivi visuel et de la commande vis-à-vis à de tel mouvement de cible.

8.2 Cible ayant un mouvement périodique

Nous commençons par positionner manuellement le robot face à un banc linéaire mobile sur lequel on accroche une cible planaire texturée. Sur cette cible, nous sélectionnons une région de dimensions (100×100) par rapport à laquelle nous voulons le repositionner. La position relative au moment de la sélection correspond à la position de référence. À partir de ce moment là, le suivi et l'asservissement visuels sont lancés.

Le mouvement appliqué à la cible est linéaire et périodique. La vitesse du banc est de 3 centimètres par seconde. La caméra utilisée acquiert des images de dimensions (384×288). Elle est calibrée et la matrice des paramètres intrinsèques est celle de l'équation (7.28). Les gains de la loi de commande décrite par l'équation (7.17) utilisés sont : $\lambda_\nu = \lambda_\omega = 1$.

Au cours de cette expérience, le suivi visuel fournit de très bon résultats. En effet, la position de la zone sélectionnée est très bien estimée dans les images acquises. Nous pouvons le voir à partir de quelques extraits des images acquises au cours de l'asservissement visuel (voir Figure 8.2).

Le mouvement du robot permet de compenser le mouvement de la cible. La vitesse dans l'espace Cartésien du robot est tracée dans la Figure 8.2(a) et la Figure 8.2(b). Étant donné que le mouvement de la cible est un mouvement de translation, nous observons une vitesse en rotation très faible variant entre -2 et $+2$ degrés par seconde (voir la Figure 8.2(b)). La vitesse de la translation est périodique et atteint parfois la vitesse de la cible, c'est-à-dire, 3 centimètres par seconde. La rotation effectuée par le robot est également périodique mais faible entre -3 et $+2$ degrés (voir la Figure 8.2(d)). La translation quant à elle est clairement périodique (voir la Figure 8.2(c)) avec une amplitude de 30 centimètres suivant $\vec{x}$ et d'environ 10 centimètres suivant $\vec{y}$ et suivant $\vec{z}$.

Dans cette expérience, nous avons appliqué la loi de commande présentée dans le chapitre précédent d'une manière brute, c'est-à-dire, sans rajouter de filtres, d'intégrateurs ou de dérivateurs. Or cette commande est destinée à des tâches de positionnement par rapport à une cible fixe. Ce qui fait qu'une erreur de traînage peut être observée lors de cette expérience. Cette erreur est statique quand la vitesse de la cible est constante. Cette erreur est maximale quand la cible est au milieu du banc linéaire puisqu'elle atteint sa vitesse maximale. L'erreur est minimale quand la cible atteint un bout du banc parce que sa vitesse s'annule dans ce cas là (voir les images de la Figure 8.2). Afin de réduire, voire annuler cette erreur, rajouter un intégrateur dans la loi de commande est nécessaire.

8.3 Cible ayant un mouvement quelconque

Dans la deuxième expérience, le mouvement de la cible est effectué manuellement. La vérification sera uniquement visuelle, c'est-à-dire, nous observerons l'attitude du suivi visuel et de la commande vis-à-vis à d'un mouvement quelconque de la cible.

En effet, nous positionnons le robot de façon à ce que la caméra soit en face d'une cible planaire texturée. Il est possible de faire faire manuellement un mouvement quelconque à cette cible. D'une manière similaire, l'objectif est de vérifier que le suivi visuel présenté dans le chapitre 4 et que l'asservissement visuel présenté dans le chapitre 7 sont bien coordonnées et permettent de garantir un positionnement correct du robot par rapport à un mouvement quelconque.

Le robot est commandé de manière à avoir une position relative constante par rapport à la

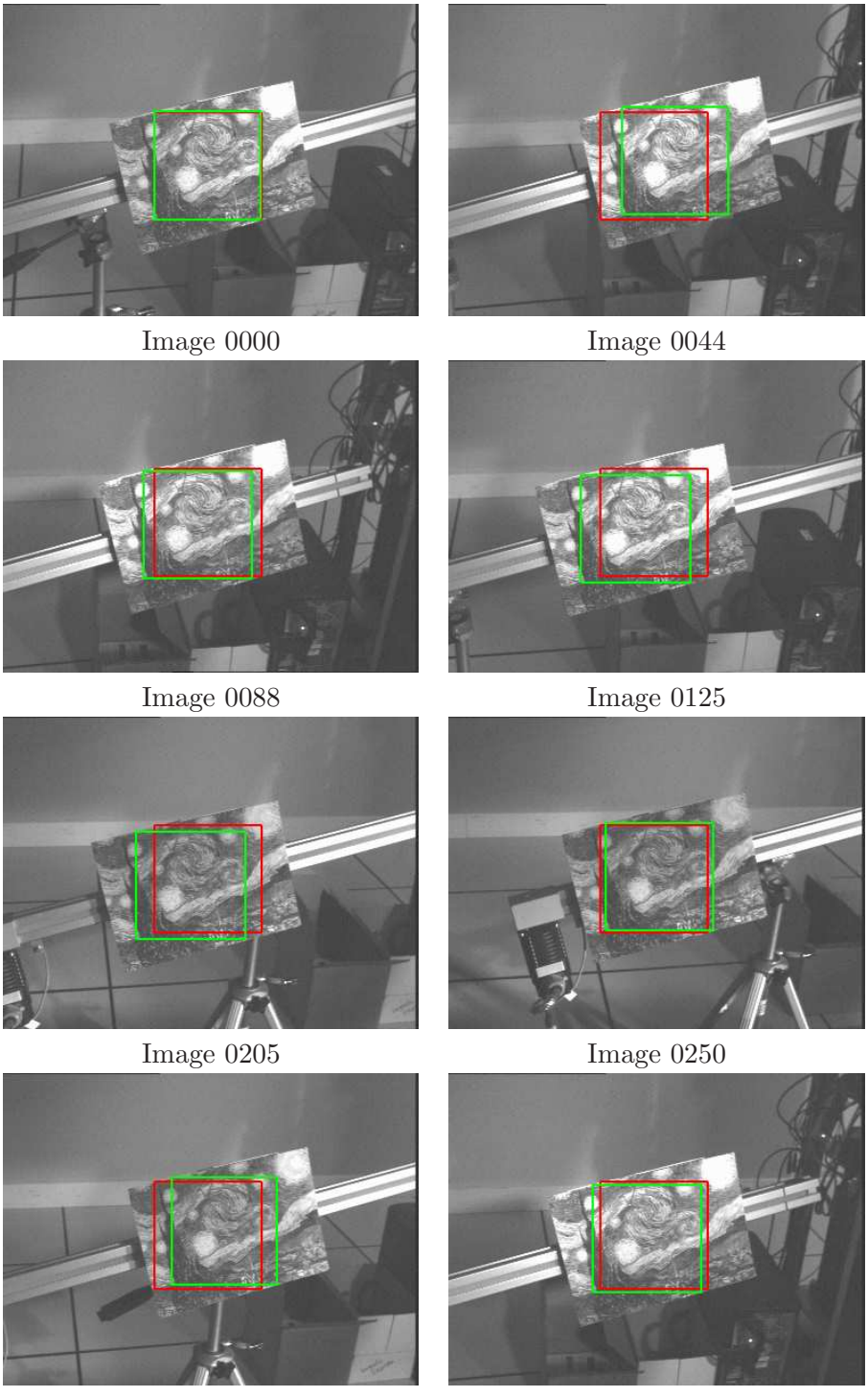


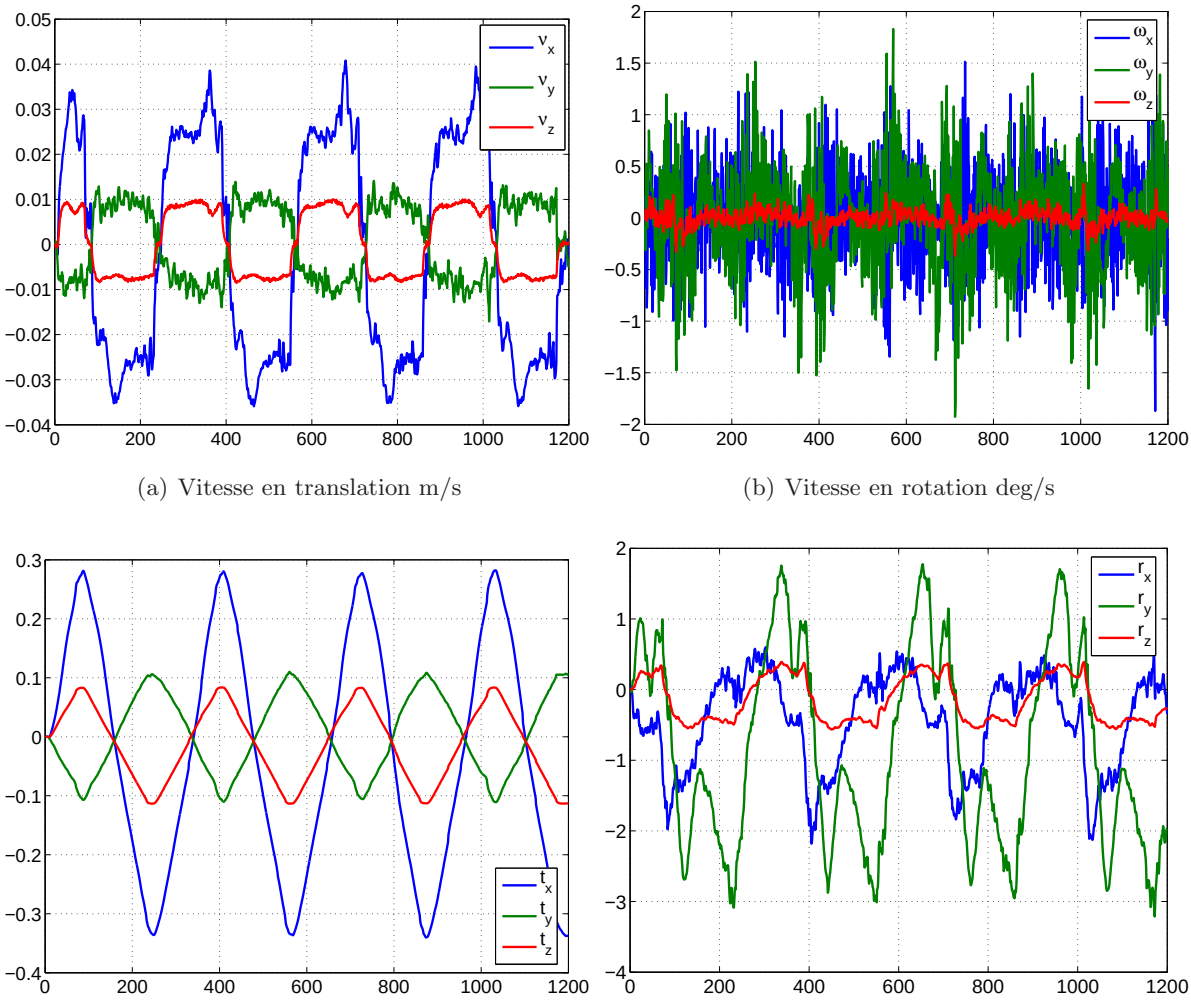
Image 0000 Image 0044

Image 0088 Image 0125

Image 0205 Image 0250

Image 0295 Image 0409

Figure 8.1 – Quelques images de l'asservissement visuel



(a) Vitesse en translation m/s

(b) Vitesse en rotation deg/s

(c) Erreur en translation m

(d) Erreur en rotation deg

Figure 8.2 – Courbes de la vitesse et de la commande en translation et en rotation

cible. Cette position correspond à celle où l'image de référence a été acquise. Au cours de cette expérience, nous n'avons pas enregistré les mouvements du robot. Néanmoins, nous pouvons voir, visuellement, à partir de la Figure 8.3 que la commande permet bien d'assurer la tâche désirée.

Figure 8.3 – Quelques images de l'application du suivi et asservissement visuel

Chapitre 9

Accrochage de véhicule par vision

9.1 Présentation du système

Nous présentons dans ce chapitre une application directe de l'algorithme de suivi visuel avec la méthode ESM. Il s'agit d'une application de suivi de véhicule basée uniquement sur de la vision mono-caméra ou encore une application d'"accrochage immatériel". Deux véhicules électriques du type "Cycab" (voir la Figure 9.1) sont utilisés. Le premier véhicule (le bleu) est le véhicule leader et le deuxième (le rouge) est le véhicule asservi. Le véhicule leader est conduit manuellement à l'aide d'une manette (ou "joystick").

Figure 9.1 – Scénario de l'application de suivi de véhicule par vision

Le véhicule asservi est équipé d'une caméra CCD permettant l'acquisition d'images de dimensions (640×480) à une fréquence de 25 images/seconde via une carte d'acquisition. La caméra est montée sur une tourelle pan-tilt située derrière le pare-brise avant (voir la Figure 9.1).

Figure 9.2 – Position de la tourelle pan-tilt

Deux ordinateurs standards (chacun équipé d'un processeur Pentium III à 700 MHz) sont utilisés. Sur le premier ordinateur s'effectuent l'algorithme de vision et la commande de la tourelle pan-tilt via une liaison série RS232. Sur le deuxième ordinateur, la loi de commande haut-niveau de la vitesse longitudinale et de la vitesse angulaire de braquage des roues est calculée. Cette commande est ensuite transférée via une liaison par bus CAN vers le contrôle bas-niveau des effecteurs du "Cycab". Un réseau Ethernet est mis en place entre les deux ordinateurs et le transfert de données se fait grâce à une communication par socket TCP/IP. Pour une vision globale du système, se référer à la Figure 9.3.

La commande de la tourelle et la commande de la vitesse longitudinale et de la vitesse angulaire de braquage des roues du véhicule asservi sont effectuées grâce seulement aux images acquises par la caméra en temps-réél, à la fréquence vidéo de 25Hz (soit toutes les 40 millisecondes). Étant donné que le pare-brise arrière du véhicule leader est transparent, un poster y a été accroché pour servir de cible de référence au suivi visuel. Le système d'acquisition d'images (la caméra et la carte d'acquisition), l'algorithme de suivi visuel, le contrôle de la tourelle pan-tilt et l'estimation de pose agissent en tant que capteur unique permettant de fournir l'erreur en position $\mathbf{e}_q = \mathbf{q} - \mathbf{q}^*$ et l'erreur en orientation $e_\psi = \psi - \psi^*$ entre les deux véhicules. L'erreur en position est l'erreur entre la position $\mathbf{q}$ du repère attaché au véhicule asservi et la position $\mathbf{q}^*$ d'un repère déporté du repère du véhicule leader (translaté d'une distance d^*) et rigidement lié à celui-ci. En pratique, nous considérons que le repère du véhicule asservi coïncide avec la base immobile de la tourelle et la distance entre ce repère et celui du véhicule leader est d^* qui n'est autre que la distance entre le centre de projection de la caméra et le poster au début du suivi visuel, c'est-à-dire, la distance à la position initiale. La position des repères et les erreurs régulées sont illustrées dans la Figure 9.4. Afin d'avoir une reconstruction métrique et des gains

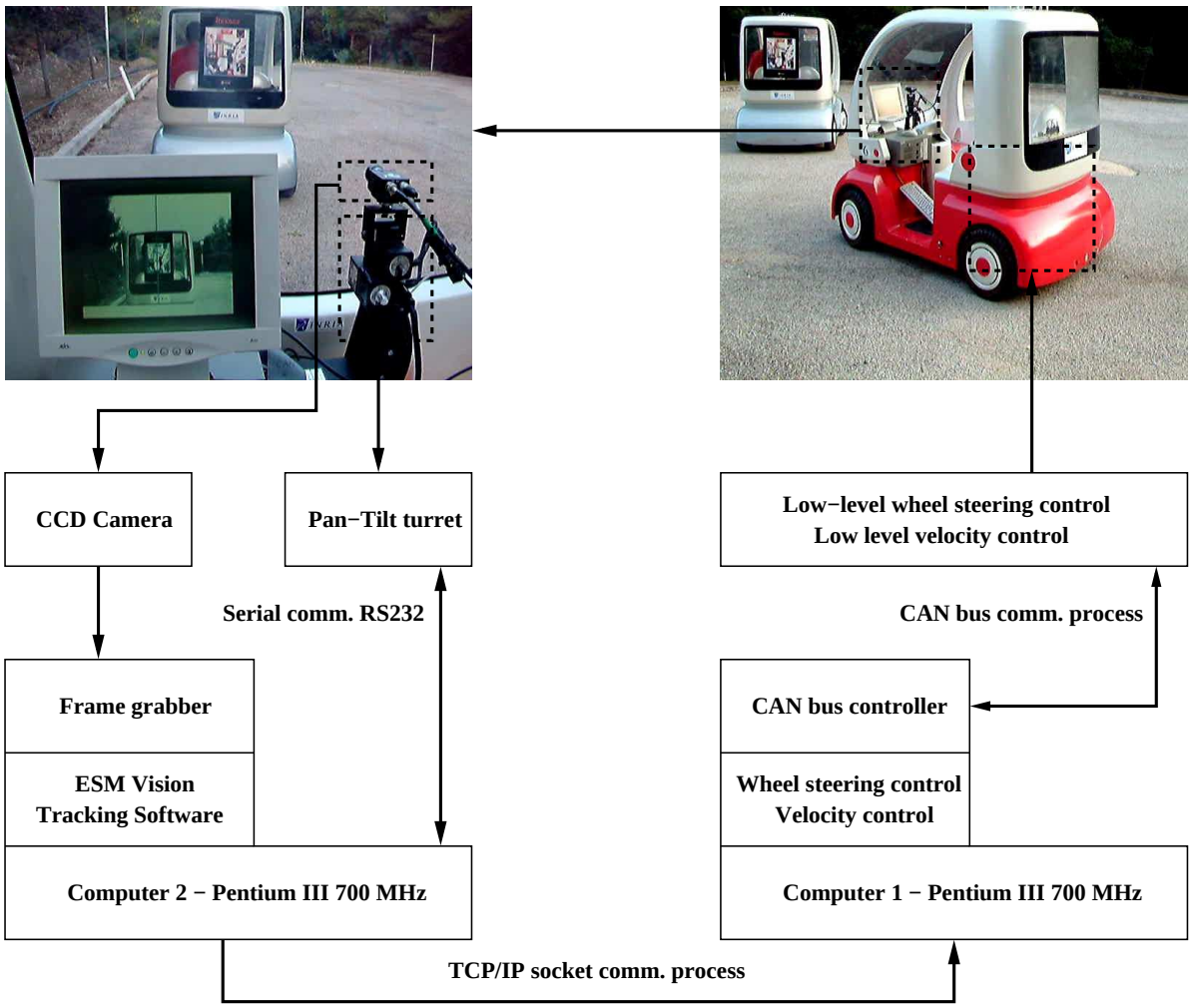


Figure 9.3 – Système de suivi de véhicule par vision

raisonnables, la caméra a été grossièrement calibrée et une approximation de la distance entre les deux Cycabs à la position initiale est donnée à l'algorithme de contrôle.

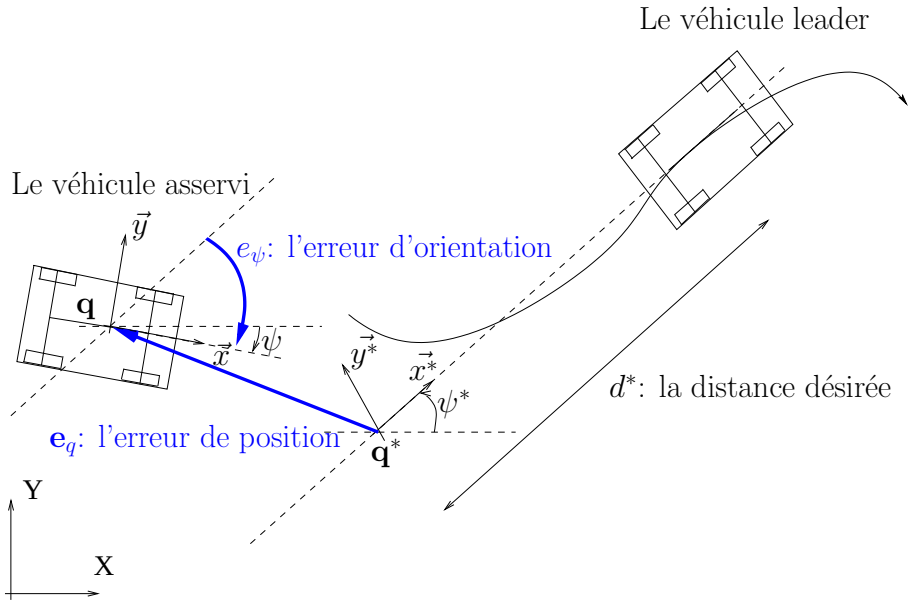


Figure 9.4 – Erreur régulée lors du suivi de véhicule

9.2 Description de la commande

À la position initiale, où $\mathbf{e}_q = \mathbf{0}$ et $e_\psi = 0$, le véhicule leader est dans le champ de vision de la caméra. Une imagette de référence de dimensions (100×100) pixels est sélectionnée manuellement sur le poster collé sur le véhicule leader (voir Figure 9.5). Dès cet instant, l'algorithme

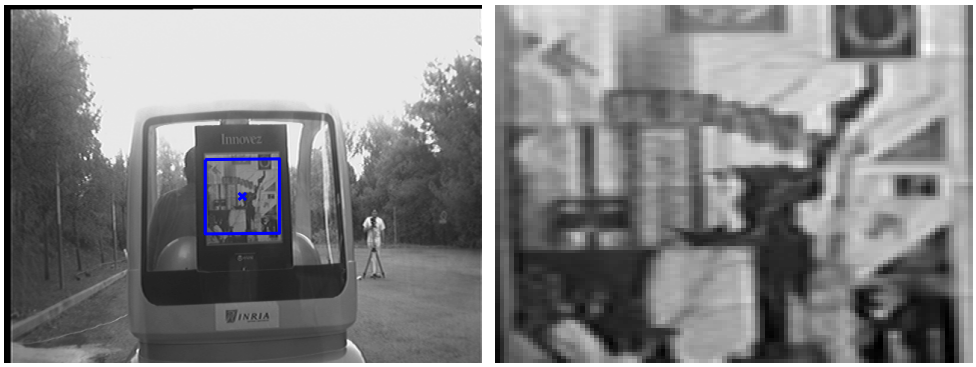


Figure 9.5 – Sélection de la fenêtre de référence

de suivi visuel estime la transformation projective $\mathbf{G}$ de l'imagette dans l'image.

La tourelle est commandée de façon à garder le véhicule leader dans le champ de vue de la

caméra. La commande de la vitesse du pan et du tilt se fait à l'aide d'un simple retour d'état proportionnel sur l'erreur entre le centre de la fenêtre d'intérêt (u, v) et le centre de l'image (u_c, v_c) dans la position courante, les vitesses appliquées au pan v_{pan} et au tilt v_{tilt} vérifient :

$$\begin{pmatrix} v_{pan} \\ v_{tilt} \end{pmatrix} = - \begin{pmatrix} k_{pan} & 0 \\ 0 & k_{tilt} \end{pmatrix} \begin{pmatrix} u - u_c \\ v - v_c \end{pmatrix} \tag{9.1}$$

où k_{pan} et k_{tilt} sont des gains strictement positifs. Une telle commande permet de faire coïncider le centre de la fenêtre d'intérêt et le centre de l'image.

La variation de la position de la caméra par rapport au véhicule leader est mesurée en temps-réel grâce à la matrice de transformation projective ${}^c\mathbf{G}_r$ de l'imagette. Nous rappelons que la matrice ${}^c\mathbf{G}_r$ s'écrit :

$${}^c\mathbf{G}_r = \mathbf{K} \left({}^c\mathbf{R}_r + {}^c\mathbf{t}_r \frac{\mathbf{n}^*}{d^*} \right) \mathbf{K}^{-1}$$

Donc, ayant une approximation de la matrice des paramètres intrinsèques de la caméra $\mathbf{K}$, du vecteur normal au plan du poster $\mathbf{n}^*$ et de la distance d^*, il est possible de calculer ${}^c\mathbf{R}_r$ et ${}^c\mathbf{t}_r$ qui sont respectivement la matrice de rotation et le vecteur de translation (exprimés dans le repère de référence) entre la position de référence et la position courante de la caméra. L'algorithme utilisé pour trouver ${}^c\mathbf{R}_r$ et ${}^c\mathbf{t}_r$ à partir de la matrice ${}^c\mathbf{G}_r$ est décrit dans (Faugeras and Lustman, 1988).

Nous notons par ${}^a\mathbf{T}_c$ la matrice de transformation homogène entre le repère du véhicule asservi et la position courante de caméra. Cette matrice dépend de la géométrie de la tourelle et de la position courante du pan et du tilt. Nous notons par ${}^c\mathbf{T}_r$ la matrice de transformation homogène entre la position de référence et la position courante de la caméra. Cette matrice est construite à l'aide de ${}^c\mathbf{R}_r$ et de ${}^c\mathbf{t}_r$:

$${}^c\mathbf{T}_r = \begin{bmatrix} {}^c\mathbf{R}_r & {}^c\mathbf{t}_r \\ \mathbf{0} & 1 \end{bmatrix}$$

Les erreurs à réguler $\mathbf{e}_q$ et e_ψ peuvent être obtenues à partir de la matrice de la transformation homogène ${}^a\mathbf{T}_r$ entre le repère du véhicule asservi et la position de référence de caméra :

$${}^a\mathbf{T}_r = {}^a\mathbf{T}_c \, {}^c\mathbf{T}_r$$

Plusieurs approches de commandes de la vitesse et de l'angle de braquage des roues peuvent être envisagées. Dans (Benhimane et al., 2005), nous décrivons une méthode possible basée sur un suivi de trajectoire. Il est également possible d'utiliser une loi de commande qui approche celle du suivi de trajectoire. Nous pouvons contrôler la vitesse longitudinale $\dot{x}$ à l'aide d'un retour d'état proportionnel à l'erreur $\mathbf{e}_q(1,1) = x - x^*$. Le braquage des roues δ dépend de l'erreur

transversale $\mathbf{e}_q(2,1) = y - y^*$ et de l'erreur en orientation $e_\psi = \psi - \psi^*$. La loi de commande simplifiée au voisinage de la position d'équilibre peut être écrite de la manière suivante :

$$\left\{ \begin{array}{rcl} \dot{x} & = & -k_x(x - x^*) \\ \delta & = & -k_y(y - y^*) - k_\psi(\psi - \psi^*) \end{array} \right. \tag{9.2}$$

où k_x, k_y et k_ψ sont des gains positifs.

9.3 Résultats expérimentaux

La Figure 9.3 montre les résultats expérimentaux de l'application d'accrochage immatériel. Nous pouvons voir, en haut à gauche, la trajectoire des deux véhicules (celle du véhicule leader en bleu et celle du véhicule asservi en vert). Les deux trajectoires (d'environ 80 mètres) ne sont pas parfaitement identique. En effet, la loi de commande appliquée fait que le véhicule asservi a tendance à couper les virages. Pour palier à ce problème, nous pensons qu'une recherche plus poussée sur les lois de commande du véhicule suiveur serait nécessaire.

Les courbes en haut à gauche montrent les mesures de la position relative telle qu'elles sont fournies par le système de vision. Les courbes en vert correspondent à la distance longitudinale entre les deux véhicules. Nous observons que cette distance est régulée autour de 3 mètres. Les courbes en bleu et en rouge correspondent respectivement à la distance transversale et à l'erreur d'orientation entre les deux véhicules. Ils sont régulés autour de zéro. À cause de la cinématique de la commande du suivi, lorsque le véhicule leader effectue un virage, les valeurs absolues des erreurs augmentent.

Les deux courbes en bas à droite montrent le résultat des mesures de la distance longitudinale et de l'erreur de l'orientation entre les deux véhicules après avoir appliqué un filtrage des mesures obtenues par le système de vision.

Finalement, en bas à droite, nous pouvons voir la commande de la vitesse d'avancement (en bleu) et le braquage des roues (en vert) du véhicule asservi. La vitesse d'avancement a été saturée à 1 mètre par seconde et cette saturation à été atteinte à la fin de la trajectoire. Nous pouvons voir également qu'elle est bruitée et un filtrage serait nécessaire dans de futures expériences. Le braquage des roues, quant à lui, est moins bruité.

Dans la Figure 9.7, nous pouvons voir, dans la première colonne, des images acquises par une caméra extérieure lors de l'expérience. Dans la deuxième colonne, nous pouvons voir des images acquises par la caméra embarquée dans le véhicule asservi. Sur ces images, nous avons représenté, par un rectangle bleu, la position (estimée par l'algorithme de suivi visuel) de l'imagette utilisée en tant que référence pour déterminer la position relative entre les deux véhicules. Grâce à la tourelle pan-tilt, la position dans l'imagette est toujours plus ou moins centrée dans l'image au cours de l'expérience. Malgré le fait que l'expérience a été effectuée dans un environnement extérieur (avec ombres portées et réflexions lumineuses dues au soleil) le suivi visuel a donné de

bons résultats. En effet, en observant la troisième colonne de la Figure 9.7, la reprojection de l'imagette suivie reste la même tout au long de l'expérience.

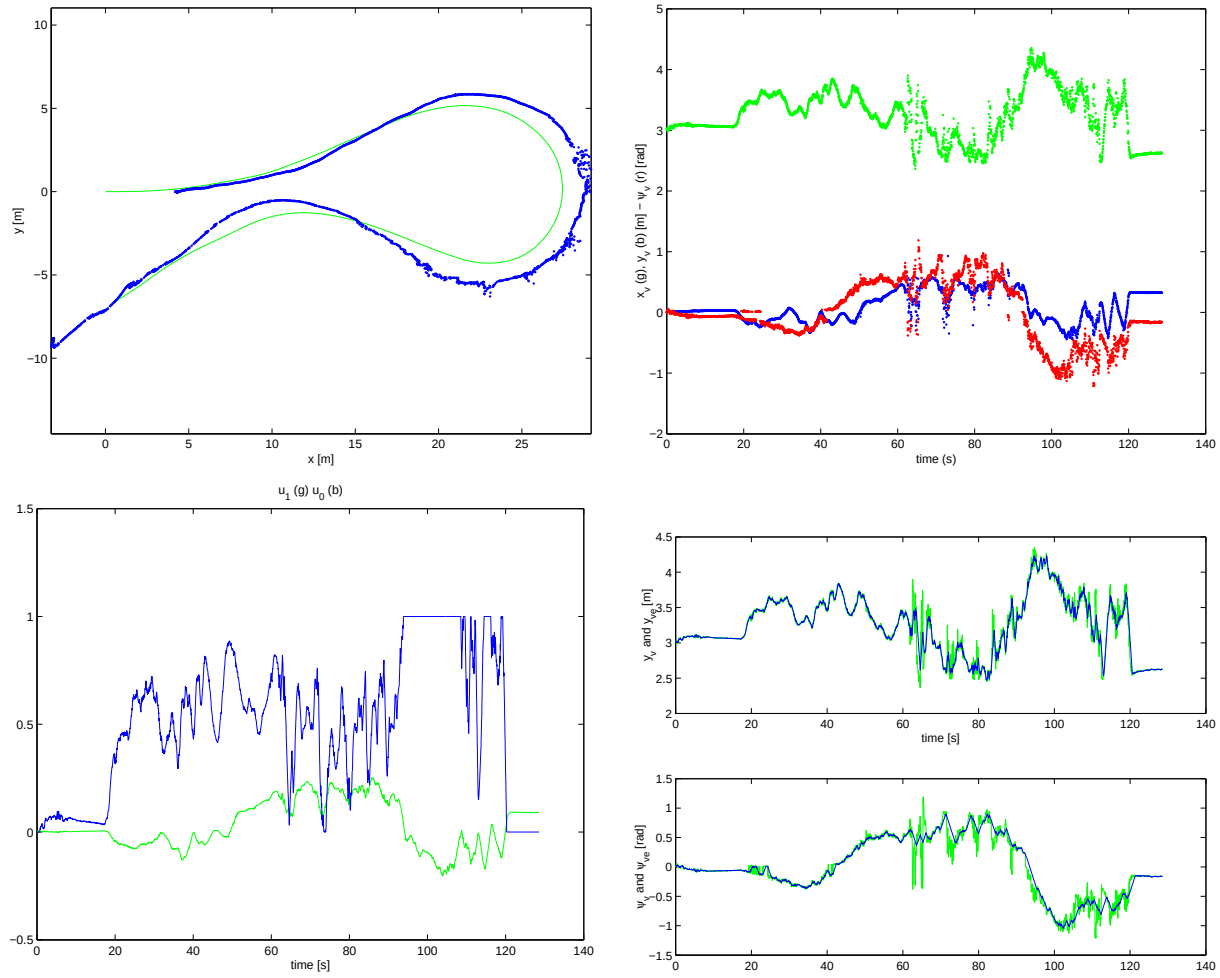


Figure 9.6 – Résultats expérimentaux de l'accrochage immatériel

Dans la Figure 9.8, nous montrons quelques photos d'une expérience effectuée à Nancy dans le cadre du projet MobiVIP (Véhicules Individuels Publics pour la Mobilité en centre ville). Comme le montrent les deux premières images, ici, la tourelle pan-tilt et la caméra sont montées sur le toit du Cycab (au lieu d'être à l'intérieur comme précédemment). Dans les deux images de la deuxième ligne de la Figure 9.8, nous pouvons voir l'algorithme du suivi visuel permettant d'estimer la position relative des deux véhicules. Au cours de cette expérience, il est possible de voir la robustesse du suivi visuel par rapport aux conditions d'éclairement de la cible. Par exemple, dans la première image de la deuxième ligne, le véhicule est à l'ombre, puis dans la deuxième image de la deuxième ligne, le véhicule est au soleil. Cette expérience a été réalisée sur une place publique. Ceci nous a permis de voir que si nous choisissons l'imagette de référence assez large dans l'image, il est possible de continuer la tâche d'accrochage par vision

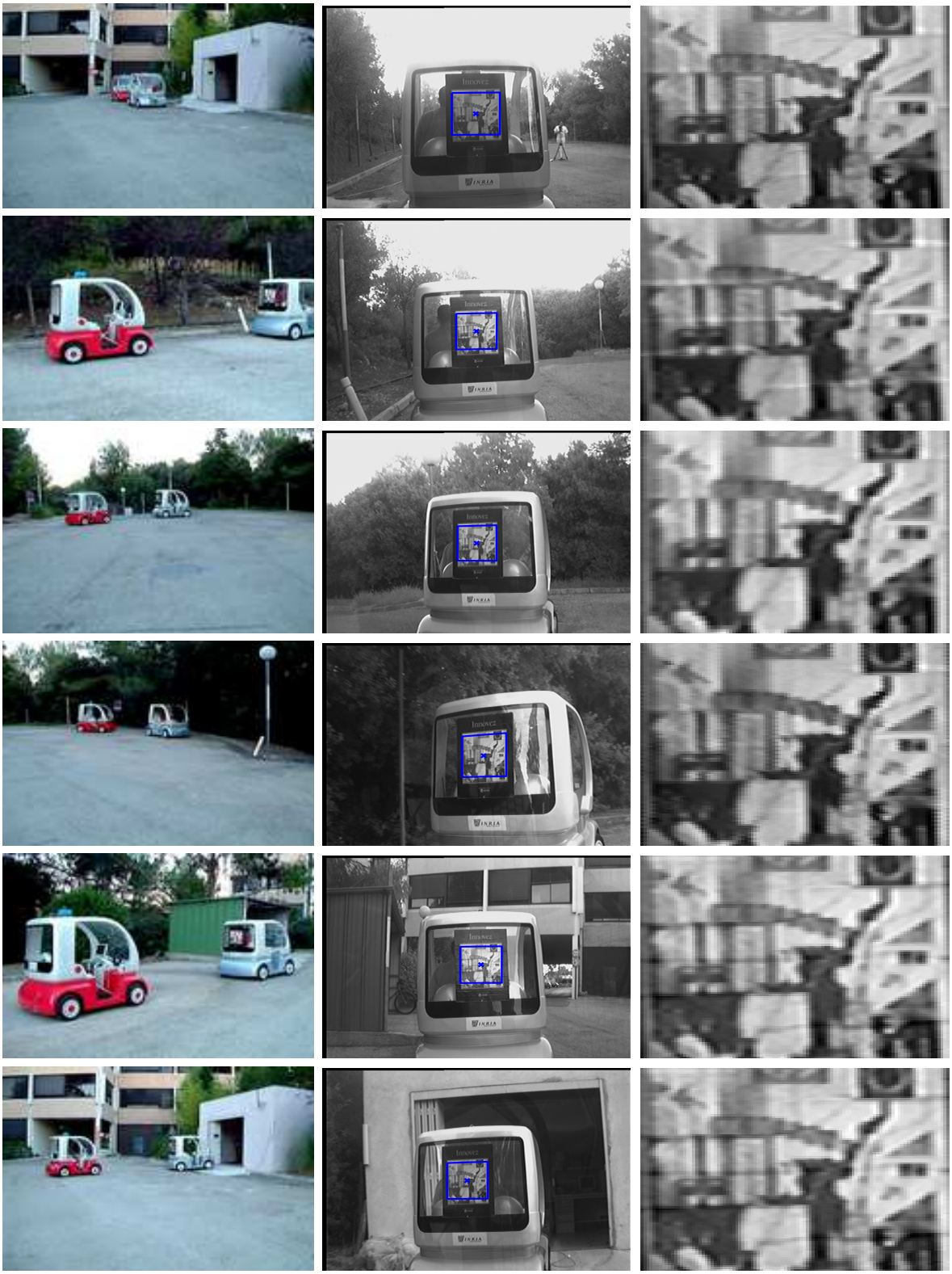


Figure 9.7 – Quelques images de l'application d'accrochage immatériel

même dans le cas où un piéton passe entre les deux véhicules. Voir la troisième ligne d'images de la Figure 9.8 où à chaque instant, le piéton cache une partie de la cible.

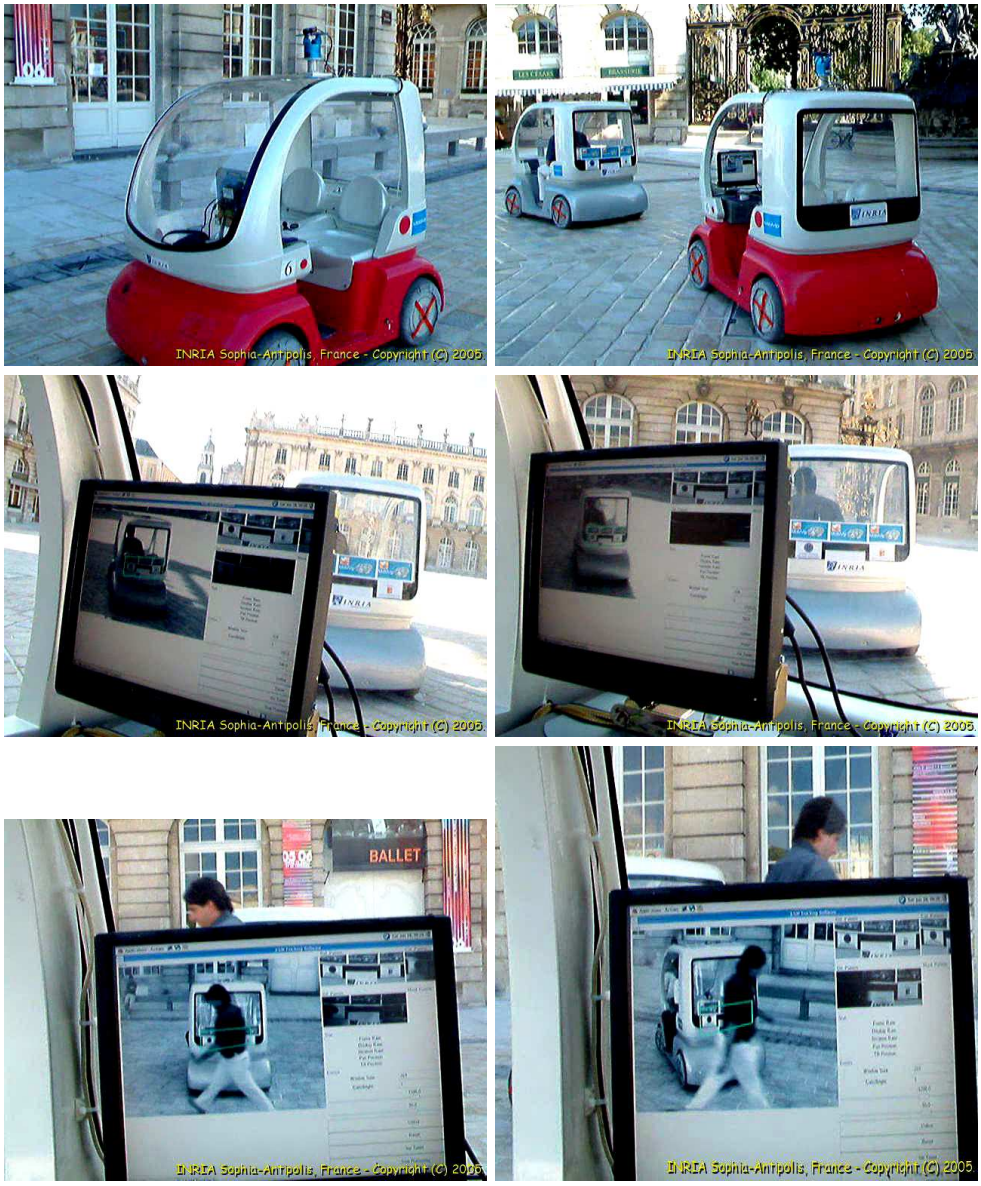


Figure 9.8 – Quelques photos d'accrochage immatériel à Nancy

Quatrième partie

Conclusions et perspectives

Conclusions

Dans ce manuscrit, nous avons présenté certains travaux effectués lors de cette thèse. Il s'agit de contributions qui vont permettre de concevoir des systèmes de commande par vision efficaces où la partie vision par ordinateur et la partie commande interagissent d'une manière unifiée. Le suivi visuel fonctionne avec tout type d'objets naturels sans qu'il y ait besoin de rajouter des marqueurs artificiels. En effet, la seule contrainte nécessaire pour les deux méthodes de suivi visuel introduites est que l'image de la cible contienne de la texture. La structure ou bien la mesure de cette texture n'est pas nécessaire et aucun type particulier de primitives visuelles n'est indispensable. Les deux algorithmes de suivi visuel sont rapides et peuvent être utilisés dans des applications de commande par vision temps-réel. La robustesse aux conditions d'éclairement et aux occultations partielles est intrinsèquement assez grande. Mais, aucune étude quantitative n'a été réalisée jusque là.

Dans ce manuscrit, nous avons traité deux cas différents : le suivi visuel dans l'image et le suivi visuel dans l'espace Cartésien. Dans le premier cas, le modèle de la cible est inconnu et dans le deuxième cas, le modèle de la cible est fourni. Il est possible de concevoir deux approches différentes de commande par vision :

1. Le suivi visuel dans l'image avec la méthode ESM, présenté dans le Chapitre 4, couplé avec la commande par asservissement visuel direct, présentée dans le Chapitre 7, permet un positionnement temps réel du robot précis, robuste et stable et ce sans connaissance de mesures du modèle 3D de la cible par rapport à laquelle l'asservissement visuel est effectué (voir les exemples dans le paragraphe 7.6 et dans le chapitre 8). Nous avons démontré que cette commande est stable localement. Différentes simulations et expériences ont montré que le domaine de stabilité est assez grand. Nous avons présenté des applications de suivi et d'asservissements visuels par rapport à des objets plans. Cependant, il est possible de généraliser l'approche pour des objets non plans (car il est possible de définir des homographies par rapport à des objets non plans, si l'on rajoute le terme de parallaxe).

2. Le suivi visuel dans l'espace Cartésien avec la méthode ESM, présenté dans le Chapitre 7, peut être utilisé avec la commande par asservissement visuel 2D 1/2. En effet, en ayant la normale au plan de la cible, ce suivi visuel permet d'estimer la position dans l'image de cette cible, tout en fournissant la rotation nécessaire au calcul de la fonction de tâche de l'asservissement visuel 2D 1/2 (sans avoir à décomposer l'homographie). Le suivi visuel est alors plus rapide car seuls six degrés de liberté sont à estimer, et l'homographie calculée est toujours cohérente avec le modèle de la cible.

Nous pensons que ce travail ouvre un très grand nombre de perspectives de recherche dans

les deux domaines : celui de la vision et celui de la commande.

Perspectives

Perspectives pour la vision

Dans ce manuscrit, nous avons traité deux cas différents : le suivi visuel dans l'image et le suivi visuel dans l'espace Cartésien. Dans le premier cas, le modèle de la cible est inconnu et dans le deuxième cas, le modèle de la cible est fourni. Il nous semble qu'il serait très intéressant d'essayer de reconstruire le modèle de la cible au cours du suivi visuel. Nous avons obtenu récemment des résultats très encourageants dans le cas d'un modèle planaire par morceaux (Mei et al., 2006b). Cette reconstruction peut être faite d'une manière récursive ou bien une fois pour toute, une fois que la cible aurait effectué un mouvement assez grand.

Par ailleurs, comme nous l'avons déjà dit dans l'introduction, la robustesse aux conditions d'éclairement et aux occultations partielles est intrinsèquement assez grande. Cependant, il est possible d'améliorer cette robustesse à l'aide d'outils standards d'optimisation et de vision par ordinateur. Par exemple, la gestion des occultations partielles peut être effectuée si nous utilisons des estimateurs robustes tels que les M-estimateurs. Cependant, d'une manière générale, la vitesse de convergence de la minimisation se verra réduite avec de tels estimateurs. Pour ce qui est du changement d'éclairement, quand celui-ci est linéaire et uniforme sur toute l'image, sa gestion est évidente (il suffit de normaliser les erreurs en retranchant la moyenne et en divisant par l'écart-type). Quand les variations d'intensité deviennent non-linéaire, alors un modèle de réponse de la cible aux sources lumineuses doit être établi et une minimisation sur les paramètres du modèle doit être effectuée. Si le suivi visuel est effectué par rapport à une cible immobile et des sources d'éclairage fixes, il est possible également d'effectuer une calibration photométrique, c'est-à-dire, reconstruire d'une manière itérative la position, l'orientation et l'intensité de ces sources lumineuses.

Nous pensons qu'un filtrage peut améliorer la qualité et la vitesse du suivi. Un filtre permettra d'avoir une estimation de mouvement plus conforme à la dynamique des objets suivis et permettra d'avoir à chaque nouvelle image acquise une prédiction de mouvement de la cible. Pour ce qui du suivi dans l'espace Cartésien, la réalisation d'un tel filtre est facile. Cependant, pour le suivi dans l'image, sa réalisation n'est pas très évidente. Une étude théorique serait nécessaire pour traduire le mouvement dans l'espace Cartésien en variations des paramètres dans l'algèbre de Lie $\mathfrak{sl}(3)$.

Une autre perspective logique à ce travail est de pouvoir effectuer le suivi visuel par rapport à des objets non rigides : des objets déformables ou bien des objets articulés. La tâche du suivi visuel serait non seulement d'estimer la position et l'orientation de ce type d'objets, mais aussi, les déformations effectuées. Pour que le problème soit bien posé, il faut se restreindre aux déformations qui ne présentes pas d'ambiguïté avec l'estimation de la pose.

Perspectives pour la commande

La commande proposée dans ce manuscrit permet de positionner un robot par rapport à une cible sans avoir de mesure sur le modèle 3D de cette cible. La commande proposée est stable localement. Le domaine de stabilité s'est avéré, grâce aux diverses simulations réalisées très grand. Il serait très intéressant d'effectuer une étude analytique pour pouvoir définir les limites de stabilité de cette commande. La commande est également assez robuste aux erreurs de calibration des paramètres intrinsèques de la caméra utilisée. Il faudrait envisager en tant que perspectives de recherche une étude de robustesse par rapport aux erreurs commises sur ces paramètres.

Étant donné que la commande présentée est couplée, c'est-à-dire, une translation pure (resp. rotation pure) peut induire une commande en rotation (resp. en translation), la visibilité de la cible dans l'image n'est pas garantie au cours de l'asservissement visuel et ce malgré le fait que la commande en translation est complètement définie dans l'espace Cartésien. Pour garantir la visibilité de la cible, il est possible d'effectuer une planification de trajectoire. Nous avons testé (sans les présenter) quelques stratégies de planification de trajectoire, mais faute de temps, nous n'avons pas effectué une étude poussée sur ce sujet. Une perspective possible serait de tester et évaluer différentes stratégies de planification pour cette commande.

Cinquième partie

Annexes

Annexe A

Diverses réalisations

Un site Web a été créé décrivant l'approche du suivi visuel dans l'image de surfaces planaires avec la méthode ESM :
http ://www-sop.inria.fr/icare/personnel/malis/software/ESM.html. Sur le site Web, une implémentation en C peut être téléchargée (une version compilée) et testée pour des applications temps-réel ou pour des applications de suivi visuel hors-ligne [1].

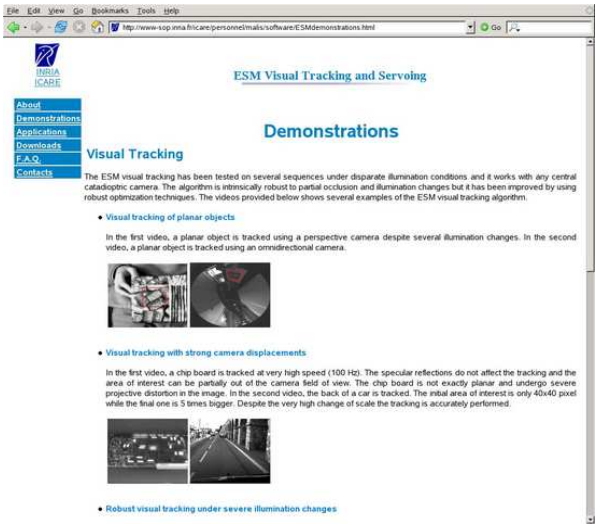


Figure A.1 – Capture d'écran du site Web de l'algorithme ESM

Nous avons également développé une interface utilisateur pour des applications temps-réel capable de gérer plusieurs type de caméras. Cette interface a été réalisée en GTK [2]. Dans la Figure A.2, nous pouvons voir des captures d'écran de cette interface.

1. Deux versions de la librarie existent pour deux systèmes d'exploitation différents : ESMlib.a pour Linux (compilée avec gcc 3.3.3.) et ESMlib.dll pour Windows® (compilée avec Visual .net 7.1.)
2. GTK : Gimp Tool Kit

(a) Selection de la fenêtre (b) Suivi visuel de la fenêtre

Figure A.2 – Capture d'écran de l'interface graphique

Annexe B

Preuve de la formule 4.46

Nous utilisons la notation introduite dans (2.1). Grâce à l'expression explicite du Jacobien $\mathbf{J_G}$ de l'équation (4.34), nous avons :

$$\mathbf{J_G}\widetilde{\mathbf{x}} = \sum_{k=1}^{8} \widetilde{x}_k \left[\mathbf{A}_k\right]_v = \left[\sum_{k=1}^{8} \widetilde{x}_k \mathbf{A}_k\right]_v = \left[\mathbf{A}(\widetilde{\mathbf{x}})\right]_v$$

Grâce à la propriété (4.24) de la paramétrisation à l'aide de l'algèbre de Lie $\mathfrak{sl}(3)$, le Jacobien $\mathbf{J}_{\widetilde{\mathbf{G}}}$ peut s'écrire :

$$\mathbf{J}_{\widetilde{\mathbf{G}}} = \nabla_{\mathbf{x}} \mathbf{G}(\widetilde{\mathbf{x}})^{-1} \mathbf{G}(\mathbf{x})\big|_{\mathbf{x}=\widetilde{\mathbf{x}}} = \nabla_{\mathbf{x}} \mathbf{G}(-\widetilde{\mathbf{x}}) \mathbf{G}(\mathbf{x})\big|_{\mathbf{x}=\widetilde{\mathbf{x}}}$$

Soit la variable intermédiaire $\mathbf{z} = \mathbf{x} - \widetilde{\mathbf{x}}$ ou encore $\mathbf{x} = \mathbf{z} + \widetilde{\mathbf{x}}$. Sachant que $\nabla_{\mathbf{x}}\mathbf{z}\big|_{\mathbf{x}=\widetilde{\mathbf{x}}} = \mathbf{I}$, le Jacobien $\mathbf{J}_{\widetilde{\mathbf{G}}}$ peut donc s'écrire :

$$\mathbf{J}_{\widetilde{\mathbf{G}}} = \nabla_{\mathbf{z}} \mathbf{G}(-\widetilde{\mathbf{x}}) \mathbf{G}(\mathbf{z} + \widetilde{\mathbf{x}})\big|_{\mathbf{z}=\mathbf{0}}$$

En utilisant la généralisation du gradient aux matrices, nous pouvons écrire :

$$\mathbf{J}_{\widetilde{\mathbf{G}}} = \left[\left[\frac{\partial(\mathbf{G}(-\widetilde{\mathbf{x}})\mathbf{G}(\mathbf{z}+\widetilde{\mathbf{x}}))}{\partial z_1}\Big|_{\mathbf{z}=\mathbf{0}}\right]_v \quad \left[\frac{\partial(\mathbf{G}(-\widetilde{\mathbf{x}})\mathbf{G}(\mathbf{z}+\widetilde{\mathbf{x}}))}{\partial z_2}\Big|_{\mathbf{z}=\mathbf{0}}\right]_v \quad \cdots \quad \left[\frac{\partial(\mathbf{G}(-\widetilde{\mathbf{x}})\mathbf{G}(\mathbf{z}+\widetilde{\mathbf{x}}))}{\partial z_8}\Big|_{\mathbf{z}=\mathbf{0}}\right]_v \right]$$

Par conséquent, en multipliant $\mathbf{J}_{\widetilde{\mathbf{G}}}$ par $\widetilde{\mathbf{x}}$, nous obtenons :

$$\mathbf{J}_{\widetilde{\mathbf{G}}}\widetilde{\mathbf{x}} = \sum_{k=1}^{8} \widetilde{x}_k \left[\frac{\partial(\mathbf{G}(-\widetilde{\mathbf{x}})\mathbf{G}(\mathbf{z}+\widetilde{\mathbf{x}}))}{\partial z_k}\Big|_{\mathbf{z}=\mathbf{0}}\right]_v = \left[\sum_{k=1}^{8} \widetilde{x}_k \frac{\partial(\mathbf{G}(-\widetilde{\mathbf{x}})\mathbf{G}(\mathbf{z}+\widetilde{\mathbf{x}}))}{\partial z_k}\Big|_{\mathbf{z}=\mathbf{0}}\right]_v$$

Maintenant, nous factorisons la somme par $\mathbf{G}(-\widetilde{\mathbf{x}})$:

$$\mathbf{J}_{\widetilde{\mathbf{G}}}\widetilde{\mathbf{x}} = \left[\mathbf{G}(-\widetilde{\mathbf{x}}) \sum_{k=1}^{8} \widetilde{x}_k \frac{\partial(\mathbf{G}(\mathbf{z}+\widetilde{\mathbf{x}}))}{\partial z_k}\Big|_{\mathbf{z}=\mathbf{0}}\right]_v$$

Si nous utilisons la définition de la parmétrisation à l'aide de l'algèbre de Lie $\mathfrak{sl}(3)$ (voir équations (4.21) et (4.22)), nous avons alors :

$$\left.\frac{\partial(\mathbf{G}(\mathbf{z}+\widetilde{\mathbf{x}}))}{\partial z_k}\right|_{\mathbf{z}=\mathbf{0}} = \left.\frac{\partial \exp(\sum_{j=1}^{8}(z_j+\widetilde{x}_j)\mathbf{A}_j)}{\partial z_k}\right|_{\mathbf{z}=\mathbf{0}} = \left.\frac{\partial \exp(z_k\mathbf{A}_k+\mathbf{A}(\widetilde{\mathbf{x}}))}{\partial z_k}\right|_{z_k=0}$$

Or, nous avons la propriété suivante : $\forall\,(\mathbf{X},\mathbf{Y}) \in \mathbb{M}_n(\mathbb{R}) \times \mathbb{M}_n(\mathbb{R})$ et $\forall t \in \mathbb{R}$:

$$\left.\frac{\partial \exp(t\mathbf{Y}+\mathbf{X})}{\partial t}\right|_{t=0} = \exp(\mathbf{X})\frac{\mathbf{I}-\exp(-\mathrm{ad}\mathbf{X})}{\mathrm{ad}\mathbf{X}}(\mathbf{Y})$$

où l'application linéaire ad$\mathbf{X}$ est définie grâce au crochet de Lie $[.,.]$ de la manière suivante :

$$\mathrm{ad}\mathbf{X}(\mathbf{Y}) = [\mathbf{X},\mathbf{Y}] = \mathbf{XY}-\mathbf{YX}$$

et l'application linéaire $\dfrac{\mathbf{I}-\exp(-\mathrm{ad}\mathbf{X})}{\mathrm{ad}\mathbf{X}}$ est définie de la manière suivante :

$$\frac{\mathbf{I}-\exp(-\mathrm{ad}\mathbf{X})}{\mathrm{ad}\mathbf{X}}(\mathbf{Y}) = \mathbf{Y} - \frac{1}{2!}[\mathbf{X},\mathbf{Y}] + \frac{1}{3!}[\mathbf{X},[\mathbf{X},\mathbf{Y}]] - ...$$

Par conséquent, nous avons :

$$\left.\frac{\partial(\mathbf{G}(\mathbf{z}+\widetilde{\mathbf{x}}))}{\partial z_k}\right|_{\mathbf{z}=\mathbf{0}} = \exp(\mathbf{A}(\widetilde{\mathbf{x}}))\frac{\mathbf{I}-\exp(-\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}}))}{\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}})}(\mathbf{A}_k)$$

Or, par définition, nous avons : $\exp(\mathbf{A}(\widetilde{\mathbf{x}})) = \mathbf{G}(\widetilde{\mathbf{x}})$. D'où :

$$\mathbf{J}_{\widetilde{\mathbf{G}}}\widetilde{\mathbf{x}} = \left[\mathbf{G}(-\widetilde{\mathbf{x}})\sum_{k=1}^{8}\widetilde{x}_k\mathbf{G}(\widetilde{\mathbf{x}})\frac{\mathbf{I}-\exp(-\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}}))}{\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}})}(\mathbf{A}_k)\right]_v$$

En factorisant, dans la somme, par $\mathbf{G}(\widetilde{\mathbf{x}})$, et en utilisant la propriété (4.24), nous obtenons :

$$\mathbf{J}_{\widetilde{\mathbf{G}}}\widetilde{\mathbf{x}} = \left[\sum_{k=1}^{8}\widetilde{x}_k\frac{\mathbf{I}-\exp(-\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}}))}{\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}})}(\mathbf{A}_k)\right]_v$$

L'application $\dfrac{\mathbf{I}-\exp(-\mathrm{ad}\mathbf{X})}{\mathrm{ad}\mathbf{X}}$ est linéaire, donc il est possible d'écrire :

$$\sum_{k=1}^{8}\widetilde{x}_k\frac{\mathbf{I}-\exp(-\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}}))}{\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}})}(\mathbf{A}_k) = \frac{\mathbf{I}-\exp(-\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}}))}{\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}})}(\mathbf{A}(\widetilde{\mathbf{x}}))$$

Finalement, d'après la définition, de l'application $\dfrac{\mathbf{I} - \exp(-\mathrm{ad}\mathbf{X})}{\mathrm{ad}\mathbf{X}}$ et celle du crochet de Lie, nous obtenons :

$$\mathbf{J}_{\widetilde{\mathbf{G}}}\widetilde{\mathbf{x}} = \left[\frac{\mathbf{I} - \exp(-\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}}))}{\mathrm{ad}\mathbf{A}(\widetilde{\mathbf{x}})}(\mathbf{A}(\widetilde{\mathbf{x}}))\right]_v = [\mathbf{A}(\widetilde{\mathbf{x}})]_v$$

Par conséquent, nous avons bien :

$$\mathbf{J}_{\widetilde{\mathbf{G}}}\widetilde{\mathbf{x}} = \mathbf{J}_{\mathbf{G}}\widetilde{\mathbf{x}}$$

Bibliographie

Baker, S. and Matthews, I. (2001). « Equivalence and efficiency of image alignment algorithms ». In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 1090–1097.

Baker, S. and Matthews, I. (2004). « Lucas-Kanade 20 years on : a unifying framework ». *International Journal of Computer Vision*, 56(3) :221–255.

Baker, S. and Nayar, K. (1999). « A theory of single-viewpoint catadioptric image formation ». *International Journal of Computer Vision*, 35(2) :1–22.

Baker, S., Patil, R., Cheung, K., and Matthews, I. (2004). « Lucas-Kanade 20 Years On : Part 5 ». Technical report CMU-RI-TR-04-64, Robotics Institute, Carnegie Mellon University.

Barreto, J. and Araujo, H. (2002). « Geometric properties of central catadioptric line images ». In *European Conf. on Computer Vision*, pages 237–251.

Barreto, J.-P., Martin, F., and Horaud, F. (2003). « *Visual Servoing/Tracking Using Central Catadioptric Images, Springer Tracts in Advanced Robotics 5* », Chapitre IV, pages 245–254. Springer Verlag, experimental robotics, viii édition.

Bascle, B., Bouthemy, P., Deriche, R., and Meyer, F. (1994). « Tracking complex primitives in an image sequence ». In *Int. Conf. on Pattern Recognition*.

Basri, R., Rivlin, E., and Shimshoni, I. (1998). « Visual Homing : Surfing on the Epipoles ». In *IEEE Int. Conf. on Computer Vision*, pages 863–869.

Bayro-Corrochano, E. and Ortegon-Aguilar, J. (2004). « Lie algebra for template tracking ». In *Int. Conf. on Pattern Recognition*, pages 55–59.

Benhimane, S. and Malis, E. (2003). « Vision-based control with respect to planar and non-planar objects using a zooming camera ». In *IEEE Int. Conf. on Advanced Robotics*, pages 991–996.

Benhimane, S. and Malis, E. (2004a). « Mise en correspondance d'images à différentes résolutions à l'aide d'invariants aux paramètres intrinsèques ». In *14ème Congrès Francophone AFRIF-AFIA de Reconnaissance des Formes et Intelligence Artificielle*.

Benhimane, S. and Malis, E. (2004b). « Real-time image-based tracking of planes using efficient second-order minimization ». In *IEEE Int. Conf. on Intelligent Robots and Systems*, pages 943–948.

Benhimane, S. and Malis, E. (2004c). « Self-calibration of the distortion of a zooming camera by matching points at different resolutions ». In *IEEE Int. Conf. on Intelligent Robots and Systems*, pages 2307–2312.

Benhimane, S. and Malis, E. (2006a). « Homography-based 2D Visual Servoing ». In *IEEE International Conference on Robotics and Automation*.

Benhimane, S. and Malis, E. (2006b). « Integration of Euclidean constraints in template-based visual tracking of piecewise-planar scenes ». In *IEEE/RSJ International Conference on Intelligent Robots Systems*.

Benhimane, S. and Malis, E. (2006c). « A new approach to vision-based robot control with omni-directional cameras ». In *IEEE International Conference on Robotics and Automation*.

Benhimane, S. and Malis, E. (2006d). « Une approche directe pour l'asservissement visuel 2D ». In *15eme Congres Francophone AFRIF-AFIA de Reconnaissance des Formes et Intelligence Artificielle*.

Benhimane, S. and Malis, E. (2007). « Homography-based 2D Visual Tracking and Servoing ». *International Journal of Computer Vision*.

Benhimane, S., Malis, E., Rives, P., and Azinheira, J.-R. (2005). « Vision-based Control for Car Platooning using Homography Decomposition ». In *IEEE Int. Conf. on Robotics and Automation*, pages 2173–2178.

Black, M. and Jepson, A. (1998). « Eigen-tracking : robust matching and tracking of articulated objects using a view-based reprsentation ». *IEEE Journal of Computer Vision*, 26(1) :63–84.

Bouguet, J.-Y. (1999). « *Pyramidal Implementation of the Lucas-Kanade Feature Tracker* ». OpenCV Documentation, Microprocessor Research Labs, Intel Corporation.

Buenaposada, J. and Baumela, L. (2002). « Real-time tracking and estimation of planar pose ». In *Int. Conf. on Pattern Recognition*, pages 697–700.

Canny, J. F. (1986). « A computational approach to edge detection ». *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6) :679–698.

Chaumette, F. (1998). Potential problems of stability and convergence in image-based and position-based visual servoing. In Kriegman, D., Hager, G., and Morse, A., editors, *The confluence of vision and control*, volume 237 of *LNCIS Series*, pages 66–78. Springer Verlag.

Chaumette, F. (2004). « Image moments : a general and useful set of features for visual servoing ». *IEEE Trans. on Robotics*, 20(4) :713–723.

Chiba, N. and Kanade, T. (1998). « A Tracker for Broken and Closely Spaced Lines ». In *Proceedings of the 1996 International Society for Photogrammetry and Remote Sensing Conference*, pages 676 – 683.

Chiuso, A., Favaro, P., Jin, H., and Soatto, S. (2002). « Structure from Motion Causally Integrated Over Time ». *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4) :523–535.

Cobzas, D. and Jagersand, M. (2004). « 3D SSD Tracking from Uncalibrated Video ». In *Proc. of Spatial Coherence for Visual Motion Analysis, in conjunction with ECCV*.

Comport, A., Marchand, E., and Chaumette, F. (2004). « Robust model-based tracking for robot vision ». In *IEEE/RSJ Int. Conf. on Intelligent Robots Systems*, pages 692–697.

Cootes, T., Edwards, G., and Taylor, C. (1998). « Active appearence models ». In *European Conf. on Computer Vision*, pages 484–498.

Corke, P. and Hutchinson, S. (2001). « A new partitioned approach to image-based visual servo control ». *IEEE Transactions on Robotics and Automation*, 14(4) :507–515.

Cowan, N. J. and Chang, D. E. (2002). Toward Geometric Visual Servoing. In Bicchi, A., Christensen, H., and Prattichizzo, D., editors, *Control Problems in Robotics*, volume 4 of *STAR, Springer Tracks in Advanced Robotics*, pages 233–248. Springer Verlag, Berlin Heidelberg.

Deguchi, K. (1998). « Optimal motion control for image-based visual servoing by decoupling translation and rotation ». In *IEEE Int. Conf. on Intelligent Robots and Systems*, pages 705–711.

Dementhon, D. and Davis, L. S. (1995). « Model-Based Object Pose in 25 Lines of Code ». *International Journal of Computer Vision*, 15(1/2) :123–141.

Deriche, R. (1987). « Using Canny's criteria to derive a recursively implemented optimal edge detector ». *International Journal of Computer Vision*, 1(2) :167–187.

Drummond, T. and Cipolla, R. (2002). « Real-time visual tracking of complex structures ». *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7) :932–946.

Espiau, B., Chaumette, F., and Rives, P. (1992). « A New Approach to Visual Servoing in Robotics ». *IEEE Trans. on Robotics and Automation*, 8(3) :313–326.

Faugeras, O. (1993). *Three-dimensionnal computer vision : a geometric viewpoint*. MIT Press, Cambridge, MA.

Faugeras, O. and Lustman, F. (1988). « Motion and Structure From Motion in a Piecewise Planar Environment ». *Int. Journal of Pattern Recognition and Artificial Intelligence*, 2(3) :485–508.

Forstner, W. (1994). « A Framework for Low-level Feature Extraction ». In *European Conf. on Computer Vision*, pages 383–394.

Geyer, C. and Daniilidis, K. (2000). « A unifying theory for central panoramic systems and practical applications ». In *European Conf. on Computer Vision*, pages 445–461.

Geyer, C. and Daniilidis, K. (2003). « Mirrors in motion : Epipolar geometry and motion estimation. ». In *IEEE Int. Conf. on Computer Vision*, pages 766–773.

Gleicher, M. (1997). « Projective registration with difference decomposition ». In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 331–337.

Hadj Abdelkader, H., Mezouar, Y., Andreff, N., and Martinet, P. (2005). « 2 1/2 D visual servoing with central catadioptric cameras ». In *IEEE/RSJ Int. Conf. on Intelligent Robots Systems*.

Hager, G. and Belhumeur, P. (1998). « Efficient region tracking with parametric models of geometry and illumination ». *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(10) :1025–1039.

Hager, G. and Toyama, K. (1998). « The XVision system : a general purpose substrate for portable real-time vision applications ». *Computer Vision and Image Understanding*, 69(1) :23–37.

Harris, C. and Stephens, M. (1988). « A combined corner and Edge Detector ». In *Proceedings of the 4th Alvey Vision Conf.*, pages 147–151.

Hartley, R. (1992). « Estimation of Relative Camera Positions for Uncalibrated Cameras ». In *European Conf. on Computer Vision*, pages 579–587.

Hartley, R. and Zisserman, A. (2000). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

Hashimoto, K. (1993). *Visual Servoing : Real Time Control of Robot manipulators based on visual sensory feedback*. World Scientific Series in Robotics and Automated Systems. World Scientific Press, Singapore.

Horaud, R. and Monga, O. (1993). *Vision par ordinateur, outils fondamentaux*. Traité des Nouvelles Technologies, série Informatique. Hermès, Paris, France.

Hutchinson, S., Hager, G. D., and Corke, P. I. (1996). « A tutorial on Visual Servo Control ». *IEEE Trans. on Robotics and Automation*, 12(5) :651–670.

Jin, H., Favaro, P., and Soatto, S. (2003). « A semi-direct approach to structure from motion ». In *The Visual Computer*, pages 343–356.

Jurie, F. and Dhome, M. (2002). « Hyperplane approximation for template matching ». *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7) :996–1000.

Kollnig, H. and Nagel, H.-H. (1997). « 3D pose estimation by directly matching polyhedral models to gray values gradients ». *International Journal of Computer Vision*, 23(3) :113–122.

La Cascia, M., Sclaroff, S., and Athitsos, V. (2000). « Fast, reliable head tracking under varying illumination : an approach based on registration of textured-mapped 3D models ». *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(4) :322–336.

Lapreste, J.-T. and Mezouar, Y. (2004). « A Hessian Approach to Visual Servoing ». In *IEEE Int. Conf. on Intelligent Robots and Systems*, pages 998–1003.

Longuet-Higgins, H. C. (1981). « A computer algorithm for reconstructing a scene from two projections ». *Nature*, 293 :133–135.

Longuet-Higgins, H. C. (1984). « The reconstruction of a scene from two projections : configurations that defeat the 8-point algorithm ». In *Proceedings of the 1st Conference on Artificial Intelligence Applications*, pages 395–397.

Lowe, D. (1991). « Fitting Parameterized Three-Dimensional Models to Images ». *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(5) :441–450.

Lowe, D. (1992). « Robust model-based motion tracking through the integration of search and estimation ». *International Journal of Computer Vision*, 8(2) :113–122.

Lowe, D. (2004). « Distinctive image features from scale-invariant keypoints ». *International Journal of Computer Vision*, 60(2) :91–110.

Lucas, B. and Kanade, T. (1981). « An iterative image registration technique with application to stereo vision ». In *Int. Joint Conf. on Artificial Intelligence*, pages 674–679.

Malis, E. (2004). « Improving vision-based control using efficient second-order minimization techniques ». In *IEEE Int. Conf. on Robotics and Automation*, pages 1843–1848.

Malis, E. and Benhimane, S. (2004). « A unified framework for real-time visual tracking and servoing ». In *IEEE/RSJ International Conference on Intelligent Robots and Systems. Workshop WWF4 Advances in Robot Vision - From Domestic Environments to Medical Applications.*

Malis, E. and Benhimane, S. (2005). « A unified approach to visual tracking and servoing ». *Robotics and Autonomous Systems*, 52(1) :39–52.

Malis, E. and Chaumette, F. (2002). « Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods ». *IEEE Trans. on Robotics and Automation*, 18(2) :176–186.

Malis, E., Chaumette, F., and Boudet, S. (1997). « Positioning a coarse-calibrated camera with respect to an unknown planar object by 2D 1/2 visual servoing ». In *Proc. 5th IFAC Symposium on Robot Control (SYROCO'97)*, pages 517–523.

Malis, E., Chaumette, F., and Boudet, S. (1999). « 2 1/2 D Visual Servoing ». *IEEE Trans. on Robotics and Automation*, 15(2) :234–246.

Malis, E., Chaumette, F., and Boudet, S. (2000). « 2 1/2 D Visual Servoing with Respect to Unknown Objects Through a New Estimation Scheme of Camera Displacement ». *International Journal of Computer Vision*, 37(1) :79–97.

Malis, E. and Rives, P. (2003). « Robustness of Image-Based Visual Servoing with Respect to Depth Distribution Errors ». In *IEEE Int. Conf. on Robotics and Automation*, pages 1056–1061.

Marchand, E. (1999). « ViSP : A Software Environment for Eye-in-Hand Visual Servoing ». In *IEEE Int. Conf. on Robotics and Automation*, pages 3224–3229.

Marchand, E., Bouthemy, P., and Chaumette, F. (2001a). « A 2D-3D model-based approach to real-time visual tracking ». *Image and Vision Computing*, 19(13) :941–955.

Marchand, E., Bouthemy, P., and Chaumette, F. (2001b). « A 2D-3D model-based approach to real-time visual tracking ». *Image and Vision Computing*, 19(13) :941–955.

Martinet, P., Daucher, N., Gallice, J., and Dhome, M. (1997). « Robot control using monocular pose estimation ». In *Workshop on New Trends In Image-Based Robot Servoing (IROS'97)*, pages 1–12.

Masson, L., Dhome, M., and Jurie, F. (2004a). « Robust Real Time Tracking of 3D Objects ». In *Int. Conf. on Pattern Recognition*, pages 252–255.

Masson, L., Dhome, M., and Jurie, F. (2004b). « Suivi de motifs texturés : approche hybride texture/contours ». In *14ème Congrès Francophone AFRIF-AFIA de Reconnaissance des Formes et Intelligence Artificielle*.

Matthews, I., Ishikawa, T., and Baker, S. (2003). « The Template Update Problem ». In *Proceedings of the British Machine Vision Conference*.

Mei, C., Benhimane, S., Malis, E., and Rives, P. (2006a). « Homography-based Tracking for Central Catadioptric Cameras ». In *IEEE Int. Conf. on Intelligent Robots and Systems*.

Mei, C., Malis, E., Benhimane, S., and Rives, P. (2006b). « Constrained Multiple Planar Template Tracking for Central Catadioptric Cameras ». In *British Machine Vision Conference*.

Mezouar, Y. and Chaumette, F. (2002). « Path planning for robust image-based control ». *IEEE Trans. on Robotics and Automation*, 18(4) :534–549.

Mezouar, Y., Haj-Abdelkader, H., Martinet, P., and Chaumette, F. (2004). « Central catadioptric visual servoing from 3D straight lines. ». In *IEEE/RSJ Int. Conf. on Intelligent Robots Systems*, pages 343–349.

Pressigout, M. and Marchand, E. (2005). « Real-time planar structure tracking for visual servoing : a contour and texture approach ». In *IEEE/RSJ Int. Conf. on Intelligent Robots Systems*.

Romdhani, S. and Vetter, S. (2003). « Efficient, Robust and Accurate Fitting of a 3D Morphable Model ». In *IEEE Int. Conf. on Computer Vision*, pages 59–66.

Samson, C., Le Borgne, M., and Espiau, B. (1991). *Robot Control : the Task Function Approach*. Oxford Engineering Science Series. Clarendon Press, Oxford, UK.

Schmid, C., Mohr, R., and Bauckhage, C. (2000). « Evaluation of Interest Point Detectors ». *International Journal of Computer Vision*, 37(2) :151–172.

Sclaroff, S. and Isidoro, J. (1998). « Active blobs ». In *IEEE Int. Conf. on Computer Vision*, pages 1146–1153.

Sepp, W. (2005). « A Direct Method for Real-Time Tracking in 3-D Under Variable Illumination ». In *DAGM Symposium Pattern Recognition*, Lecture Notes in Computer Science, pages 246–253. Springer-Verlag.

Sepp, W. and Hirzinger, G. (2003). « Real-Time Texture-Based 3-D Tracking. ». In *DAGM Symposium*, volume 2781 of *Lecture Notes in Computer Science*, pages 330–337. Springer-Verlag.

Shi, J. and Tomasi, C. (1994). « Good Features to Track ». In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 593–600.

Shum, H. and Szeliski, R. (2000). « Construction of panoramic image mosaics with global and local alignment ». *IEEE Journal of Computer Vision*, 16(1) :63–84.

Smith, S. and Brady, J. (1997). « SUSAN - A New Approach to Low Level Image Processing ». *International Journal of Computer Vision*, 23(1) :45–78.

Svoboda, T. and Pajdla, T. (2002). « Epipolar geometry for central catadioptric cameras ». *International Journal of Computer Vision*, 49(1) :23–37.

Tahri, O. and Chaumette, F. (2005). « Point-based and region-based image moments for visual servoing of planar objects ». *IEEE Trans. on Robotics*, 21(6) :1116–1127.

Taylor, C., Ostrowski, J., and Jung, S. (2000). « Robust Vision-based Pose Control ». In *IEEE Int. Conf. on Robotics and Automation*, pages 2734–2740.

Vacchetti, L., Lepetit, V., and Fua, P. (2004). « Combining Edge and Texture Information for Real-Time Accurate 3D Camera Tracking ». In *3rd IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 48–57.

Vargas, M. and Malis, E. (2005). « Visual servoing based on an analytical homography decomposition ». In *Joint 44th IEEE Conference on Decision and Control and European Control Conference, Seville, Spain.*

Wilson, W. J., Hulls, C. C. W., and Bell, G. S. (1996). « Relative End-Effector Control Using Cartesian Position-Based Visual Servoing ». *IEEE Trans. on Robotics and Automation*, 12(5) :684–696.

Xiao, J., Baker, S., Matthews, I., and Kanade, T. (2004). « Real-Time Combined 2D+3D Active Appearance Models ». In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 535–542.

Résumé

Mis à part les domaines traditionnels de la manipulation et de la saisie robotisées, la commande référencée vision offre aujourd'hui des applications dans des contextes variés qui englobent la vision par ordinateur, l'automatique et la théorie de la commande. Afin de concevoir des méthodes de commande référencée vision, il est possible d'intégrer des techniques de vision et de commande qui ont été conçues séparément.

Au lieu de considérer séparément les techniques de vision et de commande, dans cette thèse, nous les avons intégrées dans une approche unifiée. Nous avons conçu un système générique, flexible et robuste qui peut être utilisé pour une grande variété d'applications robotiques. Au cours de cette thèse, nous avons contribué dans divers domaines allant dans le sens de la conception d'un système complet. Deux contributions majeures sont présentées :

- Une approche de suivi visuel d'objets plans grâce à leurs textures dans l'image en utilisant une minimisation efficace au second-ordre appelée la méthode ESM. Le suivi visuel obtenu a des propriétés de convergence meilleures que celles des méthodes de suivi visuel existantes (domaine, taux et fréquence de convergence). Cette approche a été généralisée à l'estimation du mouvement directement dans l'espace Cartésien.

- Une nouvelle commande par asservissement visuel 2D est introduite. Cette commande est stable localement et, contrairement à toutes les méthodes existantes, elle ne nécessite pas de mesure du modèle de l'objet par rapport auquel la commande est effectuée. Seules des informations issues des images de référence et courante suffisent pour calculer la loi de commande.

Mots clés : Suivi temps-réel, asservissement visuel, commande référencée vision, vision par ordinateur, théorie de la commande, géométrie projective, optimisation, traitement d images.