N° d'ordre : 4248 ANNÉE 2010

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Traitement du signal

École doctorale Matisse

présentée par

Céline Teulière

préparée à l'unité mixte de recherche 6074 IRISA
Institut de Recherche en Informatique et Systèmes Aléatoires

Approches déterministes et bayésiennes pour un suivi robuste : application à l'asservissement visuel d'un drone.

Thèse soutenue à Rennes
le 15 décembre 2010

devant le jury composé de :

Laurent ECK
Ingénieur de recherche au CEA LIST / examinateur

Frédéric LERASLE
Professeur à l'Université Paul Sabatier / rapporteur

Éric MARCHAND
Professeur à l'Université de Rennes 1 / directeur de thèse

Youcef MEZOUAR
Maître de conférence à l'Université Blaise Pascal/ rapporteur

Patrick PÉREZ
Directeur de recherche Technicolor / examinateur

Patrick RIVES
Directeur de recherche à l'INRIA / examinateur

Remerciements

Je voudrais tout d'abord remercier tous les membres du jury pour avoir porté un regard attentif sur ce travail, pour le temps qu'ils ont bien voulu y consacrer, et pour avoir rejoint Rennes depuis trois ou quatre coins de France à l'occasion de la soutenance de cette thèse. Merci à Patrick Pérez d'avoir présidé ce jury. Je remercie les rapporteurs, Frédéric Lerasle et Youcef Mezouar, pour leur lecture et leurs commentaires avisés sur ce document. Merci également à Patrick Rives pour ses réflexions et son enthousiasme.

Pour me guider et me soutenir pendant ces années de thèse, j'ai eu la chance de pouvoir toujours compter sur mes deux encadrants. Je leur dois un grand merci. Merci Éric pour la confiance que tu m'as accordée, pour tes conseils et ta disponibilité dans toutes les étapes clés de cette thèse. Merci Laurent pour ton calme et ta grande réactivité à chacune de mes sollicitations.

Cette thèse s'est déroulée dans le cadre d'une collaboration entre l'équipe Lagadic de l'INRIA Rennes et celle du LTC (puis LISA) du CEA LIST à Fontenay-aux-Roses. Je tiens à remercier leurs responsables, François Chaumette et Yvan Measson, pour m'avoir accueillie dans leurs équipes respectives, m'offrant à la fois d'excellentes conditions matérielles et une grande liberté pour mener à bien mes recherches.

Merci à Veronica Teichrieb, et à travers elle à toute l'équipe du Virtual Reality and Multimedia Research Group, pour son accueil chaleureux lors de mon trop court séjour à Recife.

Je tiens également à remercier vivement tous ceux dont le travail permet de simplifier grandement la mise en place d'applications robotiques. Merci en particulier à Fabien, Éric, Anthony, Nicolas et Romain pour leur effort de maintien et de développement de la bibliothèque ViSP, et pour leur indulgence face à mes questions.

Je remercie aussi tous ceux qui ont bravé le froid des sous-sols du CEA pour m'assister pendant mes expériences sur le drone. Merci à Alexandre pour ses talents de pilote de Formule 1 et à Mariette et Bruno pour avoir immortalisé ces moments. Et comme parfois la gravité fut la plus forte, je dois un merci tout particulier à Laurent, Nicolas et Alexandre pour leur patience et leur savoir-faire lors des réparations, montages et démontages qui ont suivi.

Merci encore à tous les relecteurs officieux de ce document : Caroline, Nicolas, Olivier, Jérôme, mes parents, et surtout mon frère qui a relu avec attention l'ensemble du manuscrit en un temps record.

Pour toutes ces petites choses qui font que j'ai eu plaisir à les retrouver chaque matin, je voudrais également remercier ici les membres de l'équipe Lagadic et du CEA LIST qui contribuent par leur présence à l'atmosphère agréable qui y règne. Merci en particulier à mes compagnons du bureau 17, Bruno, Julie, Xavier, Greg, Alexandre pour les bons moments passés, en espérant qu'il y en ait encore beaucoup

dans le futur. Merci aux locataires de la mezzanine pour les discussions surréalistes autour d'une boisson chaude. Merci aux non francophones de m'avoir fait pratiquer mon anglais. Merci Caroline pour M. Darcy et les crêpes beurre/sucre, merci Olivier d'avoir vu des ours, merci Nicolas pour les SEGFAULTS et les blagues, merci Alexandre pour les histoires racontées avec passion, merci Andrea pour ta gentillesse, merci à tous les amateurs occasionnels ou réguliers des jeudis soirs. Et bien sûr, un remerciement spécial à Amaury : merci pour ton optimisme à toute épreuve, qui fait du bien dans les mauvais jours, merci pour les coups de main, la musique, la jungle, El Christo et les strokes.

Merci aux organisateurs d'ICRA d'avoir choisi de si belles destinations pour les conférences de 2009 et 2010.

Pour l'évasion bienfaitrice qu'ils m'ont procurée je remercie aussi tous les danseurs rencontrés à Paris ou à Rennes sur un air de swing.

Un grand merci également à mes amis de m'avoir sortie de ma thèse de temps en temps.

Je remercie affectueusement ma famille, en particulier mon cousin qui a fait le déplacement pour ma soutenance, mon frère pour son aide et ses interrogations qui ont souvent alimenté mes réflexions, et mes parents pour la curiosité qu'ils ont su me transmettre et leur soutien constant.

Enfin, merci Jérôme de m'avoir supportée, dans tous les sens du terme, tout au long de cette thèse.

Table des matières

Notations

Règles générales

x minuscule	:	scalaire
$\mathbf{x}$ minuscule gras	:	vecteur
$\mathbf{X}$ majuscule gras	:	matrice
$\widehat{X}$	:	valeur estimée de la variable X

Cas particulier : le vecteur des coordonnées d'une primitive visuelle 3D est noté en majuscule pour le différencier de sa projection 2D notée en minuscule.

Algèbre

$\mathbb{R}^*$	:	espace des réels non nuls
$\mathbb{R}^n$	:	espace des réels de dimension n
$\mathbb{E}^n$	:	espace Euclidien de dimension n
$\mathbb{P}^n$	:	espace projectif de dimension n
$SO(3)$	:	groupe spécial orthogonal des matrices de rotations
$SE(3)$	:	groupe spécial Euclidien des matrices homogènes
$so(3)$	:	algèbre spéciale orthogonale
$se(3)$	:	algèbre spéciale Euclidienne
$[\mathbf{A}_1\ \mathbf{A}_2\ \ldots\ \mathbf{A}_n]$	:	concaténation horizontale de matrices, vecteurs ou scalaires
$(\mathbf{A}_1,\ \mathbf{A}_1,\ \ldots,\ \mathbf{A}_n)$	:	concaténation verticale de matrices, vecteurs ou scalaires
$\mathbf{A}_{i,j}$	:	accès à l'élément de la i-ème ligne et j-ème colonne de la matrice $\mathbf{A}$
$\mathbf{A}^\top$	:	matrice transposée de $\mathbf{A}$
$\mathbf{A}^{-1}$	:	matrice inverse de $\mathbf{A}$
$\mathbf{A}^+$	:	matrice pseudo-inverse de $\mathbf{A}$
$det(\mathbf{A})$	:	déterminant de la matrice $\mathbf{A}$
$diag(\mathbf{a})$	:	matrice de diagonale $\mathbf{a}$
$[\mathbf{v}]_\times$	:	matrice de préproduit vectoriel associée au vecteur $\mathbf{v}$
$\mathbf{0}_{n\times m}$	:	matrice nulle de dimension $n \times m$
$\mathbf{I}_n$	:	matrice identité de dimension $n \times n$

Géométrie

$\mathcal{P}$	:	point de l'espace Euclidien
${}^c\mathbf{P} = ({}^cX,\ {}^cY,\ {}^cZ)$	:	vecteur de coordonnées de $\mathcal{P}$ dans le repère $\mathcal{R}_c$
$\mathcal{R}_c$	:	repère cartésien orthonormé associé à c
$\mathcal{F}$	:	espace des informations visuelles
$\mathcal{E}$	:	espace des paramètres d'état

Statistiques

$p(x\|z)$	:	densité de probabilité de x sachant z

Opérateurs

$\propto$	:	opérateur d'égalité à un facteur d'échelle près
$\approx$	:	opérateur d'approximation

La vision statique n'existe pas ; nul ne peut voir sans explorer.

Arthur Koestler

Introduction

Les systèmes mécatroniques (mêlant notamment mécanique, électronique, automatique et informatique) occupent aujourd'hui une place importante dans nos sociétés. Que la motivation première de leur développement soit scientifique, économique ou humaniste, ces systèmes se sont imposés notamment dans la réalisation de tâches répétitives, difficiles ou dangereuses pour l'homme ou demandant une grande précision et/ou une vitesse d'exécution élevée. Depuis les premiers robots industriels utilisés dans la peinture de carrosseries en atmosphère toxique jusqu'à des robots mobiles pionniers de l'exploration sous-marine ou spatiale, en passant par des bras manipulateurs téléopérés, reproduisant les mouvements qu'un opérateur humain exerce sur un bras maître plus facile à manipuler, une grande variété de systèmes ont ainsi été conçus.

Parmi toutes les caractéristiques qui les différencient, l'une des notions les plus essentielles est leur degré d'*autonomie*. L'autonomie d'un système est sa capacité à « se débrouiller » tout seul, c'est-à-dire agir ou réagir sans l'aide d'un opérateur humain, et ce avec le moins de connaissance *a priori* que possible sur l'environnement dans lequel il évolue. C'est cette autonomie qui différencie le simple automate du robot dit « intelligent ». Un système téléopéré ne faisant que reproduire des consignes en provenance d'un opérateur humain ne dispose à ce titre d'aucune autonomie ; un bras manipulateur industriel, conçu pour répéter une tâche bien précise dans un environnement parfaitement maîtrisé, sera incapable de s'adapter automatiquement à un changement imprévu de cet environnement.
À l'inverse, certains systèmes sont capables de prendre en compte leur environnement dans les tâches qu'ils exécutent. C'est le cas par exemple de robots aspirateurs qui construisent par eux-mêmes une carte de l'appartement dans lequel ils sont placés, mais aussi de systèmes plus évolués tels que la voiture autonome *Junior* développée par l'université de Stanford en réponse au *Darpa Challenge* ou encore au robot *Spirit* du *Jet Propulsion Laboratory* (JPL), qui a exploré la surface martienne pendant cinq ans. La question de l'autonomie est un enjeu particulièrement important en robotique mobile où l'environnement peut varier au cours du déplacement. En particulier, les véhicules terrestres ou aériens, traditionnellement pilotés par un humain sont progressivement robotisés avec pour objectif la réalisation de certaines fonctions de manière autonome : localisation, navigation, détection et évitement d'obstacles par exemple.

L'autonomie résulte de la capacité de percevoir, et d'utiliser « intelligemment »

les informations perçues pour agir ou réagir de manière adaptée afin d'accomplir une tâche ou une mission donnée. Derrière la question de l'autonomie se trouve donc celle de la perception. Pour être capable de s'adapter à son environnement, un robot doit avant tout être capable de percevoir cet environnement. À l'image des cinq sens dont l'homme est doté, un robot devra donc être équipé d'un ou plusieurs capteur(s) lui fournissant des informations sur ce qui l'entoure (capteurs extérioceptifs tels les télémètres, les caméras, etc.) ou sur lui-même (capteurs proprioceptifs, tels les capteurs gyroscopiques ou les accéléromètres). Parmi ces différents sens, la vision est une source d'information particulièrement riche, passive (contrairement aux télémètres elle ne nécessite pas d'action sur l'environnement par l'envoi d'un signal), et relative à l'environnement (à l'inverse du GPS par exemple, qui fournit une position absolue sans indication sur l'environnement).

La vision s'est ainsi imposée comme un outil majeur dans la quête d'une plus grande autonomie. Dans le cas des véhicules terrestres, des systèmes de navigation autonome basés vision font l'objet de nombreuses études. Dans cette thèse nous considèrerons un autre champ applicatif, celui des mini-drones volants. Ces dernières années, le développement de mini-drones à voilure tournante capables de vol stationnaire a suscité un intérêt grandissant [1] pour un large champ d'applications dans lesquelles un retour visuel est souvent nécessaire : surveillance de feux de forêt, d'ouvrages d'art, du trafic routier, inspection de bâtiments, recherche de victimes dans des zones difficiles d'accès ou dangereuses (catastrophes naturelles, zones de combat,...). Il est dès lors naturel d'envisager les possibilités offertes par les informations visuelles acquises en vue d'accomplir des tâches de manière autonome sur ce type d'appareils.

À l'heure actuelle les mini-drones sont majoritairement utilisés par des pilotes spécialement formés pour téléopérer ce type d'appareils. L'objet de cette thèse est de franchir un pas vers une plus grande autonomie de tels engins en étudiant la possibilité de réaliser certaines tâches de manière automatique à partir des informations fournies par un capteur de vision. Concevoir des fonctions autonomes en utilisant des informations visuelles pose un certain nombre de défis scientifiques et techniques. Tout d'abord il s'agit de pouvoir extraire les informations pertinentes des images acquises. Dans le contexte des mini-drones, il faut pour cela faire face à des contraintes particulièrement fortes : déplacements rapides et importants, images parfois de mauvaise qualité, occultations possibles et nécessité de gérer les échecs potentiels, etc. Une fois ces informations extraites, il faut alors les utiliser pour accomplir au mieux la tâche voulue : c'est l'objet de l'asservissement visuel.

Dans cette thèse nous nous intéresserons à ces deux aspects, et nous considèrerons deux types de tâches :

- une tâche de poursuite, dans laquelle on demande à un drone de suivre une cible mobile dans un environnement inconnu ;
- une tâche de positionnement et navigation par points de passage dans un environnement structuré dont un modèle est disponible.

Pour ces deux types de situations, des algorithmes de suivi robuste sont proposés et

1. Les études présentées à l'occasion du salon *UAV show Europe* organisé en septembre dernier à Bordeaux, annonçaient que ce secteur d'activité représentaient la plus forte progression du secteur de l'industrie aéronautique et spatiale sur ces dix dernières années.

utilisés pour la commande d'un drone. Des résultats expérimentaux sur un drone de type quadrirotor viennent valider ces approches.

Cette thèse est née d'une collaboration entre le CEA LIST qui développe un drone quadrirotor depuis 2003, et l'équipe Lagadic de l'IRISA dont les principaux axes de recherches concernent la vision robotique et l'asservissement visuel.

Organisation du manuscrit et contributions

La suite de ce manuscrit s'organise de la façon suivante. Dans un premier temps, le chapitre 1 présente les notions fondamentales de vision par ordinateur ainsi que les outils élémentaires de suivi visuel nécessaires à la compréhension de nos travaux. Le chapitre 2 dresse un rapide panorama de l'état de l'art en termes de commande de drone à l'aide d'informations visuelles.

La suite du document s'articule alors autour de deux parties, correspondant à chacune des deux tâches que nous avons considérées : la partie I se focalise sur une tâche de poursuite. Dans ce cadre, le chapitre 3 propose un algorithme de suivi robuste basé sur une représentation à noyaux multiples. Ce suivi permet d'estimer la position d'un objet mobile dans une séquence d'images. Les aspects de détection et réinitialisation automatique en cas d'échec du suivi sont également abordés. Le chapitre 4 montre comment les informations obtenues peuvent être utilisées pour l'asservissement d'un mini-drone. Nous y présentons le modèle du drone quadrirotor utilisé, ainsi que les lois de commande mises en place et les résultats expérimentaux obtenus dans le cadre de nos travaux.

Dans la partie II nous considérons une tâche de positionnement ou navigation. En environnement urbain encombré ou à l'intérieur de bâtiments, les données de position classiquement obtenues par GPS ne sont généralement pas disponibles. La vision peut alors être utilisée pour estimer la position du drone par rapport à son environnement. Nous proposons ainsi dans le chapitre 5 des algorithmes de calcul de pose robustes destinés à ce type de contexte, en utilisant un modèle simplifié de l'environnement. Le chapitre 6 décrit alors l'utilisation de l'information de pose pour asservir en position un mini-drone. La précision de la localisation obtenue est évaluée, et des expériences de navigation par points de passage viennent valider l'approche proposée.

Publications

Les travaux présentés dans ce manuscrit ont fait l'objet de publications que nous référençons ci-après.

Conférences nationales

[CN1] **C. Teulière**. – Approche multi-noyaux pour le suivi temps-réel d'objet mobile. – in *Congrès de jeunes chercheurs en vision par ordinateur, ORASIS 2009*, Trégastel, France, mai 2009.

Conférences internationales

[C1] **C. Teulière**, E. Marchand, L. Eck. – A combination of particle filtering and deterministic approaches for multiple kernel tracking. – in *IEEE Int. Conf. on Robotics and Automation, ICRA'09*, Kobe, Japan, May 2009.

[C2] **C. Teulière**, E. Marchand, L. Eck. – Using multiple hypothesis in model-based tracking. – in *IEEE Int. Conf. on Robotics and Automation, ICRA'10*, Anchorage, Alaska, May 2010.

[C3] **C. Teulière**, L. Eck, E. Marchand, N. Guénard. – 3D model-based tracking for UAV position control. – in *IEEE Int. Conf. on Robotics and Automation, IROS'10*, Taipei, Taiwan, October 2010.

Chapitre 1

Fondamentaux de vision par ordinateur

Ce chapitre présente les notions fondamentales de vision par ordinateur sur lesquelles se basent les travaux décrits dans ce mémoire. La section 1.1 décrit le modèle de formation des images numériques et la géométrie de la prise de vue. La section 1.2 décrit les transformations rigides de l'espace, ainsi que les transformations projectives du plan image. Enfin, les notions fondamentales du suivi visuel sont présentées section 1.3.

1.1 Géométrie de la prise de vue

Les caméras, au même titre que les appareils photographiques, sont composées d'un système optique, constitué de lentilles et de diaphragmes, d'un boîtier et d'une surface sensible à la lumière. Dans le cas des caméras numériques, la surface sensible est formée d'un pavage plus ou moins dense de capteurs élémentaires, sensibles aux photons [Slama 80]. Ce paragraphe décrit le modèle géométrique de projection perspective de formation des images numériques qui sera utilisé dans le cadre de cette thèse.

1.1.1 Modèle de projection

Il existe plusieurs façons de modéliser la formation des images numériques. Le modèle *sténopé* ou *pin-hole* est le plus couramment utilisé en vision par ordinateur car il constitue une bonne approximation du modèle physique de la caméra et il permet d'établir une relation simple entre les coordonnées 3D des éléments d'une scène et leur projection dans l'image en 2D.

Géométriquement le modèle sténopé repose sur une projection centrale ou perspective [Faugeras 93] [Hartley 01]. Selon ce modèle, tous les rayons lumineux émis

ou réfléchis depuis la scène et captés par la caméra passent par un unique point $\mathcal{C}$, le *centre de projection* ou *centre optique*.

Pour chaque point $\mathcal{P}$ de la scène on considère ainsi la droite passant par $\mathcal{P}$ et par le centre optique $\mathcal{C}$ comme étant le trajet de la lumière perçue par la caméra en provenance de $\mathcal{P}$. L'intersection de cette droite avec le plan de projection ou *plan image* est la projection $\mathbf{p}$ du point $\mathcal{P}$. Ce modèle de projection est illustré Figure 1.1.

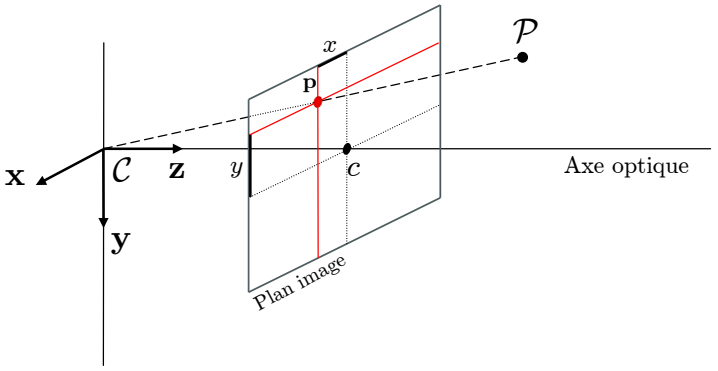


FIGURE 1.1 – *Le modèle sténopé.*

Le modèle sténopé associe à la caméra un repère cartésien $\mathcal{R}_c = (\mathcal{C},\ \mathbf{x},\ \mathbf{y},\ \mathbf{z})$ dont l'origine coïncide avec le centre de projection. Le plan image est parallèle aux axes $\mathbf{x}$ et $\mathbf{y}$. Il est situé à une distance f de l'origine $\mathcal{C}$, appelée *distance focale* de la caméra. L'*axe optique*, porté par $\mathbf{z}$, coupe le plan image en un point $\mathbf{c}$ appelé *point principal*.

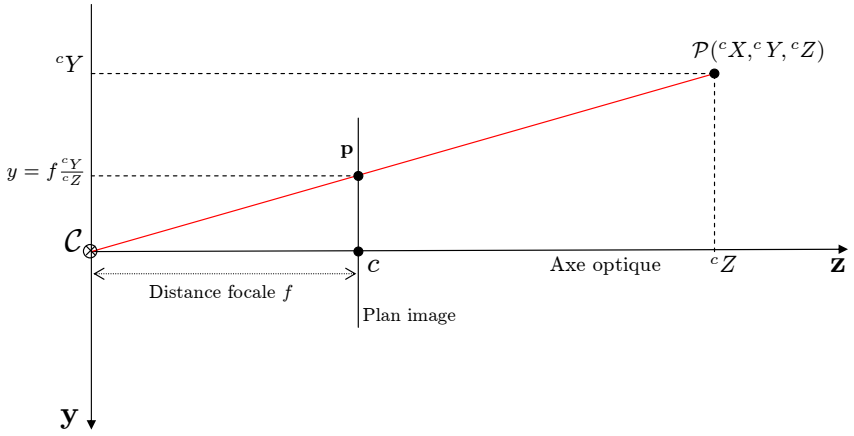


FIGURE 1.2 – *Le modèle projectif du sténopé.*

Un point $\mathcal{P}$ de l'espace, de coordonnées ${}^c\mathbf{P} = ({}^cX,\ {}^cY,\ {}^cZ)$ dans le repère $\mathcal{R}_c$, se projette en un point $\mathbf{p}$ de coordonnées $(x,\ y)$ dans le plan image, avec :

$$x = f\frac{{}^cX}{{}^cZ},\ y = f\frac{{}^cY}{{}^cZ}. \tag{1.1}$$

Espace projectif et coordonnées homogènes. La relation (1.1) lie de manière non linéaire les coordonnées d'un point de l'espace dans $\mathbb{R}^3$ avec les coordonnées de sa projection dans le plan image. Pour une direction donnée, quelle que soit la distance entre le point $\mathcal{P}$ et le centre de projection, sa projection $\mathbf{p}$ dans le plan image reste la même. Autrement dit, quel que soit $\lambda \in \mathbb{R}^*$, le point de coordonnées $\lambda^c\mathbf{P}$ donne la même projection $\mathbf{p}$ dans le plan image. L'ensemble des points de coordonnées $\lambda^c\mathbf{P} \in \mathbb{R}^3$ correspond ainsi à un unique point $\mathcal{P}'$ de l'espace projectif $\mathbb{P}^3$ dont les coordonnées homogènes s'écrivent $(\lambda^c X,\ \lambda^c Y,\ \lambda^c Z,\ \lambda)$, ou de façon normalisée $({}^cX,\ {}^cY,\ {}^cZ,\ 1)$.

L'utilisation des coordonnées homogènes permet de réécrire la projection perspective sous forme linéaire, comme une transformation de l'espace projectif $\mathbb{P}^3$ vers l'espace projectif $\mathbb{P}^2$:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{pmatrix} \tag{1.2}$$

où $({}^cX,\ {}^cY,\ {}^cZ,\ 1)$ (resp. $(x,\ y,\ 1)$) sont les coordonnées homogènes normalisées du point de l'espace projectif $\mathbb{P}^3$ (resp. $\mathbb{P}^2$) correspondant au point $\mathcal{P}$ (resp. $\mathbf{p}$).

1.1.2 Image numérique

L'image numérique est une discrétisation du plan image en un pavage généralement rectangulaire d'éléments élémentaires appelés pixels. Les pixels sont caractérisés par leur position sur le pavage et leur couleur, qui correspond à l'intensité lumineuse reçue dans les trois composantes principales (rouge, vert et bleu). Les coordonnées pixelliques définissant la position d'un pixel sur ce pavage sont exprimées par des valeurs entières, et leur origine est généralement prise en haut à gauche de l'image et non au point principal (voir Figure 1.3). Ainsi, la transformation permet-

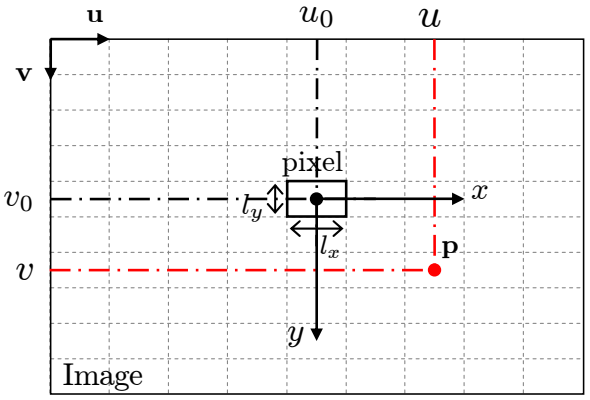


FIGURE 1.3 – *Coordonnées pixelliques.*

tant de passer des coordonnées métriques (x, y) d'un point $\mathbf{p}$ du plan image à ses

coordonnées pixelliques (u, v) dépend des dimensions (l_x, l_y) d'un pixel ainsi que des coordonnées pixelliques (u_0, v_0) du point principal dans l'image :

$$u = u_0 + \frac{x}{l_x} \tag{1.3}$$

$$v = v_0 + \frac{y}{l_y} \tag{1.4}$$

Cette relation peut s'écrire dans une représentation homogène sous la forme :

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{l_x} & 0 & u_0 \\ 0 & \frac{1}{l_y} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{1.5}$$

Ainsi, d'après les relations (1.2) et (1.5), un point $\mathcal{P}$ de l'espace, de coordonnées métriques ${}^c\mathbf{P} = ({}^cX,\ {}^cY,\ {}^cZ)$ dans $\mathcal{R}_c$ se projette dans l'image numérique selon la relation suivante :

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{l_x} & 0 & u_0 \\ 0 & \frac{1}{l_y} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{pmatrix} \tag{1.6}$$

$$= \begin{pmatrix} p_x & 0 & u_0 & 0 \\ 0 & p_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} {}^cX \\ {}^cY \\ {}^cZ \\ 1 \end{pmatrix} \tag{1.7}$$

où $p_x = \frac{f}{l_x}$ et $p_y = \frac{f}{l_y}$.

Le modèle sténopé de la caméra dépend donc de quatre paramètres, dits *paramètres intrinsèques* : les coordonnées pixelliques (u_0, v_0) du point principal dans l'image, et les deux paramètres de changement d'échelle p_x et p_y.

1.1.3 Modèle de distorsion

Le modèle de sténopé présenté ci-dessus suppose une caméra parfaite. En réalité, des déformations ou distorsions de l'image peuvent apparaître, dues principalement à l'optique de la caméra. Ces distorsions sont d'autant plus fortes que la distance focale f est petite. Lorsque leur effet est important (voir Figure 1.4) il devient nécessaire de les prendre en compte dans la modélisation de la caméra. La déformation prépondérante est généralement la distorsion radiale, qui peut être modélisée de la façon suivante [Faugeras 93] [Hartley 01] :

$$x = x_d(1 + k_1 r^2 + k_2 r^4) \tag{1.8}$$

$$y = y_d(1 + k_1 r^2 + k_2 r^4) \tag{1.9}$$

Cette équation relie les coordonnées (x, y) d'un point du plan image obtenues par un modèle de projection perspective (1.1), avec les coordonnées (x_d, y_d) du point

correspondant obtenues en prenant en compte la distorsion radiale. $r^2 = x^2 + y^2$, et k_1 et k_2 sont les paramètres de cette distorsion. L'effet de la distorsion radiale est d'autant plus grand que la distance radiale r, c'est-à-dire la distance entre le point considéré et le centre de l'image, augmente. Un phénomène de distorsion dite tangentielle peut également apparaître mais son effet étant infime pour la grande majorité des objectifs, on le négligera ici.


(a)

(b)

FIGURE *1.4 – Exemple de vue avec distorsion (a) et après correction de la distorsion (b).*

Les paramètres intrinsèques de la caméra sont alors $(u_0, v_0, p_x, p_y, k_1, k_2)$. Différentes méthodes d'étalonnage sont disponibles pour estimer ces différents paramètres [Brown 71] [Faugeras 87] [Tsai 86] [Stein 97].

En résumé, le passage des coordonnées 3D d'un point $\mathcal{P}$ de la scène, vers les coordonnées pixelliques de sa projection dans l'image résulte des transformations suivantes :

$$({}^cX, {}^cY, {}^cZ) \overset{Projection}{\longrightarrow} (x, y) \overset{Distorsion}{\longrightarrow} (x_d, y_d) \overset{Pixellisation}{\longrightarrow} (u, v)$$

Dans la suite de cette thèse, les paramètres intrinsèques des caméras utilisées sont supposés connus.

1.2 Transformations de l'espace, transformations du plan

Dans la section précédente nous avons décrit le lien entre les coordonnées d'un point d'une scène 3D, exprimées dans le repère de la caméra, et les coordonnées de sa projection dans l'image, grâce au modèle de projection perspective.

Lorsque la position relative entre la caméra et la scène change, la projection de la scène dans l'image subit également une transformation. L'enjeu d'un algorithme de suivi consiste à identifier cette transformation dans l'image (suivi 2D) ou à utiliser des informations sur la transformation ayant lieu dans l'image pour reconstituer la position relative entre la caméra et la scène ou l'objet suivi (suivi 3D). Cette section présente les notions géométriques nécessaires à la paramétrisation des transformations 2D et 3D.

1.2.1 Changement de repère

La transformation qui permet de passer d'un référentiel orthonormé de référence $\mathcal{R}_b$ à un référentiel $\mathcal{R}_a$ est une transformation rigide, caractérisée de façon unique par une rotation et une translation de l'espace $\mathbb{R}^3$. Soit $\mathcal{P}$ un point de l'espace, et ${}^a\mathbf{P} = ({}^aX,\ {}^aY,\ {}^aZ) \in \mathbb{R}^3$, resp. ${}^b\mathbf{P} = ({}^bX,\ {}^bY,\ {}^bZ) \in \mathbb{R}^3$ le vecteur contenant ses coordonnées exprimées dans le repère $\mathcal{R}_a$, resp. $\mathcal{R}_b$. La relation de changement de repère est donnée par :

$$ {}^a\mathbf{P} = {}^a\mathbf{R}_b\,{}^b\mathbf{P} + {}^a\mathbf{t}_b \tag{1.10} $$

où ${}^a\mathbf{t}_b$ est la translation qui permet de passer de l'origine de $\mathcal{R}_a$ à l'origine de $\mathcal{R}_b$, et ${}^a\mathbf{R}_b$ est la matrice de rotation du repère $\mathcal{R}_b$ dans le repère $\mathcal{R}_a$.

Cette relation peut s'écrire sous forme linéaire en utilisant les coordonnées homogènes :

$$ \begin{pmatrix} {}^aX \\ {}^aY \\ {}^aZ \\ 1 \end{pmatrix} = \begin{bmatrix} {}^a\mathbf{R}_b & {}^a\mathbf{t}_b \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \begin{pmatrix} {}^bX \\ {}^bY \\ {}^bZ \\ 1 \end{pmatrix} \tag{1.11} $$

La matrice ${}^a\mathbf{M}_b = \begin{bmatrix} {}^a\mathbf{R}_b & {}^a\mathbf{t}_b \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}$ représente une transformation homogène.

1.2.2 Paramétrisation des rotations

Toute rotation de $\mathbb{R}^3$ s'écrit sous la forme d'une matrice orthogonale $\mathbf{R} \in SO(3)$.

$$ SO(3) = \left\{ \mathbf{R} \in \mathbb{R}^{3\times 3} | \mathbf{R}^\top \mathbf{R} = \mathbf{I}_3, det(\mathbf{R}) = 1 \right\} $$

Les neufs éléments de cette matrice ne sont évidemment pas indépendants puisqu'ils doivent vérifier la contrainte $\mathbf{R}^\top\mathbf{R} = \mathbf{I}_3$ où $\mathbf{I}_3$ est la matrice identité de dimension 3×3. Ils ne constituent donc pas une représentation minimale de l'ensemble des rotations. Cette relation impose six contraintes indépendantes sur les neufs paramètres, et l'espace $SO(3)$ est donc de dimension 3. Différentes paramétrisations sont couramment utilisées pour représenter une rotation. Les angles d'Euler correspondant aux angles de rotations selon chacun des trois axes constituent une paramétrisation simple et très répandue. D'usage courant dans le domaine aéronautique ils seront notamment utilisés dans ce mémoire pour caractériser l'orientation d'un drone (voir chapitre 4). Les angles d'Euler présentent cependant des singularités dans certaines configurations. En vision par ordinateur ou en commande, on leur préfère souvent d'autres représentations, telles que les *quaternions* ou les *coordonnées exponentielles canoniques*. Les quaternions constituent une représentation non minimale des rotations sous la forme d'un quadruplet. Dans ce mémoire, nous utiliserons la représentation par les coordonnées exponentielles canoniques, dont la signification géométrique sous forme de vecteur de rotation instantanée est plus intuitive et dont les singularités sont atteintes pour des valeurs de paramètres qu'il est facile d'éviter [Grassia 98]. Ses caractéristiques sont décrites dans le paragraphe suivant.

Représentation exponentielle canonique. Toute rotation peut s'écrire sous la forme d'un vecteur de rotation indiquant l'axe de rotation instantané et dont la norme représente la valeur de l'angle de cette rotation. Ce vecteur est noté $\theta\mathbf{u}$, avec $\mathbf{u} = (u_x,\ u_y,\ u_z)$ le vecteur unitaire de l'axe de rotation et θ l'angle de rotation.

Cette représentation est appelée représentation exponentielle canonique car elle est liée à la représentation matricielle par une relation exponentielle :

$$\mathbf{R} = e^{[\theta\mathbf{u}]_\times} = \sum_{n=0}^{\infty} \frac{[\theta\mathbf{u}]_\times^n}{n!}. \tag{1.12}$$

La notation $[\mathbf{v}]_\times$ d'un vecteur $\mathbf{v}$ désigne la matrice anti-symétrique associée, à savoir :

$$[\mathbf{v}]_\times = \begin{pmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{pmatrix}. \tag{1.13}$$

En pratique, la matrice de rotation $\mathbf{R}$ associée à un vecteur de rotation $\theta\mathbf{u}$ est calculée plus simplement à l'aide de la formule de Rodrigues :

$$\mathbf{R} = \mathbf{I}_3 + (1 - \cos\theta)\ \mathbf{u}\mathbf{u}^\top + \sin\theta\ \ [\mathbf{u}]_\times . \tag{1.14}$$

Inversement, la représentation $\theta\mathbf{u}$ d'une matrice de rotation $\mathbf{R}$ est obtenue à partir des relations :

$$\cos\theta = \frac{1}{2}\ (\ \mathrm{trace}(\mathbf{R}) - 1) \tag{1.15}$$

et

$$\sin\theta\ \ [\mathbf{u}]_\times = \frac{1}{2}(\mathbf{R} - \mathbf{R}^\top). \tag{1.16}$$

D'après (1.15) on obtient :

$$\theta = arccos\left(\frac{\mathbf{R}_{11} + \mathbf{R}_{22} + \mathbf{R}_{33} - 1}{2}\right) \tag{1.17}$$

où $\mathbf{R}_{ij}$ désigne l'élément (i, j) de la matrice $\mathbf{R}$.

En fixant $\theta > 0$, si $\sin\theta \neq 0$, $\theta\mathbf{u}$ est déterminé de manière unique par (1.16) :

$$\theta\mathbf{u} = \frac{1}{2\ sinc(\theta)} \begin{pmatrix} \mathbf{R}_{32} - \mathbf{R}_{23} \\ \mathbf{R}_{13} - \mathbf{R}_{31} \\ \mathbf{R}_{21} - \mathbf{R}_{12} \end{pmatrix}. \tag{1.18}$$

où $sinc$ désigne la fonction sinus cardinal $sinc(\theta) = \frac{\sin(\theta)}{\theta}$. Si θ est un multiple de π, $\mathbf{u}$ est le vecteur propre unitaire de la matrice $\mathbf{R}$ associé à la valeur propre 1.

1.2.3 Paramétrisation des transformations rigides de l'espace

Une transformation rigide étant caractérisée par une rotation et une translation, l'ensemble des transformations rigides s'écrit :

$$SE(3) = \left\{ \mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \mid \mathbf{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3 \right\}.$$

D'après la paramétrisation des rotations présentée ci-dessus, une transformation rigide pourra donc être représentée par un vecteur $\mathbf{r}$ de 6 paramètres : 3 paramètres de translation et 3 paramètres de rotation :

$$\mathbf{r} = (\mathbf{t}, \theta\mathbf{u}). \tag{1.19}$$

1.2.4 Transformations dans l'image

Les transformations rigides entre la caméra et la scène observée engendrent des transformations de la projection de cette scène dans le plan image. Les transformations ayant lieu entre deux images successives I_k et I_{k+1} dépendent à la fois du type de déplacement de la caméra (rotation pure, translation, mouvement complexe...) et de la structure de la scène (en particulier si elle peut être considérée comme planaire ou non). Identifier les paramètres de transformation dans l'image peut fournir des informations sur le déplacement 3D de la caméra.

Il n'existe pas de modèle de transformation prenant en compte tout type de mouvement pour tout type de scène. Selon le type de configuration et les connaissances *a priori* dont on dispose sur le type de déplacement qui peut avoir lieu, un modèle de déplacement 2D simplifié pourra être utilisé pour caractériser une transformation, par exemple afin de prédire la position d'un objet suivi d'une image à l'autre.

Soit $\mathbf{p}_k = (x_k,\ y_k) \in \mathbb{R}^2$ un point d'une image I_k. On note $\mathbf{p}_{k+1} = tr(\mathbf{p}_k)$ sa nouvelle position dans l'image I_{k+1}. La fonction $tr(.)$ désigne le modèle de transformation 2D utilisé. Les modèles les plus courants sont décrits ci-après.

Translation pure. Le modèle le plus simple consiste à représenter le déplacement par une simple translation dans l'image. On a alors $tr(\mathbf{p}_k) = \mathbf{p}_k + \mathbf{t}$, où $\mathbf{t} \in \mathbb{R}^2$ est un vecteur de translation.

Similitude. La similitude permet de prendre en compte quatre paramètres de déplacement, à savoir la translation, la rotation dans le plan et un facteur d'échelle : $tr(\mathbf{p}_k) = s\mathbf{R}\mathbf{p}_k + \mathbf{t}$, où $\mathbf{R}$ est une matrice de rotation de dimension 2 et $s \in \mathbb{R}^*$ un facteur d'échelle.

Transformation affine. Pour prendre en compte les transformations affines, on utilise une relation de la forme : $tr(\mathbf{p}_k) = \mathbf{A}\mathbf{p}_k + \mathbf{t}$, où $\mathbf{A} \in \mathbb{R}^{2\times 2}$.

Transformation projective ou homographie[1]**.** Pour définir la relation d'homographie on suppose que l'image I_k est la projection de points 3D appartenants à un même plan Π. Soit $\mathcal{P}$ le point dont la projection dans le plan de l'image I_k est $\mathbf{p}_k$ (voir Figure 1.5). On note $\mathbf{P}_k = (X_k,\ Y_k,\ Z_k)$ le vecteur des coordonnées de $\mathcal{P}$ dans le repère de la caméra $\mathcal{R}_c^k$ à l'instant de l'acquisition de l'image I_k. Soient ${}^k\mathbf{n}$ la normale au plan Π exprimée dans $\mathcal{R}_c^k$, et d_k la distance du plan Π au centre de projection $\mathbf{c}_k$. L'appartenance de $\mathcal{P}$ au plan Π se traduit par la relation :

$${}^k\mathbf{n}^\top \mathbf{P}_k = d_k. \tag{1.20}$$

1. Le terme *homographie* désigne toute transformation bijective entre deux espaces projectifs. Dans ce mémoire ce terme concernera uniquement les transformations bijectives de $\mathbb{P}^2$ vers $\mathbb{P}^2$.

L'équation de changement de repère (1.10) entre $\mathcal{R}_c^k$ et $\mathcal{R}_c^{k+1}$ devient alors :

$$\mathbf{P}_{k+1} = \left({}^{k+1}\mathbf{R}_k + \frac{{}^{k+1}\mathbf{t}_k {}^k\mathbf{n}^\top}{d_k} \right) \mathbf{P}_k. \tag{1.21}$$

De plus, $\mathbf{p}_k$ et $\mathbf{p}_{k+1}$ étant les projections respectives de $\mathcal{P}$ dans I_k et I_{k+1}, on a également : $\mathbf{p}_k = \lambda_k \mathbf{P}_k$ et $\mathbf{p}_{k+1} = \lambda_{k+1}\mathbf{P}_{k+1}$. On en déduit la relation d'homographie :

$$\begin{aligned} tr(\mathbf{p}_k) &= \mathbf{p}_{k+1} && (1.22)\\ &= \frac{\lambda_{k+1}}{\lambda_k} \left({}^{k+1}\mathbf{R}_k + \frac{{}^{k+1}\mathbf{t}_k {}^k\mathbf{n}^\top}{d_k} \right) \mathbf{p}_k && (1.23)\\ &\propto {}^{k+1}\mathbf{H}_k \mathbf{p}_k && (1.24) \end{aligned}$$

où

$${}^{k+1}\mathbf{H}_k = \left({}^{k+1}\mathbf{R}_k + \frac{{}^{k+1}\mathbf{t}_k {}^k\mathbf{n}^\top}{d_k} \right) \tag{1.25}$$

${}^{k+1}\mathbf{H}_k$ est appelée matrice d'homographie.

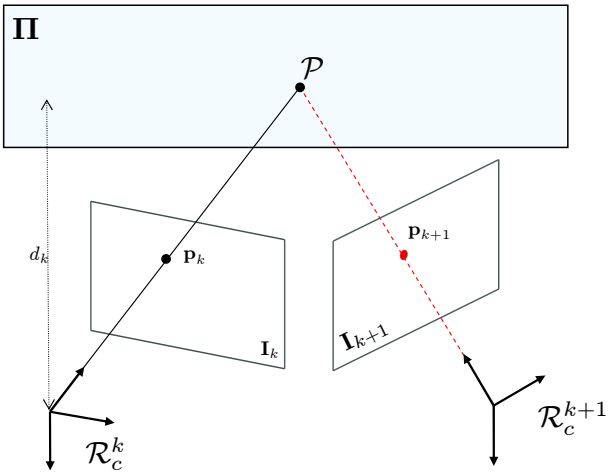


FIGURE *1.5 – Relation d'homographie.*

Les différentes transformations décrites dans ce paragraphe sont illustrées Figure 1.6. Chaque type de modélisation permet de prendre en compte certains types de transformations. Le choix de la modélisation utilisée devra donc être adapté au problème considéré.

1.3 Suivi dans une séquence d'images

Dans la section précédente nous avons décrit le lien géométrique entre le déplacement relatif d'une caméra par rapport à un objet réel, et la transformation de la projection de cet objet dans le plan image. Le problème du suivi à partir d'une unique caméra consiste à utiliser les informations contenues dans une séquence d'images pour identifier la nouvelle position 2D (dans le plan image) ou 3D (dans l'espace)

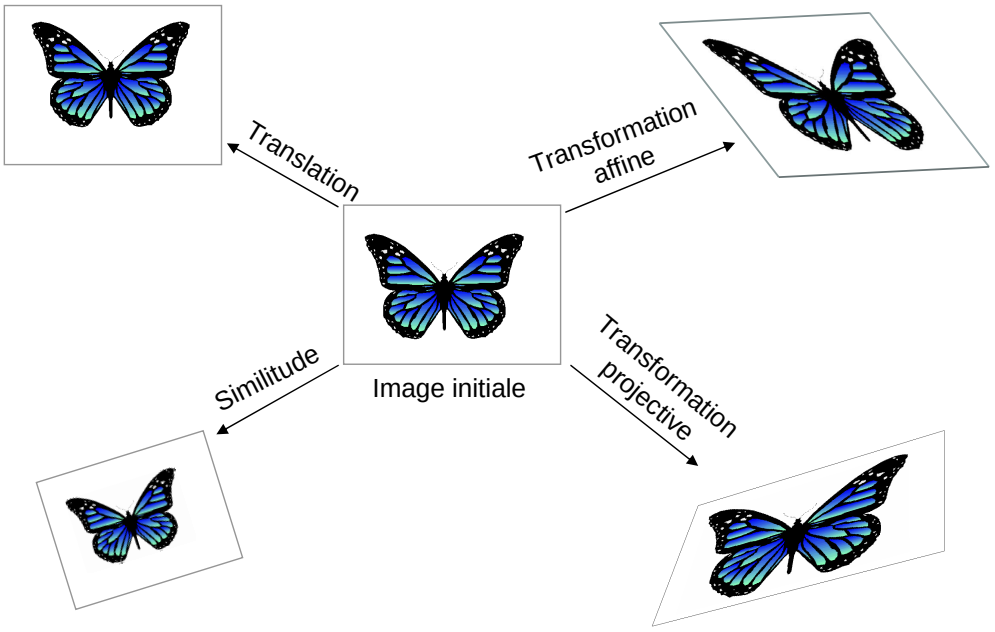


FIGURE *1.6 – Exemples de transformations 2D.*

de l'objet suivi.

Le suivi 2D vise à estimer les paramètres de la transformation 2D d'objets, ou zones d'intérêt dans une séquence d'images. Il sera particulièrement adapté lorsqu'on n'a pas de connaissance *a priori* sur la scène et sur l'objet. Un modèle est construit directement à partir d'une image de l'objet, ou appris à partir d'un ensemble de vues de l'objet, et sert de référence pour le suivi. Aucune connaissance *a priori* sur la scène 3D n'étant nécessaire, l'utilisation de ces approches est peu contraignante. Cependant, l'absence d'information 3D rend plus difficile la gestion de mouvements complexes ou d'occultations, et peut entraîner des dérives.

Le suivi 3D quant à lui vise à identifier les six degrés de liberté qui définissent la position et l'orientation de la caméra par rapport à la scène, ou de manière équivalente le déplacement 3D d'un objet par rapport à la caméra. Ce déplacement 3D peut dans certains cas être déduit des paramètres de transformation 2D, ou estimé directement à partir des primitives visuelles et d'un modèle 3D de l'objet.

Pour réaliser ce suivi un grand nombre d'approches existent, en fonction du type d'objet, des degrés de liberté de la caméra et de l'objet, et bien sûr de l'application considérée. De manière générale, un algorithme de suivi suppose que certains descripteurs de l'apparence de l'objet d'intérêt restent *consistants* dans le temps. On recherchera ainsi d'une image à l'autre un objet possédant une couleur, une texture ou une forme similaire à celui qui nous intéressait dans l'image précédente.

Cependant, différentes contraintes viennent s'opposer à cette hypothèse de consistance et font du suivi un problème complexe : changements d'illumination, « flou de bougé », occultations de l'objet, etc. Ces difficultés devront être prises en compte dans un algorithme de suivi robuste. Il existe une littérature abondante sur le suivi visuel. [Yilmaz 06] fournit un état de l'art clair sur le sujet. Dans cette section on

présentera un aperçu des différents critères à prendre en compte dans le choix d'un algorithme de suivi dans le cas de mouvements rigides (objets non déformables). On décrira également les principaux cadres mathématiques de résolution de problèmes de suivi, déterministes et bayésiens.

1.3.1 Critères de choix d'un algorithme de suivi

De manière générale le choix d'un algorithme de suivi dépendra :

- des connaissances *a priori* dont on dispose. On pourra ainsi se demander si l'environnement est connu, s'il peut être équipé de marqueurs et si un modèle de l'objet est disponible ;
- du type d'informations que l'on cherche à reconstituer : quels sont les degrés de liberté à prendre en compte ? Quel type de mouvement doit-on être capable de détecter ?
- des contraintes vis-à-vis desquelles on veut assurer la robustesse de l'algorithme : des occultations sont-elles à envisager ? La qualité de réception des images, l'illumination de la scène sont-elles variables ?
- éventuellement de besoins spécifiques : est-ce que l'algorithme doit être capable de tourner en temps réel ? Doit-il se réinitialiser automatiquement ? Faut-il pouvoir suivre plusieurs objets en même temps ?

Ces différents critères, fortement liés à l'application visée, doivent permettre de sélectionner les éléments clés de l'algorithme, à savoir :

- **le mode de représentation de l'objet** : la forme de l'objet peut être représentée simplement par un point (son centre de gravité dans l'image), par une forme géométrique (ellipse, triangle, rectangle...), par son contour, ou encore par sa silhouette (la zone à l'intérieur du contour). L'apparence de l'objet quant à elle pourra être représentée sous forme d'une imagette de référence, ou de densité de probabilité de présence, paramétrique ou non (gaussienne, histogramme...), d'éléments caractéristiques ;
- **la transformation à estimer** : comme nous l'avons vu dans la section 1.2, il existe différentes façons de modéliser la transformation 2D ou 3D à estimer (changement de repère complet, homographie, mouvement affine, translation pure...). On notera que le type de transformation que l'on pourra estimer dépendra du type de représentation choisi. Ainsi, si l'objet est représenté par un point, on ne pourra détecter que des mouvements de translation dans l'image, tandis qu'une ellipse permettra d'identifier un mouvement de translation, de changement d'échelle et de rotation dans le plan ;
- **les informations visuelles utilisées** : les primitives visuelles sont les éléments de base que l'algorithme de suivi va chercher à retrouver d'une image à l'autre. Elles peuvent être de natures variées : couleur, contours, texture, flot optique, ... Chaque primitive dispose de caractéristiques propres qui la rendent plus ou moins adaptée à une application donnée. Par exemple, des primitives de type contour seront particulièrement utilisées dans des environnements structurés où ceux-ci sont abondants ;
- **la méthode de mise en correspondance spatio-temporelle** : une fois définis le type de représentation de l'objet ainsi que les primitives visuelles,

il s'agit de déterminer une méthode pour les suivre d'une image à l'autre, et donc de reconstituer la trajectoire spatio-temporelle de l'objet. Les méthodes de suivi utilisent généralement une fonction de coût qui permet d'évaluer un critère de ressemblance avec l'objet recherché. Différents cadres mathématiques permettent alors de déterminer la position qui optimise cette fonction de coût (optimisation déterministe, cadre statistique...). Les principales approches sont présentées plus en détail dans la section suivante.

1.3.2 Mise en correspondance spatio-temporelle

Différents formalismes existent pour exprimer le problème du suivi. On pourra les regrouper autour de deux catégories principales : les méthodes de type optimisation déterministe, et les méthodes statistiques. Dans le cadre déterministe, la position de l'objet est supposée déterminée de manière unique à un instant k et on cherche sa position à l'instant $k+1$. De manière générale, on va chercher à effectuer un recalage (2D ou 3D) sous forme de minimisation d'une fonction de coût adéquatement modélisée. Cette minimisation sera réalisée par un algorithme d'optimisation linéaire (moindres carrés) ou non linéaire (Gauss-Newton, Levenberg-Marquardt). Ce type d'approche permet généralement une bonne précision, mais ne prend pas en compte les incertitudes et peut être mis en échec en présence de minima locaux.

Une autre façon de considérer le problème de suivi consiste à utiliser une représentation probabilistique de la position de l'objet. Au lieu de considérer de manière déterministe que l'objet est à une position que l'on connaît avec certitude, on recherche la densité de probabilité associée à la position de l'objet. Autrement dit, à un état (généralement une position) possible de l'objet est associée une certaine probabilité en fonction des mesures dont on dispose.

Les notions fondamentales d'optimisation déterministe et de filtrage statistique nécessaires à la compréhension de nos travaux sont décrites dans les sections suivantes.

1.3.3 Formalisation du problème de suivi dans le cadre déterministe

Dans le cadre déterministe, le principe du suivi consiste à effectuer un recalage entre un modèle de référence de l'objet et des mesures issues des images. Le problème du suivi peut alors s'exprimer de la façon suivante. Soit $\boldsymbol{\mu}_k \in \mathcal{E}$ le vecteur des paramètres qui caractérise la position de l'objet dans l'image I_k à l'instant k. $\mathcal{E}$ désigne ainsi l'espace des vecteurs de paramètres. Si $n_{\boldsymbol{\mu}}$ est le nombre de paramètres considérés, $\mathcal{E} \subset \mathbb{R}^{n_{\boldsymbol{\mu}}}$. Le suivi consiste à trouver les nouveaux paramètres $\boldsymbol{\mu}_{k+1}$ qui minimisent une distance dans l'espace $\mathcal{F}$ des mesures à partir de l'image suivante I_{k+1}. On suppose que l'on dispose d'une fonction $g : \boldsymbol{\mu} \in \mathcal{E} \rightarrow g(\boldsymbol{\mu}) \in \mathcal{F}$ qui associe à chaque position un vecteur de mesure, de dimension n_m. Si l'espace $\mathcal{E}$ est muni d'une distance, le problème peut alors s'écrire dans le cas d'une distance Euclidienne :

$$\boldsymbol{\mu}_{k+1} = arg \min_{\boldsymbol{\mu}} \|g(\boldsymbol{\mu}) - \mathbf{b}\|^2 \tag{1.26}$$

où $\mathbf{b}$ est un vecteur de mesure de référence, qui peut être pris dans l'image précédente $\mathbf{b} = g(\boldsymbol{\mu}_k)$, à la position initiale $\mathbf{b} = g(\boldsymbol{\mu}_0)$, ou encore à partir d'un modèle connu $\mathbf{b} = g(\boldsymbol{\mu}^*)$. La distance à minimiser est également appelée *fonction de coût*, ou *fonction objectif*. Un exemple classique de fonction de coût consiste à utiliser directement l'intensité lumineuse des pixels de l'image : cette approche connue sous le nom de SSD (*Sum of Squared Differences*) sera présentée à titre d'exemple dans la Figure 3.2.

Méthode des moindres carrés. Lorsque la fonction g est linéaire, (1.26) peut s'écrire sous la forme :

$$\boldsymbol{\mu}_{k+1} = arg\min_{\boldsymbol{\mu}} \|\mathbf{A}\boldsymbol{\mu} - \mathbf{b}\|^2. \tag{1.27}$$

On cherche alors $\boldsymbol{\mu}$ tel que $\mathbf{A}\boldsymbol{\mu} = \mathbf{b}$. On a généralement $n_m > n_{\boldsymbol{\mu}}$, et on estime $\boldsymbol{\mu}$ tel que $\boldsymbol{\mu} = \mathbf{A}^+\mathbf{b}$ où $\mathbf{A}^+$ désigne la pseudo inverse de $\mathbf{A}$ définie par :

$$\mathbf{A}^+ = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top \tag{1.28}$$

Si l'on souhaite attribuer un poids plus important à certaines mesures qu'à d'autres, on pourra utiliser la méthode des moindres carrés pondérés. On cherche alors à résoudre l'équation :

$$\mathbf{W}\mathbf{A}\boldsymbol{\mu} = \mathbf{W}\mathbf{b} \tag{1.29}$$

où $\mathbf{W}$ est une matrice de poids, généralement diagonale. La solution est donnée par :

$$\boldsymbol{\mu} = (\mathbf{A}^\top\mathbf{W}^\top\mathbf{W}\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{W}^\top\mathbf{W}\mathbf{b}. \tag{1.30}$$

En particulier, pour réduire l'influence d'éventuelles mesures aberrantes dans le cadre d'une estimation robuste, les poids peuvent être calculés à l'aide d'un *M-estimateur* [Huber 81]. Si l'on note $\mathbf{W} = diag(\dots w_i \dots)$ et que l'on réécrit la fonction à minimiser sous la forme d'une somme de résidus :

$$\|\mathbf{A}\boldsymbol{\mu} - \mathbf{b}\|^2 = \sum_{i=1}^{n_m} {r_i}^2, \tag{1.31}$$

les poids w_i correspondant à des résidus r_i non nuls s'expriment alors de la façon suivante :

$$w_i = \frac{\rho(r_i)^{1/2}}{r_i}. \tag{1.32}$$

La fonction ρ est une fonction d'estimation robuste. Pour plus d'information sur la théorie des M-estimateurs et le choix de la fonction ρ on pourra se référer à [Huber 81].

Méthodes de type Gauss-Newton. Lorsque la fonction g n'est pas linéaire, ce qui est souvent le cas de par le caractère non linéaire de la projection perspective, il est possible de minimiser l'erreur de manière itérative. C'est ce que proposent les algorithmes de Gauss-Newton ou Levenberg-Marquardt. Partant d'une estimation initiale $\boldsymbol{\mu}_0$ on pose ainsi :

$$\boldsymbol{\mu}_{i+1} = \boldsymbol{\mu}_i + \delta_i. \tag{1.33}$$

Les deux algorithmes diffèrent dans la manière dont est calculé le pas δ_i dont on se déplace à chaque itération. Dans l'algorithme de Gauss-Newton, δ_i est choisi de manière à minimiser le résidu $\varepsilon_i = g(\boldsymbol{\mu}_i) - \mathbf{b}$ en faisant un développement limité de g au premier ordre : $g(\boldsymbol{\mu}_i + \delta_i) \approx g(\boldsymbol{\mu}_i) + \mathbf{J}\delta_i$ où $\mathbf{J}$ est la matrice Jacobienne [2] de g calculée en $\boldsymbol{\mu}_i$. On obtient ainsi :

$$\begin{aligned} \delta_i &= arg\min_{\delta} \|g(\boldsymbol{\mu}_i) + \mathbf{J}\delta - \mathbf{b}\|^2 && (1.34)\\ &= -\mathbf{J}^+\varepsilon_i && (1.35)\\ &= -(\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top\varepsilon_i && (1.36) \end{aligned}$$

La procédure de Gauss-Newton est résumée dans l'algorithme 1.

Algorithme de Gauss-Newton
Minimisation itérative de la fonction $\|g(\boldsymbol{\mu}) - \mathbf{b}\|^2$ à partir d'une position initiale $\boldsymbol{\mu}_0$.
Itérer :
1. calcul de la Jacobienne $\mathbf{J} = \left.\frac{\partial g}{\partial \boldsymbol{\mu}}\right|_{\boldsymbol{\mu}_i}$;
2. calcul du pas $\delta_i = -\mathbf{J}^+(g(\boldsymbol{\mu}_i) - \mathbf{b})$ (équation (1.36)) ;
3. mise à jour des paramètres $\boldsymbol{\mu}_{i+1} = \boldsymbol{\mu}_i + \delta_i$;

tant que $\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_{i-1}\|^2 > \varepsilon$.

Algorithme 1 – Gauss-Newton.

La méthode de Gauss-Newton est un cas particulier de la méthode de Newton-Raphson qui consiste à approcher localement la fonction g par une parabole et à se déplacer vers le minimum de cette parabole (Figure 1.7), dans lequel la matrice Hessienne [3] est approchée par la matrice $\mathbf{J}^\top\mathbf{J}$.

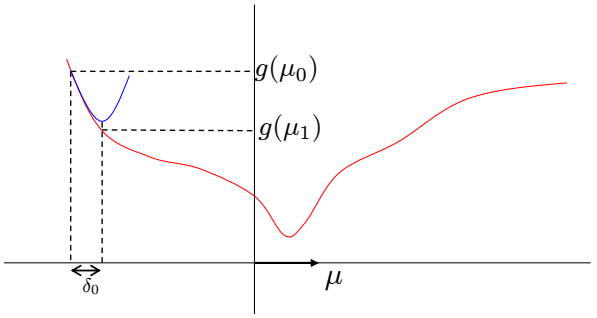


FIGURE *1.7 – Méthode de Newton.*

Dans le cas de la méthode de Levenberg-Marquardt, le pas est calculé comme une interpolation entre le pas d'un algorithme de Gauss-Newton et celui d'une « simple » descente de gradient :

$$\delta_i = -(\mathbf{J}^\top\mathbf{J} + \lambda\mathbf{I})^{-1}\mathbf{J}^\top\varepsilon_i. \quad (1.37)$$

2. La matrice Jacobienne d'une fonction est la matrice de ses dérivées partielles du premier ordre.
3. La matrice Hessienne (ou simplement la Hessienne) d'une fonction est la matrice carrée de ses dérivées partielles secondes.

Le terme additionnel $\lambda \mathbf{I}$ agit comme un terme d'amortissement qui permet de réguler l'amplitude du pas en fonction de l'évolution de l'erreur. Si le pas calculé à l'étape i induit une diminution de l'erreur, le paramètre λ peut être réduit, et l'algorithme se rapproche d'un algorithme de Gauss-Newton (1.36), dans le cas contraire, on augmente la valeur de λ rapprochant cette fois l'algorithme d'une descente de gradient.

La prise en compte des dérivées d'ordre 2 (matrice Hessienne) dans l'approximation de g peut améliorer la performance dans certains cas, mais implique une augmentation de la complexité algorithmique et du temps de calcul.

1.3.4 Formalisation du problème de suivi dans le cadre bayésien

Nous considérons à présent le problème du suivi comme un problème d'estimation statistique. Dans ce cadre, on cherche ici à estimer l'état de notre système au cours du temps, en fonction d'observations ou mesures. Dans le cas du suivi, l'état contient généralement les paramètres $\boldsymbol{\mu}_k$ de position 2D ou 3D de l'objet suivi, mais il peut également contenir d'autres paramètres (vitesse, accélération,...). Ainsi, supposons que l'on dispose d'un modèle f de l'évolution de l'état $\mathbf{x}_k$ du système qui permet à partir de l'état précédent $\mathbf{x}_{k-1}$ de donner une prédiction du nouvel état du système $\mathbf{x}_k$. Ce modèle est une approximation du comportement du système, dont l'incertitude est modélisée par un bruit $\boldsymbol{n}_k$.

Considérons à titre d'exemple que l'on souhaite suivre un mouvement de translation d'un objet dans une séquence d'images. Avec les notations de la section 1.3.3, l'espace $\mathcal{E}$ est alors l'espace des vecteurs de translation du plan image, de dimension $n_{\boldsymbol{\mu}} = 2$. Lorsque l'on ne dispose d'aucune information *a priori* sur le mouvement de l'objet, on peut modéliser de façon simplifiée son mouvement sous la forme d'un modèle à *position constante* par exemple :

$$f(\mathbf{x}_{k-1}, \boldsymbol{n}_k) = \mathbf{x}_{k-1} + \boldsymbol{n}_k. \tag{1.38}$$

Dans ce cas, l'état $\mathbf{x}_k$ correspond au vecteur de paramètres $\boldsymbol{\mu}_k = (x_k \ y_k)^\top$ définissant la position du centre de l'objet.

Si l'on considère plutôt un modèle à *vitesse constante* on pourra inclure la vitesse $(v_{x_k} \ v_{y_k})^\top$ dans l'état : $\mathbf{x}_k = (x_k \ y_k \ v_{x_k} \ v_{y_k})^\top$. L'évolution de l'état peut alors être modélisée par l'équation d'état linéaire suivante :

$$f(\mathbf{x}_{k-1}, \boldsymbol{n}_k) = \mathbf{A}\mathbf{x}_{k-1} + \boldsymbol{n}_k \tag{1.39}$$

où pour un intervalle de temps δt :

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & \delta t & 0 \\ 0 & 1 & 0 & \delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{1.40}$$

On suppose également que l'on dispose d'un modèle de mesure, ou modèle d'observation

$$\mathbf{z}_k = h(\mathbf{x}_k, \boldsymbol{w_k}) \tag{1.41}$$

avec $\boldsymbol{w_k}$ le bruit associé à ce modèle.

Dans le cadre statistique considéré dans cette section, l'état et la mesure sont des informations bruitées et à l'inverse des méthodes déterministes, ce bruit est ici pris en compte dans le processus d'estimation. L'état et la mesure sont ainsi considérés comme des variables aléatoires au sens statistique du terme. Estimer l'état $\mathbf{x}_k$ du système à partir des mesures $\mathbf{z}_{1:k} = (\mathbf{z}_1, .., \mathbf{z}_k)$ connues jusqu'à l'instant k équivaut alors à déterminer la fonction de densité de probabilité $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$ qui représente la densité de probabilité de l'état $\mathbf{x}_k$ connaissant les mesures $\mathbf{z}_{1:k}$.

Dans la section 1.3.4.1 nous présentons le formalisme du filtrage bayésien générique. Les sections 1.3.4.2 et 1.3.4.3 décrivent plus particulièrement deux types de filtres bayésiens classiques correspondant à des approximations de ce filtre générique, à savoir le filtre de Kalman et le filtre particulaire.

1.3.4.1 Filtrage bayésien

Le filtrage bayésien repose sur la loi de Bayes sur les probabilités conditionnelles :

Théorème 1.3.1 (*Bayes*)

Pour toutes variables aléatoires A, B

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} \tag{1.42}$$

où $P(A \mid B)$ désigne la probabilité de A sachant B.

Dans le cadre qui nous intéresse, en supposant que les mesures sont indépendantes entre elles conditionnellement aux états, la loi de Bayes nous permet d'écrire la relation récursive suivante :

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k \mid \mathbf{x}_k)p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k \mid \mathbf{z}_{1:k-1})} \tag{1.43}$$

On note que $p(\mathbf{z}_k \mid \mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k \mid \mathbf{x}_k)p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})d\mathbf{x}_k$ est un simple facteur de normalisation, indépendant de l'état. Ainsi la relation (1.43) indique que la densité *a posteriori* de l'état est proportionnelle au produit de la densité de probabilité *a priori* de l'état $p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})$ et de la vraisemblance de la mesure $p(\mathbf{z}_k \mid \mathbf{x}_k)$.

En supposant que l'on est dans le cadre d'un processus markovien pour lequel l'état $\mathbf{x}_k$ ne dépend que de l'état précédent $\mathbf{x}_{k-1}$, la relation (1.43) devient :

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) \propto p(\mathbf{z}_k \mid \mathbf{x}_k) \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1})p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1})d\mathbf{x}_{k-1}. \tag{1.44}$$

Ainsi le cadre bayésien fournit une solution récursive optimale à ce problème basée sur deux étapes :

$$p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1}) \overset{prediction}{\longrightarrow} p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1}) \overset{correction}{\longrightarrow} p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) \tag{1.45}$$

- **l'étape de prédiction** utilise l'équation de dynamique et la densité de probabilité précédente $p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1})$ pour obtenir la densité de probabilité *a*

priori $p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})$:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1}) d\mathbf{x}_{k-1}; \tag{1.46}$$

– **l'étape de correction** permet alors de calculer la densité de probabilité postérieure $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$ à partir de la fonction de vraisemblance $p(\mathbf{z}_k \mid \mathbf{x}_k)$ grâce à la formule de Bayes récursive (1.44).

Cette solution récursive est valable pour tout processus markovien pour lequel les observations sont indépendantes conditionnellement aux états (ce qui est généralement le cas dans un problème de suivi). Ce filtre est donc très général dans la mesure où aucune hypothèse n'est faite sur la forme de la densité de probabilité que l'on cherche à estimer.

En revanche, il n'est pas toujours possible de résoudre les équations (1.44) et (1.46) explicitement. En fonction de l'application visée, on utilisera diverses simplifications pour approcher la solution optimale. Dans le cas linéaire gaussien, une solution analytique est fournie par le filtre de Kalman. Dans le cas général on fera appel à des méthodes d'approximations :

– linéarisation du système autour de l'estimée courante (filtre de Kalman étendu ou EKF),
– approximation numérique par discrétisation ou maillage de l'espace d'état,
– représentation du filtre par des échantillons : filtres particulaires.

Les sections suivantes présentent les cas du filtrage de Kalman et du filtrage particulaire qui sont les plus couramment utilisés pour des problèmes de suivi visuel.

1.3.4.2 Le filtre de Kalman étendu

Si le filtre présenté dans la section précédente ne faisait aucune hypothèse sur la forme des densités de probabilité, dans cette section on considèrera que les bruits d'état et de mesure du système suivent une loi normale gaussienne. La loi gaussienne est fréquemment utilisée pour représenter les variables aléatoires sur lesquelles on dispose de peu d'information *a priori* .

Une loi normale gaussienne $\mathcal{N}_{\boldsymbol{\mu},\boldsymbol{\Sigma}}$ sur un espace de dimension N est définie par sa moyenne $\boldsymbol{\mu}$ et sa dispersion autour de la moyenne définie par la matrice de covariance $\boldsymbol{\Sigma}$:

$$\mathcal{N}_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{N/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}. \tag{1.47}$$

La principale restriction de cette représentation est son caractère unimodal. Elle est cependant suffisante pour un grand nombre d'applications.

Si l'on représente l'état du système par une variable aléatoire suivant une loi normale $\mathcal{N}(\boldsymbol{x})$ de covariance $\boldsymbol{P}$, le filtre optimal peut être calculé explicitement. On donnera ici directement les équations obtenues dans le cas d'un filtre de Kalman étendu (EKF). L'EKF est une extension du filtre de Kalman classique au cas où les modèles d'évolution et de mesure sont non linéaires mais peuvent être linéarisés localement [BarShalom 93].

On considère que l'on dispose d'un modèle f de l'évolution de l'état $\mathbf{x}_k$ du système qui permet à partir de l'état précédent $\mathbf{x}_{k-1}$ de prédire le nouvel état du système $\mathbf{x}_{k|k-1}$. L'incertitude de cette prédiction est modélisée par un bruit $\boldsymbol{n}_k$ de loi $\mathcal{N}_{0,\boldsymbol{Q}}$:

$$\mathbf{x}_{k|k-1} = f(\mathbf{x}_{k-1}, \boldsymbol{n}_k), \tag{1.48}$$

ainsi que d'un modèle de mesure, ou modèle d'observation :

$$\mathbf{z}_k = h(\mathbf{x}_k, \boldsymbol{w_k}) \tag{1.49}$$

avec $\boldsymbol{w_k}$ le bruit associé à ce modèle, supposé suivre une loi normale centrée de covariance $\boldsymbol{R}$.

À l'initialisation du filtre on détermine l'état initial $\mathbf{x}_0$ et la covariance initiale $\boldsymbol{P}_0$ en fonction des informations dont on dispose. Les étapes suivantes sont ensuite appliquées de manière itérative :

Prédiction. Le modèle d'évolution (1.48) est d'abord utilisé pour prédire l'état $\mathbf{x}_{k|k-1}$ du système. La matrice de covariance de l'état est alors mise à jour en utilisant :

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_{\mathbf{x}} \boldsymbol{P}_{k-1} \boldsymbol{F}_{\mathbf{x}}^\top + \boldsymbol{F}_{\boldsymbol{n}} \boldsymbol{Q} \boldsymbol{F}_{\boldsymbol{n}}^\top \tag{1.50}$$

où $\boldsymbol{F}_{\mathbf{x}} = \frac{\partial f(\mathbf{x}_{k-1}, 0)}{\partial \mathbf{x}_{k-1}}$ et $\boldsymbol{F}_{\boldsymbol{n}} = \frac{\partial f(\mathbf{x}_{k-1}, 0)}{\partial \boldsymbol{n}_k}$.

Le modèle de mesure (1.49) est utilisé pour prédire la mesure en utilisant l'état prédit. Cette mesure prédite pourra alors être comparée avec la mesure réellement obtenue. La différence entre les deux est l'innovation :

$$\tilde{\mathbf{z}}_k = \mathbf{z}_k - h(\mathbf{x}_{k|k-1}). \tag{1.51}$$

La covariance $\boldsymbol{S}$ de l'innovation est calculée à l'aide de la Jacobienne du modèle de mesure $\boldsymbol{H} = \frac{\partial h(\mathbf{x}_{k|k-1}, 0)}{\partial \mathbf{x}_{k|k-1}}$

$$\boldsymbol{S} = \boldsymbol{H} \boldsymbol{P}_{k|k-1} \boldsymbol{H}^\top + \boldsymbol{R}. \tag{1.52}$$

Correction. Dans un deuxième temps, l'étape de correction permet de mettre à jour l'état et sa covariance en fonction des prédictions et de l'innovation. La matrice de gain $\boldsymbol{K}$ traduit l'importance plus ou moins grande qui est donnée à la mesure ou à la prédiction.

$$\boldsymbol{K} = \boldsymbol{P}_{k|k-1} \boldsymbol{H}^\top \boldsymbol{S}^{-1} \tag{1.53}$$

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \boldsymbol{K} \tilde{\mathbf{z}}_k \tag{1.54}$$

Chaque nouvelle mesure apporte une information supplémentaire sur l'état, dont la covariance est mise à jour par l'équation :

$$\boldsymbol{P}_k = \boldsymbol{P}_{k|k-1} - \boldsymbol{K} \boldsymbol{S} \boldsymbol{K}^\top. \tag{1.55}$$

Le filtre de Kalman permet une bonne approximation du filtre optimal lorsque le système est linéaire ou faiblement non linéaire et que les densités de probabilité peuvent être assimilées à des lois normales. En revanche, il peut être insuffisant lorsque le modèle est fortement non linéaire ou dans le cas des densités multimodales (présence de plusieurs objets d'apparence similaire par exemple dans le cas du suivi). Le filtrage particulaire est une approximation du filtre bayésien optimal qui ne fait pas d'hypothèse sur la forme de la densité ou la linéarité du système.

1.3.4.3 Filtrage particulaire

Dans cette section nous montrerons comment le filtre particulaire peut fournir une approximation du filtre optimal présenté précédemment. Contrairement au cas du filtrage de Kalman, aucune hypothèse n'est faite sur la forme de la densité de probabilité ou la linéarité du système [Arulampalam 02] [Doucet 01]. La théorie du filtrage particulaire est basée sur l'approximation particulaire décrite ci-après.

Approximation particulaire : principe de Monte Carlo. Ne sachant pas calculer analytiquement les intégrales apparaissant dans (1.44) et (1.46), les méthodes de Monte-Carlo utilisent une approximation permettant d'approcher ces intégrales. Le principe de l'approximation particulaire est de représenter la densité de probabilité que l'on cherche à estimer par un ensemble d'échantillons. Ces échantillons, les particules, sont supposés être indépendants et identiquement distribués [4] selon la densité de probabilité à estimer. Comme illustré sur la Figure 1.8 dans le cas d'un espace d'états de dimension 1, de tels échantillons sont ainsi statistiquement plus nombreux au niveau des pics de la loi de probabilité considérée.

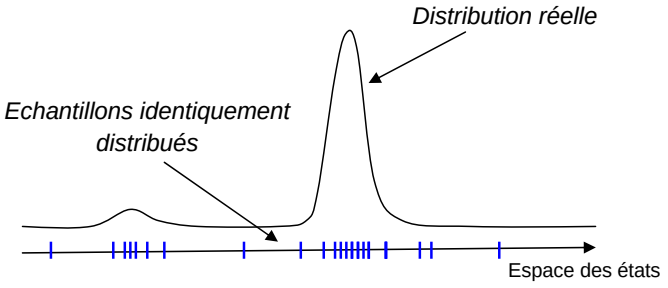


FIGURE 1.8 – *Approximation de Monte-Carlo.*

L'idée de l'approximation particulaire repose sur la loi forte des grands nombres selon laquelle l'espérance calculée sur les échantillons est un estimateur de l'espérance de la densité réelle. Plus formellement, si on note $p(\mathbf{x})$ la densité de probabilité que l'on cherche à approcher, et $\{s^{(i)}\}_{i=1}^{N}$ un ensemble de N échantillons indépendants et identiquement distribués selon p, alors pour toute fonction ϕ continue et bornée

4. Des variables indépendantes et identiquement distribuées (i.i.d) sont des variables aléatoires qui ont toutes la même loi de probabilité et qui sont mutuellement indépendantes.

on a :

$$\frac{1}{N}\sum_{i=1}^{N}\phi(s^{(i)}) \underset{N\to\infty}{\longrightarrow} E_p[\phi(\mathbf{x})] = \int \phi(\mathbf{x})p(\mathbf{x})d\mathbf{x} \tag{1.56}$$

où E_p désigne l'espérance prise par rapport à la densité p.
Soit la quantité $p^N(\mathbf{x})$ définie par :

$$p^N(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}\delta_{s^{(i)}}(\mathbf{x}) \tag{1.57}$$

avec $\delta_{s^{(i)}}$ la distribution de Dirac centrée en $s^{(i)}$. $p^N(\mathbf{x})$ est une approximation particulaire de la loi $p(\mathbf{x})$. Plus généralement, une approximation particulaire sera de la forme :

$$p^N(\mathbf{x}) = \sum_{i=1}^{N}\pi^{(i)}\delta_{s^{(i)}}(\mathbf{x}) \tag{1.58}$$

avec $\pi^{(i)} \geq 0$ et $\sum_{i=1}^{N}\pi^{(i)} = 1$.
Cette écriture permet d'associer un poids $\pi^{(i)}$ à chaque particule $s^{(i)}$. On a alors pour tout ϕ :

$$E_p[\phi(\mathbf{x})] = \int \phi(\mathbf{x})p(\mathbf{x})d\mathbf{x} \approx \int \phi(\mathbf{x})p^N(\mathbf{x})d\mathbf{x} = \sum_{i=1}^{N}\pi^{(i)}\phi(s^{(i)}). \tag{1.59}$$

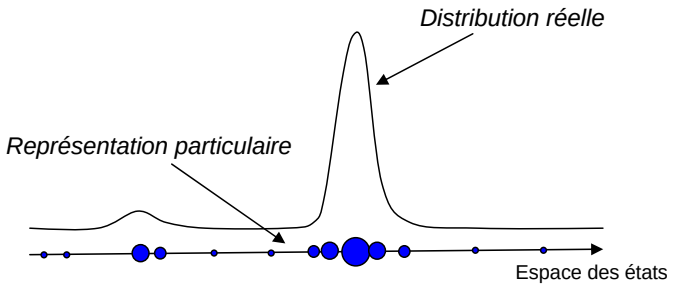


FIGURE *1.9 – Approximation particulaire.*

Avant de décrire comment cette approximation s'applique dans le cadre bayésien que nous avons présenté précédemment, on peut s'interroger sur la façon d'obtenir une telle représentation lorsque la densité p n'est pas directement accessible. Ainsi, lorsque l'on souhaite estimer une distribution p, il n'est pas toujours possible d'obtenir directement un ensemble d'échantillons identiquement distribués selon p. L'échantillonnage d'importance décrit dans le paragraphe suivant pourra être utilisé pour y parvenir.

Échantillonnage d'importance. Supposons que l'on souhaite obtenir une représentation de la forme (1.58) d'une densité de probabilité p, p n'étant pas directement accessible. Supposons connue une distribution q, dite fonction d'importance ou loi

de proposition, telle que $p = 0 \Rightarrow q = 0$, et que l'on sache obtenir N réalisations $s^{(i)}$ de q. On a alors :

$$\int \phi(\mathbf{x})p(\mathbf{x})d\mathbf{x} = \int \phi(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})}q(\mathbf{x})d\mathbf{x} \tag{1.60}$$

$$\int \phi(\mathbf{x})p^N(\mathbf{x})d\mathbf{x} = \frac{1}{N}\sum_{i=1}^{N}\frac{p(s^{(i)})}{q(s^{(i)})}\phi(s^{(i)}), \tag{1.61}$$

où $p^N(\mathbf{x})$ est donné par l'expression (1.57). Les facteurs $\pi^{(i)} = \frac{p(s^{(i)})}{q(s^{(i)})}$ sont appelés les poids d'importance. En pratique on préfèrera les normaliser pour se ramener à l'écriture (1.58) avec $\sum_{i=1}^{N}\pi^{(i)} = 1$.

Filtrage particulaire. Le filtrage particulaire consiste à déterminer des approximations de type particulaire du filtre bayésien récursif. Ainsi, on souhaite obtenir une approximation de la densité $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$ sous la forme :

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) \approx p^N(\mathbf{x}_k \mid \mathbf{z}_{1:k}) = \sum_{i=1}^{N}\pi_k^{(i)}\delta_{s_k^{(i)}}(\mathbf{x}_k). \tag{1.62}$$

Supposons que l'on dispose à l'instant $k-1$ d'une telle approximation :

$$p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1}) \approx p^N(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1}) = \sum_{i=1}^{N}\pi_{k-1}^{(i)}\delta_{s_{k-1}^{(i)}}(\mathbf{x}_{k-1}). \tag{1.63}$$

Les équations du filtre de Bayes récursif (1.44) et (1.46) deviennent :

- **Prédiction** :

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1}) \approx p^N(\mathbf{x}_k \mid \mathbf{z}_{1:k-1}) = \sum_{i=1}^{N}\pi_{k-1}^{(i)}\delta_{s_k^{(i)}}(\mathbf{x}_k) \tag{1.64}$$

  où les $s_k^{(i)}$ sont échantillonnés selon $p(\mathbf{x}_k \mid \mathbf{x}_{k-1} = s_{k-1}^{(i)})$.
- **Correction** : d'après (1.44) on obtient :

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) \approx \sum_{i=1}^{N}\pi_k^{(i)}\delta_{s_k^{(i)}}(\mathbf{x}_k) \tag{1.65}$$

  avec

$$\pi_k^{(i)} = \frac{\pi_{k-1}^{(i)}p(\mathbf{z}_k \mid \mathbf{x}_k = s_k^{(i)})}{\sum_{i=1}^{N}\pi_{k-1}^{(i)}p(\mathbf{z}_k \mid \mathbf{x}_k = s_k^{(i)})}. \tag{1.66}$$

On obtient ainsi une expression explicite de l'approximation particulaire à l'instant k à partir de son expression à l'instant $k-1$.

On s'est restreint dans cette section au cas fréquent où les particules sont propagées selon la loi $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$. Dans le cas général où on utilise une fonction d'importance pour faire évoluer les particules, on fera appel au principe de l'échantillonnage d'importance présenté dans le paragraphe précédent (voir [Arulampalam 02] par exemple).

En pratique, le filtrage particulaire se déroule selon le principe suivant : un ensemble de particules est créé, chaque particule représentant un état possible du système. Ces particules évoluent ensuite dans l'espace d'état selon un modèle de prédiction. On obtient ainsi un nouvel ensemble de particules, qui représente la densité de probabilité de la prédiction. Dans un deuxième temps, la vraisemblance de chacune de ces particules prédites est évaluée à l'aide d'une mesure. Ainsi un poids est attribué à chaque particule prédite en fonction de sa vraisemblance par rapport à la mesure. Le nouvel ensemble de particules ainsi pondérées représente alors la nouvelle densité de probabilité *a posteriori* . Cette méthode est résumée par l'algorithme 2.

Algorithme SIS

Initialisation : générer N particules $\left\{s_0^{(i)}\right\}_{i=1}^{N}$ et poser $\pi_0^{(i)} = \frac{1}{N}$.

Pour $k = 1, ..., T$:
1. Prédiction : générer N particules $s_k^{(i)}$ selon $p(\mathbf{x}_k \mid \mathbf{x}_{k-1} = s_{k-1}^{(i)})$
2. Correction : mettre à jour les poids des particules prédites en utilisant (1.66)

Espérance : $E[\mathbf{x}_k] = \sum_{i=1}^{N} \pi_k^{(i)} s_k^{(i)}$

Algorithme 2 – SIS.

Dégénérescence. L'algorithme SIS (pour *Sampling Importance Sampling*), basé uniquement sur ces étapes de prédiction/correction (voir l'algorithme 2), n'est pas viable en pratique. En effet au bout de quelques itérations, la majorité des particules se voient attribuer un poids proche de zéro, comme illustré sur la Figure 1.10. Un nombre très restreint de particules correspond ainsi à une possibilité vraisemblable de l'état. L'algorithme passe alors l'essentiel de son temps à mettre à jour des particules qui contribuent de façon négligeable à l'estimation de la densité de probabilité. Ce problème est connu sous le nom de dégénérescence. Pour y remédier,

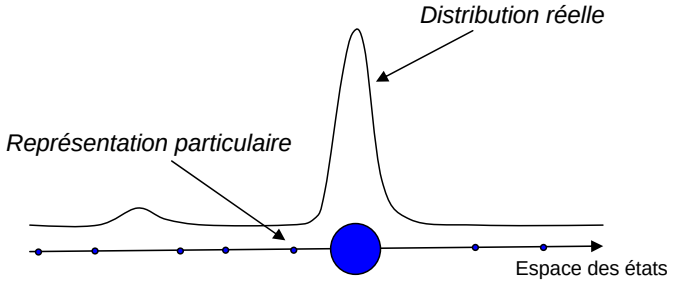


FIGURE *1.10 – Dégénérescence.*

les filtres particulaires classiques utilisent une phase de rééchantillonnage, qui vise à redistribuer les particules vers les zones de forte vraisemblance.

Rééchantillonnage. Pour éviter le problème de dégénérescence, le principe du rééchantillonnage consiste à favoriser les particules ayant une forte vraisemblance en les

dupliquant, au détriment des particules de poids négligeable [Smith 92], [Gordon 93]. L'algorithme obtenu en ajoutant une étape de rééchantillonnage est connu dans la littérature sous le nom de SIR (pour *Sampling Importance Resampling*), bootstrap [Gordon 93] ou encore CONDENSATION [Isard 98a]. Il est résumé dans l'algorithme 3 et illustré Figure 1.11. L'étape cruciale du rééchantillonnage consiste à effectuer

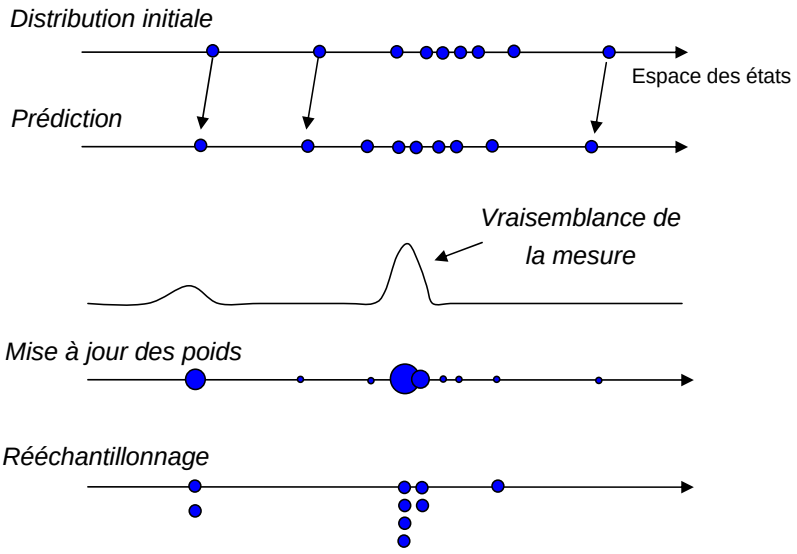


FIGURE 1.11 – *Illustration de la première itération de l'algorithme SIR.*

un tirage pondéré avec remise de N particules dans l'ensemble des particules courantes. La probabilité qu'une particule soit tirée est proportionnelle à son poids. Les particules de faible vraisemblance auront donc peu de chance d'être sélectionnées par l'étape de rééchantillonnage, tandis que celles de poids fort seront dupliquées. Pour conserver une bonne approximation de la densité de probabilité, les nouvelles particules auront toutes le même poids.

Algorithme SIR dans sa version CONDENSATION [Isard 98a]

Initialisation : générer N particules $\left\{s_0^{(i)}\right\}_{i=1}^{N}$ et poser $\pi_0^{(i)} = \frac{1}{N}$.

Pour $k = 1, ..., T$:

1. Prédiction : générer N particules $s_k^{\prime(i)}$ selon $p(\mathbf{x}_k \mid \mathbf{x}_{k-1} = s_{k-1}^{(i)})$
2. Correction : mettre à jour les poids des particules prédites en utilisant (1.66)
3. Rééchantillonnage : tirage aléatoire pondéré de N particules dans l'ensemble $\left\{(s_{k-1}^{\prime(i)}, \pi_{k-1}^{(i)})\right\}_{i=1}^{N}$

   On obtient $\left\{(s_{k-1}^{(i)}, \frac{1}{N})\right\}_{i=1}^{N}$

Espérance : $E[\mathbf{x}_k] = \sum_{i=1}^{N} \pi_k^{(i)} s_k^{\prime(i)}$

Algorithme 3 – SIR.

Appauvrissement des états. Un autre problème classique du filtrage particulaire peut apparaître notamment lorsque la vraisemblance est très localisée (pic étroit). Un très petit nombre de particules auront alors un poids important, et au fur et à mesure des itérations, le nuage de particules converge vers une partie très restreinte de l'espace d'état. Le filtre perd alors sa flexibilité et ne rend plus compte de ce qui se situe dans le reste de l'espace d'état (voir Figure 1.12).

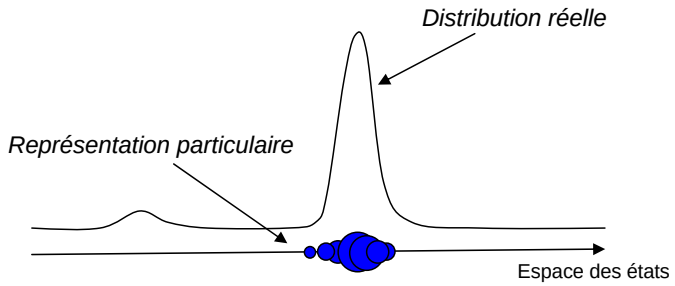


FIGURE *1.12 – Appauvrissement des états. Les particules ne couvrent pas suffisamment l'espace d'état et les pics secondaires éventuels ne sont plus représentés.*

L'un des enjeux de la mise en place d'un filtre particulaire sera donc de guider les particules vers les régions d'intérêt tout en conservant une bonne diversité de la représentation de l'état.

Le filtre particulaire est très largement utilisé en vision par ordinateur. Il permet une grande flexibilité. Comme nous l'avons vu, le filtre particulaire tend vers le filtre bayésien optimal lorsque le nombre de particules tend vers l'infini. En pratique cela signifie que la précision de l'estimation obtenue dépend directement du nombre de particules utilisées. Si l'espace d'état est de grande dimension, un grand nombre de particules sera nécessaire. Or chaque itération implique de calculer la prédiction et la correction de chaque particule. L'utilisation d'un nombre important de particules peut donc induire des temps de calcul considérables.

Le choix d'une fonction d'importance sera alors crucial pour maintenir une bonne représentation de l'état avec un nombre réduit de particules. La fonction d'importance aura pour rôle de permettre l'exploration des zones pertinentes de l'espace d'état.

Pour un aperçu plus complet du filtrage particulaire et des variantes proposées pour améliorer ses performances dans des applications variées on pourra se référer à [Doucet 01].

1.4 Conclusion

Dans ce chapitre nous avons présenté les concepts théoriques fondamentaux de vision nécessaires à la compréhension de nos travaux. La première section a présenté le modèle de formation des images numériques par projection perspective et les différents paramètres qui caractérisent une caméra à projection perspective. Dans un

deuxième temps, nous avons introduit les paramétrisations qui permettent de décrire les transformations géométriques dans l'espace 3D et dans le plan image 2D. Enfin, la troisième section a présenté les outils génériques classiquement utilisés dans le domaine du suivi visuel, et décrit en particulier les formalismes d'optimisation déterministe et de filtrage bayésien qui fournissent deux cadres de résolution différents pour aborder le problème du suivi.

Chapitre 2

Vision et commande de drones

Si la vue d'un hélicoptère dans le ciel ne nous surprend plus aujourd'hui, de tels engins capables de s'élever dans les airs en défiant les lois de la pesanteur n'en demeurent pas moins d'une grande complexité technique. Contrairement aux véhicules à *voilure fixe* tels que les avions qui volent grâce à leur vitesse, les véhicules à *voilure tournante* parviennent à se sustenter dans les airs et s'y maintenir quasiment immobiles, leur poids étant compensé par les forces aérodynamiques générées par la rotation de leurs pales. Cette capacité de vol stationnaire présente un intérêt tout particulier pour effectuer des missions d'inspection ou de surveillance. La possibilité de décoller et atterrir verticalement [1] permet leur utilisation dans des zones relativement encombrées.

Ces dernières années, les progrès technologiques notamment en termes de miniaturisation des capteurs ont permis le développement de petits véhicules à voilure tournante pilotés à distance, candidats particulièrement adaptés pour évoluer dans des zones difficiles ou dangereuses pour l'Homme. Les applications de ces drones sont nombreuses à la fois dans les domaines militaire et civil : surveillance de convois, de trafic routier ou de manifestations, inspection d'ouvrages d'art (ponts par exemple), de lignes électriques aériennes ou de pales d'éoliennes, surveillance d'incendies, ou encore recherche de victimes et assistance dans le cadre de catastrophes naturelles, etc. Ces tâches sont, à ce jour, essentiellement confiées à des opérateurs expérimentés pilotant les drones à distance. L'un des enjeux des développements actuels sur ce type d'appareils consiste à les rendre plus autonomes. En particulier, il s'agira de leur permettre de décoller, se localiser, se positionner par rapport à une zone d'intérêt, naviguer, et atterrir en toute sécurité. Dans cette optique, les informations visuelles nécessairement acquises pour les missions envisagées (surveillance par exemple) ont un rôle essentiel à jouer, et de nombreuses recherches ont été menées ces dix dernières années pour utiliser ces informations dans la commande de drones.

Ce chapitre vise à présenter les bases de la commande de drones par la vision. Bien que les algorithmes de vision utilisés dans la suite de cette thèse soient transposables

1. On parlera également de véhicules de type VTOL, pour *Vertical Take Off and Landing*.

à d'autres types de robots, nous restreindrons notre propos aux drones à voilure tournante. Tout d'abord, la section 2.1 dressera un rapide panorama des drones à voilure tournante existant à l'heure actuelle et de leur principe de fonctionnement. Dans la section 2.2 nous présenterons plus particulièrement les différentes techniques de commande de ce type d'engins à partir d'informations visuelles.

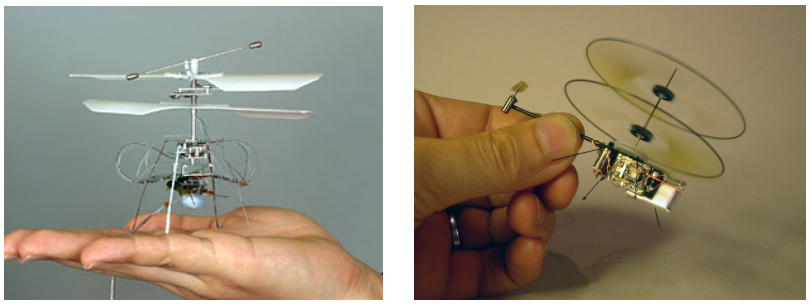
2.1 Les drones à voilure tournante

On appelle drone un appareil aérien sans pilote humain à bord, capable d'effectuer des missions de façon plus ou moins autonome. Ce terme inclut donc *a priori* une grande variété d'appareils : ballons dirigeables, avions, hélicoptères ou encore drones à ailes battantes inspirés du vol des insectes ou des oiseaux. Nous nous intéressons plus particulièrement ici aux drones à voilure tournante tels que les hélicoptères. Contrairement aux engins à ailes fixes, les drones à voilure tournante sont capables de vol stationnaire, ce qui leur permet d'évoluer dans des espaces limités et de se positionner au-dessus d'une zone d'intérêt. Cette section vise à donner les principes généraux du fonctionnement de ce type d'appareils afin de mieux appréhender les problématiques spécifiques de leur commande par la vision.

2.1.1 Généralités

La force d'élévation d'un drone à voilure tournante est générée par le mouvement de ses pales tournant autour de l'axe d'un (ou plusieurs) rotor(s). Cette force, appelée *portance*, dépend de l'angle d'attaque, ou *incidence*, des pales ainsi que de la vitesse de rotation du (ou des) rotor(s). De manière générale, le déplacement d'un drone à voilure tournante est induit par son inclinaison, qui oriente la force de poussée globale du (ou des) rotor(s) dans la direction dans laquelle on souhaite avancer, provoquant alors la translation de l'appareil. Il existe donc un *couplage* entre la dynamique de rotation et la dynamique de translation.

Une grande variété de drones utilisent ce mode de sustentation et de déplacement, allant des hélicoptères classiques similaires à ceux que l'on aperçoit parfois dans le ciel aux hélicoptères miniatures dont la taille atteint quelques centimètres à peine (voir Figure 2.1).



(a) μFR (b) Picoflyer

FIGURE 2.1 – *Les plus petits hélicoptères mesurent seulement quelques centimètres. (a) le μFR (Micro Flying Robot) d'Epson, (b) le Picoflyer développé par Proxflyer.*

Indépendamment de leur taille, ils diffèrent par le nombre de rotors utilisés et la statégie employée pour permettre un vol stable. Nous présentons ici quelques exemples de drones à voilure tournante existants. Pour un panorama plus complet, on pourra se référer par exemple à [Pflimlin 06].

2.1.2 Les systèmes à voilure tournante existants

Hélicoptères classiques

Les hélicoptères « classiques » (voir Figure 2.2) utilisent un rotor à axe vertical. Ce rotor est une pièce mécanique complexe, qui assure la portance et le déplacement du véhicule. Le rotor tourne à vitesse constante, c'est l'*incidence* (ou *le pas*) des pales qui est variable et détermine la direction et l'intensité de la force de poussée. Pour faire avancer l'appareil vers l'avant par exemple, on modifiera l'incidence des pales de manière à ce que leur portance soit maximale lorsqu'elles passent à l'arrière de l'appareil, et minimale à leur passage à l'avant. On parle alors de commande de pas cyclique. Pour prendre de l'altitude ou descendre, on diminuera ou on augmentera globalement l'incidence de toutes les pales (commande de pas général), faisant ainsi varier l'intensité de la force de poussée.

Ce rotor n'est cependant pas suffisant pour stabiliser un hélicoptère. En effet, la rotation des pales induit un couple de réaction qui tend à faire tourner l'appareil sur lui-même dans le sens inverse de la rotation du rotor. Les hélicoptères sont donc également équipés d'un système *anti-couple* ayant pour rôle de contrer cet effet indésirable. La technique la plus courante consiste à utiliser un rotor secondaire plus petit situé à la queue de l'appareil et dont l'axe est horizontal comme c'est le cas pour les exemples de la Figure 2.2. Ce rotor permet également à l'hélicoptère de changer de direction (ou de *cap*). Pour des notions plus avancées sur le fonctionnement des hélicoptères on pourra se référer à [Raletz 10].

Hélicoptères à rotors contrarotatifs

Une deuxième classe d'hélicoptères utilise des rotors contrarotatifs, c'est-à-dire tournant en sens inverse, afin d'annuler l'effet du couple de réaction tout en participant à la sustentation du véhicule. Leur mécanique est plus complexe que celle des hélicoptères à un rotor principal, mais ils permettent de faire l'économie d'un système anti-couple dédié et ont généralement un vol stationnaire plus stable. On trouve notamment dans cette catégorie des hélicoptères à rotors coaxiaux ou des hélicoptères à rotors en tandem (voir Figure 2.3).

Drones multi-rotors à pas fixe

Une autre catégorie de drones à voilure tournante est constituée par des véhicules à plusieurs rotors entraînant des pales à pas constant. La stabilisation et le contrôle du véhicule ne sont alors plus générés par l'incidence des pales mais par la vitesse de rotation des différents rotors. Le mouvement de translation est par exemple obtenu par l'inclinaison de l'appareil en créant une différence de vitesse (et donc une différence de force de poussée) entre les moteurs opposés.

(a)

(b)

(c)

FIGURE 2.2 – *Exemples de drones hélicoptères à rotor principal unique. (a) L'hélicoptère Rmax de Yamaha utilisé par l'ONERA dans le cadre du projet ReSSAC. (b) L'hélicoptère Marvin de l'université TUB de Berlin (projet COMETS). (c) Drone Copter commercialisé par la société Survey Copter.*

(a)

(b)

FIGURE 2.3 – *Exemple de drone à rotors contrarotatifs : (a) le drone de la société Infotron. (b) L'hélicoptère à rotors en tandem Bristol T-173 construit en 1952 qui permettait le transport de 10 passagers.*

Sur ce type d'appareils le couple de réaction est annulé en utilisant un nombre pair de rotors tournant en sens inverse deux-à-deux selon la même idée que pour les hélicoptères contrarotatifs à pas variable. L'utilisation de rotors à pas fixe facilite la conception de la mécanique du drone. Leur alimentation est essentiellement électrique, ce type d'énergie étant plus adapté pour créer des variations de vitesse rapides. Les multi-rotors à pas fixe sont donc équipés de batteries et réservés à des structures de taille modeste. Ils sont ainsi particulièrement adaptés pour évoluer dans des espaces réduits et pour des missions d'inspection de précision (intérieur de bâtiments, etc.). L'envergure des pales étant plus faible, ils ont un caractère moins dangereux et sont plus faciles à caréner.

Parmi les configurations existantes la plus répandue est celle des quadrirotors (voir Figure 2.4). Également appelés *X4-flyers*, les quadrirotors disposent de 4 rotors en opposition disposés à chaque extrémité d'une structure en forme de croix. On trouve une grande variété d'autres configurations possibles, telles que des structures à 6 ou 8 branches (voir Figure 2.5 b et c). Le drone AR70 de la société AirRobot® (voir Figure 2.5 a) utilise 6 rotors, sur une structure à 3 branches, chaque branche étant équipée de 2 moteurs en opposition. Une telle configuration offre un encombrement réduit, et la redondance obtenue permet au drone de supporter la panne d'un de ses moteurs.

On peut citer également le cas des drones à pas fixe qui utilisent des volets mobiles pour orienter la force de poussée de leurs rotors comme le drone caréné HoverEye de Bertin Technologies (voir Figure 2.6).

(a) (b)

FIGURE 2.4 – *Exemples de quadrirotors. (a) Le drone commercialisé par la société Parrot, (b) le drone de la société Novadem lors d'une inspection d'ouvrage d'art.*

2.2 L'utilisation de la vision dans la commande de drones

Les drones présentés dans la section précédente sont des systèmes sous-actionnés [2], non linéaires et qui peuvent être soumis à des perturbations importantes (comme

2. Un système est dit *sous-actionné* lorsqu'il possède moins de degrés de commande que de degrés de liberté.

(a) AR70 (b) Hexacopter (c) Octocopter

FIGURE 2.5 – *Drones à rotors multiples.*

FIGURE 2.6 – *Drone caréné HoverEye de la société Bertin Technologies.*

le vent par exemple). Ces éléments sont autant de difficultés qui rendent leur commande complexe. De nos jours, ces appareils sont encore principalement pilotés par des opérateurs expérimentés, et les rendre facilement utilisables par des utilisateurs non avertis demande un certain degré d'automatisation. Ils sont pour cela équipés de capteurs destinés à leur permettre de se localiser, de naviguer, de percevoir leur propre déplacement ou encore de percevoir la présence d'obstacles : outre la centrale inertielle quasi-incontournable, le GPS (*Global Positioning System*) est couramment utilisé pour connaître la position absolue de l'appareil et effectuer des tâches de navigation en extérieur. Il n'est cependant d'aucune aide pour se positionner par rapport à un objet d'intérêt, éventuellement mobile, dont la position est *a priori* inconnue, ou encore pour s'adapter à la présence d'obstacles. De plus, dans le cadre d'une application en environnement urbain encombré ou à l'intérieur de bâtiments, le signal GPS n'est pas toujours fiable ni même disponible. D'autres capteurs peuvent être envisagés, tels que les capteurs télémétriques (radar, laser ou ultrasons). Ils entrainent cependant certaines contraintes en termes de poids et de coût. À cet égard, l'information fournie par une caméra s'avère un choix pertinent. La caméra est en effet un capteur passif qui fournit une information riche relative à l'environnement observé, et son faible poids la rend parfaitement adaptée pour l'embarqué, même sur des drones de faible envergure.

L'utilisation d'informations visuelles dans la commande d'un système, ou *asservissement visuel*, est un domaine de recherche très actif de la robotique. Son champ d'application couvre des domaines très variés, tels que la micromanipulation [Feddema 98][Tamadazte 09], la robotique médicale [Mebarki 10] [Bachta 10]

et bien sûr la robotique mobile : véhicules terrestres [Masutani 94][Mariottini 07] [Courbon 08], sous-marins [Rives 97] [Lots 01] et, dans le cas qui nous intéresse, aériens. Cette section présente les notions clés de l'asservissement visuel.

2.2.1 L'asservissement visuel : notions élémentaires

L'objectif de l'asservissement visuel est de contrôler un système dynamique (un robot, un drone, etc.) à l'aide d'informations extraites des images acquises par une ou plusieurs caméra(s). Il existe une littérature abondante sur le sujet et on pourra se référer par exemple à [Chaumette 06] et [Chaumette 07] pour des tutoriaux récents qui abordent à la fois les fondamentaux et des notions plus avancées de l'asservissement visuel. On se contente ici de rappeler quelques notions fondamentales en rapport avec les techniques qui ont pu être appliquées à la commande de véhicules aériens. En particulier, on considère ici le cas où la caméra est embarquée sur le véhicule. L'usage d'une caméra déportée est en effet très contraignant et ne va pas de pair avec une application pratique.

De manière générale on définit une tâche d'asservissement visuel sous la forme d'une erreur à réguler entre des primitives visuelles courantes $\mathbf{s}$ et désirées $\mathbf{s}^*$:

$$\boldsymbol{\varepsilon} = \mathbf{s} - \mathbf{s}^*. \tag{2.1}$$

Ces informations visuelles peuvent être de différentes natures comme nous le verrons dans la section 2.2.2.

Notons tout d'abord que ces informations visuelles peuvent intervenir à différents niveaux dans la commande d'un robot. On distingue ainsi classiquement les asservissements visuels directs et indirects :

- dans l'asservissement visuel direct les informations visuelles sont utilisées pour asservir le robot au bas-niveau. Le correcteur de l'asservissement visuel communique directement avec les actionneurs du robot(voir Figure 2.7). Cette approche nécessite une cadence image élevée ;

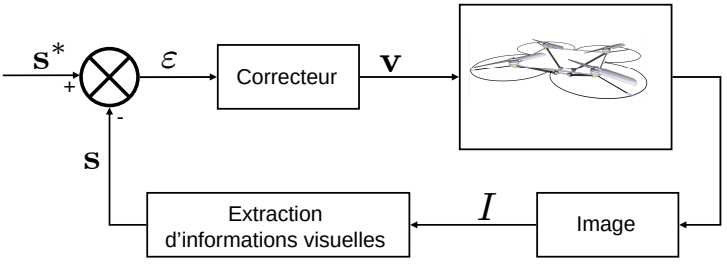


FIGURE 2.7 – *Schéma-bloc de l'asservissement visuel direct.*

- dans le cas de l'asservissement visuel indirect, on suppose que la dynamique du robot est déjà régulée en interne à une fréquence élevée. L'asservissement visuel joue alors le rôle d'un correcteur de plus haut niveau qui envoie des consignes à un contrôleur interne à une cadence image moins élevée, généralement de l'ordre de quelques dizaines de Hertz (voir Figure 2.8).

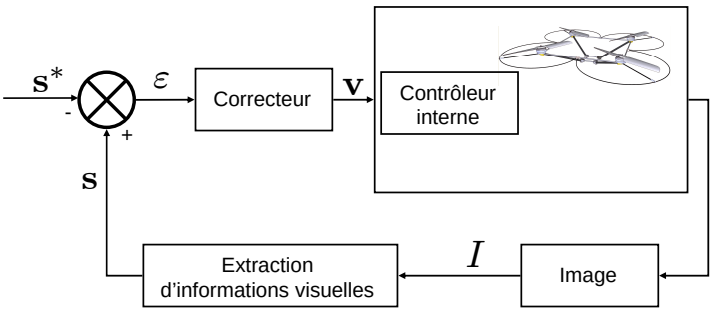


FIGURE 2.8 – *Schéma-bloc de l'asservissement visuel indirect.*

Dans le cas de l'asservissement indirect l'approche la plus classique consiste à définir le lien entre le torseur cinématique $\mathbf{v}$ de la caméra et la variation des primitives visuelles $\dot{\mathbf{s}}$ à l'aide d'une matrice appelée *matrice d'interaction* $\mathbf{L}$ telle que :

$$\dot{\mathbf{s}} = \mathbf{L}\mathbf{v}. \tag{2.2}$$

Le correcteur le plus simple est un correcteur proportionnel qui assure la décroissance exponentielle de l'erreur :

$$\dot{\boldsymbol{\varepsilon}} = -\lambda\boldsymbol{\varepsilon}, \lambda > 0. \tag{2.3}$$

La consigne de vitesse est alors donnée par :

$$\mathbf{v} = -\lambda\mathbf{L}^{+}\boldsymbol{\varepsilon} \tag{2.4}$$

et $\mathbf{v}$ est régulée par le contrôleur interne du robot. Cette approche indirecte présente l'avantage d'être plus facilement adaptable d'un robot à l'autre. Elle suppose cependant qu'une mesure de vitesse fiable est disponible sur le système considéré. Ce n'est malheureusement pas toujours le cas pour les véhicules aériens, en particulier en intérieur, lorsque le signal GPS n'est pas disponible, ou lorsque les contraintes de charge utile limitent l'équipement en termes de capteurs. Il pourra alors être nécessaire soit d'estimer cette vitesse à l'aide des informations visuelles, soit d'utiliser une information de vitesse relative ou de *flot optique* - basée sur le mouvement apparent dans l'image - [Hérissé 10] pour mettre en place la commande.

2.2.2 Différentes approches pour l'asservissement visuel de drones

Les informations visuelles $\mathbf{s}$ utilisées dans la commande (équation (2.1)) peuvent être de nature variée. Les approches les plus répandues utilisent des informations extraites de l'image pour reconstruire tout ou partie de la pose de la caméra. Les informations 3D ainsi reconstruites servent alors à définir la loi de commande. Une autre approche consiste à exprimer la tâche d'asservissement directement dans l'espace image 2D. Les sections suivantes décrivent l'utilisation de ce type d'approche pour l'asservissement visuel de véhicules aériens.

2.2.2.1 La commande de drones par reconstruction visuelle de la pose

Le principe de l'asservissement visuel 3D consiste à utiliser des informations visuelles extraites des images pour reconstituer les paramètres 3D définissant la position de la caméra par rapport à son environnement. Ces informations sont alors utilisées dans la commande.

Cependant, comme nous l'avons vu dans le chapitre 1, la caméra ne fournit qu'une projection 2D de la scène observée. Pour estimer la pose complète de la caméra, une possibilité consiste à utiliser plusieurs caméras. On peut citer les travaux de [Altug 02] [Altug 03] qui utilisent une caméra embarquée et une caméra déportée pour estimer la pose d'un quadrirotor. La caméra déportée de type PTZ (*Pan-Tilt-Zoom*) suit des taches de couleur placées sous le quadrirotor. Plus récemment [Klose 10] a proposé une approche sans marqueur pour suivre la pose d'un quadrirotor à partir de son modèle CAO, en minimisant l'erreur de reprojection des arêtes de ce modèle dans les images acquises par deux caméras déportées. [Amidi 99] [Rock 98] [Mejías 07] utilisent plus classiquement une paire de caméras embarquées pour bénéficier d'une vision stéréoscopique. Dans [Amidi 99] [Mejías 07] la vision stéréoscopique est utilisée pour réaliser un odomètre visuel. [Amidi 99] utilise également des données inertielles pour aider à éliminer les ambiguïtés entre les mouvements de rotation et ceux de translation qui ont des effets proches dans le plan image. Dans [Rock 98] un objet de couleur unique connue est détecté et mis en correspondance dans l'image de chacune des deux caméras, et une simple triangulation permet de déterminer sa position.

L'utilisation d'une paire de caméras est cependant contraignante en termes de charge à embarquer et ces approches nécessitent la connaissance précise de la position relative entre les deux caméras.

Une autre approche consiste à n'estimer qu'une partie de la pose à l'aide de la vision. [Demonceaux 07] estime ainsi l'orientation d'un drone par rapport à l'horizon à l'aide d'une caméra omnidirectionnelle. D'autres capteurs peuvent être utilisés pour compléter l'estimation de la pose.

Enfin, une approche possible réside dans l'utilisation d'une connaissance *a priori* sur l'environnement. [Shakernia 99] [Sharp 01] [Saripalli 03] [Saripalli 07] utilisent un hélipad de forme et dimensions connues pour reconstituer la pose de la caméra. Cette information est alors utilisée pour assurer l'atterrissage d'un hélicoptère. [Kemp 05] propose d'estimer la pose à l'aide d'un modèle de l'environnement, pour permettre à un quadrirotor de se localiser uniquement par la vision. Plus récemment, [Artieda 09] décrit un algorithme de SLAM (*Simultaneous Localization And Mapping*) validé sur des séquences d'images acquises depuis un hélicoptère. Ce type d'approches permet de reconstruire une pose 3D sans connaissance *a priori* sur l'environnement. Cependant l'information reconstruite tend à dériver au cours du temps.

Dans [Courbon 10] la navigation d'un drone quadrirotor utilise une mémoire visuelle sous la forme d'images-clés pour reproduire de façon automatique un trajet déjà effectué. La pose de la caméra est alors reconstruite à un facteur d'échelle près à partir de la transformation homographique entre les images.

2.2.2.2 La commande de drones à partir d'informations visuelles 2D

Une autre façon de considérer le problème de l'asservissement visuel consiste à exprimer la tâche (2.1) directement dans l'espace image. L'erreur à réguler est alors une erreur entre des informations visuelles 2D $\mathbf{s}$ directement issues de l'image. Un avantage majeur de cette formulation est qu'elle se passe de l'étape de reconstruction 3D.

Le choix des informations visuelles utilisées dépend naturellement des caractéristiques de la scène observée. Des structures linéaires ont été utilisées pour modéliser les lignes de pistes d'atterrissage d'avions [Rives 02] ou la ligne d'horizon [Rives 04]. Dans [Zhang 99] un dirigeable est asservi sur une balle de couleur à partir de sa position dans l'image. [Mejías 06] propose d'asservir un hélicoptère par rapport à une fenêtre à partir de la position de celle-ci dans l'image acquise par une caméra frontale. Le contrôle concerne uniquement l'altitude et le mouvement latéral afin que la fenêtre soit centrée dans l'image.

On observe toutefois qu'assurer la bonne réalisation d'une tâche exprimée dans l'espace 2D du capteur (la caméra) ne permet pas toujours d'assurer un comportement satisfaisant pour le robot dans l'espace 3D, en particulier lorsqu'il existe un fort couplage entre les différentes primitives visuelles. La présence d'un tel couplage se traduit par le fait que la matrice d'interaction $\mathbf{L}$ (voir équation (2.2)) n'est pas diagonale, ce qui est malheureusement le cas le plus fréquent. L'objectif est alors de choisir les informations visuelles pour avoir un découplage maximal. Dans [Guenard 08] l'information visuelle utilisée est le vecteur moyen des coordonnées sphériques d'un ensemble de points de l'image (centroïde). [Bourquardez 09] propose d'utiliser des moments sphériques d'ordre 1 pour la stabilisation d'un quadrirotor au-dessus d'une cible. [Hrabar 03] utilise comme information visuelle la centroïde obtenue à partir des images d'une caméra omnidirectionnelle.

On peut citer également les approches basées sur le mouvement [Crétual 01], qui utilisent directement le flot optique mesuré dans l'image pour la commande de drones [Hérissé 10]. Inspirées de la navigation de certains animaux (les abeilles notamment) ces travaux proposent de réguler directement la vitesse apparente dans l'image (la « vitesse de défilement » des informations visuelles) pour se maintenir à une distance constante d'un plan, ou encore attérir sans connaître la distance entre le drone et le sol.

2.2.3 L'extraction d'informations visuelles depuis un drone

Lorsque l'on souhaite utiliser des informations visuelles dans un but de commande, comme c'est le cas des travaux présentés dans ce chapitre, il est crucial que ces informations puissent être extraites de façon robuste. Qu'il s'agisse du suivi d'une cible, de la reconstruction de la pose, ou encore d'une estimation de mouvement, l'extraction d'informations visuelles fiables est soumise à des contraintes particulièrement fortes lorsqu'elle est réalisée sur un drone. On observe notamment :

- des transformations importantes entre deux images consécutives dues à des mouvements rapides ;
- des images particulièrement bruitées (voir Figure 2.9) générées occasionnellement par des erreurs de transmission, lorsque les calculs sont déportés sur une

station au sol (c'est encore souvent le cas pour des raisons de puissance de calcul limitée en embarqué) ;
- des occultations qui peuvent survenir notamment dans des environnements encombrés.

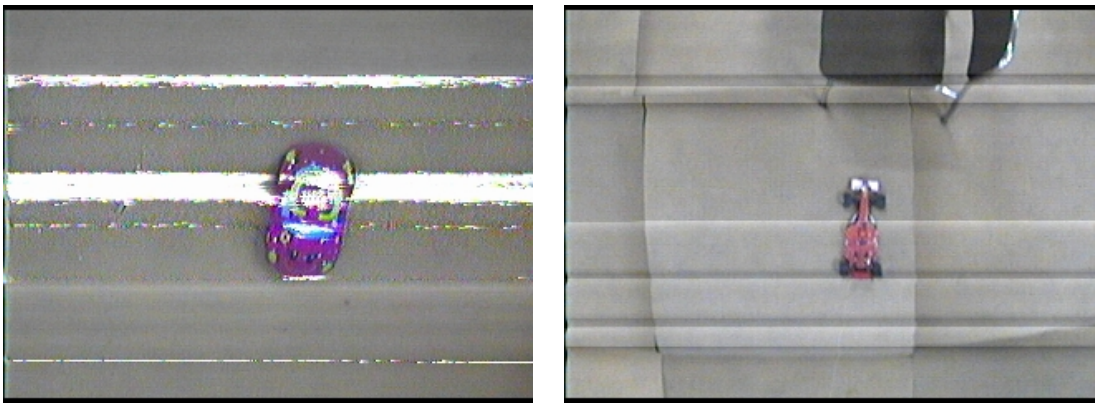


FIGURE 2.9 – *Exemples d'images bruitées en provenance d'un mini-drone.*

Les travaux qui utilisent la vision pour la commande de drones se placent généralement dans des conditions particulièrement avantageuses pour assurer le succès de l'étape d'extraction. En particulier la zone d'atterrissage ou la cible à suivre sont supposées toujours visibles [Saripalli 07] [Watanabe 10]. À un autre niveau, le dispositif proposé dans [Michael 10] utilise un système de vision déporté VICON [3], qui fournit la position de marqueurs avec une précision de 50 microns à une cadence de 375Hz.

2.3 Conclusion

Dans ce chapitre nous avons introduit la problématique de la commande de drones par la vision. Après avoir dressé un rapide panorama des drones à voilure tournante, capables de vol stationnaire, et de leur mode de fonctionnement, nous avons décrit les notions fondamentales de l'asservissement visuel en orientant notre propos sur les approches déjà proposées sur ce type d'appareils. Ce chapitre a mis en évidence les difficultés inhérentes à l'utilisation de drones, aussi bien en termes de commande (nécessité d'estimer également la vitesse de l'engin, couplage entre les différents degrés de liberté) qu'en termes d'extraction d'informations visuelles robuste.

Les travaux que nous présentons dans la suite de ce manuscrit s'attachent à proposer des approches répondant aux difficultés rencontrées en situation réelle en particulier concernant l'extraction d'informations visuelles.

Dans cette thèse nous avons considéré deux types de tâches :
- une tâche de poursuite, dans laquelle on demande à un drone de suivre une cible mobile (en l'occurrence un véhicule) dans un environnement inconnu,

3. http ://www.vicon.com

– une tâche de positionnement et navigation par points de passage dans un environnement structuré.

Les deux parties suivantes s'articulent naturellement autour de ces deux tâches.

Première partie

Poursuite

La première tâche que nous avons considérée dans cette thèse est une tâche de poursuite. On s'intéresse ainsi à une situation dans laquelle un objet se déplace et l'on souhaite qu'un drone le suive à l'aide de données visuelles. Ce type de situation concerne aussi bien des applications de sécurité civile telles que la poursuite d'un véhicule [Watanabe 10] ou d'un convoi, que des applications ludiques telles qu'on peut les imaginer avec l'arrivée sur le marché grand public de mini-drones pilotables depuis un téléphone portable [4].

Dans tous les cas, la tâche de poursuite peut se décomposer en deux parties principales. D'une part, il s'agit pour le drone de localiser l'objet qu'il doit poursuivre à partir des images qu'il acquiert : c'est l'objet du chapitre 3. D'autre part, il doit être capable d'utiliser ces informations pour se déplacer et poursuivre la cible. Cet aspect sera abordé dans le chapitre 4.

4. À l'heure où ces lignes sont écrites, des affiches publicitaires pour ce type de mini-drones tapissent en effet les couloirs de métro parisiens.

Chapitre 3

Suivi visuel d'un objet mobile

Pour le type d'applications visé dans le cadre de cette thèse, à savoir l'utilisation d'informations visuelles pour la commande d'un mini-drone, nous nous sommes intéressés dans un premier temps à la possibilité de suivre un objet mobile dans un environnement inconnu. En effet, pour des tâches d'inspection ou de surveillance, notamment en extérieur, on ne dispose pas toujours d'un modèle de l'environnement dans lequel le drone doit évoluer, ni des objets que l'on va y rencontrer.

Les conditions considérées dans ce chapitre sont donc les suivantes :

- l'environnement est supposé inconnu,
- on ne dispose pas de modèle préalable de l'objet,
- la caméra et l'objet sont susceptibles de se déplacer et ces déplacements peuvent être très importants entre deux acquisitions d'images successives,
- le suivi doit être robuste à d'éventuelles occultations, et au bruit provoqué par des erreurs passagères de transmission (voir la section 2.2.3),
- le suivi doit être temps réel.

Il doit permettre au drone de se placer au-dessus de la cible et de réaliser des mouvements similaires dans un plan situé à une altitude fixée.

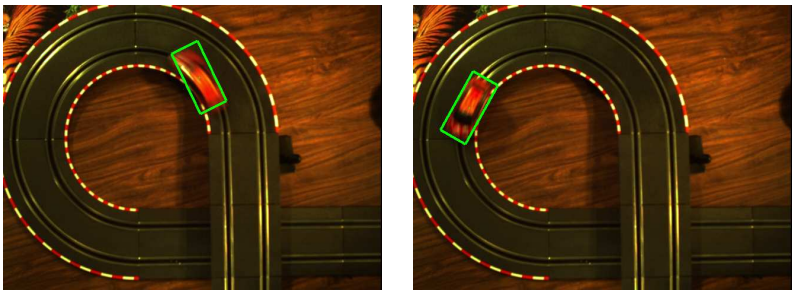


FIGURE *3.1 – Suivi d'objet mobile.*

Ce chapitre s'organise comme suit : la section 3.1 justifie le choix d'une description non paramétrique de l'objet sous forme d'histogrammes de couleur et présente un état de l'art des principales méthodes de suivi basé couleur. La section 3.2 propose

une configuration multi-noyaux et décrit les méthodes de suivi qui en découlent, à la fois dans un cadre déterministe et dans une approche par filtrage particulaire, pour le suivi de la position, l'orientation et la taille de l'objet dans l'image. Une approche hybride entre ces deux méthodes est proposée dans la section 3.3. Les questions d'initialisation et de réinitialisation sont discutées dans la section 3.4. Enfin, la section 3.5 présente des résultats expérimentaux comparatifs entre les approches introduites dans ce chapitre.

3.1 Suivi basé couleur

Étant données les contraintes spécifiques auxquelles nous souhaitons répondre, nous nous sommes orientés dans un premier temps vers des méthodes de suivi basées sur une représentation par histogrammes. Le choix de ce type de descripteur est justifié plus précisément dans le paragraphe suivant, et ses caractéristiques y sont décrites. Un état de l'art des principales méthodes de suivi utilisant ce type de descripteur est présenté section 3.1.2.

3.1.1 Représentation par histogrammes

Motivation. Dans la section 1.3, nous avons vu qu'il existe un large éventail de descripteurs possibles pour caractériser l'apparence d'un objet à suivre. L'une des représentations les plus simples et largement répandues consiste à utiliser directement l'imagette constituée par l'ensemble des pixels de l'objet [Hager 98] [Irani 98a] [Jurie 02] [Malis 07] [Dame 10]. Le modèle de l'objet est alors composé de la valeur de l'intensité lumineuse ou de la couleur de ces pixels et de leur position dans l'image. Cette description au niveau pixellique est précise et permet de détecter une grande variété de transformations du plan image (voir section 1.2.4) en utilisant des fonctions de similarité telles que la SSD [Hager 98] [Jurie 02] [Malis 07] (voir la Figure 3.2), des fonctions de corrélation [Irani 98a], ou encore l'information mutuelle dérivée de l'entropie de l'image [Dowson 08] [Dame 10].

Cependant, ce type de descripteur est parfois sensible aux modifications de l'apparence de l'objet, qu'elles soient dues à des changements d'illumination, des occultations partielles ou des angles de vue différents. Ces changements d'apparence peuvent alors mettre en échec le suivi. La zone de convergence des méthodes utilisant ce type de représentation étant souvent réduite, la position initiale de ces algorithmes doit également être proche de la position réelle de l'objet. Cette zone de convergence dépend cependant de plusieurs facteurs, dont le modèle de mouvement considéré et le processus de minimisation choisi.

Des solutions ont été proposées pour mettre à jour le modèle au cours du déplacement [Matthews 03] [Zhou 04], ou faire face aux occultations [Hager 98] [Jepson 03] et aux changements d'angle de vue [Black 96]. Cependant, la connaissance complète du modèle au niveau pixellique n'est pas toujours indispensable à une tâche de suivi : le mode de représentation doit être suffisamment discriminant pour permettre de différencier l'objet du reste de la scène, tout en autorisant une certaine flexibilité vis-à-vis de changements d'apparence.

Exemple de l'erreur quadratique (SSD)

Supposons qu'une zone d'intérêt que l'on cherche à suivre est constituée de n pixels $\{\boldsymbol{l}_i\}_{i=1..n}$ dans une image de référence I^*. Soient $\boldsymbol{\mu}_k$ les paramètres de la transformation $tr_{\boldsymbol{\mu}_k}$ qui permet de passer de l'image I^* à une image I_k. L'hypothèse de la conservation lumineuse [Horn 87] s'écrit alors :

$$\forall i, I^*(\boldsymbol{l}_i) = I_k(tr_{\boldsymbol{\mu}_k}(\boldsymbol{l}_i)) \tag{3.1}$$

où $I_k(\boldsymbol{l})$ désigne l'intensité lumineuse de pixel $\boldsymbol{l}$ dans l'image I_k.
Ici, on considère que l'espace des mesures $\mathcal{F}$ est l'espace de dimension n des vecteurs d'intensités lumineuses. En reprenant les notations de la section 1.3.3 on rappelle ici l'équation (1.26) :

$$\boldsymbol{\mu}_k = arg\min_{\boldsymbol{\mu}} \|g(\boldsymbol{\mu}) - \mathbf{b}\|^2. \tag{3.2}$$

Dans notre exemple :

$$\forall \boldsymbol{\mu}, g(\boldsymbol{\mu}) = (I(tr_{\boldsymbol{\mu}}(\boldsymbol{l}_1)), ..., I(tr_{\boldsymbol{\mu}}(\boldsymbol{l}_n)))$$

et

$$\mathbf{b} = (I^*(\boldsymbol{l}_1), ..., I^*(\boldsymbol{l}_n)).$$

L'équation (1.26) peut donc s'écrire en utilisant la norme 2 :

$$\boldsymbol{\mu}_k = arg\min_{\boldsymbol{\mu}} \sum_{i=1}^{n} (I(tr_{\boldsymbol{\mu}}(\boldsymbol{l}_i)) - I^*(\boldsymbol{l}_i))^2. \tag{3.3}$$

Cette erreur quadratique entre les valeurs des intensités lumineuses est une fonction de coût classique appelée SSD [Lucas 81] [Tomasi 91] [Hager 98] [Benhimane 04]. La figure ci-dessous est un exemple de l'allure de la SSD, où le minimum à atteindre est très marqué.

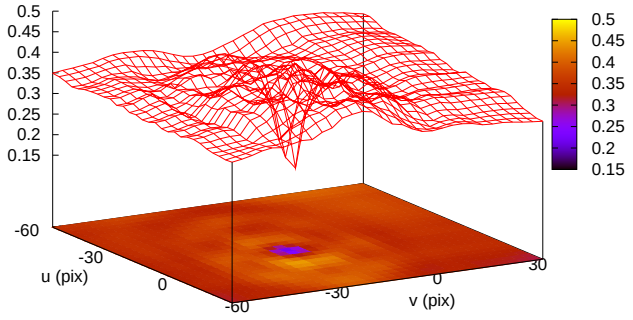


FIGURE 3.2 – *Exemple de fonction de coût classique : la SSD (Sum of Squared Differences).*

Dans cette optique, une alternative intéressante consiste à utiliser une description plus globale de l'objet, en considérant la densité de probabilité associée à une information visuelle. Soit $\mathcal{F}$ l'espace des informations visuelles considérées, $\mathcal{F}$ peut désigner l'espace des couleurs, des intensités lumineuses ou encore des gradients. Une représentation par densité de probabilité associe alors à chaque information visuelle dans $\mathcal{F}$ sa probabilité. Ces représentations incluent des densités de probabilité paramétriques, où l'apparence de l'objet est caractérisée typiquement sous forme d'une combinaison de fonctions gaussiennes [Stauffer 99], ou plus généralement, des densités non paramétriques, telles que des histogrammes.

Dans le cadre du suivi d'objet, les histogrammes de couleur (voir Figure 3.3) sont particulièrement populaires [Comaniciu 03], [Pérez 02], [Nummiaro 03], [Hager 04], [Maggio 05b]. La couleur est une information intuitive pour l'homme lorsqu'il s'agit de décrire un objet, et accessible directement dans l'image sans besoin de traitement d'image (contrairement au gradient par exemple). Dans le cadre de cette thèse les expériences de suivi d'objet basées sur des histogrammes que nous considèrerons utilisent ainsi une information de couleur ($\mathcal{F}$ désigne alors l'espace des couleurs) mais on gardera en mémoire que les descripteurs sous forme de densité de probabilité peuvent utiliser différents types d'informations visuelles [Maggio 07] [Jeyakar 08].

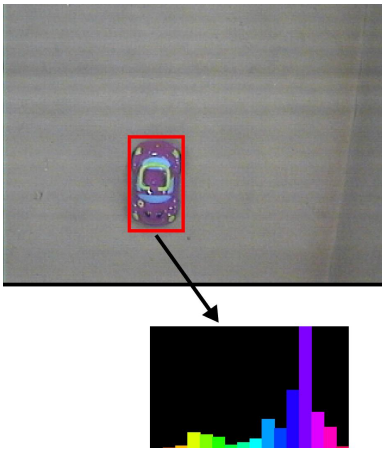


FIGURE 3.3 – *Représentation d'un objet par un histogramme de couleur.*

Dans une représentation sous forme d'histogramme, l'espace de mesure $\mathcal{F}$ est divisé en un nombre fini m de classes (*bins* dans la littérature anglophone). Un histogramme $\mathbf{q}$ stocke pour chaque classe u sa probabilité d'apparition q_u, calculée sur l'ensemble des pixels de l'objet. Supposons que l'objet est constitué de n pixels $\{\boldsymbol{l}_i\}_{i=1..n}$ dans une image I donnée. Soit $\mathbf{q} = \{q_u\}_{u=1...m}$ son histogramme. Chaque classe u s'exprime sous la forme :

$$q_u = \frac{1}{n} \sum_{i=1}^{n} \delta_u(b_I(\boldsymbol{l}_i)) \tag{3.4}$$

où $b_I(\boldsymbol{l}_i)$ désigne la classe à laquelle appartient la couleur du pixel $\boldsymbol{l}_i$ dans I, et δ est la fonction de Kronecker définie par :

$$\delta_a(b) = \begin{cases} 1 \text{ si } a = b \\ 0 \text{ sinon.} \end{cases} \tag{3.5}$$

Le caractère global des descripteurs par densité de probabilité, tels que les histogrammes de couleur, les rend peu sensibles à des changements locaux de l'apparence de l'objet. Le flou d'une image par exemple modifiera localement l'intensité des pixels qui appartiennent à un objet, mais modifiera peu son histogramme global. De plus, les fonctions de similarité utilisant ces descripteurs ont une zone de convergence plus étendue : la position optimisant la fonction de similarité pourra donc être atteinte à partir d'une position éloignée. Avoir une grande zone de convergence autorise de plus grands déplacements entre deux images successives. La Figure 3.4 présente ainsi l'allure d'une fonction de similarité basée sur une représentation par histogramme et une fonction de similarité basée sur une représentation sous forme d'imagette (la SSD, voir Figure 3.2) pour une même image de référence. Dans le cas de la SSD la position recherchée correspond à un pic étroit de la fonction de similarité, tandis que la représentation par histogramme permet de converger depuis une position initiale beaucoup plus éloignée. La fonction de similarité utilisée dans le cas de la représentation par histogramme est ici la distance de Hellinger [Kanazawa 93], dont l'expression sera définie dans le paragraphe suivant (3.8).

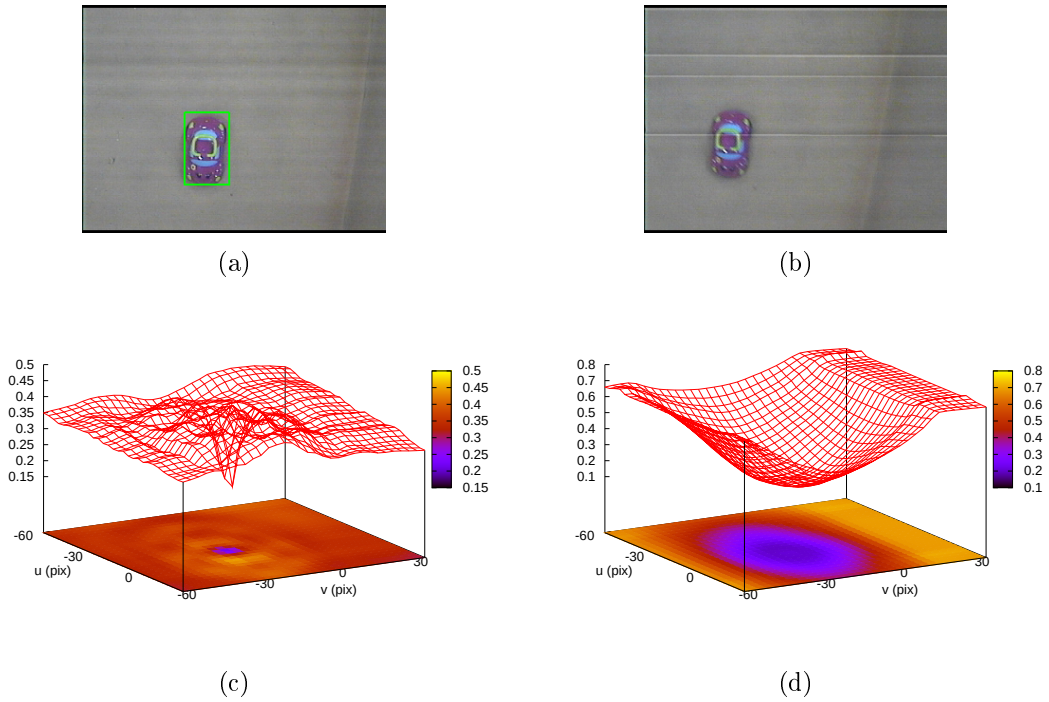


FIGURE 3.4 – *Comparaison entre une fonction de similarité basée sur l'intensité de chaque pixel, ici la SSD (c) et une fonction de coût basée sur une représentation par histogramme de couleur (d). Dans les deux cas l'objet de référence est celui de la Figure (a), et la fonction de similarité est calculée dans l'image (b). La zone de convergence est beaucoup plus grande dans le cas de la représentation par histogramme.*

Critères de similarité dans l'espace des histogrammes. Le principe d'un algorithme de suivi basé sur une représentation par histogramme est de déterminer dans chaque image quelle est la position de l'objet pour laquelle l'histogramme « ressemble » le plus à un histogramme de référence $\mathbf{q}^*$. L'hypothèse de conservation

des intensités lumineuses de l'équation (3.1) se traduit ici par une hypothèse de conservation des histogrammes :

$$\mathbf{q}^*\left(\{\boldsymbol{l}_i\}_{i=1..n}\right) = \mathbf{q}(\{tr_{\boldsymbol{\mu}}(\boldsymbol{l}_i)\}_{i=1..n}) \tag{3.6}$$

où selon les notations de la section 1.2.4 la fonction $tr_{\boldsymbol{\mu}}(.)$ désigne la transformation de paramètres $\boldsymbol{\mu}$ appliquée aux pixels pour passer de l'image de référence I^* à l'image I. Dans cette expression, $\mathbf{q}^*(\{\boldsymbol{l}_i\}_{i=1..n})$ désigne donc l'histogramme calculé à partir des pixels $\{\boldsymbol{l}_i\}_{i=1..n}$ de l'objet dans l'image I^*, et $\mathbf{q}(\{tr_{\boldsymbol{\mu}}(\boldsymbol{l}_i)\}_{i=1..n})$ désigne l'histogramme calculé à partir des pixels $\{tr_{\boldsymbol{\mu}}(\boldsymbol{l}_i)\}_{i=1..n}$ de l'objet dans une image I.

Plusieurs critères peuvent être utilisés pour évaluer la similarité entre deux histogrammes [Cha 08]. Le plus courant repose sur le coefficient de corrélation de Bhattacharyya [Bhattacharyya 43][Comaniciu 00] :

$$\rho(\mathbf{q}^*, \mathbf{q})) = \sum_{u=1}^{m} \sqrt{q_u^* q_u}. \tag{3.7}$$

Ce coefficient permet de définir des distances dans l'espace des histogrammes, telles que *la distance de Hellinger* [Kanazawa 93]

$$d_{\text{Hellinger}}(\mathbf{q}^*, \mathbf{q}) = \sqrt{1 - \rho(\mathbf{q}^*, \mathbf{q})} \tag{3.8}$$

ou *la distance de Matusita*

$$d_{\text{Matusita}}(\mathbf{q}^*, \mathbf{q}) = \|\sqrt{\mathbf{q}^*} - \sqrt{\mathbf{q}}\| \tag{3.9}$$

où la notation $\sqrt{\mathbf{q}}$ signifie que l'on applique la racine carrée à chaque classe de l'histogramme. Ces distances sont liées par la relation :

$$d^2_{\text{Matusita}}(\mathbf{q}^*, \mathbf{q}) = 2 - 2\rho(\mathbf{q}^*, \mathbf{q}) = 2d^2_{\text{Hellinger}}(\mathbf{q}^*, \mathbf{q}). \tag{3.10}$$

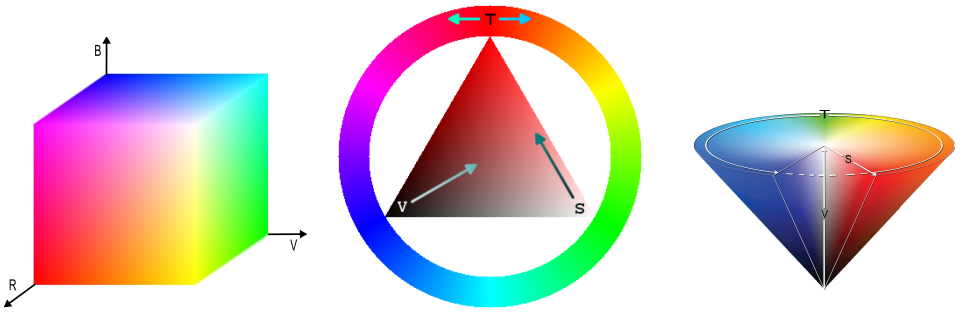
D'autres critères de similarité peuvent être utilisés, tels que *la divergence de Kullback-Leibler* [Chen 01], également appelée *entropie relative* et définie par :

$$\text{KL}(\mathbf{q}^*, \mathbf{q}) = \sum_{u=1}^{m} q_u \log\left(\frac{q_u}{q_u^*}\right). \tag{3.11}$$

Cette fonction est une mesure à valeurs dans $[0, 1]$ mais ce n'est pas une distance ($\text{KL}(\mathbf{q}^*, \mathbf{q}) \neq \text{KL}(\mathbf{q}, \mathbf{q}^*)$) et elle n'est pas définie pour les classes vides de l'histogramme de référence et non vides dans l'histogramme candidat.
Un algorithme de suivi basé couleur s'attachera donc à trouver la transformation 2D $tr_{\boldsymbol{\mu}}(.)$ dans l'équation 3.6 qui optimise l'un de ces critères de similarité.

Espaces de couleur. Un espace colorimétrique ou espace de couleur permet de représenter une couleur visible en lui associant des valeurs numériques. La vision humaine possède trois types de récepteurs de couleurs, les cônes, sensibles aux longueurs d'ondes du rouge, du vert et du bleu. Les espaces colorimétriques sont donc généralement tridimensionnels. Parmi les espaces de couleur les plus utilisés on peut citer notamment :

- l'espace RVB (RGB en anglais), qui décompose les couleurs selon leurs composantes de rouge, vert et bleu (voir Figure 3.5 (a)). C'est la représentation la plus utilisée car la plus directement accessible [Comaniciu 00] [Maggio 05b] [Nummiaro 03] [Babu 07] ;
- l'espace TSV (HSV en anglais), qui offre une représentation plus intuitive en séparant la teinte, la saturation et la valeur. La teinte représente la couleur au sens commun du terme, que l'on peut visualiser sur le cercle chromatique (Figure 3.5 (b)). La saturation correspond à la pureté de la couleur, et la valeur ou intensité représente la quantité de noir. L'espace TSV est utilisé notamment lorsque la zone d'intérêt du suivi possède une teinte connue dont l'intensité et la saturation peuvent varier (détection de visages par exemple) ou pour limiter l'impact des variations d'illumination en réduisant le nombre de classes correspondant à la composante V dans l'histogramme [Bradski 98] [Pérez 02] ;
- les espaces XYZ, xyY, YUV et L*a*b* définis par souci de normalisation par la Commission Internationale des Éclairages (CIE) séparent luminance et chrominance. L*a*b* a été conçu pour que les distances calculées entre couleurs correspondent aux différences perçues par l'œil humain.



(a) (b) (c)

FIGURE 3.5 – *Les espaces de couleur RVB (a) et TSV (b) ou (c).*

Dans la suite de ce document, sauf mention contraire, l'espace de couleur $\mathcal{F}$ que nous utilisons désigne l'espace RVB. Le nombre m de classes de l'histogramme caractérise le sous-échantillonnage de l'espace $\mathcal{F}$ que l'on utilise. Un faible nombre de classes correspond à un sous-échantillonnage important, mais permet de réduire la quantité d'information nécessaire pour représenter l'objet.

La section suivante présente les principales méthodes de suivi possibles à partir d'une représentation de l'objet sous forme d'histogramme de couleurs.

3.1.2 Méthodes de suivi

Algorithme du *Mean Shift* pour le suivi d'objet. Historiquement, le suivi d'objet basé sur des histogrammes de couleurs s'est développé à partir du succès de l'application de l'algorithme du *Mean Shift* au problème du suivi d'objet [Bradski 98] [Comaniciu 00]. L'algorithme du *Mean Shift* est une procédure itérative d'estimation des modes [1] d'une densité de probabilité non paramétrique à partir d'un en-

1. En statistiques un *mode* désigne un maximum global d'une densité de probabilité.

semble d'échantillons. Supposons ainsi que l'on dispose d'un ensemble d'échantillons $\{\mathbf{x}_i\}_{i=1..n}$. La densité de probabilité sous-jacente est alors approchée sous la forme :

$$p(\mathbf{x}) = C \sum_{i=1}^{n} K(\mathbf{x} - \mathbf{x}_i) \tag{3.12}$$

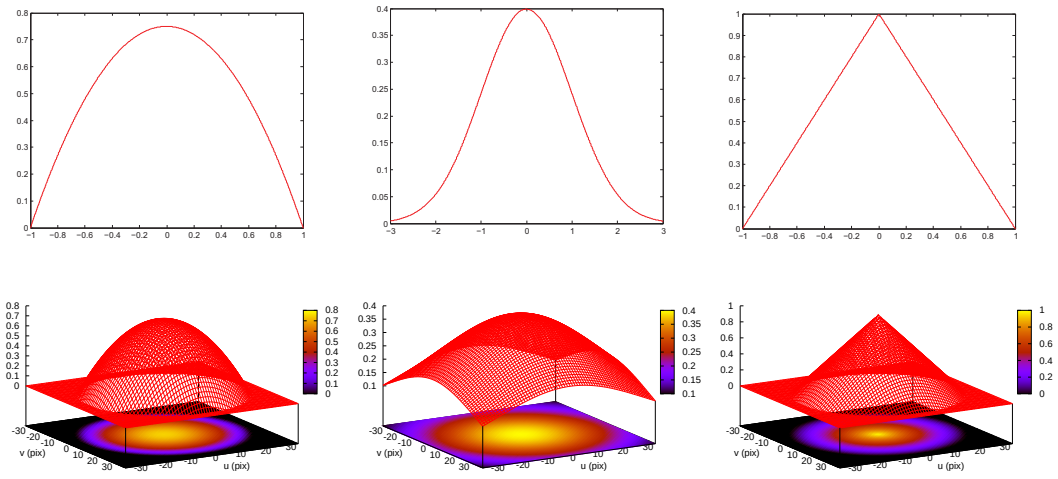
où la fonction K est une fonction dite « noyau » différentiable, qui pondère la contribution de chaque échantillon à la densité de probabilité p, et C une constante de normalisation. Plus généralement, un noyau défini sur un espace $\mathcal{E}$ est une fonction à valeurs réelles positives qui vérifie les propriétés suivantes :

$$\int_{\mathcal{E}} K(\mathbf{x}) d\mathbf{x} = 1 \tag{3.13}$$

$$\forall \mathbf{x} \in \mathcal{E}, K(-\mathbf{x}) = K(\mathbf{x}). \tag{3.14}$$

La Figure 3.6 présente des exemples de fonctions noyaux à symétrie radiale couramment utilisées, ainsi que leur *profil* k, défini par :

$$k : \mathbb{R}^+ \to \mathbb{R}, K(\mathbf{x}) = k(\|\mathbf{x}\|^2)$$



(a) (b) (c)

FIGURE 3.6 – *Exemples de profils de fonctions noyaux : Epanechnikov (a), noyau normal (b) et noyau conique (c) et représentation de leur allure 3D.*

Le principe du *Mean Shift* est similaire à une descente de gradient pour la recherche des modes d'une densité de probabilité de la forme (3.12). Dans cette section on présente l'application de l'algorithme du *Mean Shift* au cas du suivi d'objet [Comaniciu 00]. Dans le cas du suivi basé couleur, la densité non paramétrique dont on recherche les modes découle d'une approximation du coefficient de corrélation de Bhattacharyya (3.7). Considérons un objet ou une zone d'intérêt à suivre, et supposons qu'elle est constituée des pixels $\{\boldsymbol{l}_i\}_{i=1...n}$ dans l'image considérée. Les classes

de l'histogramme correspondant à cette zone de l'image peuvent alors s'écrire :

$$q_u(\mathbf{c}) = \sum_{i=1}^{n} K\left(\frac{\boldsymbol{l}_i - \mathbf{c}}{h}\right) \delta_u \left(b_I(\boldsymbol{l}_i)\right). \tag{3.15}$$

où $K : \mathbb{R}^2 \to \mathbb{R}^+$ est une fonction noyau centrée en $\mathbf{c} \in \mathbb{R}^2$, et h définit son étendue et donc la zone sur laquelle elle agit : dans la Figure 3.6 par exemple on a tracé les représentations 3D de fonctions $K\left(\frac{\mathbf{x}}{h}\right)$ en fonction de $\mathbf{x}$ pour $h = 30$ pix. Sauf mention contraire, K est supposée normalisée de sorte que $\sum_{u=1}^{m} q_u = 1$. L'introduction d'une fonction noyau permet d'appliquer des méthodes différentielles aux méthodes de suivi qui utilisent des représentations sous forme d'histogrammes. D'autre part, les fonctions noyaux généralement utilisées pondèrent la contribution de chaque pixel à l'histogramme de couleur, et permettent ainsi de donner une plus grande importance aux pixels du centre de l'objet, qui sont moins susceptibles d'être occultés.

La recherche de l'objet dans l'image consiste à chercher la position $\mathbf{c}$ qui maximise le coefficient de Bhattacharyya $\rho(\mathbf{c})$ (défini dans l'équation (3.7)) entre l'histogramme courant $\mathbf{q}(\mathbf{c})$ et un histogramme de référence $\mathbf{q}^*$. L'approximation au premier ordre de $\rho(\mathbf{c})$ en $\mathbf{q}(\mathbf{c}_0)$ s'écrit :

$$\rho(\mathbf{c}) = \rho(\mathbf{q}^*, \mathbf{q}(\mathbf{c})) \approx \sum_{u=1}^{m} \sqrt{q_u^* q_u(\mathbf{c}_0)} + \frac{1}{2} q_u(\mathbf{c}) \sqrt{\frac{q_u^*}{q_u(\mathbf{c}_0)}} \left(q_u(\mathbf{c}) - q_u(\mathbf{c}_0)\right) \tag{3.16}$$

$$\approx \frac{1}{2}\left(\sum_{u=1}^{m} \sqrt{q_u^* q_u(\mathbf{c}_0)} + \sum_{u=1}^{m} q_u(\mathbf{c}) \sqrt{\frac{q_u^*}{q_u(\mathbf{c}_0)}}\right) \tag{3.17}$$

Le premier terme de (3.17) étant constant, maximiser $\rho(\mathbf{c})$ équivaut à maximiser le deuxième terme, à savoir :

$$O(\mathbf{c}) = \sum_{u=1}^{m} q_u(\mathbf{c}) \sqrt{\frac{q_u^*}{q_u(\mathbf{c}_0)}} = \sum_{i=1}^{n} w_i K\left(\frac{\mathbf{c} - \boldsymbol{l}_i}{h}\right) \tag{3.18}$$

où

$$w_i = \sum_{u=1}^{m} \delta_u \left(b_I(\boldsymbol{l}_i)\right) \sqrt{\frac{q_u^*}{q_u(\mathbf{c}_0)}} \tag{3.19}$$

L'optimisation de $O(\mathbf{c})$ est alors obtenue dans [Comaniciu 03] en annulant la dérivée de l'expression (3.18).

On suppose en outre que le profil k du noyau K est dérivable par morceaux sur $\mathbb{R}^+$. On obtient alors :

$$\mathbf{c} = \frac{\sum_{i=1}^{n} \boldsymbol{l}_i \; w_i \; g(\|\frac{\mathbf{c}_0 - \boldsymbol{l}_i}{h}\|^2)}{\sum_{i=1}^{n} w_i \; g(\|\frac{\mathbf{c}_0 - \boldsymbol{l}_i}{h}\|^2)} \tag{3.20}$$

où g est définie à partir de la dérivée du profil $\dot{k}$ par $g(\boldsymbol{l}) = -\dot{k}(\boldsymbol{l})$.

Cette solution résulte de l'approximation au premier ordre du coefficient de Bhattacharyya et n'est donc pas une solution exacte. Comme une méthode du gradient classique, l'algorithme du *Mean Shift* procède donc par itérations successives pour approcher le mode de la densité de probabilité. L'algorithme du suivi d'objet par *Mean Shift* [Comaniciu 00] (voir l'algorithme 4) est illustré Figure 3.7.

On remarque dans l'équation (3.20) que la zone dans laquelle l'objet peut être localisé est limitée par la taille du noyau h. Par conséquent, si le déplacement de l'objet entre deux images consécutives est plus important que la taille du noyau, le vecteur (3.20) n'est plus corrélé avec l'objet et le suivi est amené à échouer.

Algorithme de suivi par *Mean Shift*

Soit $\mathbf{q}^* = \{q_u^*\}_{u=1\ldots m}$ l'histogramme de référence de l'objet d'intérêt.

1. Initialiser la position $\mathbf{c}_0$ dans l'image courante, calculer la distribution de probabilité $\{q_u(\mathbf{c}_0)\}_{u=1\ldots m}$, et évaluer

$$\rho(\mathbf{c}_0) = \sum_{u=1}^{m} \sqrt{q_u^* q_u(\mathbf{c}_0)}.$$

2. Calculer les poids w_i selon l'équation (3.19).
3. En déduire la nouvelle position

$$\mathbf{c}_1 = \frac{\sum_{i=1}^{n} \boldsymbol{l}_i w_i g(\|\frac{\mathbf{c}_0 - \boldsymbol{l}_i}{h}\|^2)}{\sum_{i=1}^{n} w_i g(\|\frac{\mathbf{c}_0 - \boldsymbol{l}_i}{h}\|^2)}.$$

   Mettre à jour $\{q_u(\mathbf{c}_1)\}_{u=1\ldots m}$ et évaluer

$$\rho(\mathbf{c}_1) = \sum_{u=1}^{m} \sqrt{q_u^* q_u(\mathbf{c}_1)}.$$

4. Tant que $\rho(\mathbf{c}_1) < \rho(\mathbf{c}_0)$ Faire : $\mathbf{c}_1 \leftarrow \frac{1}{2}(\mathbf{c}_0 + \mathbf{c}_1)$
5. Si $\|\mathbf{c}_1 - \mathbf{c}_0\| < \varepsilon$ arrêter
   Sinon : $\mathbf{c}_0 \leftarrow \mathbf{c}_1$ et retourner à l'étape 1.

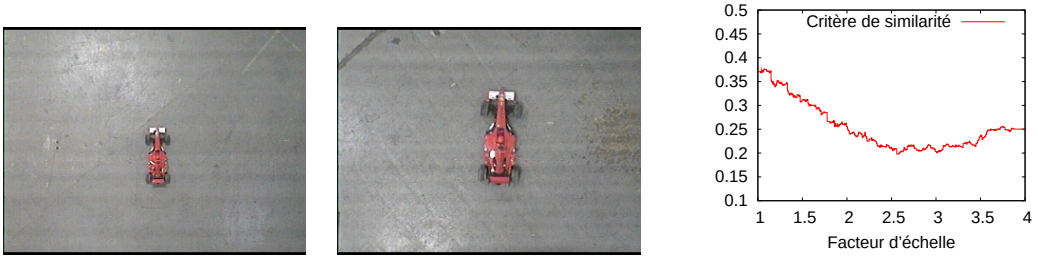
Algorithme 4 – Suivi par Mean Shift.

FIGURE *3.7 – Exemple de suivi par* Mean Shift. *Les différents rectangles représentent les itérations successives, en partant de la position initiale en rouge.*

Cet algorithme de suivi est rapide et bénéficie de la robustesse de la représentation non paramétrique de l'objet. Il a ouvert la voie à de nombreux travaux sur le

suivi basé couleur.

État de l'art en matière de suivi basé couleur. Si l'utilisation d'une représentation globale permet une bonne flexibilité face à certains changements d'apparence, le manque d'information sur la configuration spatiale de l'objet présente des inconvénients. En particulier, un tel histogramme est peu sensible à certains déplacements, notamment les rotations ou les changements d'échelle (voir Figure 3.8). Une représentation par un histogramme global peut également s'avérer insuffisante pour différencier plusieurs objets de couleurs similaires.



(a) (b) (c)

FIGURE 3.8 – *La courbe (c) représente les valeurs d'un critère de similarité entre l'histogramme de référence de l'objet, calculé sur l'image (a) et les histogrammes calculés pour différents facteurs d'échelle dans l'image (b). On remarque que le minimum du critère de similarité n'apparait pas de façon marquée. Les histogrammes de couleurs sont peu sensibles aux changements d'échelle.*

Pour pallier ces inconvénients, de nombreuses tentatives ont été faites ces dernières années pour définir une description de l'objet qui bénéficie à la fois de la flexibilité propre aux représentations par histogramme, et d'une information sur la configuration spatiale de l'objet. [Birchfield 05] propose d'ajouter des informations spatiales aux histogrammes, en construisant des *spatiogrammes* qui contiennent les moments spatiaux associés à chaque classe de l'histogramme.

L'utilisation de noyaux pour pondérer les pixels en fonction de leur position constitue en soi un apport d'information spatiale. Cependant, comme nous l'avons vu, les noyaux symétriques ne permettent pas de déterminer les mouvements de rotation, et les changements d'échelle ne sont pas traités directement dans [Comaniciu 00]. [Collins 03] propose de calculer les noyaux à différentes échelles pour permettre de s'adapter aux changements de taille de l'objet. Plus récemment, des travaux ont proposé l'utilisation de plusieurs noyaux, utilisés pour calculer l'histogramme de différentes parties de l'objet [Hager 04] [Fan 05] [Fan 07] [Guskov 06] [Megret 06]. L'idée commune à ces travaux est d'obtenir une représentation suffisamment discriminante pour déterminer la position de l'objet, tout en étant suffisamment insensible aux variations autres que le mouvement recherché.

Une alternative pour améliorer le caractère discriminant du modèle de l'objet peut consister à combiner plusieurs types d'informations visuelles. Dans [Birchfield 98] [Maggio 07] [Jeyakar 08] une information de gradient est utilisée en complément de l'information de couleur. Cependant, le calcul de plusieurs types d'informations pour chaque position candidate peut être coûteux en temps de calcul pour des applications temps réel.

Dans ces approches basées noyaux, le suivi est réalisé par un processus déterministe de type descente de gradient ou Gauss-Newton [Comaniciu 00] [Hager 04]. Comme nous l'avons vu dans la section 1.3.3, ces méthodes nécessitent d'être initialisées à une position proche de la position recherchée pour converger, et sont sensibles à la présence de minima locaux.

Parallèlement, les approches basées histogrammes se sont également développées dans le cadre bayésien, notamment au sein d'un filtrage particulaire [Pérez 02] [Nummiaro 03] [Okuma 04] [Maggio 05b], avec la préoccupation là encore d'ajouter des informations sur la configuration spatiale de l'objet [Pérez 02] [Okuma 04] [Maggio 05b]. Le filtre particulaire est également adapté pour fusionner des informations visuelles de différentes natures : [Brèthes 10] a proposé récemment dans ce cadre de fusionner des informations de couleur, forme et mouvement pour le suivi de personnes.

Ces approches restent souvent limitées à la détection de mouvements de translation ou de changement d'échelle, souvent suffisants pour des applications d'indexation de vidéos ou de vidéosurveillance.

Dans le cas qui nous intéresse, nous souhaitons pouvoir estimer à la fois les translations et la rotation dans le plan image, ainsi que les variations d'échelle de l'objet, tout en étant robuste à certaines modifications de l'apparence de l'objet. Nous nous sommes donc orientés vers une représentation multi-noyaux de l'objet.

3.2 Suivi multi-noyaux

3.2.1 Choix d'une configuration multi-noyaux

Espace d'état. Avant de décrire la représentation choisie pour caractériser l'apparence de l'objet, il est nécessaire de définir les paramètres d'état utilisés pour décrire la transformation 2D que l'on cherchera à estimer.

Dans le cadre d'un suivi d'objet depuis un drone de type quadrirotor, les déplacements sont majoritairement situés dans un plan. On a donc choisi un modèle de transformation simplifié adapté à ce type de mouvements. La projection de l'objet dans l'image est ainsi définie dans la suite par un rectangle de rapport fixe $r = \frac{h}{w}$ caractérisé par la position de son centre $\mathbf{c} = (x, y)$, son orientation ψ et sa surface $S = hw$ (voir Figure 3.9). L'état $\mathbf{x}_k$ de l'objet dans l'image d'indice temporel k est alors défini par :

$$\mathbf{x}_k = (x_k, y_k, \psi_k, s), \tag{3.21}$$

où $s = \sqrt{\frac{S}{S_0}}$ est un facteur d'échelle exprimé par rapport à une taille de référence S_0 supposée mesurée dans l'image initiale. Soient $\{\boldsymbol{l}_i\}_{i=1..n}$ les positions pixelliques initiales de l'objet. La transformation $tr_{\mathbf{x}}$ entre l'état initial de l'objet et l'état $\mathbf{x}$ est une similitude définie pour chaque position pixellique $\boldsymbol{l}_i$ par :

$$tr_{\mathbf{x}}(\boldsymbol{l}_i) = tr_{\mathbf{x}}(\mathbf{c}) + s\mathbf{R}_{\psi}(\boldsymbol{l}_i - \mathbf{c}) \tag{3.22}$$

où $\mathbf{R}_{\psi}$ est la matrice de rotation 2×2 d'angle $\psi - \psi_0$.

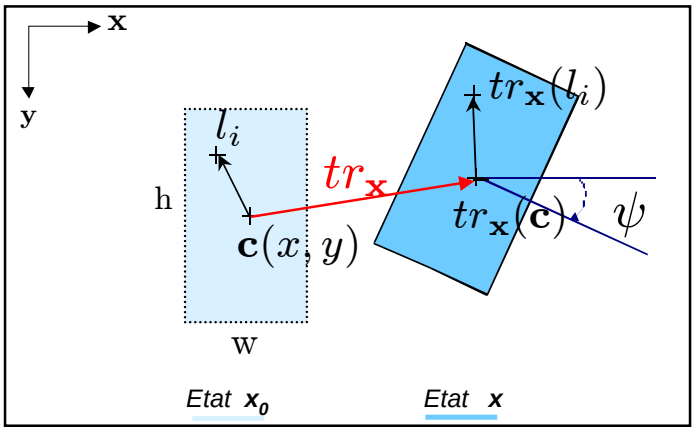


FIGURE 3.9 – *Paramètres de transformation 2D utilisés.*

Comme souligné dans [Hager 04], l'utilisation de plusieurs noyaux (et donc plusieurs histogrammes) augmente l'espace de mesure, ce qui rend le suivi sensible à une plus grande variété de déplacements. Un moyen de concevoir un algorithme de suivi sensible aux déplacements considérés ici consiste à concevoir des noyaux spécifiques à chaque mouvement. [Hager 04] fournit dans cette optique des exemples de noyaux sensibles aux rotations, mais aucune configuration générique n'est proposée. De plus, l'utilisation de noyaux différents implique le calcul de plusieurs histogrammes sur l'objet entier, calcul qui peut être coûteux dans une perspective d'application temps-réel.

La configuration que nous avons choisie ici consiste à utiliser plusieurs noyaux identiques, positionnés en une grille régulière comme illustré sur la Figure 3.10. La pondération des pixels selon ces noyaux leur donne une importance différente en fonction de leur position dans l'objet. Dans ce cas, l'augmentation du nombre de noyaux n'entraîne pas une augmentation du temps de calcul dans la mesure où les histogrammes peuvent être calculés sur des sous-parties de l'objet. De plus, ce type de configuration peut être adapté pour détecter un plus large panel de transformations.

Plus formellement, un état $\mathbf{x}_k$ est associé à un ensemble de n_h histogrammes $\{\mathbf{q}_j(\mathbf{x}_k)\}_{j=1..n_h}$. Chaque histogramme $\mathbf{q}_j(\mathbf{x}_k)$ est calculé selon (3.15) en utilisant le noyau K_j centré en $\mathbf{c}_j$.

Mise en évidence de l'apport de cette configuration sur une fonction de similarité. Pour mettre en évidence l'avantage de cette configuration sur le caractère discriminant de la représentation par histogramme associée on étend ici la distance de Hellinger à la configuration multi-noyaux. Pour définir la similarité entre deux histogrammes $\mathbf{q}^*$ et $\mathbf{q}$ à partir de cette représentation on définit d'abord pour chaque noyau K_j les distances $d_j(\mathbf{x}_k) = d(\mathbf{q}_j^*, \mathbf{q}_j(\mathbf{x}_k))$, où d désigne la distance d'Hellinger (voir équation (3.8)).La Figure 3.11.b montre l'allure de la fonction de distance :

$$d_m(\mathbf{x}_k) = \frac{1}{n_h} \sum_{j=1}^{n_h} d_j(\mathbf{x}_k) \tag{3.23}$$

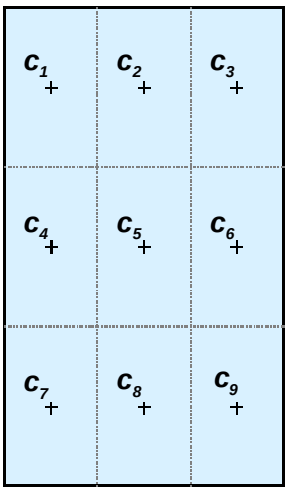
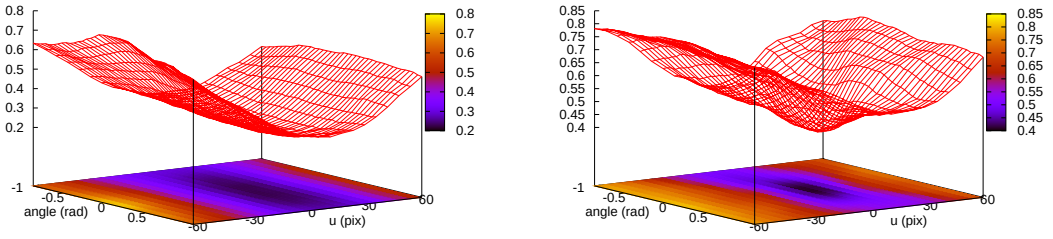


FIGURE 3.10 – *Configuration multi-noyaux avec* $n_h = 9$.

calculée sur l'image de la Figure 3.4-b. La représentation multi-noyaux étant plus discriminante, la sensibilité de la méthode de suivi aux déplacements tels que la rotation est améliorée. Il est bien sûr possible de jouer sur le nombre de noyaux n_h pour augmenter la précision du suivi.



(a) (b)

FIGURE 3.11 – *Fonction de distance en fonction de la rotation et de la position sur l'axe* x*, en utilisant une représentation de l'objet à un seul (a) ou plusieurs (b) noyau(x). La fonction de coût varie très peu avec les rotations dans le cas de la représentation à noyau unique, le minimum apparaît de façon marquée en utilisant la configuration proposée.*

3.2.2 Suivi déterministe

Cette section montre comment la configuration multi-noyaux présentée dans la section précédente (voir Figure 3.10) peut être utilisée dans une approche de suivi déterministe. Suivant le formalisme général proposé dans [Hager 04], les histogrammes sont à présent considérés sous leur forme vectorielle,

$$\mathbf{q} = ({q_1}^{(1)}, .., {q_m}^{(1)}, ..., {q_1}^{(n_h)}, .., {q_m}^{(n_h)}),$$

où ${q_u}^{(j)}$ désigne la valeur de la classe u dans l'histogramme calculé avec le noyau K_j centré en $\mathbf{c}_j$. Pour se ramener à une expression de la forme (1.26), la fonction de coût choisie ici est la distance de Matusita (3.9) :

$$O(\mathbf{x}) = d^2_{\text{Matusita}}(\mathbf{q}^*, \mathbf{q}(\mathbf{x})) = \|\sqrt{\mathbf{q}(\mathbf{x})} - \sqrt{\mathbf{q}^*}\|^2 \tag{3.24}$$

où la racine carrée est appliquée à chaque composant des vecteurs.

La définition de chaque fonction noyau K_j est étendue pour mettre en évidence la dépendance aux paramètres de la transformation $tr_{\mathbf{x}}$ qui permet de passer de l'état $\mathbf{x}_0$ à l'état $\mathbf{x}$. Pour chaque pixel $\boldsymbol{l}$:

$$K_j(\boldsymbol{l}, \mathbf{c}_j, \mathbf{x}) = CK(tr_{\mathbf{x}}(\boldsymbol{l}) - \mathbf{c}_j) \tag{3.25}$$

C étant un facteur de normalisation donné par :

$$C = \frac{1}{\sum_i K(tr_{\mathbf{x}}(\boldsymbol{l}_i) - \mathbf{c}_j)} \tag{3.26}$$

et $\mathbf{c}_j$ étant le centre du noyau K_j. On note alors $\mathbf{K}_j(\mathbf{x}) \in \mathbb{R}^n$ le vecteur de noyau :

$$\mathbf{K}_j(\mathbf{x}) = (K_j(\boldsymbol{l}_1, \mathbf{c}_j, \mathbf{x}), ..., K_j(\boldsymbol{l}_n, \mathbf{c}_j, \mathbf{x}))$$

Le gradient de $\mathbf{K}_j(\mathbf{x})$ est défini par :

$$\mathbf{J}_{\mathbf{K}j} = \left[\frac{\partial \mathbf{K}_j}{\partial x}, \frac{\partial \mathbf{K}_j}{\partial y}, \frac{\partial \mathbf{K}_j}{\partial \psi}, \frac{\partial \mathbf{K}_j}{\partial s}\right].$$

Soit $\mathbf{U}$ la matrice définie par $\mathbf{U} = \{u_{i,j}\}_{i=1..n, j=1..m}$ avec $u_{i,j} = \delta_j(b(\boldsymbol{l}_i))$. $\mathbf{U}$ lie les n points avec la classe de couleur du pixel correspondant dans l'image considérée. Alors, la minimisation de la fonction de similarité (3.24) en utilisant la méthode de Gauss-Newton conduit à

$$\Delta\mathbf{x} = -\mathbf{J}^+_{\mathbf{U}}\mathbf{e} \tag{3.27}$$

où $\mathbf{e} = \sqrt{\mathbf{q}(\mathbf{x})} - \sqrt{\mathbf{q}^*}$, $\mathbf{J}^+_{\mathbf{U}}$ désigne la pseudo-inverse du Jacobien $\mathbf{J}_{\mathbf{U}}$:

$$\mathbf{J}_{\mathbf{U}} = \mathbf{d}(\mathbf{q}(\mathbf{x}))^{-\frac{1}{2}} \begin{bmatrix} \mathbf{U}^\top \mathbf{J}_{\mathbf{K}1} \\ ... \\ \mathbf{U}^\top \mathbf{J}_{\mathbf{K}n_h} \end{bmatrix} \tag{3.28}$$

$\mathbf{d}(\mathbf{q}(\mathbf{x}))$ représente la matrice de diagonale $\mathbf{q}$.

Les noyaux utilisés sont ici des noyaux d'Epanechnikov (voir Figure 3.6-a). Pour chaque pixel $\boldsymbol{l}$:

$$K_j(\boldsymbol{l} - \mathbf{c}_j) = \begin{cases} C\left(1 - \frac{\|\boldsymbol{l}-\mathbf{c}_j\|^2}{h^2}\right) & \text{si } \|\frac{\boldsymbol{l}-\mathbf{c}_j}{h}\| \leqslant 1 \\ 0 & \text{sinon.} \end{cases} \tag{3.29}$$

D'après (3.25), la dépendance du noyau vis-à-vis des paramètres de l'état apparaît en écrivant pour chaque pixel $\boldsymbol{l}$:

$$K_j(\boldsymbol{l}, \mathbf{c}_j, \mathbf{x}) = C\left(1 - \frac{\|tr_{\mathbf{x}}(\boldsymbol{l}) - \mathbf{c}_j)\|^2}{h^2}\right)$$

où $tr_{\mathbf{x}}(\boldsymbol{l})$ est défini dans (3.22).

L'algorithme se déroule alors de manière itérative comme résumé dans l'algorithme 5.

Algorithme de suivi multi-noyaux
Soit $\mathbf{q}^* = \left(q_1^{*(1)}, .., q_m^{*\,(1)}, ..., q_1^{*(n_h)}, .., q_m^{*\,(n_h)}\right)$ l'histogramme de référence de l'objet d'intérêt, calculé avec n_h noyaux.

1. Initialiser la position $\mathbf{x}_0$ dans l'image courante, calculer l'histogramme courant $\mathbf{q} = \left(q_1{}^{(1)}, .., q_m{}^{(1)}, ..., q_1{}^{(n_h)}, .., q_m{}^{(n_h)}\right)$, et évaluer $\mathbf{e} = \sqrt{\mathbf{q}(\mathbf{x})} - \sqrt{\mathbf{q}^*}$.
2. Calculer $\mathbf{J_U} = \mathbf{d}(\mathbf{q}(\mathbf{x}))^{-\frac{1}{2}} \begin{bmatrix} \mathbf{U}^\top \mathbf{J}_{\mathbf{K}1} \\ ... \\ \mathbf{U}^\top \mathbf{J}_{\mathbf{K}n_h} \end{bmatrix}$ selon l'équation (3.28).
3. En déduire le déplacement
$$\Delta\mathbf{x} = -\mathbf{J}_\mathbf{U}^+ \mathbf{e}.$$
4. Si $\|\Delta\mathbf{x}\| < \varepsilon$ arrêter
Sinon : $\mathbf{x}_0 \leftarrow \mathbf{x}_0 + \Delta\mathbf{x}$ et retourner à l'étape 1.

Algorithme 5 – Suivi multi-noyaux.

3.2.3 Approche particulaire

Si historiquement les méthodes de suivi basées couleur ont été développées dans le cadre de méthodes de recalage déterministe, le succès des approches bayésiennes dans le monde de la vision a amené à revisiter le problème du suivi dans ce cadre [Pérez 02] [Nummiaro 02]. Cette section présente l'adaptation de la configuration multi-noyaux choisie, dans le cadre d'un suivi par filtrage particulaire. L'algorithme proposé est une application de l'algorithme SIR (voir l'algorithme 3) dans laquelle la fonction de vraisemblance est basée sur notre configuration multi-noyaux. La densité de probabilité de l'état est représentée sous la forme d'un ensemble de particules, chaque particule correspondant à un état possible (3.21).

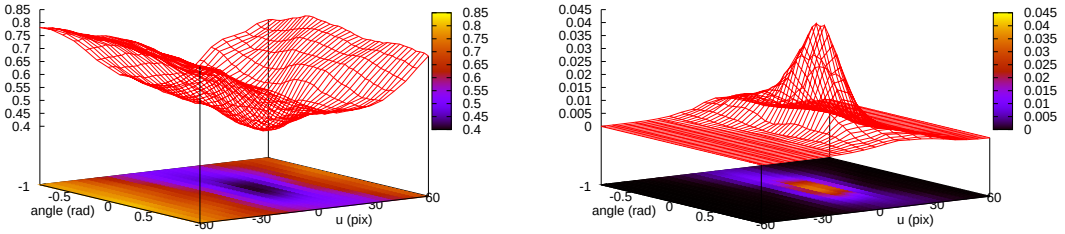
Fonction de vraisemblance. La fonction de vraisemblance est déduite de l'expression de la distance étendue à une configuration multi-noyaux (voir équation (3.23)). Soit $\mathbf{x}_k$ un état candidat, c'est-à-dire une position possible de l'objet dans l'image. On représente alors sa vraisemblance sous la forme :

$$p(\mathbf{z}_k \mid \mathbf{x}_k) \propto exp(-\lambda\, {d_m}^2(\mathbf{x}_k)) \tag{3.30}$$

où λ est un paramètre constant, qui détermine la sensibilité de la vraisemblance aux variations de la fonction de distance d_m. Ainsi une valeur élevée de λ conduit à une vraisemblance très sélective, qui ne donnera un poids élevé que dans une zone restreinte de l'espace de mesure, ce qui peut conduire à un appauvrissement des états lorsque peu de particules se trouvent dans des zones à forte vraisemblance (voir page 28). Le paramètre λ représente alors la confiance que l'on attribue à la mesure. Sa valeur est déterminée empiriquement dans les expérimentations.

La Figure 3.12 montre la valeur de la vraisemblance pour l'exemple de la Figure 3.11, où l'on fait varier un paramètre de translation et un paramètre de rotation dans le plan image.

Évolution. Durant l'étape de prédiction, les particules sont propagées selon un modèle d'évolution. Lorsqu'aucune information *a priori* sur le déplacement de l'objet



(a) (b)

FIGURE 3.12 – *Fonction de distance (3.23) en fonction de la rotation dans le plan et de la position sur l'axe x (a) et vraisemblance associée pour $\lambda = 20$ (b).*

dans l'image n'est disponible, le modèle le plus simple consiste à supposer la position constante et modéliser l'incertitude de ce modèle sous la forme d'une densité de probabilité gaussienne centrée autour de la position précédente :

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \boldsymbol{v}_k \tag{3.31}$$

où $\boldsymbol{v}_k$ est généré selon un loi normale $\mathcal{N}_{\mathbf{x}_k,\boldsymbol{\Sigma}}$. Σ détermine la confiance attribuée au modèle.
Pour la tâche de poursuite considérée dans nos travaux nous avons choisi un modèle à vitesse constante pour les composantes de $\mathbf{c}$, afin de mieux prendre en compte les grands déplacements inter-images :

$$\mathbf{c}_{k+1} - \mathbf{c}_k = \mathbf{c}_k - \mathbf{c}_{k-1} + \boldsymbol{w}_k. \tag{3.32}$$

où $\boldsymbol{w}_k$ est un bruit gaussien.

Les dynamiques des différentes composantes du vecteur d'état sont supposées indépendantes. L'évolution du facteur d'échelle est représentée quant à elle par un modèle à position constante.

Initialisation du filtre. Pour initialiser l'ensemble de particules, on suppose ici que la position de l'objet dans l'image initiale est connue, qu'elle soit sélectionnée manuellement par un opérateur humain ou détectée automatiquement. La question de la détection automatique sera discutée dans la section 3.4. La position initiale étant connue, l'ensemble de particules est généré à l'initialisation selon une densité de probabilité gaussienne autour de cette position initiale (Figure 3.13).

Position estimée. L'ensemble de particules pondérées représente un échantillonnage de la densité de probabilité de l'état que l'on cherche à estimer. L'un des avantages de cette représentation est sa capacité à représenter des densités multimodales. Dans le cadre du suivi, la position estimée à chaque instant sera extraite de la représentation courante. Différents estimateurs peuvent être considérés, les plus utilisés étant l'espérance : $E[\mathbf{x}_k] = \frac{1}{N}\sum_{n=1}^{N} s_k^{(i)}$ ou le maximum de vraisemblance,

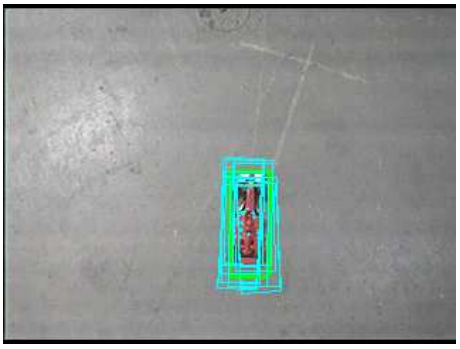


FIGURE *3.13 – Exemple d'initialisation des particules autour de la position de l'objet. Par souci de clarté seules* 30 *particules ont été représentées.*

c'est-à-dire la position de la particule ayant la plus forte vraisemblance.
L'algorithme de suivi multi-noyaux par filtrage particulaire est résumé dans l'algorithme 6.

Algorithme de suivi multi-noyaux par filtrage particulaire
Soit $\mathbf{q}^* = \left(q_1^{*(1)}, .., q_m^{*\,(1)}, ..., q_1^{*(n_h)}, .., q_m^{*\,(n_h)}\right)$ l'histogramme de référence de l'objet d'intérêt, calculé avec n_h noyaux. Initialisation : générer N particules $\left\{s_0^{(i)}\right\}_{i=1}^{N}$ et poser $\forall i, \pi_0^{(i)} = \frac{1}{N}$.
Pour $k = 1, ..., T$:
1. Prédiction : générer N particules $s_{k^-}^{(i)}$ selon $p(\mathbf{x}_k \mid \mathbf{x}_{1:k-1} = s_{k-1}^{(i)})$.
2. Correction : mettre à jour les poids des particules prédites en utilisant (3.30).
3. Rééchantillonnage : tirage aléatoire pondéré de N particules dans l'ensemble $\left\{(s_{k^-}^{(i)}, \pi_{k^-}^{(i)})\right\}_{i=1}^{N}$
   On obtient $\left\{(s_k^{(i)}, \frac{1}{N})\right\}_{i=1}^{N}$.

Espérance : $E[\mathbf{x}_k] = \sum_{i=1}^{N} \pi_k^{(i)} s_k^{(i)}$.

Algorithme 6 – Suivi multi-noyaux par filtrage particulaire.

3.3 Suivi hybride

Dans la section précédente nous avons présenté deux approches différentes pour le suivi d'objet basé couleur, utilisant une même configuration multi-noyaux. Arrivé à ce point, il est intéressant d'observer les aspects complémentaires des deux approches considérées.

Dans l'optimisation multi-noyaux déterministe, quelques itérations permettent d'atteindre le minimum de la fonction de coût avec une bonne précision, à condition que la position initiale soit proche du mode recherché. Cependant, elle ne permet pas de faire face à des occultations totales, est susceptible de tomber dans des *minima* locaux et ne prend en compte qu'une seule hypothèse sur l'état de l'objet.

Le filtrage particulaire permet le filtrage temporel de l'état, et donc de tirer parti des informations antérieures. Indépendamment de cet aspect, il permet également

d'estimer la densité de probabilité sans aucune hypothèse préalable sur sa forme, ce qui permet notamment de prendre en compte des densités multi-modales. Cependant, la précision obtenue dépend du nombre de particules. En effet, la vitesse de convergence de l'espérance calculée à partir de l'approximation particulaire vers l'espérance réelle (voir l'équation (1.59)), d'après le théorème de la limite centrale, est de l'ordre de $\frac{1}{\sqrt{N}}$. Par conséquent, pour gagner un facteur 2 dans la précision de l'estimée il est théoriquement nécessaire d'utiliser 4 fois plus d'échantillons. Le nombre de particules influant linéairement sur le temps de calcul, un compromis est souvent nécessaire entre la précision et la rapidité d'exécution.

Pour tirer parti des avantages de chacune de ces méthodes, il est intéressant d'étudier un moyen de les combiner. Cette idée a inspiré plusieurs travaux [Sullivan 01] [Maggio 05a] [Shan 07]. Dans [Maggio 05a] et [Shan 07], un filtrage particulaire avec *Mean Shift* intégré est proposé, dans lequel l'optimisation par *Mean Shift* est appliquée à chaque particule pour générer des particules dans les maxima locaux de la fonction de densité de probabilité. Cela permet d'utiliser moins de particules en conservant une bonne précision de localisation. L'étude est limitée dans ce cas à l'estimation des translations. Plus récemment, [Maggio 09] a proposé une extension de la procédure du *Mean Shift* qui utilise un noyau anisotrope [2] pour détecter spécifiquement les rotations et les changements d'échelle. Les travaux réalisés dans cette thèse s'inscrivent dans la même lignée, avec la volonté d'utiliser une configuration multi-noyaux générique, qui ne soit pas spécifique à un type de mouvement particulier.

Approche proposée

Dans le cas du suivi multi-noyaux, réaliser l'optimisation à partir de chaque particule est coûteux en termes de temps de calcul. Par conséquent, au lieu de faire converger chaque particule vers le mode le plus proche, nous avons choisi d'appliquer la procédure d'optimisation directement sur l'estimée fournie par le filtre particulaire. Ainsi, la diminution du nombre de particules qui introduit un manque de précision est compensée par la procédure d'optimisation qui trouve le mode de la fonction de distance.

Tandis que l'optimisation seule échoue en cas de déplacements importants, la combiner avec un filtrage particulaire lui permet de tirer profit de l'estimée comme étant une approximation proche de l'état réel de l'objet. En outre, cette estimée préalable permet de réduire le nombre d'itérations requises par la minimisation de type Newton.

Pour éviter les échecs dus aux occultations temporaires, celles-ci doivent être détectées. La détection des échecs et la réinitialisation est abordée dans la section suivante.

2. Une fonction noyau K est dite anisotrope lorsque sa valeur $K(\mathbf{x})$ dépend de la direction du vecteur $\mathbf{x}$.

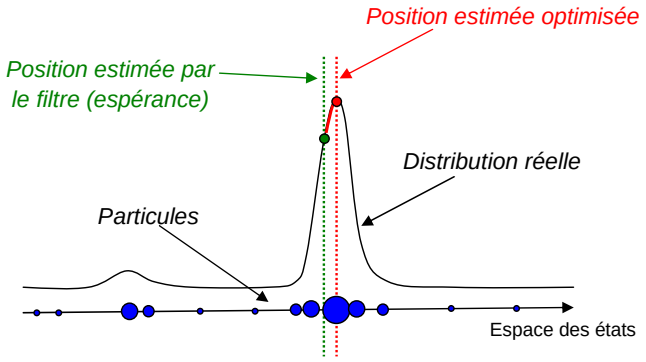


FIGURE 3.14 – *L'optimisation déterministe permet de converger vers le mode de la densité de probabilité sous-jacente.*

3.4 Détection des échecs et réinitialisation du suivi

Dans les sections précédentes on a présenté des algorithmes de suivi en supposant que la position initiale de l'objet était connue. En pratique, dans le cadre applicatif qui nous intéresse la détection initiale est un enjeu important. En outre, quel que soit l'algorithme de suivi visuel envisagé, des cas d'échec peuvent se produire, nécessitant la réinitialisation de l'algorithme. Cette section vise à étudier ces cas critiques.

3.4.1 Gestion des échecs

Détection des échecs. Avant d'aborder la question de l'initialisation ou réinitialisation du suivi, on s'intéresse ici à la question de la détection des échecs.

Lorsque l'objet du suivi n'est plus visible à cause d'une occultation totale (passage sous un pont par exemple, voir Figure 3.15-a) ou d'une sortie de champ, le filtrage particulaire SIR présenté dans l'algorithme 6 continue d'estimer la position de l'objet en faisant évoluer l'ensemble de particules. Cependant, la mesure de distance n'est plus pertinente et les particules promues par le filtre lors de l'étape de rééchantillonnage comme étant les meilleures ne correspondent plus à l'objet. L'estimée résultante peut alors être attirée par des éléments de l'image éloignés de la position réelle alors que l'on souhaiterait baser l'estimation uniquement sur le modèle de prédiction.

En outre, dans le cas de l'algorithme hybride présenté dans la section 3.3, l'étape d'optimisation déterministe est évidemment vouée à l'échec dans une situation d'occultation.

Même si l'algorithme de filtre particulaire peut généralement retrouver l'objet lorsqu'il réapparait dans l'image, il est donc nécessaire de détecter ces situations d'échecs. Pour cela le critère utilisé est ici un seuil sur la distance (3.23).

$$d_{min} = min_{\mathbf{x}}\{d_m(\mathbf{x})\} \tag{3.33}$$

On rappelle que la distance d_m est la moyenne des distances calculées pour chaque

noyau :

$$d_m(\mathbf{x}) = \frac{1}{n_h} \sum_{j=1}^{n_h} d_j(\mathbf{x}) \tag{3.34}$$

Ainsi, un objet est considéré temporairement perdu lorsque la distance minimale d_{min} obtenue sur l'ensemble des particules est supérieure à un seuil ε. Ce seuil est déterminé de façon empirique et affiné au moment de l'initialisation du filtre, où l'objet est supposé encore visible, en prenant : $\varepsilon = 1.3\ d_{min}$. Dans le cadre de nos travaux l'utilisation d'un seuil constant a permis d'obtenir des résultats satisfaisants (voir section 3.5). On note cependant que pour de longues séquences avec de forts changements d'illumination, l'adaptation du modèle et du seuil au cours du suivi est à envisager.

Comportement en cas d'échec. Lorsqu'un échec est détecté, le filtre particulaire passe dans un mode dit *critique* jusqu'à ce que l'une des particules passe à nouveau sous le seuil de détection. Dans ce mode critique, la mesure de distance n'est plus considérée comme pertinente et le suivi évolue selon le modèle de prédiction seul. En pratique, la mesure de vraisemblance est toujours effectuée pour déterminer si l'objet est perdu ou non, mais l'étape de rééchantillonnage du filtre n'est plus appliquée. Cela permet de conserver une représentation de l'espace d'état qui ne soit pas concentrée par erreur vers des zones de faible intérêt, et d'améliorer ainsi les chances de retrouver l'objet.

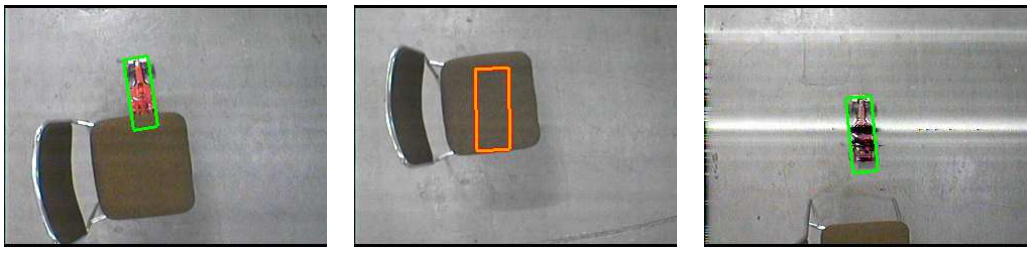
Dans le cas de l'algorithme hybride, l'estimée considérée pendant les phases critiques sera l'estimée du filtre particulaire non optimisée. Ainsi, si $\overline{\mathbf{x}}_k{}^{PF}$, et $\overline{\mathbf{x}}_k{}^{PFOPT}$ représentent respectivement l'estimée donnée par le filtre particulaire et l'algorithme combiné dans l'image k, la procédure hybride proposée dans le paragraphe précédent peut être formulée comme suit :

Si $\left(d_m\left(\overline{\mathbf{x}}_k{}^{PF}\right) \leq \varepsilon\right)$ **Alors**
  calculer $\overline{\mathbf{x}}_k{}^{PFOPT}$ à partir de $\overline{\mathbf{x}}_k{}^{PF}$ en utilisant l'optimisation multi-noyaux
Sinon
  $\overline{\mathbf{x}}_k{}^{PFOPT} \leftarrow \overline{\mathbf{x}}_k{}^{PF}$
Fin Si

La détection des occultations permet à l'algorithme d'endurer des occultations de courte durée ou d'éventuels pics de bruit dans les images (voir Figure 3.15). Si toutefois l'algorithme de suivi échoue (occultation longue, changement brutal de direction...) une procédure de réinitialisation est alors nécessaire.

3.4.2 Initialisation et réinitialisation du suivi

Création du modèle. Les algorithmes de suivi basé couleur présentés dans ce chapitre supposent qu'un modèle de référence de l'objet est disponible, sous la forme



(a) (b) (c)

FIGURE *3.15 – Exemples de situations rencontrées : occlusion partielle (a) ou totale (b) et bruit de transmission (c).*

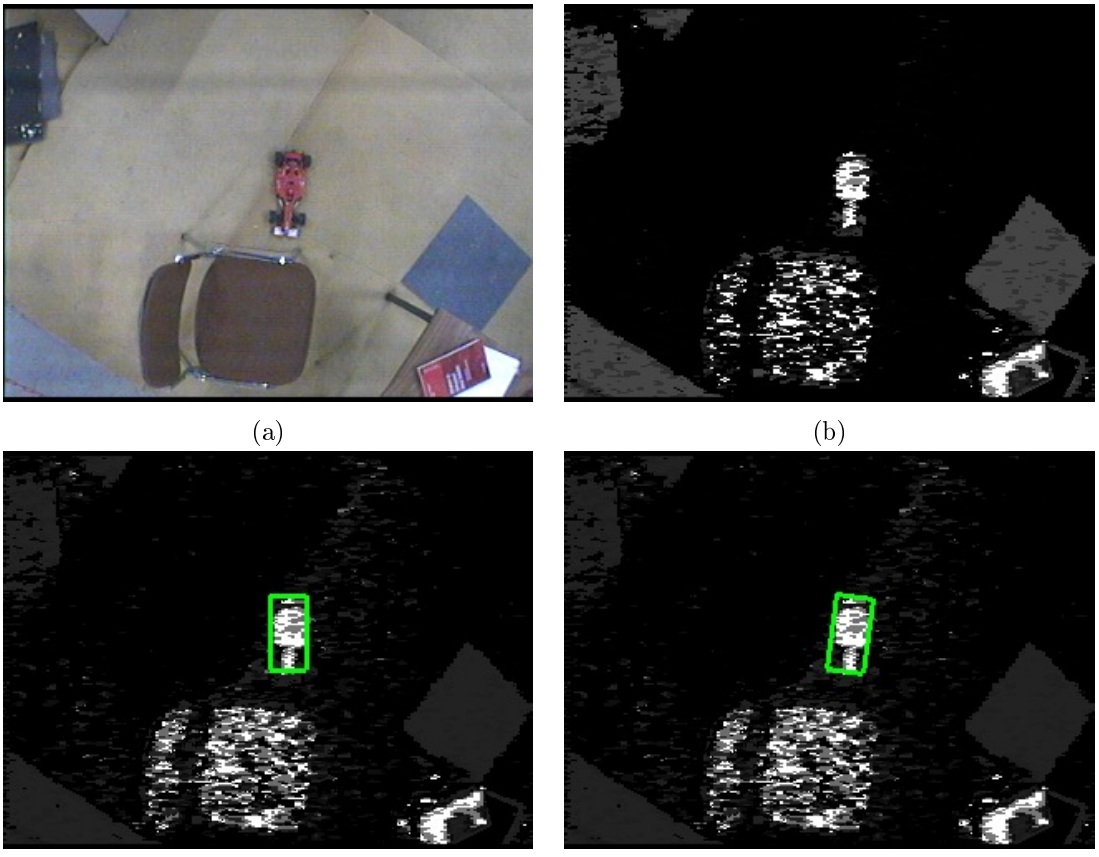
d'un histogramme de référence $\mathbf{q}^*$. En l'absence de connaissances *a priori* sur l'apparence de l'objet il existe plusieurs possibilités pour obtenir ce modèle. Certains travaux proposent d'initialiser le suivi à partir de la détection des zones en mouvement par rapport au fond [Haritaoglu 98] [Irani 98b]. Les objets immobiles ne seront donc pas identifiés. Ces méthodes sont particulièrement adaptées pour des applications de type vidéosurveillance, avec une caméra fixe ou dont les mouvements sont connus. Lorsque les déplacements de la caméra sont importants entre deux images successives, et que des interférences peuvent survenir, les méthodes d'estimation des zones de mouvement sont plus lourdes à mettre en œuvre et le plus souvent peu efficaces.

Dans les applications pour lesquelles une interaction humaine est possible, le modèle peut être calculé en ligne dans une région de l'image sélectionnée manuellement au début de la séquence [Azrad 10]. Cette méthode est plus adaptée à notre application, dans la mesure où la tâche de suivi envisagée est déclenchée par un opérateur humain.

On note toutefois que pendant la sélection de l'objet par l'opérateur, le drone continue à se déplacer, et si le modèle reste valide, la position de l'objet dans l'image servant à l'initialisation du modèle peut être très différente de celle que l'objet occupe une fois le modèle calculé, lorsque l'on cherche à initialiser le suivi en lui-même. Une méthode de détection de l'objet à partir de son modèle reste donc nécessaire.

Détection automatique. À la différence d'un algorithme de suivi qui utilise les positions passées de l'objet pour réduire sa zone de recherche, le module de détection automatique ne dispose pas de connaissance *a priori* sur la position de l'objet. La méthode proposée ici est basée sur l'algorithme Camshift [Bradski 98]. [Bradski 98] propose de calculer l'image de *rétro-projection* du modèle de l'objet dans l'image de recherche. L'image de rétro-projection est construite en remplaçant chaque pixel de l'image de recherche par la valeur de la classe de l'histogramme de référence à laquelle le pixel appartient. L'image obtenue représente donc la probabilité de chaque pixel d'appartenir à l'objet (voir Figure 3.16). Un algorithme de *Mean Shift* est alors appliqué sur cette image de rétro-projection pour converger d'une position initiale vers le maximum local de cette densité de probabilité. L'algorithme Camshift fournit également une estimation de l'orientation et de la taille de l'objet en calculant les moments spatiaux d'ordre 1 de la zone obtenue par *Mean Shift* .

Dans nos travaux, cette méthode est utilisée pour la détection automatique de



(a) (b)

(c) (d)

FIGURE 3.16 – *Algorithme du Camshift. L'image de rétro-projection (b) de l'histogramme de référence sur l'image courante (a) représente la probabilité de chaque pixel d'appartenir à l'objet. L'algorithme du Mean Shift permet de détecter le mode le plus proche d'une position initiale donnée, pour une taille de fenêtre fixe (c). Le calcul des moments spatiaux permet d'obtenir également l'orientation et la taille de l'objet détecté (d).*

l'objet, en calculant l'image de rétro-projection à partir de l'histogramme global de l'objet, et non la configuration multi-noyaux, et en appliquant la recherche de modes à partir de plusieurs positions initiales différentes dans l'image de recherche. Les positions candidates résultant de ces recherches de modes sont alors testées à l'aide de la distance de Hellinger sur les histogrammes de couleur, afin de déterminer si l'une d'elles est susceptible de correspondre à l'objet (voir Figure 3.17). Le cas échéant, l'algorithme de suivi pourra alors être initialisé ou réinitialisé à partir de cette position. Cette procédure de réinitialisation pourra être initiée manuellement par l'opérateur, ou déclenchée automatiquement lorsque le système demeure en échec sur un intervalle de temps trop important.

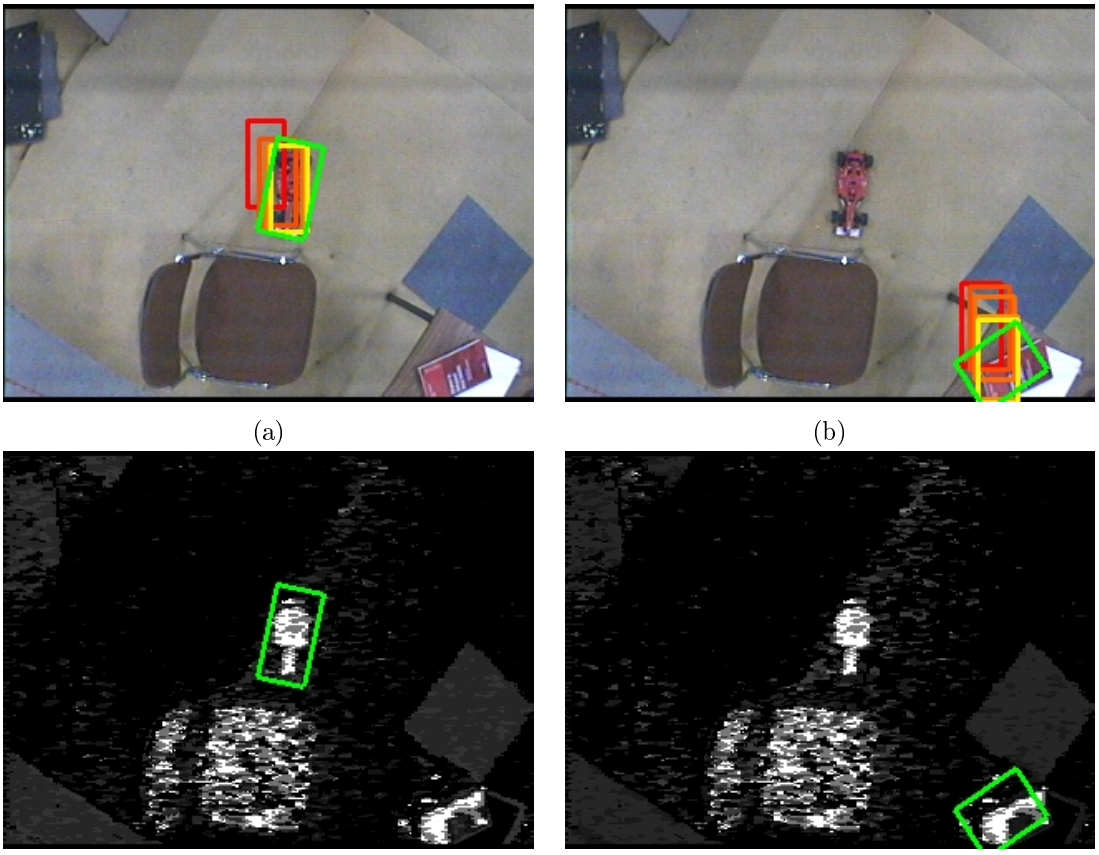


FIGURE *3.17 – Exemple de détection avec plusieurs positions candidates testées. Pour une position initiale indiquée en rouge, la position finale est représentée ici en vert.*

La Figure 3.18 illustre le processus de suivi avec gestion des échecs.

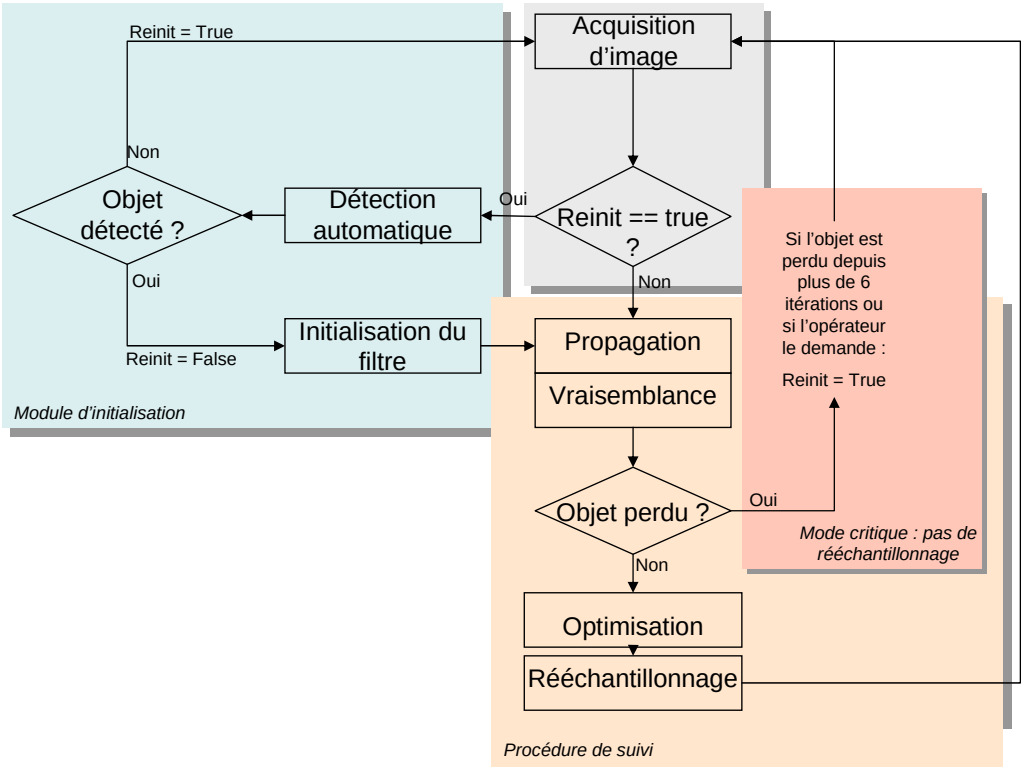


FIGURE 3.18 – *Algorithme de suivi hybride avec gestion des échecs.*

3.5 Résultats comparatifs

Les approches présentées dans ce chapitre ont été testées sur de nombreuses séquences d'images afin d'évaluer leur comportement face à certaines contraintes, en particulier la capacité à surmonter des occultations, de grands déplacements, des mouvements rapides et des changements d'apparence. Cette section présente les résultats comparatifs des différentes approches.

Conditions expérimentales

Pour chaque méthode, les histogrammes sont calculés dans l'espace de couleur RVB divisé en $8 \times 8 \times 8$ classes, et les noyaux utilisés sont des noyaux d'Epanechnikov (Figure 3.6 (a)). La position initiale fournie est identique pour tous les algorithmes.

Pour réduire le temps d'exécution, le calcul de la vraisemblance du filtre particulaire est parallélisée : les calculs effectués pour deux particules différentes sont indépendants et peuvent donc s'exécuter en parallèle. On utilise pour cela les 2 cœurs d'un PC *dualcore* en faisant appel à l'interface de programmation *openmp*[3].

Translations et rotations

Les Figures 3.19 et 3.20 présentent les résultats comparatifs entre :

3. http ://openmp.org/

1. un filtre particulaire à noyau unique, avec 500 particules ;
2. l'approche déterministe multi-noyaux ;
3. le filtre particulaire multi-noyaux, avec 500 particules ;
4. le filtre particulaire multi-noyaux, avec 75 particules ;
5. l'approche combinée avec 75 particules ;

pour une séquence avec un mouvement rapide de l'objet (1-2m/s), comportant une occultation complète, pour un objet de taille fixée.

Le filtre particulaire à noyau unique donne des résultats insuffisants notamment dans l'estimation de la rotation de l'objet (voir Figure 3.20 (1) et première ligne de la Figure 3.19). La méthode d'optimisation seule est précise, mais échoue lors de l'occultation (Figure 3.19 (2-c), ce qui correspond à la rupture qui apparaît dans l'image de la Figure 3.20 (2)). Le filtre particulaire avec 75 particules ((4) dans les Figures 3.19 et 3.20) est robuste à l'occultation mais moins précis qu'avec 500 particules (3). Les résultats expérimentaux montrent que l'approche hybride (5) atteint la même précision que le filtre particulaire seul (3), en utilisant beaucoup moins de particules, ce qui la rend plus efficace du point de vue du temps de calcul. La fréquence d'exécution de notre approche était de l'ordre de 20Hz pour les applications considérées, ce qui la rend adaptée à une utilisation temps-réel.

Le fait d'appliquer l'optimisation sur l'estimée du filtrage particulaire permet de réduire le nombre de particules nécessaires et donc la finesse de la représentation de la densité de probabilité par le filtre particulaire. Il est toutefois important de souligner que le filtre particulaire est toujours garant de la robustesse de l'algorithme notamment en cas d'occultations. En outre, pour obtenir des résultats satisfaisants à partir du suivi hybride il est nécessaire que l'estimée du filtre soit suffisamment proche du mode en question, il n'est donc pas question ici de réduire de manière excessive le nombre de particules utilisées. L'idée que nous défendons ici propose plutôt de tirer partie du caractère *continu*, de l'optimisation déterministe tout en bénéficiant de la robustesse apportée par le filtrage particulaire. Les expériences présentées mettent en évidence les apports de la méthode proposée.

Prise en compte des changements d'échelle

La Figure 3.22 présente les résultats comparatifs entre un filtre particulaire à noyau unique (1^e ligne), l'approche déterministe multi-noyaux (2^e ligne) et le filtre particulaire multi-noyaux (dernière ligne) pour une séquence avec un mouvement rapide de l'objet et des effets de flou dus à la vitesse.

L'utilisation de la configuration multi-noyaux permet une meilleure précision que celle à noyau unique, notamment dans la détection des rotations. Dans cette séquence, le manque d'information dû à l'effet de flou fait converger la méthode d'optimisation vers un mauvais minimum comme illustré Figure 3.22. En l'occurence, le flou « lisse » l'apparence de l'objet (voir Figure 3.21), et lorsque la taille du rectangle diminue la corrélation avec l'histogramme de référence s'avère élevée. Le filtrage particulaire multi-noyaux permet de conserver plusieurs hypothèses correspondant à des échelles différentes et d'obtenir une bonne localisation. De plus, le filtre limite les changements pouvant apparaître entre deux images successives en prenant en compte les positions passées et à travers son modèle d'évolution.

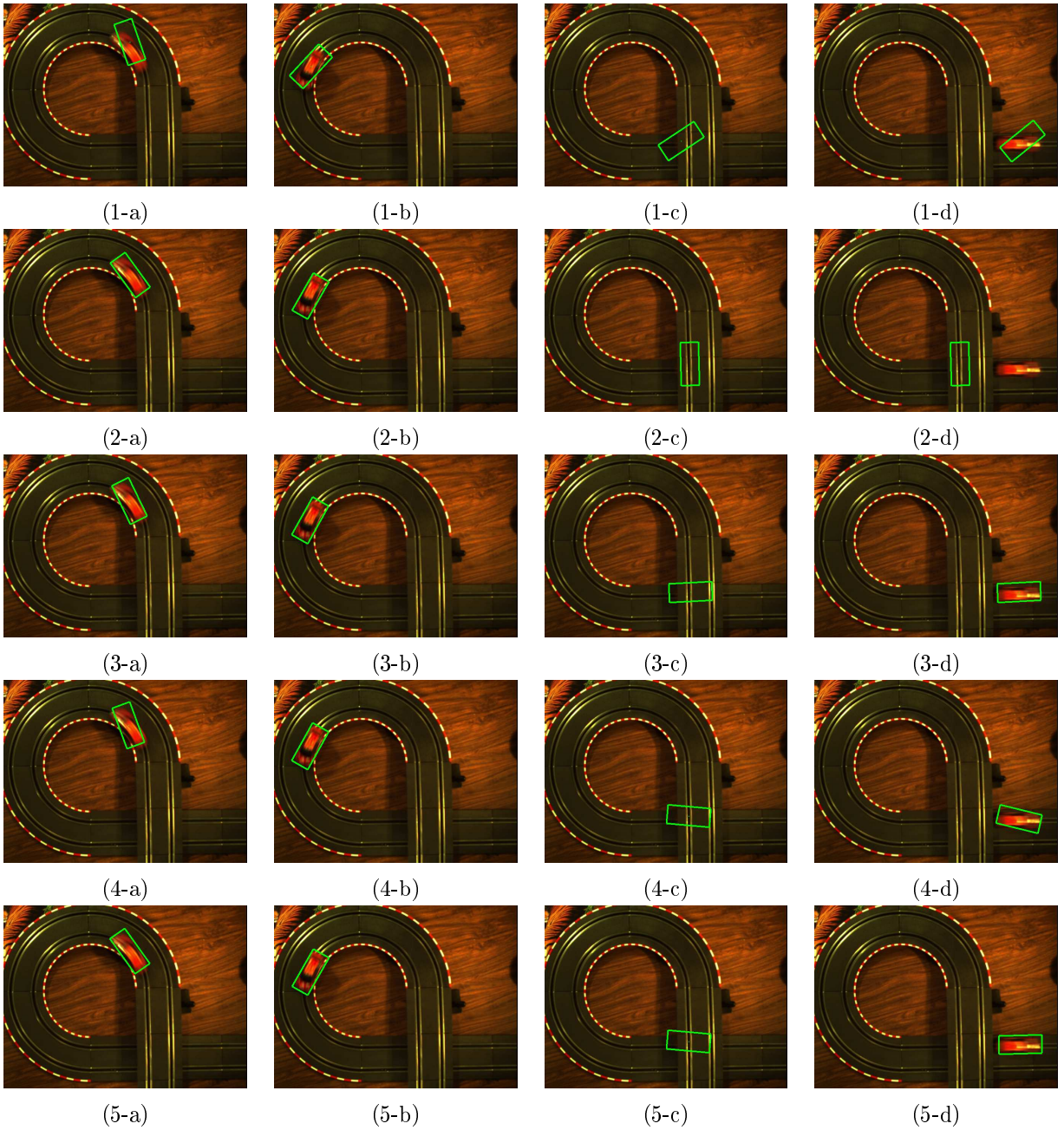


FIGURE *3.19 – Résultats comparatifs des différentes approches avec des translations et des rotations. 1e ligne : filtre particulaire à noyau unique, avec 500 particules. 2e ligne : approche déterministe multi-noyaux. 3e ligne : filtre particulaire multi-noyaux, avec 500 particules. 4e ligne : filtre particulaire multi-noyaux, avec 75 particules. 5e ligne : approche hybride avec 75 particules.*

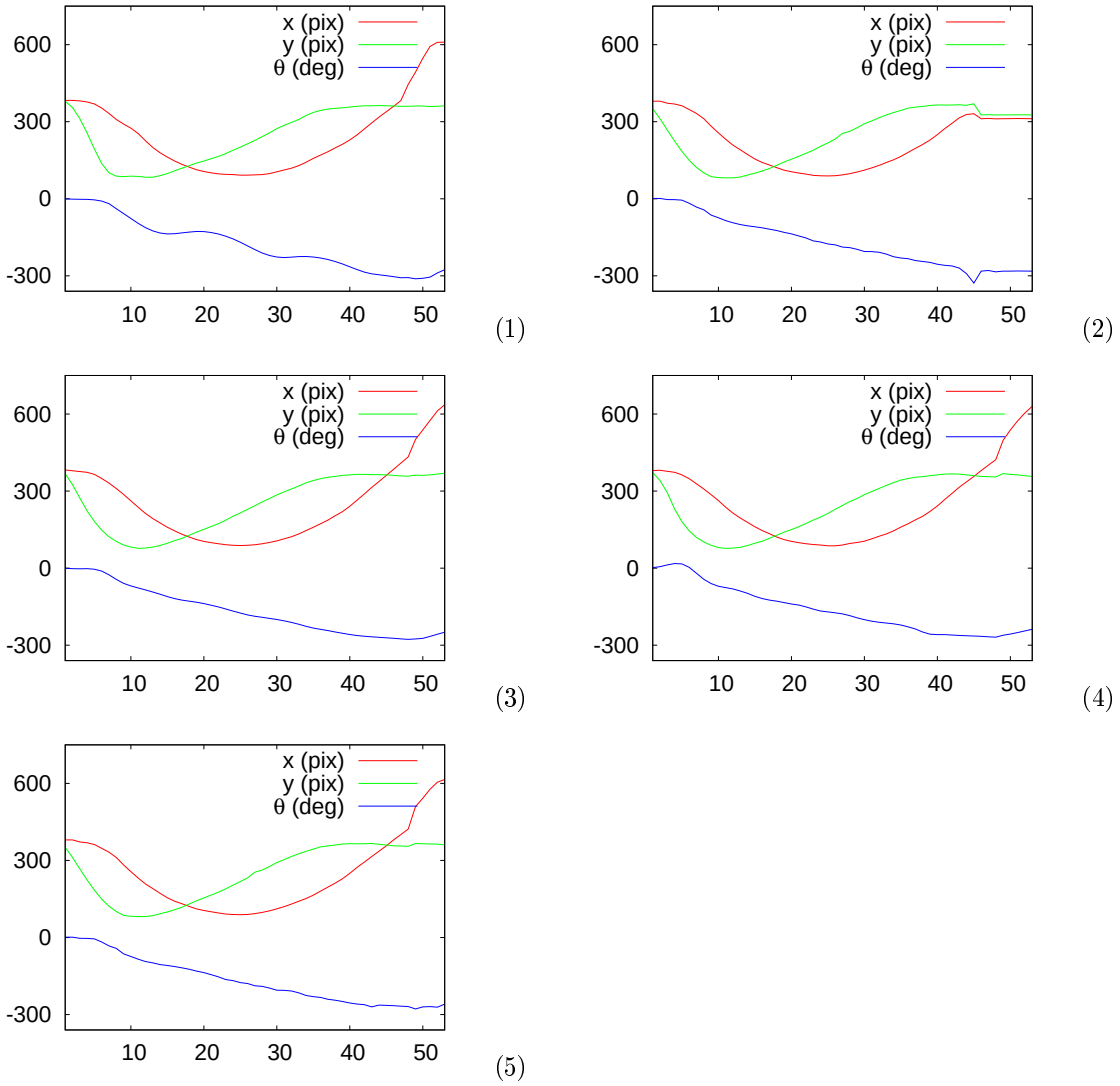


FIGURE 3.20 – *Position et orientation de l'objet dans la séquence de la Figure 3.19 obtenue en utilisant : (1) un filtre particulaire à noyau unique avec 500 particules, (2) l'approche déterministe multi-noyaux [Hager 04], (3) un filtre particulaire multi-noyaux avec 500 particules, (4) un filtre particulaire multi-noyaux avec 75 particules, (5) l'approche hybride avec 75 particules. L'abscisse indique le numéro de l'image dans la séquence.*

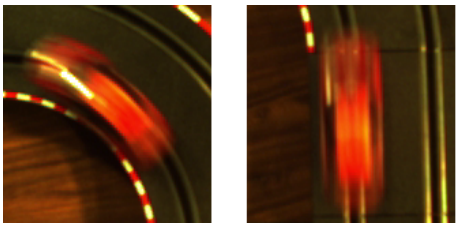


FIGURE 3.21 – *Exemple de flou dû aux mouvements rapides de l'objet.*

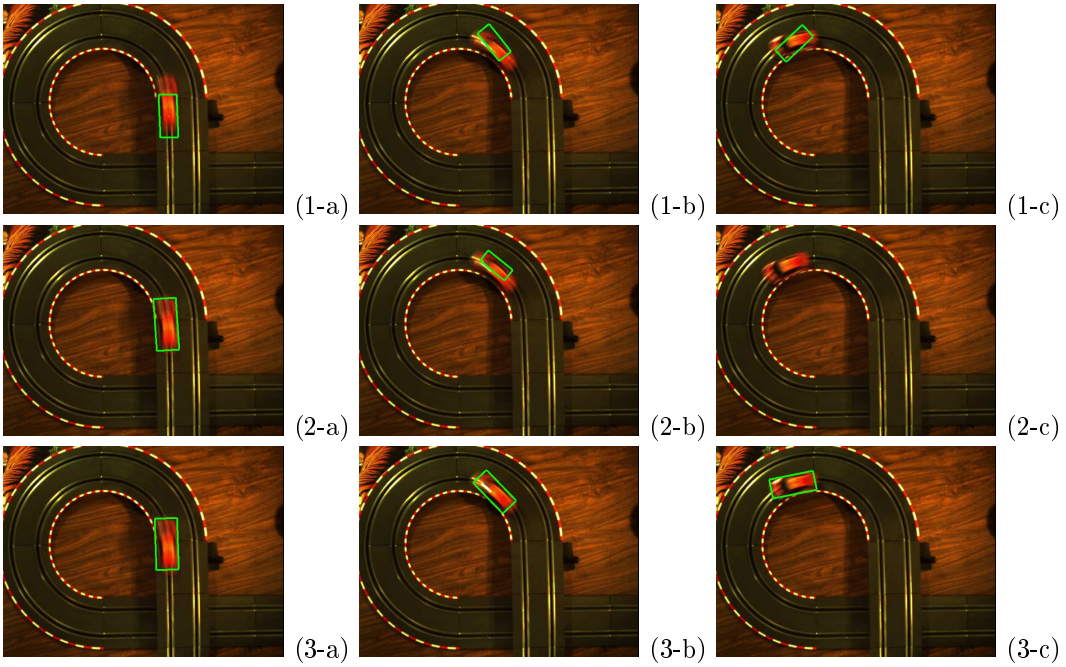


FIGURE 3.22 – *Résultats comparatifs des différentes approches avec des translations, rotations et changement d'échelle. 1e ligne : filtre particulaire avec 1 noyau, et 500 particules. 2e ligne : approche déterministe multi-noyaux [Hager 04]. 3e ligne : filtre particulaire avec 9 noyaux constants, et 500 particules.*

La Figure 3.23 montre un autre exemple de séquence présentant des translations, rotations, changements d'échelle, et du flou de bougé dû à des mouvements rapides. La position du rectangle en vert correspond à l'estimée fournie par notre algorithme de suivi hybride. Dans cette séquence 200 particules ont été utilisées et l'estimation obtenue suit avec succès les mouvements de l'objet.

FIGURE 3.23 – *Extraits d'une séquence de suivi hybride avec rotations, changements d'échelle, et flou de bougé dû à des mouvements rapides.*

3.6 Conclusion

Dans ce chapitre, nous avons étudié le problème du suivi d'objet dans une séquence d'images dans l'optique des contraintes spécifiques de robustesse liées à l'application visée. Dans la première section nous avons justifié le choix d'un mode de représentation par histogrammes, notamment pour ses propriétés de robustesse face aux modifications locales de l'apparence de l'objet, ainsi que pour la zone de convergence importante des critères de similarité associés. Un état de l'art du suivi basé couleur a été présenté, rappelant les principaux enjeux de ces méthodes, en particulier l'intérêt d'obtenir une représentation qui concilie la flexibilité des histogrammes avec un apport d'information spatiale permettant de détecter suffisamment de paramètres du mouvement. Dans cette optique nous nous sommes intéressés aux approches multi-noyaux développées ces dernières années. Nous avons proposé une configuration multi-noyaux générique adaptée aussi bien au cadre des approches multi-noyaux déterministes qu'au cadre bayésien de filtrage particulaire. Une approche hybride combinant ces deux types de méthodes a été proposée et des résultats expérimentaux ont été présentés. Afin de fournir un système de suivi complet, les questions de détection des échecs et de réinitialisation automatique du suivi ont été abordées section 3.4.

Le chapitre suivant présente la commande mise en œuvre pour permettre à un mini-drone de suivre l'objet en utilisant la mesure de vision d'un algorithme de suivi robuste.

Chapitre 4

Poursuite

Dans ce chapitre nous nous intéressons à la commande d'un mini-drone pour une tâche de poursuite de cible mobile. La cible est supposée évoluer dans un plan et notre objectif est d'utiliser les informations visuelles obtenues par notre système de suivi pour réaliser une tâche de poursuite [1].

Ce chapitre est organisé de la façon suivante. La section 4.1 décrit la modélisation du drone considéré. Le schéma de commande proposé est présenté en section 4.2. Enfin des résultats expérimentaux sont exposés et analysés dans la section 4.3.

4.1 Modèle de drone utilisé

Le principe de fonctionnement des drones de type quadrirotor a été présenté brièvement dans le chapitre 2. Cette section décrit de façon plus formelle le modèle dynamique de ce type d'appareils. Après avoir défini les repères utilisés, nous présentons les équations dynamiques du système nécessaires à la mise en place de la commande.

4.1.1 Repères et notations utilisés

Nous supposons ici que la géométrie du drone est parfaite et qu'il peut être assimilé à un corps rigide malgré la rotation de ses pales. On définit tout d'abord les répères utilisés, qui sont illustrés Figure 4.1 :

- le repère inertiel $\mathcal{R}_w$, considéré comme Galiléen, est lié à la terre et défini par le trièdre direct $(\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z)$, relatif à une origine fixe O_w. Les axes de ce repère sont fixes et $\mathbf{e}_z$ pointe vers le centre de la terre ;
- le repère $\mathcal{R}_c$, lié au drone, a pour origine son centre de gravité O_c. L'axe $\mathbf{x}_c$ dirigé vers l'avant du véhicule est appelé axe de roulis, $\mathbf{y}_c$ dirigé vers la

1. En marge des travaux sur un drone décrits dans ce chapitre, nous avons également étudié l'utilisation de notre suivi pour l'asservissement d'un robot cartésien. Les travaux correspondants sont décrits dans l'annexe A.4 de cette thèse.

droite est l'axe de tangage et $\mathbf{z}_c$ est l'axe de lacet, qui pointe vers le bas de l'appareil. Le centre de masse du drone est supposé ici coïncider avec le centre optique de la caméra. Cette hypothèse ne constitue pas une approximation forte sur le système que l'on considère dans ces travaux et suffit à l'obtention de résultats satisfaisants. Dans le cas où cette approximation n'est pas valide la transformation rigide entre les deux devra être estimée au préalable par une méthode d'étalonnage ;

– le repère projeté $\mathcal{R}_p$, dont l'origine est attachée au centre de gravité du drone, a une orientation identique à celle de $\mathcal{R}_w$.

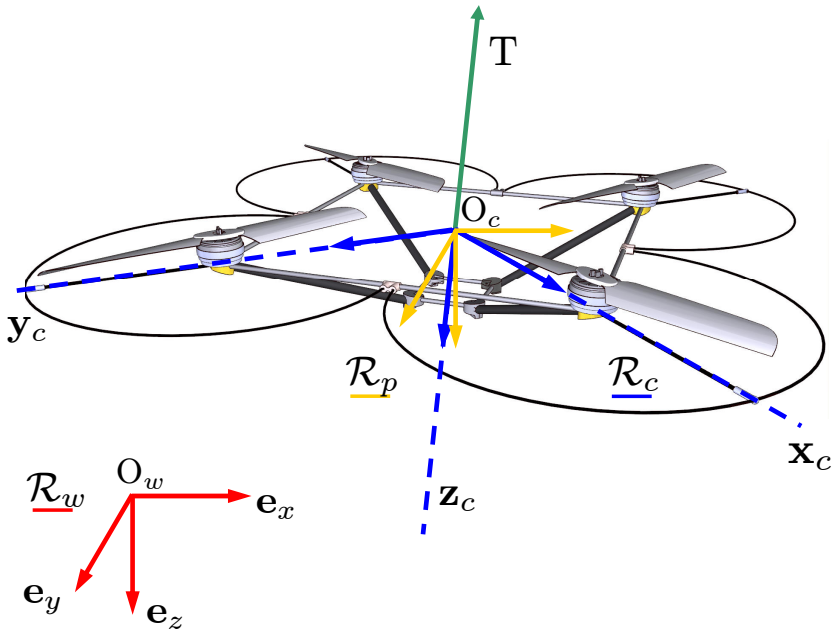


FIGURE *4.1 – Définition des repères utilisés.*

La position du drone est alors caractérisée par la position de son centre de gravité et l'orientation de ses axes par rapport à $\mathcal{R}_w$. La position du centre de gravité du véhicule par rapport au référentiel $\mathcal{R}_w$ est notée $\mathbf{p} = {}^w\mathbf{p}_c$, et sa vitesse $\mathbf{v}$, exprimée dans le même repère, vérifie alors $\mathbf{v} = \frac{d\mathbf{p}}{dt}$.

L'orientation du véhicule est représentée par la rotation ${}^w\mathbf{R}_c$ du véhicule par rapport au repère inertiel. Elle sera notée $\mathbf{R}$ dans la suite. D'après la définition du repère projeté $\mathcal{R}_p$ on a également ${}^p\mathbf{R}_c = {}^w\mathbf{R}_c = \mathbf{R}$. Dans le domaine aéronautique les angles d'Euler autour des axes $(\mathbf{x}_c, \mathbf{y}_c, \mathbf{z}_c)$ sont appelés respectivement angles de roulis, tangage et lacet (ϕ, θ, ψ).

La matrice de rotation est calculée à partir des angles d'Euler de la façon suivante :

$$\mathbf{R} = \begin{pmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi - s_\phi s_\psi \\ c_\theta c_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{pmatrix} \tag{4.1}$$

où l'on a utilisé les notations : $s_\theta = sin(\theta)$ et $c_\theta = cos(\theta)$. Cette rotation correspond à l'application successive d'une rotation de lacet, de tangage, puis de roulis. On

appelle *attitude* ou *assiette* du drone la rotation induite par les angles de roulis et de tangage. La vitesse de rotation instantanée de $\mathcal{R}_c$ par rapport à $\mathcal{R}_w$, exprimée dans $\mathcal{R}_c$, est notée $\mathbf{\Omega}$.

4.1.2 Modèle dynamique

Au niveau le plus bas, le drone est contrôlé par la vitesse de rotation des quatre moteurs entraînant les quatre rotors. Les différences de poussée de ces rotors induisent des couples de tangage (avant/arrière) et de roulis (gauche/droite), et les couples résistants déterminent le mouvement de lacet. Les couples relatifs aux angles d'Euler sont notés ici $\mathbf{\Gamma} = (\Gamma_x,\ \Gamma_y,\ \Gamma_z)$. La rotation des rotors induit une force de poussée globale T dans la direction $\mathbf{z}_c$. Les efforts impliqués dans la modélisation du drone font également intervenir le poids de l'appareil, le couple gyroscopique résultant de la rotation conjointe de la structure de l'appareil et des hélices, et les forces de trainée. Soit m la masse du véhicule et $\mathbf{I} \in \mathbb{R}^{3\times3}$ sa matrice d'inertie. Les théorèmes de la mécanique de Newton et Euler appliqués au drone dans le référentiel $\mathcal{R}_w$ supposé Galiléen s'écrivent alors :

$$\begin{cases} m \left.\frac{d\mathbf{v}}{dt}\right|_{\mathcal{R}_w} = T\mathbf{R}\mathbf{e}_z + mg\mathbf{e}_z + \mathbf{F}_{\text{pert}} \\ \mathbf{I} \left.\frac{d\mathbf{\Omega}}{dt}\right|_{\mathcal{R}_w} = \mathbf{\Gamma} + \mathbf{\Gamma}_{\text{pert}} \end{cases} \tag{4.2}$$

où g est la constante de gravité. $\mathbf{F}_{\text{pert}}$ regroupe les forces extérieures appliquées au véhicule (traînée, vent, etc.) sous la forme d'une force de perturbation. On a également défini le couple $\mathbf{\Gamma}_{\text{pert}}$ de perturbation. Pour une modélisation plus complète des différents efforts auxquels sont soumis les drones de type $X4$, on pourra se référer par exemple à [Guenard 07].

La formule de dérivée de vecteur nous permet d'écrire :

$$\left.\frac{d\mathbf{I\Omega}}{dt}\right|_{\mathcal{R}_w} = \left.\frac{d\mathbf{I\Omega}}{dt}\right|_{\mathcal{R}_c} + \mathbf{\Omega} \times \mathbf{I\Omega} \tag{4.3}$$

Ainsi, le modèle dynamique du drone est défini par le système suivant :

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{v} \\ m\dot{\mathbf{v}} = T\mathbf{R}\mathbf{e}_z + mg\mathbf{e}_z + \mathbf{F}_{\text{pert}} \\ \dot{\mathbf{R}} = [\mathbf{\Omega}]_\times \\ \mathbf{I}\dot{\mathbf{\Omega}} = -\mathbf{\Omega} \times \mathbf{I\Omega} + \mathbf{\Gamma} + \mathbf{\Gamma}_{\text{pert}} \end{cases} \tag{4.4}$$

où on a utilisé la notation $\dot{\mathbf{u}} = \frac{d\mathbf{u}}{dt}$ pour la dérivée temporelle d'un vecteur $\mathbf{u}$.

La modélisation précise des différents efforts aérodynamiques étant très complexe, les modèles utilisés dans la littérature [Bouabdallah 05] [Waslander 05] [Metni 07] sont généralement simplifiés. On négligera ici l'effet des forces $\mathbf{F}_{\text{pert}}$ et couples de perturbation $\mathbf{\Gamma}_{\text{pert}}$, en faisant notamment les hypothèses simplificatrices suivantes :

- le vol s'effectue en régime quasi-stationnaire, en particulier les angles de roulis et de tangage restent faibles, et la poussée est proche de la poussée d'équilibre (c'est-à-dire de la poussée qui permet de compenser le poids du drone) ;

– la géométrie du drone est parfaite ;
– l'effet gyroscopique, étant imperceptible, peut être négligé.

Le modèle simplifié s'écrit alors :

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{v} \\ m\dot{\mathbf{v}} = T\mathbf{R}\mathbf{e}_z + mg\mathbf{e}_z \\ \dot{\mathbf{R}} = [\mathbf{\Omega}]_\times \\ \mathbf{I}\dot{\mathbf{\Omega}} = -\mathbf{\Omega} \times \mathbf{I}\mathbf{\Omega} + \mathbf{\Gamma} \end{cases} \tag{4.5}$$

Le quadrirotor est un système sous-actionné à 4 entrées de commande : au bas niveau ces entrées correspondent aux vitesses de rotation de chacun des 4 moteurs. La poussée T et le couple $\mathbf{\Gamma}$ s'expriment à partir des vitesses des rotors de la façon suivante :

$$\begin{pmatrix} T \\ \mathbf{\Gamma}_x \\ \mathbf{\Gamma}_y \\ \mathbf{\Gamma}_z \end{pmatrix} = \begin{pmatrix} b & b & b & b \\ 0 & -db & 0 & db \\ db & 0 & -db & 0 \\ \kappa & -\kappa & \kappa & -\kappa \end{pmatrix} \begin{pmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{pmatrix} \tag{4.6}$$

où ω_i correspond à la vitesse de rotation du rotor i, $d > 0$ est la distance entre les axes des rotors et le centre de masse, et $b, \kappa > 0$ sont des paramètres aérodynamiques identifiés expérimentalement. La matrice liant les vitesses de rotation des moteurs et le vecteur $(T,\ \Gamma_x,\ \Gamma_y,\ \Gamma_z)$ étant constante et inversible, on peut utiliser directement $(T,\ \Gamma_x,\ \Gamma_y,\ \Gamma_z)$ comme entrées virtuelles de la loi de commande.

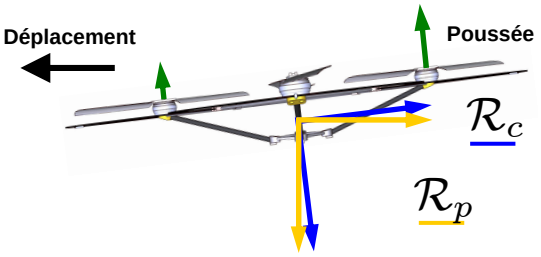


FIGURE *4.2 – Le déplacement du drone en translation est induit par son inclinaison. Cette inclinaison résulte elle-même de la différence de poussée des rotors opposés.*

La dynamique de rotation est couplée à la dynamique de translation (voir le système (4.5) et la Figure 4.2). Dans nos travaux, on considère que la dynamique de rotation est déjà contrôlée en embarqué à l'aide d'un contrôleur grand gain décrit dans [Bertrand 08]. Par conséquent, les contrôleurs de haut niveau développés ici visent à envoyer des consignes d'orientation et de poussée au contrôleur embarqué afin de contrôler la dynamique de translation et éventuellement le lacet du véhicule.

La section suivante présente le schéma de commande mis en place pour réaliser la tâche de poursuite à l'aide des informations issues du système de vision présenté au chapitre 3.

4.2 Commande du drone

Le chapitre précédent a montré comment obtenir une estimation robuste des paramètres de position, taille et orientation d'un objet mobile dans une séquence d'images. Ces mesures peuvent ensuite être utilisées pour contrôler les déplacements du drone afin de suivre une cible mobile. Cette section présente les lois de commande mises en place pour contrôler le mouvement du drone en translation, ainsi que le mouvement de lacet qui permet au drone de suivre les changements d'orientation de l'objet.

4.2.1 Contrôle du lacet

Le contrôle du lacet constitue la partie la moins critique de la commande dans la mesure où le mouvement de lacet est découplé des autres mouvements. La vitesse de rotation de lacet $\mathbf{\Omega}_z = \Omega_z \mathbf{e}_z$ étant contrôlée en embarqué à l'aide des capteurs gyrométriques, nous avons utilisé un simple contrôleur proportionnel (voir Figure 4.3) pour suivre les changements d'orientation effectués par l'objet suivi. Si $\widehat{\psi}$ désigne l'orientation de l'objet dans le plan image mesurée par l'algorithme de suivi et ψ^* l'orientation désirée, le contrôleur envoie une consigne de vitesse de lacet Ω_z^* proportionnelle à l'erreur mesurée :

$$\Omega_z^* = k_{P\psi} \varepsilon_\psi, \tag{4.7}$$

où

$$\varepsilon_\psi = \psi^* - \widehat{\psi}, \tag{4.8}$$

$k_{P\psi}$ étant le gain proportionnel utilisé.

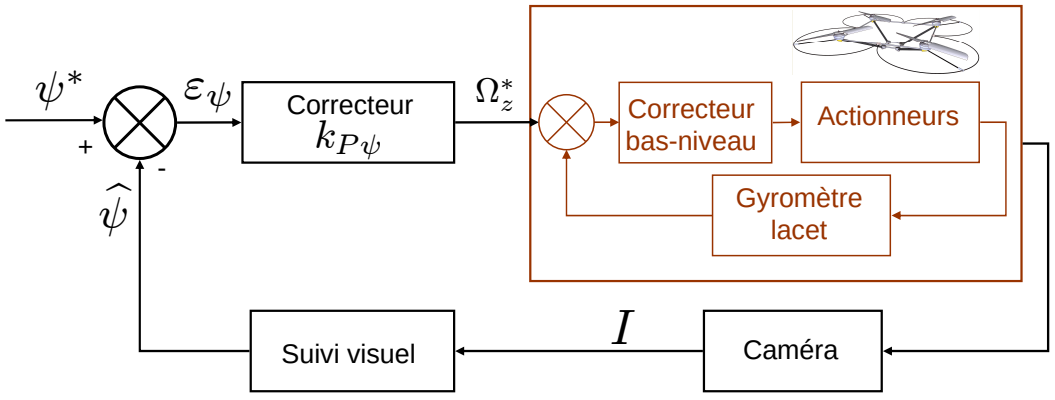


FIGURE 4.3 – *Contrôle du lacet.*

4.2.2 Contrôle du mouvement de translation

Objectif de commande. Pour contrôler les mouvements de translation du drone on utilise l'estimation de la position $\mathbf{c} = (x, y)$ du centre de l'objet dans le plan

image, obtenue par l'algorithme de vision. Cette position est exprimée sous la forme de coordonnées normalisées dans le plan image (voir page 7). Comme illustré dans la Figure 4.4, la position de l'objet observée dans l'image dépend de l'assiette du drone. Pour contrôler le mouvement de translation du drone, la position considérée comme entrée de commande correspond à la position $\mathbf{c}_p = (x_p, y_p)$ de l'objet dans le repère projeté $\mathcal{R}_p$. Cette position $\mathbf{c}_p$ est obtenue à l'aide d'une estimation de la matrice de rotation $\mathbf{R}$ (voir équation (4.1)) calculée à partir des mesures inertielles grâce à l'estimateur d'assiette embarqué du drone.

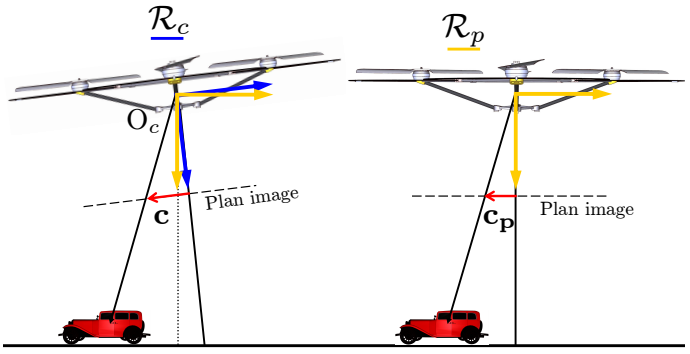


FIGURE 4.4 – *Compensation de l'assiette : $\mathbf{c}$ est la position de l'objet estimée par l'algorithme de suivi et $\mathbf{c}_p$ est la position correspondante dans le repère $\mathcal{R}_p$ où la rotation a été compensée.*

L'objectif de la commande de translation consiste alors à réguler l'erreur de position définie par :

$$\boldsymbol{\varepsilon}_{\mathbf{c}_p} = \mathbf{c}_p^* - \mathbf{c}_p \tag{4.9}$$

où $\mathbf{c}_p^*$ désigne la position désirée de l'objet dans le plan image. En pratique, pour une tâche de poursuite on aura $\mathbf{c}_p^* = 0$, ce qui signifie que l'on souhaite voir l'objet au centre de l'image.

Commande hiérarchique. Pour réguler l'erreur de position (4.9) on a choisi d'utiliser une commande hiérarchique [Guenard 08]. Comme illustré Figure 4.5, la commande hiérarchique est composée d'une boucle de contrôle en position au niveau le plus haut qui définit une vitesse de translation désirée $\mathbf{v}_p^*$ à partir de l'erreur de position estimée (4.9) :

$$\mathbf{v}_p^* = k_P \boldsymbol{\varepsilon}_{\mathbf{c}_p}, \tag{4.10}$$

k_P étant le gain proportionnel utilisé. Cette vitesse est elle-même contrôlée dans une boucle de régulation interne qui joue un rôle d'amortissement nécessaire à la stabilité du système. Le correcteur de vitesse détermine la commande d'assiette (roulis et tangage) à envoyer au drone, celle-ci étant supposée atteinte quasi intantanément à l'aide du correcteur d'assiette interne, en imposant les couples de commande $\boldsymbol{\Gamma}$. La stabilité de la régulation de l'assiette a été démontrée dans [Guenard 07]. En pratique la dynamique de translation étant lente par rapport à la dynamique de rotation, les termes de couplage entre la régulation de l'assiette et celle de la vitesse de translation peuvent être négligés.

Pour prendre en compte les perturbations internes du système, un contrôleur proportionnel intégral (PI) est utilisé pour réguler la vitesse, et un contrôleur proportionnel régule l'erreur de position.

L'erreur de vitesse est définie par

$$\boldsymbol{\varepsilon}_{\mathbf{v}_p} = \mathbf{v}_p^* - \mathbf{v}_p \tag{4.11}$$

où l'erreur courante $\mathbf{v}_p$ est obtenue par un filtre dérivateur sur la position $\mathbf{c}_p$. On utilise ici un filtre de la forme :

$$\mathbf{v}_p^{(k+1)} = (1-\alpha)\mathbf{v}_p^{(k)} + \alpha \, \frac{\mathbf{c}_p^{(k+1)} - \mathbf{c}_p^{(k)}}{\delta t}, \tag{4.12}$$

où α détermine la fréquence de coupure.
Finalement la consigne d'assiette (θ^*, ϕ^*) envoyée au drone est sous la forme :

$$(\theta^*, \phi^*) = k_{Pv}(k_P \boldsymbol{\varepsilon}_{\mathbf{c}_p} - \mathbf{v}_p) + k_{Iv} \int (k_P \boldsymbol{\varepsilon}_{\mathbf{c}_p} - \mathbf{v}_p), \tag{4.13}$$

où k_{Pv} et k_{Iv} désignent respectivement les gains proportionnels et intégrateurs du régulateur de vitesse PI.

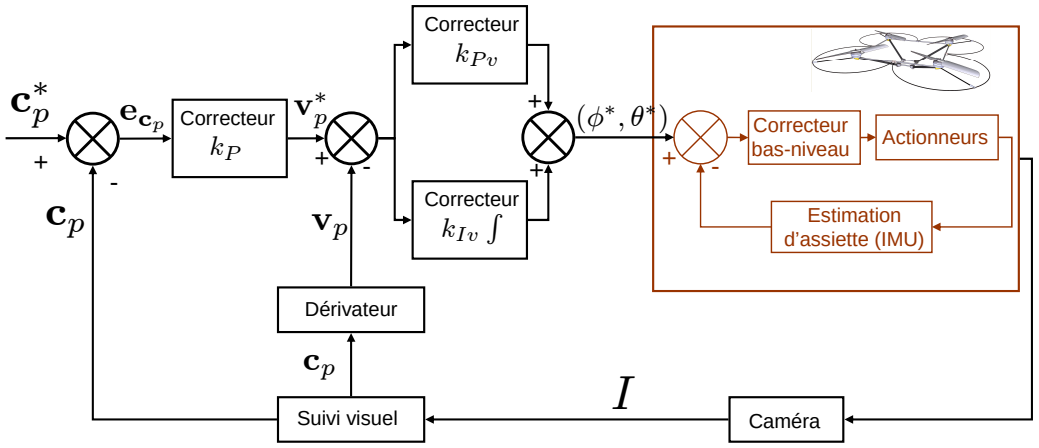


FIGURE 4.5 – *Contrôle du mouvement de translation.*

4.2.3 Altitude

Le système de vision présenté dans le chapitre 3 fournit une estimation du facteur d'échelle de l'objet dans l'image, qui constitue une information d'altitude relative $s = \sqrt{\frac{S}{S_0}}$ (voir équation (3.21)). Cependant, la vitesse obtenue par dérivation discrète de s s'est avérée trop bruitée pour être exploitable dans une commande hiérarchique du même type que celle utilisée pour la commande en translation. En l'absence de mesure de vitesse de translation selon l'axe $\mathbf{z}_c$, dans les validations expérimentales mises en place, l'altitude est contrôlée à l'aide des mesures d'un capteur barométrique comme proposé dans [Guenard 06]. L'objet suivi évoluant dans un plan, le contrôleur vise à conserver une altitude constante au cours du déplacement.

4.2.4 Conclusion

Dans cette section nous avons décrit la commande proposée pour contrôler l'orientation en lacet et la position en x et y d'un drone de type quadrirotor à partir d'informations de position relative fournies par un algorithme de vision robuste. On note que cette commande entre dans le cadre de l'asservissement visuel indirect (voir section 2.2.1), les consignes d'orientation étant supposées atteintes quasi-instantanément grâce à un contrôleur spécifique dont l'échelle de temps est plus rapide. Afin de valider l'approche proposée, la section suivante présente les résultats expérimentaux obtenus sur un système réel sur lequel cette commande a été implémentée.

4.3 Validation expérimentale

Le système complet (vision et commande) a été testé en situation réelle sur un quadrirotor développé au CEA LIST (Figure 4.6). Cette section présente le dispositif expérimental et les résultats obtenus.

4.3.1 Présentation du quadrirotor

Après avoir présenté une modélisation générique de quadrirotor dans la section 4.1 nous présentons ici plus spécifiquement le modèle de drone utilisé pour les expérimentations, à savoir un drone quadrirotor développé au CEA LIST (Figure 4.6). Le quadrirotor est constitué de 4 hélices à pas fixes situées aux extrémités d'une structure rigide en forme de croix de 80cm d'envergure. Les pales sont actionnées par 4 moteurs électriques sans balais, ou *brushless*, développés par la société AirRobot® [2] et alimentés par une batterie Lithium-Polymère. Le poids du drone est d'environ 700g avec une charge utile de 200g. Son autonomie atteint environ 20 minutes en régime quasi stationnaire avec une batterie neuve. Le drone est contrôlé à distance à partir d'un poste au sol par l'intermédiaire d'une liaison radio.

FIGURE *4.6 – Quadrirotor utilisé.*

2. http ://www.airrobot.de

Équipement embarqué. Le drone est équipé d'une architecture électronique distribuée constituée d'un empilement de 5 cartes électroniques incluant chacune un micro-contrôleur (voir Figure 4.7) :

- **la carte moteurs** gère la régulation des vitesses de rotation des moteurs et met en forme les différentes tensions d'alimentation nécessaires aux autres cartes électroniques ;
- **la carte centrale inertielle** (IMU) embarque 3 accéléromètres et 3 gyromètres qui mesurent l'accélération et la vitesse de rotation du véhicule sur les 3 axes du repère $\mathcal{R}_c$;
- **la carte capteurs** est dédiée aux autres capteurs utilisés dans certaines applications, notamment un baromètre et des capteurs ultrason et infrarouge ;
- **la carte DSP** (*Digital Signal Processor*) est le cerveau du drone. Elle reçoit les informations des différents capteurs, implémente les boucles de contrôle, et envoie les consignes à la carte moteurs. Le DSP, cadencé à 120MHz intègre en particulier les algorithmes d'estimation de l'orientation du drone [Metni 06] ainsi que la régulation de l'assiette [Guenard 06] qui s'effectue à 166Hz ;
- **la carte transmission HF** permet la réception des commandes issues du poste au sol (poussée, assiette et vitesse de rotation en lacet désirées) et l'envoi des données inertielles (assiette estimée et vitesse de rotation du véhicule). Elle utilise pour cela un module de communication sans fil utilisant le protocole ZigBee.

Les images acquises par la caméra sont envoyées vers le poste au sol par l'intermédiaire d'une liaison HF analogique 2.4GHz. La caméra est calibrée, sa focale est d'environ 3.6mm.

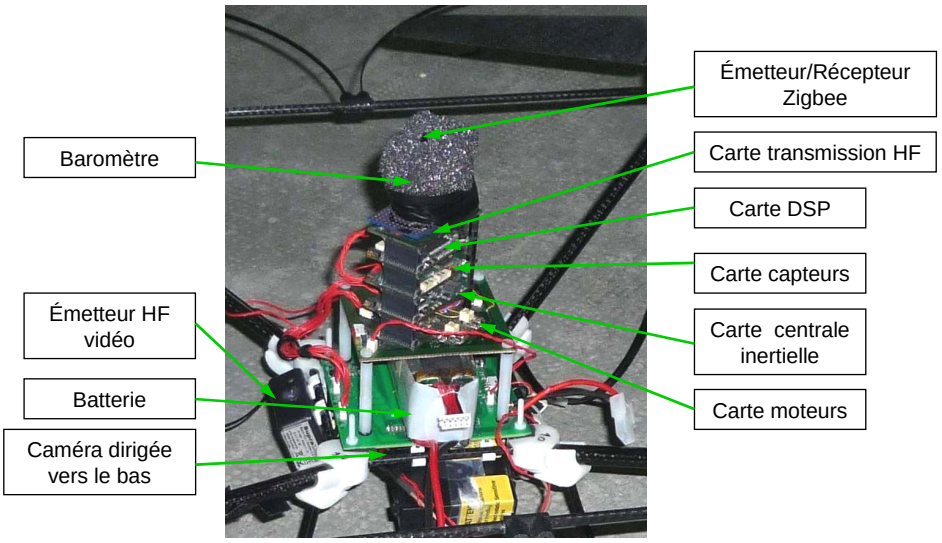


FIGURE 4.7 – *Équipement embarqué sur le drone.*

Station au sol. Le poste au sol a plusieurs fonctions. Il permet d'une part la réception des données (notamment images) du drone dans une optique de supervision

d'une mission par un opérateur humain et d'enregistrement, ainsi que le pilotage en mode manuel à l'aide d'un joystick. D'autre part, les calculs de vision étant trop complexes pour être implémentés directement sur le DSP embarqué, le poste au sol fournit la puissance de calcul nécessaire à l'exécution de ce type d'algorithmes. Cette configuration déportée présente cependant des inconvénients : les délais d'acheminement des données non négligeables engendrent des retards, et les données des différents capteurs ne sont pas synchronisées, ce qui complexifie la mise en œuvre de la commande. En pratique ces difficultés se traduisent par une limitation des gains et de la fréquence de la boucle de commande que l'on pourra mettre en place.

Les images envoyées par le drone sont reçues par un récepteur HF et transformées à l'aide d'un convertisseur analogique/numérique *GrabBeeX* avant d'être envoyées au PC. Au niveau logiciel, la réception et la mise en forme des images, de taille 320×240, utilise la bibliothèque *VideoInput*[3]. Les algorithmes développés sur le poste sont implémentés en C/C^{++} et les fonctions élémentaires de vision utilisent la bibliothèque open-source *ViSP*[4] [Marchand 05]. Une correction de la distorsion radiale est appliquée à chaque image. L'algorithme de suivi visuel d'une part et la réception des données inertielles et le calcul et l'envoi de la commande d'assiette d'autre part, tournent dans des fils d'exécution (*threads*) séparés. La cadence de l'algorithme de vision est de 20Hz.

Le dispositif expérimental complet est illustré Figure 4.8.

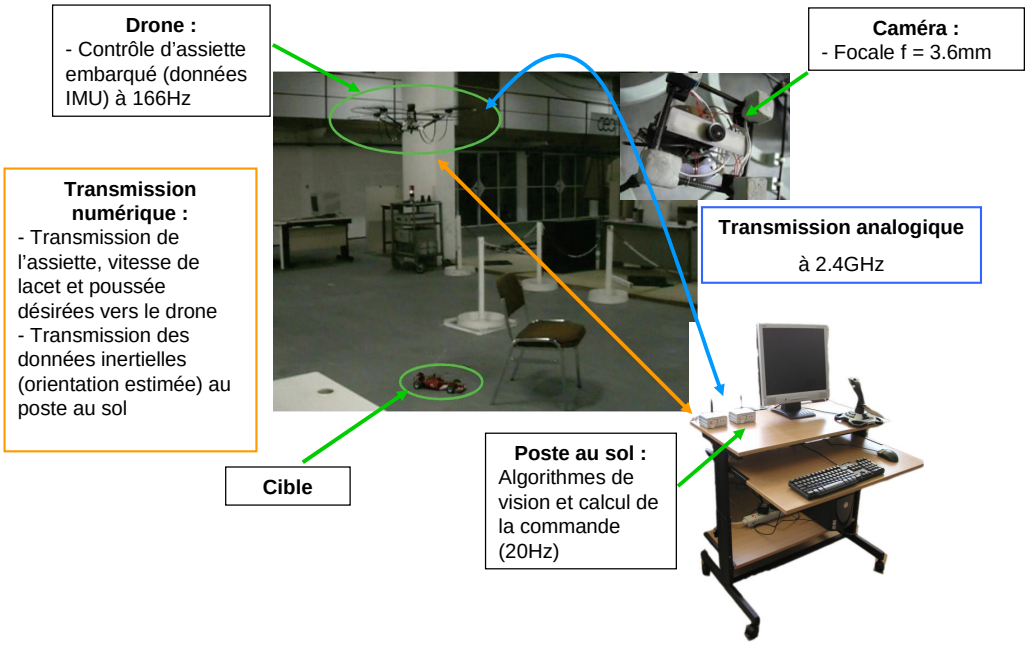


FIGURE 4.8 – *Dispositif expérimental*

3. http ://muonics.net/school/spring05/videoInput
4. http ://www.irisa.fr/lagadic/visp/visp.html

4.3.2 Résultats expérimentaux

Nous présentons maintenant les expériences réalisées pour valider le système proposé. Le système a été testé pour des tâches de stabilisation et de poursuite, l'objet à suivre étant une voiture télécommandée (Figures 4.16 et 4.17). L'algorithme de suivi est le filtre particulaire multi-noyaux présenté dans le chapitre 3. Il utilise 200 particules et s'exécute à 20Hz.

4.3.2.1 Tâche de stabilisation

Nous présentons tout d'abord une tâche de stabilisation pour valider le bon comportement du système en vol stationnaire au-dessus d'une cible fixe. La Figure 4.9 représente la trajectoire de la voiture dans le plan image de la caméra pour cette tâche de stabilisation.

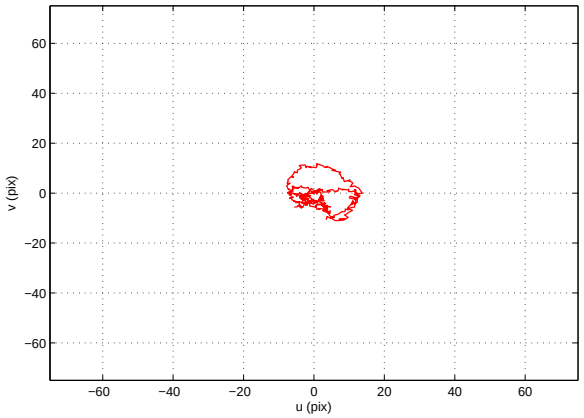


FIGURE 4.9 – *Trajectoire du centre de la voiture dans l'image, en coordonnées pixelliques. L'origine correspond à la position désirée.*

La voiture reste au centre de l'image comme spécifié par la consigne, avec une erreur moyenne de 1.3 pixels sur chaque axe et un écart-type de 5.3 pixels.

L'altitude ${}^pZ$ du drone est inconnue par l'algorithme. Cependant, pour donner une idée de l'erreur de position en coordonnées métriques $({}^pX\ {}^pY) = {}^pZ\ \mathbf{c}_p$, nous l'avons ici calculée à partir d'une estimation de l'altitude. On utilise pour cela le facteur d'échelle estimé par l'algorithme de vision. L'altitude estimée est donnée par : ${}^p\hat{Z} = \sqrt{\frac{S^*}{S}}$ où S^* est la surface réelle évaluée en mesurant les dimensions de la petite voiture et S sa surface dans le plan image normalisé, telle qu'elle est obtenue par l'algorithme de vision : $S = s^2\ S_0$. Les Figures 4.10 et 4.11 représentent ainsi respectivement l'erreur de position métrique entre le drone et l'objet et l'erreur de vitesse. L'écart-type de positionnement est de l'ordre de 3cm.

La Figure 4.12 montre que l'erreur de lacet $(\psi^* - \widehat{\psi})$ est inférieure à 3 degrés, avec un écart-type de 1 degré.

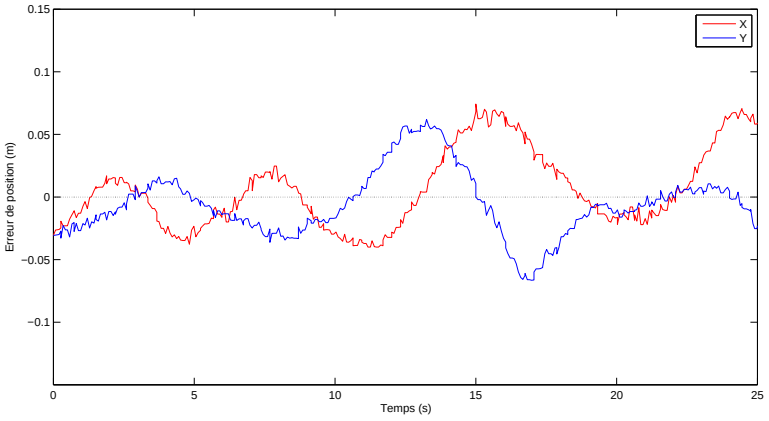


FIGURE 4.10 – *Erreur de position sur les axes x (rouge) et y (bleu) pour la tâche de stabilisation.*

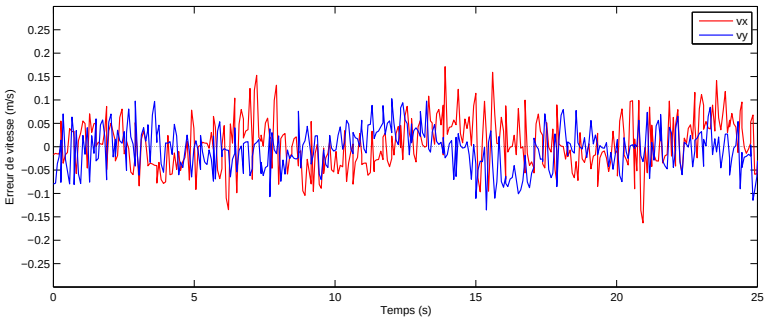


FIGURE 4.11 – *Erreur de vitesse sur les axes x (rouge) et y (bleu) pour la tâche de stabilisation.*

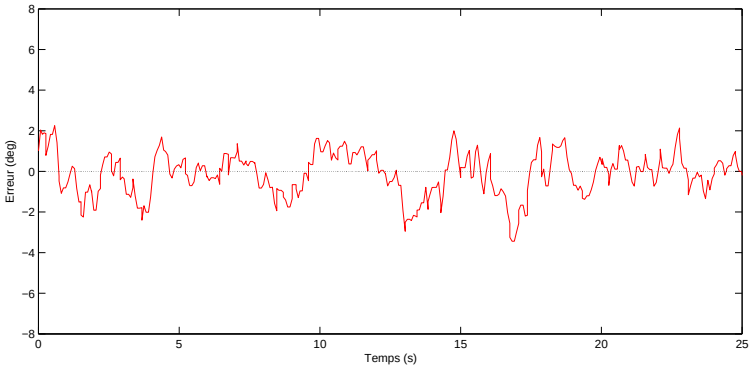


FIGURE 4.12 – *Erreur de lacet pour la tâche de stabilisation.*

4.3.2.2 Tâche de poursuite

Dans un deuxième temps le système a été testé avec une cible en mouvement sur des séquences de plusieurs minutes. Les tests réalisés comprennent différents types de déplacements, lignes droites, virages, occultations complètes (passages sous une chaise) et des variations de vitesse de la cible.

Positionnement en translation. La Figure 4.13 représente la trajectoire de la cible pour une séquence de 5 minutes. Le déplacement en translation de la voiture ayant lieu principalement sur l'axe y, on observe naturellement une erreur de position plus importante sur cet axe. L'erreur moyenne est ainsi de 2 pixels sur l'axe x et 14 pixels sur l'axe y, avec des écarts-types de 10 et 30 pixels respectivement. Dans le cas où la vitesse est quasi constante, la commande intégrale permet de compenser l'erreur de traînage. Le fait que la position moyenne soit positive sur l'axe y traduit le délai de réactivité de la commande face à des accélérations de la voiture vers l'avant. La Figure 4.13 montre cependant que la voiture reste dans le champ de vision de la caméra ce qui permet la bonne réalisation de la tâche de poursuite.

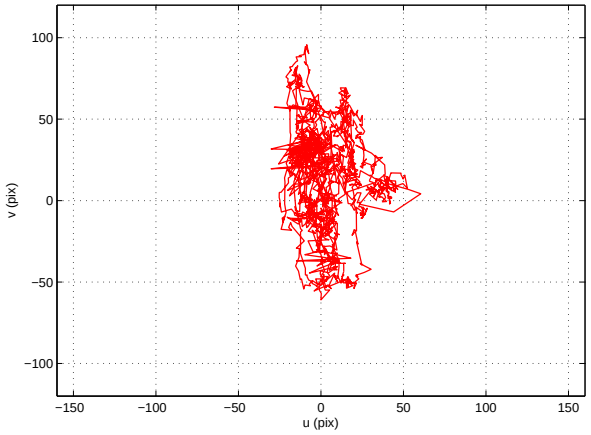


FIGURE *4.13 – Trajectoire du centre de la voiture dans l'image, en coordonnées pixelliques. L'origine correspond à la position désirée.*

De la même façon que pour la tâche de stabilisation, on a également évalué l'erreur métrique correspondant à la tâche de poursuite (voir Figure 4.14). L'erreur de position moyenne est de 1cm sur l'axe x et 7cm sur l'axe y pour des écarts-type respectifs de 7cm et 15cm. L'altitude du drone est de l'ordre de 2m.

L'erreur de vitesse est représentée Figure 4.15. Les plages constantes autour de $t = 75$s et $t = 200$s correspondent à des moments où l'objet était considéré comme temporairement perdu. En pratique l'erreur de vitesse continue à être estimée pendant ces phases mais nous souhaitions ici faire apparaître les zones d'occultation.

Lacet. La Figure 4.16 illustre la régulation du lacet lors d'un virage. L'algorithme de suivi permet d'estimer l'orientation de la voiture dans l'image et le régulateur de lacet permet au drone de suivre les virages du véhicule.

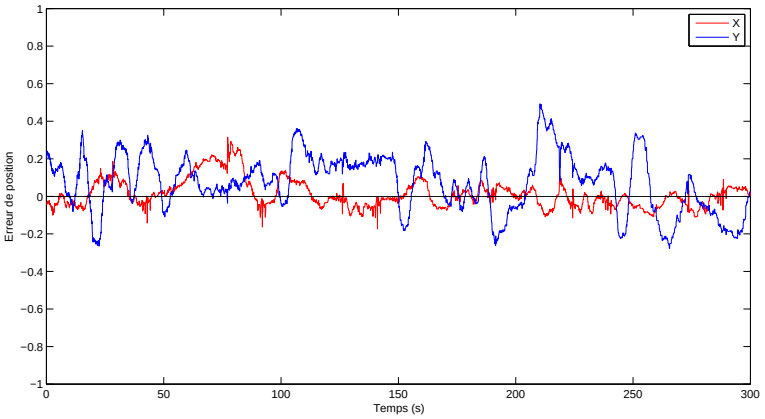


FIGURE 4.14 – *Erreur de position sur les axes x (rouge) et y (bleu) pour la tâche de poursuite.*

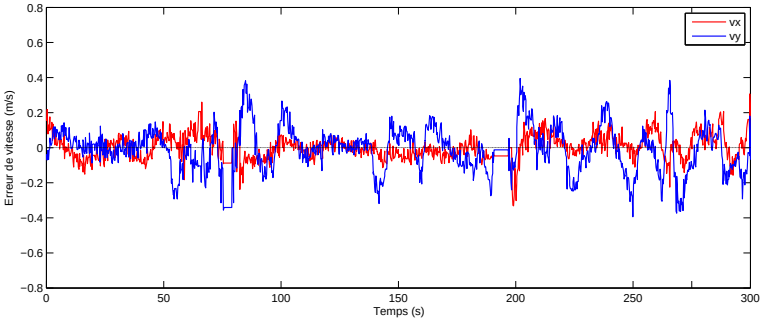
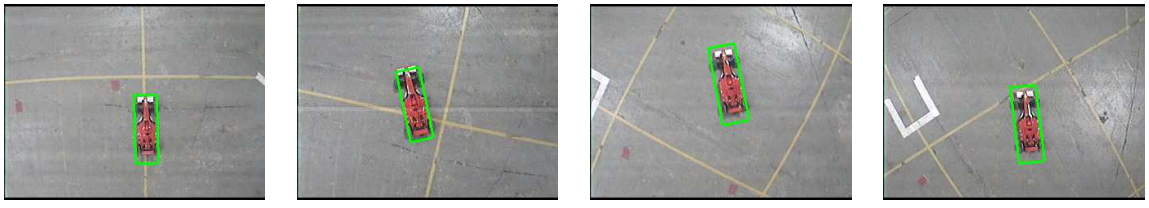


FIGURE 4.15 – *Erreur de vitesse sur les axes x (rouge) et y (bleu) pour la tâche de poursuite.*



(1) (2) (3) (4)

FIGURE 4.16 – *Exemple de virage : L'algorithme de suivi estime la rotation de lacet et le drone suit la rotation pour maintenir la même orientation de la voiture dans l'image.*

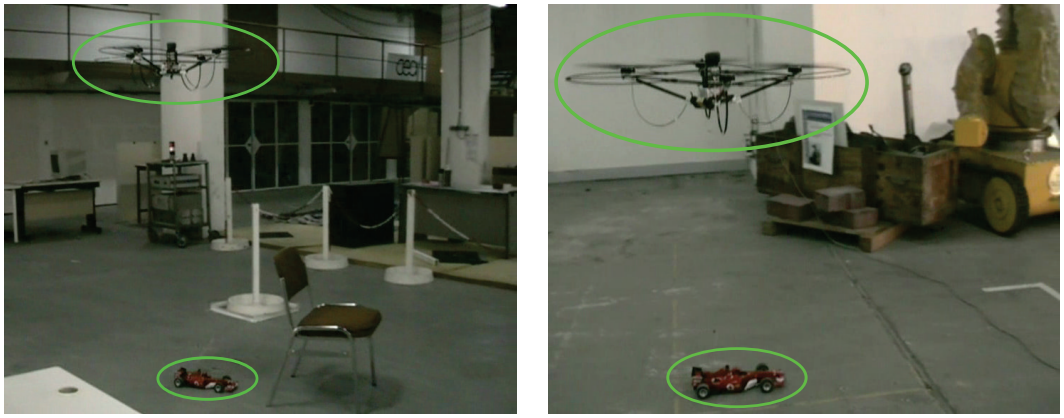
Altitude. La Figure 4.17 montre que la régulation de l'altitude à l'aide des mesures barométriques reste peu précise. Ceci est dû à la présence de perturbations (ventilation, courants d'air) auxquelles ce type de capteur est sensible, et à la variance importante de la mesure (de l'ordre du mètre). Cette imprécision est d'autant plus gênante qu'elle est du même ordre de grandeur que l'altitude de vol du drone (2m environ). Si le drone volait à une altitude plus élevée l'incidence serait beaucoup plus faible. Nos expériences mettent toutefois en évidence la robustesse de l'algorithme de suivi à des changements d'échelle importants, comme illustré Figure 4.17. La fusion avec une information relative d'altitude telle que celle obtenue par notre système de vision n'a pas été testée dans le cadre de nos travaux mais constitue une piste d'amélioration à envisager.

Occultation et gestion des échecs. Lorsque l'objet est perdu, la position estimée par le filtre particulaire est représentée en orange sur les images (Figure 4.18-b). Les occultations sont identifiées avec succès et le filtre se base alors sur la prédiction seule. Lorsque la voiture est détectée à nouveau (la position estimée est alors tracée en vert dans les images), l'algorithme de suivi utilise à nouveau le rééchantillonnage pondéré en fonction de la mesure de vraisemblance. Le modèle de prédiction utilisé étant un modèle à vitesse de translation constante, l'estimation du filtre peut être erronée si la voiture subit une forte accélération, un fort ralentissement ou un changement d'orientation important pendant une occultation. Cependant, en l'absence de rééchantillonnage, le bruit de modèle étend le nuage de particules qui couvre alors une zone plus importante de l'espace d'état, permettant le plus souvent de retrouver la cible avec succès.

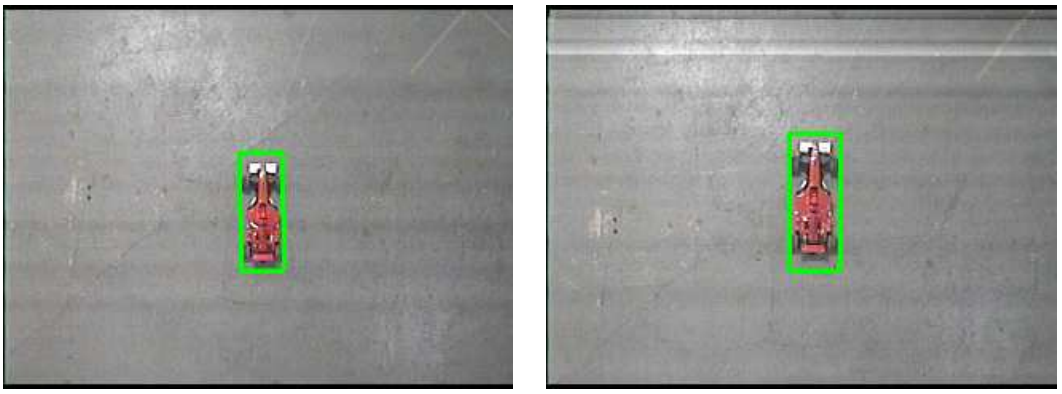
Nous présentons Figure 4.19-a un exemple d'échec dû à un changement important de l'orientation de la voiture lors d'une occultation. Dans la mesure où la cible reste dans le champ de vision de la caméra, la procédure de réinitialisation présentée section 3.4.2 permet toutefois au système de récupérer de ce type d'échecs 4.19-b.

4.4 Conclusion

Dans ce chapitre nous avons étudié la problématique de la commande d'un minidrone pour une tâche de poursuite de cible mobile. Après avoir présenté la modélisation du quadrirotor utilisé, nous avons décrit la mise en place d'une commande hiérarchique pour réguler la position et le lacet du drone. Le système proposé a été validé expérimentalement dans des tâches de positionnement avec une cible statique puis une cible mobile. Les résultats présentés comprennent des occultations, virages, changements d'échelle auxquels l'algorithme de suivi s'est montré robuste. Le régulateur a permis d'obtenir des erreurs de position faibles (15 cm d'écart-type en translation, quelques degrés en lacet) sur des séquences comprenant jusqu'à 10 000 images environ ce qui démontre la robustesse de l'approche proposée.



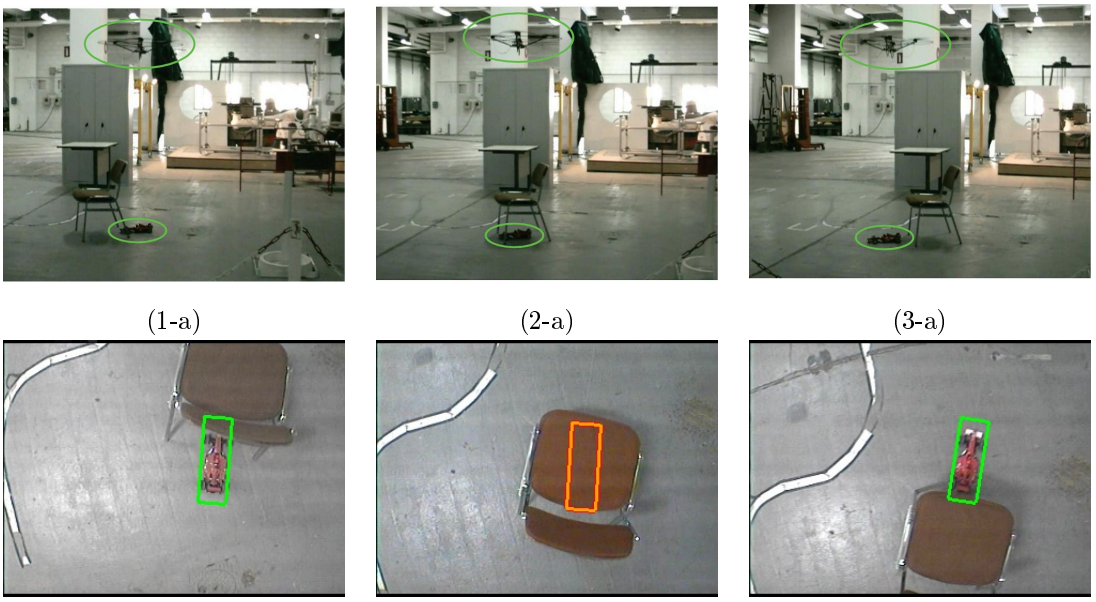
(1-a) (1-b)



(2-a) (2-b)

(2-c) (2-d)

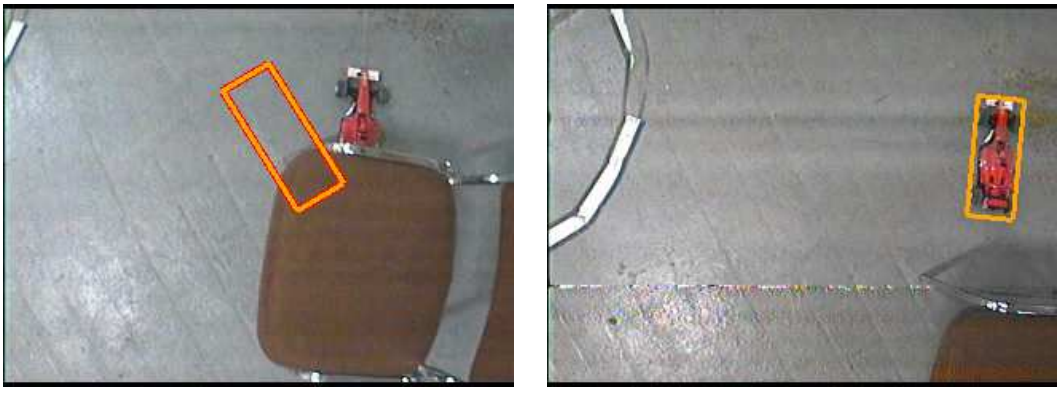
FIGURE *4.17 – Changements d'échelle.*



(1-a) (2-a) (3-a)

(1-b) (2-b) (3-b)

FIGURE 4.18 – *Exemple d'occultation totale.*



(a) (b)

FIGURE 4.19 – *Exemple d'échec (a) et réinitialisation (b).*

Deuxième partie

Positionnement

La navigation des drones repose essentiellement à l'heure actuelle sur la fusion GPS/centrale inertielle. En environnement urbain ou à l'intérieur de bâtiments, le signal GPS est toutefois rarement disponible. Pour se localiser et naviguer dans ce type d'environnement, l'utilisation d'informations visuelles constitue une alternative intéressante. Dans cette partie, nous considérons ainsi une tâche de positionnement 3D et navigation d'un drone dans un environnement structuré.

D'un point de vue vision, il ne s'agit donc plus de suivre un objet « à un facteur d'échelle près » mais d'estimer la pose complète de la caméra. La commande qui en résultera sera donc un asservissement visuel 3D (voir section 2.2.2.1). Le chapitre 5 présente la méthode de calcul de pose proposée, puis le chapitre 6 décrit l'utilisation de cette pose dans le cadre du positionnement d'un quadrirotor.

Chapitre 5

Calcul de pose

Dans ce chapitre nous nous intéressons à l'estimation de la transformation rigide complète (pose) entre la caméra et l'environnement dans lequel elle évolue. L'enjeu de l'estimation de pose par la vision consiste à utiliser des informations issues d'une projection 2D pour reconstituer le lien géométrique 3D entre la caméra et la scène qu'elle observe.

Il existe pour cela un grand nombre d'approches possibles qui dépendent à la fois du type de primitives visuelles utilisées et de la méthode de résolution proprement dite, ainsi que du niveau de connaissance *a priori* sur la scène 3D. On pourra se reporter à [Lepetit 05] pour une synthèse des travaux sur ce sujet. Sans vouloir en dresser un panorama exhaustif, on rappelle ici les orientations principales. De manière générale, l'estimation de pose par la vision se base sur la mise en correspondance entre des éléments de la scène 3D (points, arêtes d'un modèle, etc.) et leur projection dans l'image. Soient $\{{}^w\mathbf{X}_i\}_{i=1..n}$ un ensemble de n primitives 3D, et $\{\mathbf{x}_i\}_{i=1..n}$ les projections correspondantes observées dans l'image. La pose ${}^c\mathbf{M}_w$ de la caméra par rapport à la scène est alors estimée classiquement par la minimisation d'un critère de reprojection :

$$\widehat{{}^c\mathbf{M}_w} = arg\min_{{}^c\mathbf{M}_w} \sum_{i=1}^{n} d^2(\mathbf{x}_i, pr({}^c\mathbf{M}_w, {}^w\mathbf{X}_i)), \tag{5.1}$$

où d est une distance sur l'espace des primitives 2D, et $pr(\mathbf{M}, \mathbf{X})$ désigne la projection de la primitive $\mathbf{X}$ dans l'image pour une transformation rigide $\mathbf{M}$, les paramètres intrinsèques de la caméra étant supposés connus (voir la section 1.1). Dans la grande majorité des cas, les méthodes d'estimation de pose supposent que les primitives 3D $\{{}^w\mathbf{X}_i\}_{i=1..n}$ sont connues *a priori*. On parle alors d'approches « basées modèle ». Dans [Horaud 89] [Haralick 89] [Dementhon 95] [Lu 00] le modèle est composé de points dont les coordonnées 3D sont connues. [Lowe 91] [Drummond 02] [Pupilli 06] [Comport 06] utilisent les arêtes d'un modèle 3D. D'autres types de primitives ont été considérées telles que des coniques ou des objets cylindriques. Enfin, certaines approches dites « hybrides » combinent différents types de primitives 3D

[Phong 95] [Marchand 02]. En ce qui concerne le cadre de résolution, de la même façon que pour le suivi basé noyaux présenté dans le chapitre 3, le problème du calcul de pose basé modèle a été abordé dans la littérature à la fois dans le cadre d'une formulation déterministe et dans le cadre bayésien. La résolution de l'équation (5.1) peut parfois s'exprimer de façon analytique [Horaud 89] [Fischler 81]. Si certaines approches linéaires de type moindres-carrés sont utilisées dans certains cas, il s'agit toutefois d'un problème intrinsèquement non linéaire abordé le plus souvent par des méthodes d'optimisation non linéaires itératives [Lowe 91] [Drummond 02] [Vacchetti 04] [Comport 06], moins sensibles au bruit de mesure (voir section 1.3.3). L'utilisation du filtrage particulaire est apparue plus récemment dans l'estimation de pose. On peut citer notamment les travaux de [Pupilli 06] [Klein 06] [Nuske 08].

Lorsqu'aucune information *a priori* sur la scène n'est disponible, une approche possible consiste à estimer simultanément la pose de la caméra et la position 3D d'éléments $\mathbf{X}_i$ de la scène, dont la projection $\mathbf{x}_i$ est mise en correspondance dans les vues successives de la caméra. Cette approche est connue sous le nom de SLAM (*Simultaneous Localisation And Mapping*) ou *Structure From Motion* [DurrantWhyte 06] [Mei 07]. En l'absence de modèle 3D de référence, la pose et la carte des primitives 3D sont exprimées relativement à un système de coordonnées défini arbitrairement. En pratique, on se réfère souvent à une pose initiale dont une estimation peut être obtenue à l'aide d'un modèle. La caméra joue alors le rôle de capteur odométrique en estimant à un facteur près la transformation entre deux positions successives de la caméra à partir de la transformation 2D entre les images. La mise en correspondance des primitives dans l'image est utilisée pour reconstruire leur position 3D. Pour limiter les problèmes de dérive dus à l'accumulation d'erreurs d'estimation, des techniques d'ajustement de faisceau (*bundle adjustment*) permettent d'ajuster les paramètres à estimer en prenant en compte les mesures d'un ensemble d'images mais sont relativement lourdes à mettre en œuvre. Une autre approche classique pour le SLAM consiste à utiliser un filtre de Kalman étendu (EKF, voir section 1.3.4.2) dont l'état contient à la fois les paramètres de position de la caméra et les positions 3D des primitives de la carte de l'environnement reconstruite. Le modèle de prédiction permet alors de limiter la zone de recherche lors de la mise en correspondance 2D. Lorsque la mise en correspondance est réalisée, la position des primitives $\mathbf{X}_i$ est mise à jour.

Pour éviter les problèmes de dérive, et favoriser une plus grande robustesse de l'estimation de pose, nous avons choisi d'utiliser une approche basée sur un modèle 3D connu *a priori*. Disposer d'un modèle de l'environnement est une hypothèse forte dans le cadre de missions d'exploration notamment en extérieur. En revanche, pour des tâches d'inspection de bâtiments ou d'ouvrages d'art, un modèle 3D de type CAO peut souvent être disponible. Dans ce chapitre, on supposera ainsi que l'on dispose d'un modèle 3D de l'environnement ou d'un objet spécifique d'intérêt. Notre objectif consiste alors à estimer la pose relative entre la caméra et l'environnement modélisé au cours du déplacement de la caméra. En d'autres termes, il s'agit de trouver la pose qui fournit le meilleur recalage entre la projection des primitives du modèle (ici les arêtes du modèle) et les primitives observées dans l'image (les contours, caractérisés par de forts gradients) [Lowe 92] [Drummond 02] [Comport 06] (voir Figure 5.1).

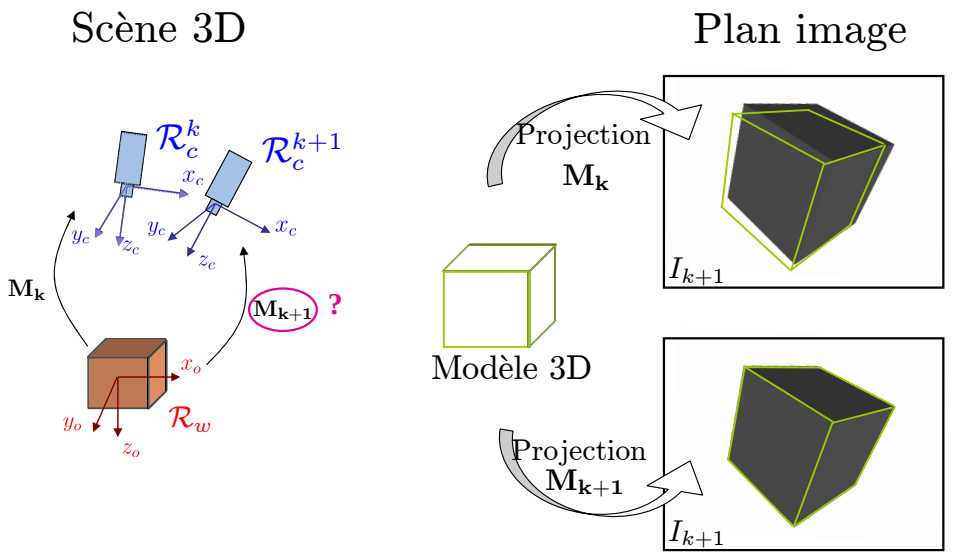


FIGURE 5.1 – *Suivi de pose basé modèle. Dans cet exemple, la pose $\mathbf{M}_k$ de la caméra à un instant k donné est connue et on souhaite déterminer la pose $\mathbf{M}_{k+1}$ à partir de la nouvelle image acquise à l'instant $k+1$. La pose $\mathbf{M}_{k+1}$ est celle qui donnera le meilleur recalage dans l'image entre les contours observés et la projection des arêtes du modèle.*

Les environnements urbains, industriels ou l'intérieur de bâtiments sont des environnements structurés dans lesquels les arêtes sont des éléments visuels fréquents. Ces informations visuelles sont, de plus, peu sensibles aux changements d'illumination ou aux changements d'angle de vue, et faciles à détecter même en présence de bruit ou de flou dans l'image, ce qui fait d'eux des éléments particulièrement intéressants dans une optique de suivi visuel. Cependant, à l'inverse des points d'intérêt qui sont généralement choisis de manière à être facilement mis en correspondance [Shi 94], les contours résultant de la projection des arêtes du modèle ont souvent des apparences très similaires. Leur mise en correspondance d'une image à l'autre peut être confrontée à des ambiguïtés, notamment lorsque les projections d'arêtes distinctes deviennent proches dans une image. Les erreurs de mise en correspondance peuvent mettre en échec l'algorithme de suivi. L'un des enjeux des travaux présentés dans ce chapitre consistera à proposer une approche d'estimation de pose robuste à ce type de difficultés.

La suite de ce chapitre s'organise de la façon suivante. Dans un premier temps la section 5.1 décrit le principe des méthodes de suivi par optimisation déterministe existantes. Les particularités des techniques d'optimisation dans l'espace $SE(3)$ sont introduites, et le cas de primitives sous forme d'arêtes est présenté. La section 5.1.3 propose alors un algorithme permettant de générer plusieurs hypothèses sur la pose à partir d'hypothèses détectées lors de la mise en correspondance des contours au bas niveau. La section 5.2.4 montre comment ces hypothèses peuvent être intégrées dans le cadre d'un filtrage particulaire. Enfin, la section 5.3 met en évidence les performances de l'approche multi-hypothèses proposée par rapport aux approches

classiques de suivi basé modèle à travers des résultats expérimentaux.

5.1 Le suivi de pose basé modèle sous forme d'optimisation déterministe

Dans cette section nous décrivons le problème du suivi de pose dans le cadre des méthodes d'optimisation déterministe.

5.1.1 Estimation de pose par optimisation déterministe

Comme introduit dans l'équation (5.1), le problème général de l'estimation de la pose ${}^c\mathbf{M}_w$ de la caméra par rapport à l'environnement peut s'écrire sous la forme :

$$\widehat{{}^c\mathbf{M}_w} = arg\min_{{}^c\mathbf{M}_w} \sum_{i=1}^{n} d^2(\mathbf{x}_i, pr({}^c\mathbf{M}_w, {}^w\mathbf{X}_i)). \tag{5.2}$$

Une approche classique pour résoudre ce type de problème et déterminer les paramètres de pose consiste à utiliser un algorithme de type Gauss-Newton. Comme présenté dans la section 1.3.3, les méthodes d'optimisation non-linéaires de type Gauss-Newton visent à minimiser itérativement une fonction de coût. À chaque itération i, un déplacement est effectué dans l'espace des paramètres :

$$\boldsymbol{\mu}_{i+1} = \boldsymbol{\mu}_i \oplus \delta_i, \tag{5.3}$$

où $\oplus$ est une loi de composition interne sur l'espace des paramètres considéré. Dans le cas du calcul de pose, l'espace des paramètres considéré correspond à $SE(3)$, espace dans lequel l'addition n'est pas une loi de composition interne (la somme arithmétique de deux rotations n'est pas une rotation). Pour définir un déplacement incrémental dans l'espace des transformations rigides $SE(3)$, on utilise une propriété caractéristique des groupes de Lie qui affirme l'existence d'une application bijective entre un groupe de Lie et son algèbre associée [Warner 83]. $SE(3)$ est en effet un groupe de Lie dont l'algèbre associée $se(3)$ est définie par :

$$se(3) = \left\{ \boldsymbol{\xi} = \begin{bmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{v} \\ 0 & 0 \end{bmatrix} \mid [\boldsymbol{\omega}]_\times \in so(3), \boldsymbol{v} \in \mathbb{R}^3 \right\} \subset \mathbb{R}^{4\times 4}.$$

On définit ainsi l'application :

$$\begin{aligned} se(3) &\mapsto SE(3) && (5.4) \\ \boldsymbol{\xi} &\mapsto \mathbf{M} = exp(\boldsymbol{\xi}), && (5.5) \end{aligned}$$

connue dans la littérature anglophone sous le nom d'*exponential map*. L'avantage de cette formulation réside dans le fait que les notions d'addition et de distance qui n'étaient pas définies dans le groupe $SE(3)$ sont en revanche définies dans l'algèbre $se(3)$. La définition de $se(3)$ ci-dessus fait apparaître une paramétrisation évidente des éléments de cet espace sous la forme de vecteurs de 6 paramètres : $\mathbf{v} = (\boldsymbol{v}, \boldsymbol{\omega})$ avec $\boldsymbol{v}$, $\boldsymbol{\omega} \in \mathbb{R}^3$. La notation choisie est justifiée par le fait que le vecteur $\mathbf{v}$ peut

être interprété comme un torseur cinématique. En effet, si $\boldsymbol{v}$ désigne une vitesse de translation et $\boldsymbol{\omega}$ une vitesse de rotation, alors la transformation rigide $\delta\mathbf{M}$ générée par l'application du torseur cinématique $\mathbf{v} = (\boldsymbol{v}, \boldsymbol{\omega})$ sur un intervalle de temps δt est donnée par :

$$\delta\mathbf{M} = exp(\mathbf{v}\delta\mathbf{t}) \tag{5.6}$$

où on s'autorise l'abus de notation suivant :

$$exp(\mathbf{v}) = exp\left(\begin{bmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{v} \\ 0 & 0 \end{bmatrix}\right). \tag{5.7}$$

L'équation (5.3) devient alors :

$$\mathbf{M}_{i+1} = \mathbf{M}_i \; exp(\mathbf{v}\delta t). \tag{5.8}$$

Le calcul de la vitesse $\mathbf{v}$ peut être réalisé en interprétant cette vitesse comme le torseur cinématique que l'on appliquerait à la caméra dans le cadre d'un asservissement visuel pour réguler l'erreur de reprojection (5.2) à zéro [Comport 06].

5.1.2 Suivi de pose basé sur un modèle CAO

Après avoir présenté la formulation générale du problème d'estimation de pose, on considère ici le cas où le modèle est un modèle CAO constitué d'arêtes de positions 3D connues dans $\mathcal{R}_w$ [Drummond 02] [Vacchetti 04] [Comport 06]. Si le modèle est composé d'un ensemble $\{L_i\}$ d'arêtes, on note E_i la projection dans l'image d'une arête L_i du modèle selon la pose ${}^c\mathbf{M}_w$ entre la caméra et la scène :

$$E_i = L_i({}^c\mathbf{M}_w). \tag{5.9}$$

Dans l'image, les arêtes des objets modélisés correspondent généralement à des zones de forts gradients. Le critère à minimiser s'exprime classiquement sous la forme d'une distance entre les arêtes projetées E_i et des éléments de l'image identifiés comme correspondant potentiellement à des contours. Dans [Lowe 92], les forts gradients de l'image sont détectés et regroupés en segments. [Drummond 02] et [Comport 06] utilisent des points extraits des contours : chaque contour projeté E_i est d'abord échantillonné en un ensemble de points $\{e_{i,j}\}$. En chacun de ces points on recherche alors les points de fort gradient le long de la normale à E_i (voir Figure 5.2). Dans [Drummond 02] le point de plus fort gradient rencontré lors de l'exploration de la normale est sélectionné, tandis que [Comport 06] retient le point correspondant à un maximum de vraisemblance par rapport au point initial $e_{i,j}$. Le point sélectionné sera noté $e'_{i,j}$ dans la suite. La pose recherchée correspond à :

$$\widehat{{}^c\mathbf{M}_w} = arg\min_{{}^c\mathbf{M}_w} \sum_{i,j} d^2_\perp(E_i, e'_{i,j}) \tag{5.10}$$

où $d_\perp(E_i, e'_{i,j}) = d_\perp(L_i(\mathbf{M}), e'_{i,j})$ est la distance du point $e'_{i,j}$ à la droite portant le segment E_i. Le critère à minimiser s'écrit alors sous la forme :

$$S(\mathbf{M}) = \frac{1}{N_e} \sum_i \sum_j d^2_\perp(L_i(\mathbf{M}), e'_{i,j}) \tag{5.11}$$

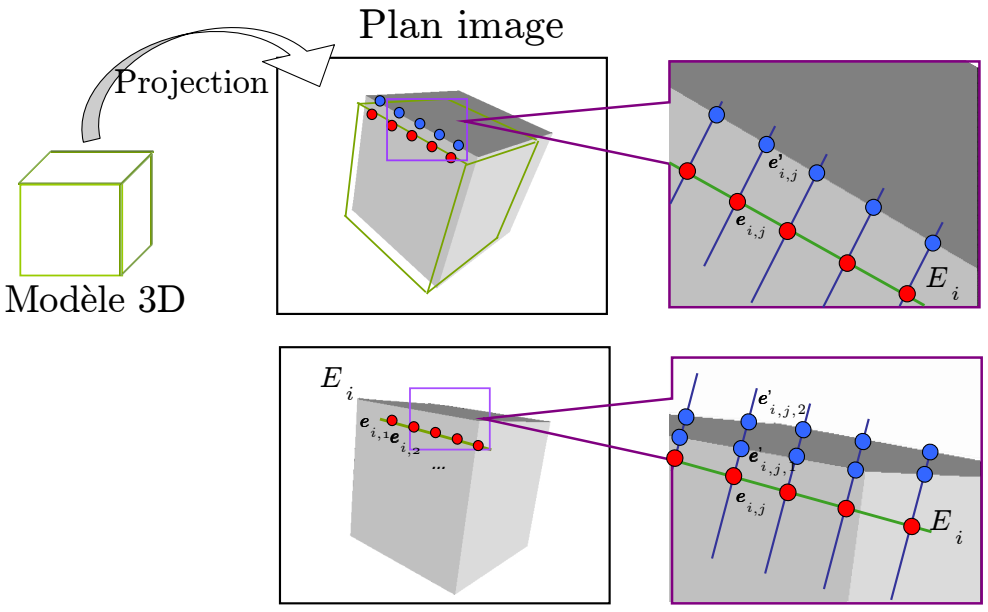


FIGURE 5.2 – *Dans une approche classique, les arêtes du modèle sont projetées dans le plan image et échantillonnées par un ensemble de points, puis une recherche est effectuée le long de la normale (en haut). Quand plusieurs contours sont proches dans l'image, des ambiguïtés apparaissent lors de la recherche le long de la normale (en bas).*

où N_e est le nombre total de points échantillonnés.

Ce critère peut alors être minimisé en fonction des paramètres de la pose $\mathbf{M}$ à l'aide d'une procédure d'optimisation non-linéaire itérative de type Gauss-Newton ou Levenberg-Marquardt (voir section 5.1.1).

Vers plus de robustesse. Un des inconvénients de l'utilisation des contours est que des contours distincts ont souvent des apparences similaires dans une image. En pratique les points $e'_{i,j}$ sélectionnés sur la normale aux contours projetés peuvent donc contenir des points aberrants (*outliers*) correspondant à des éléments de fort gradient de l'image n'appartenant pas à la projection des arêtes L_i souhaitées. En particulier lorsque plusieurs contours sont proches dans l'image, l'exploration le long de la normale à la projection de l'arête du modèle peut présenter plusieurs points vraisemblables et le critère de sélection peut être mis en défaut, avec pour conséquence une perte possible du suivi (voir Figure 5.2).

Pour améliorer la robustesse du suivi face à ces ambiguïtés et mesures aberrantes, une possibilité consiste à utiliser une autre source d'information, en fusionnant l'information de contour avec l'information fournie par un autre type de primitive visuelle (des points d'intérêt par exemple [Vacchetti 04] [Pressigout 07]) ou un autre capteur [Klein 04].

Divers travaux proposent également d'améliorer la robustesse de la méthode en faisant appel aux techniques d'estimation robuste [Malis 06] qui attribuent un poids faible aux valeurs aberrantes au cours du processus d'optimisation [Armstrong 95], [Drummond 02], [Comport 06] de type IRLS (*Iterative Reweighted Least Square*). Le critère (5.11) à minimiser devient alors :

$$S(\mathbf{M}) = \frac{1}{N_e} \sum_i \sum_j \rho \left(d^2_\perp (L_i(\mathbf{M}), e'_{i,j}) \right) \tag{5.12}$$

où ρ est un estimateur robuste (voir section 1.3.3). Cette formulation ne conserve toutefois qu'une seule hypothèse de fort gradient par échantillon de contour. Dans des situations d'ambiguïtés (Figure 5.2) trop de mauvaises sélections conduiront encore à un échec du suivi.

L'idée de prendre en compte plusieurs hypothèses dans le processus d'optimisation a été introduite dans le suivi 3D par [Vacchetti 04]. Lors de l'exploration de la normale au contour E_i à partir du point $e_{i,j}$, plusieurs hypothèses $\{e'_{i,j,l}\}$ correspondant à des points de fort gradient sont mémorisées (voir Figure 5.2). Les différentes hypothèses sont prises en compte dans le processus d'optimisation en introduisant un estimateur multi-hypothèses ρ^* défini par :

$$\forall n \in \mathbb{N}, \forall (x_1, ..., x_n) \in \mathbb{R}^n, \rho^*(x_1, ..., x_n) = \min_i \rho(x_i). \tag{5.13}$$

L'équation (5.12) devient :

$$S^*(\mathbf{M}) = \frac{1}{N_e} \sum_i \sum_j \rho^* \left(d^2_\perp \left(L_i(\mathbf{M}), e'_{i,j,1} \right), ..., d^2_\perp \left(L_i(\mathbf{M}), e'_{i,j,n_{i,j}} \right) \right) \tag{5.14}$$

où $n_{i,j}$ est le nombre d'hypothèses candidates sélectionnées pour le point $e_{i,j}$. L'estimateur robuste ρ^* détermine les hypothèses à rejeter et améliore la robustesse de

l'algorithme de suivi. Il en résulte toutefois une seule hypothèse sur la pose, et en l'absence de filtrage temporel (Kalman ou filtre particulaire par exemple) les informations provenant des hypothèses sélectionnées à un instant donné ne sont pas utilisées à l'instant suivant.

Dans cette thèse nous proposons d'utiliser également différentes hypothèses $\{e'_{i,j,l}\}$ correspondant à des contours vraisemblables dans l'image. Cependant, à la différence de [Vacchetti 04], ces hypothèses au niveau des points vont être utilisées pour remonter à différentes hypothèses sur la pose recherchée. Cette approche est décrite dans la section suivante.

5.1.3 Suivi multi-hypothèses

5.1.3.1 Classification des hypothèses bas niveau en hypothèses sur les arêtes

Pour générer différentes hypothèses sur la pose de la caméra à partir des points candidats détectés au bas niveau, il est possible d'effectuer une procédure d'optimisation classique à partir de différents jeux de points. Il n'est cependant pas envisageable de considérer toutes les combinaisons possibles parmi les points candidats pour des raisons évidentes de charge de calcul rédhibitoire et du nombre important d'*outliers* par rapport au nombre d'*inliers*.

L'idée proposée ici consiste alors à identifier dans l'ensemble de points candidats $\{e'_{i,j,l}\}$ quelles sont les droites sous-jacentes les plus probables. On utilise pour cela un algorithme de classification supervisé de type *k-mean* [MacQueen 67] [Hartigan 75].

Le principe d'un algorithme de type *k-mean* est de segmenter (partitionner) itérativement un ensemble de données en k classes, de manière à minimiser la variance au sein de chaque classe. Il repose pour cela sur l'itération de deux étapes clefs :

- le calcul du *centroïde* de chaque classe ;
- l'affectation de chaque élément à la classe dont le centroïde est le plus proche.

L'algorithme du *k-mean* est par exemple utilisé classiquement pour réaliser une segmentation basique d'images, les données considérées étant alors les pixels de l'image, classifiés en fonction de leur intensité lumineuse comme décrit dans l'algorithme 7. Le centroïde de chaque classe correspond alors simplement à la moyenne des intensités lumineuses des pixels de la classe.

Dans le cas qui nous intéresse, il s'agit de classifier pour chaque arête projetée E_i les points candidats $\{e'_{i,j,l}\}$ en k_i classes $\{\mathcal{C}_1, \ldots, \mathcal{C}_{k_i}\}$, chaque classe correspondant à une hypothèse sur la projection E_i de l'arête L_i du modèle. Le centroïde de chacune des k_i classes correspond à la droite permettant de représenter au mieux l'ensemble des points de la classe par minimisation robuste au sens des moindres carrés. Pour initialiser l'algorithme, le nombre k_i de classes pour l'arête E_i est fixé au nombre maximum de points candidats détectés, à savoir $k_i = max_j\{n_{i,j}\}$. Les classes $\{\mathcal{C}^i_1, \ldots, \mathcal{C}^i_{k_i}\}$ sont ensuite initialisées en respectant l'ordre dans lequel les hypothèses sont rencontrées lors de l'exploration de la normale. On a alors pour chaque classe : $\mathcal{C}^i_l = \{e'_{i,j,l}\}_j$. Autrement dit, la première classe est initialisée avec l'ensemble des points rencontrés en premier lors de l'exploration de la normale à l'arête projetée, etc. Les classes ne contiennent donc pas nécessairement le même nombre de points.

Algorithme *k-mean* classique pour la segmentation d'une image en niveau de gris.
Soient $\{\boldsymbol{l}_i\}_{i=1...n}$ l'ensemble des pixels de l'image à segmenter et k le nombre de classes que l'on souhaite obtenir. L'algorithme est initialisé en répartissant les valeurs aléatoirement dans chacune des k classes $\{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$. La procédure se déroule alors itérativement de la façon suivante :

1. Pour chaque classe $\mathcal{C}_j \in \{\mathcal{C}_1, \ldots, \mathcal{C}_k\}$ calculer la valeur moyenne m_j des intensités lumineuses des pixels appartenant à la classe $\mathcal{C}_j$.
2. Attribuer chaque pixel $\boldsymbol{l}_i$ de l'image à la classe $\mathcal{C}_j^*$ telle que $m_j^* = min_j\{\|I(\boldsymbol{l}_i) - m_j\|^2\}$. Ainsi :

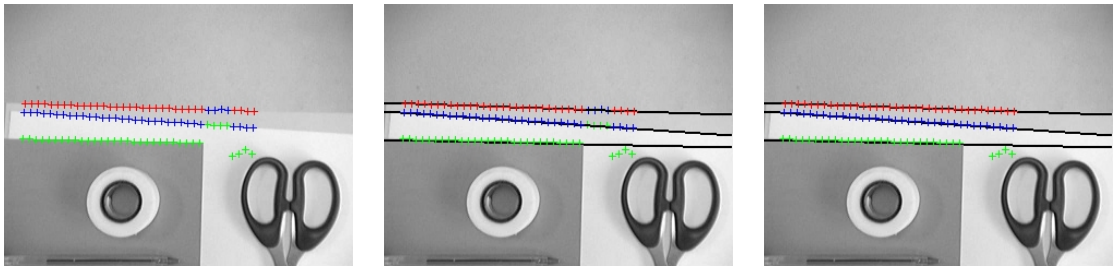
$$\forall j \in \{0, \ldots, k\}, \mathcal{C}_j = \{\boldsymbol{l}_i | \forall l \in \{0, \ldots, k\}, \|I(\boldsymbol{l}_i) - m_j\|^2 \leq \|I(\boldsymbol{l}_i) - m_l\|^2\} \quad (5.15)$$

L'algorithme s'arrête lorsqu'aucun pixel ne change de classe dans l'étape 2.

Algorithme 7 – Exemple d'utilisation de l'algorithme du k-mean *pour la segmentation d'une image.*

Cette initialisation est souvent proche de la réalité ce qui permet à l'algorithme de converger plus rapidement (voir Figure 5.3). À chaque itération de l'algorithme, la droite moyenne de chaque classe est calculée (Figure 5.3 b). Chaque point est ensuite affecté à la classe correspondant à la droite moyenne la plus proche. Comme les arêtes potentielles sont supposées ne pas être normales à l'arête initiale, on ajoute la contrainte que deux hypothèses e'_{i,j,l_1} et e'_{i,j,l_2} issues du même point initial $e_{i,j}$ ne peuvent pas appartenir à la même classe. Les itérations s'arrêtent lorsqu'il n'y a plus de changement dans l'affectation des points.

Au final, l'algorithme de *k-mean* fournit pour chaque arête considérée E_i un ensemble de classes $\mathcal{C}_l^i = (\{e'_{i,j,l}\}_j, r_l^i)$ avec r_l^i correspondant au résidu de la minimisation par moindres carrés. Ce résidu représente un critère de vraisemblance qui sera utilisé par la suite. En pratique, seules les classes qui possèdent un nombre suffisant de points sont conservées. La Figure 5.3 illustre l'ensemble du processus décrit ci-dessus à travers un exemple. L'algorithme 8 résume la procédure. Notons que dans la plupart des cas, k_i ne dépasse pas 2 ou 3.



(a) (b) (c)

FIGURE *5.3 – Un exemple d'utilisation de l'algorithme de* k-mean *avec k=3. Chaque classe est représentée avec une couleur différente. (a) Initialisation des classes de points. (b) Calcul des droites moyennes (centroïdes). (c) La segmentation finale est obtenue après une itération.*

Algorithme *k-mean* pour la classification des hypothèses vers des hypothèses sur les arêtes.
Soient $\{e'_{i,j,l}\}$ les points candidats correspondant au contour projeté E_i et $k_i = max_j\{n_{i,j}\}$ le nombre maximal de candidats détectés pour un même échantillon. L'algorithme est initialisé en respectant l'ordre dans lequel les candidats ont été rencontrés : $\mathcal{C}^i_l = \{e'_{i,j,l}\}_j$. La procédure se déroule alors itérativement de la façon suivante :

1. Pour chaque classe $\mathcal{C}^i_l \in \{\mathcal{C}^i_1, \ldots, \mathcal{C}^i_{k_i}\}$ calculer la droite correspondant à l'interpolation linéaire des points de la classe, par une méthode des moindres carrés robuste. On note r^i_l le résidu de la minimisation.
2. Pour tout i, j :
   - calculer la distance des points $e'_{i,j,l}$ à chacune des droites moyennes,
   - les points $e'_{i,j,l}$ (pour i, j fixés) sont alors associés à des classes différentes, la priorité étant donnée aux points dont la distance à la droite moyenne la plus proche est minimale.

L'algorithme s'arrête lorsque aucun pixel ne change de classe dans l'étape 2.

Algorithme 8 – Algorithme du k-mean *pour la classification des hypothèses bas niveau vers des hypothèses sur les arêtes.*

5.1.3.2 Obtention de plusieurs poses candidates

Pour chaque arête E_i considérée, l'algorithme de *k-mean* détaillé dans la section précédente permet d'obtenir un résidu associé à un ensemble de points représentatifs d'une hypothèse, ce résidu correspondant, on le rappelle, à l'erreur finale de la minimisation des moindres carrés calculée à partir de ces points. Pour l'ensemble des arêtes du modèle, on peut alors effectuer un tirage pondéré en calculant au préalable des poids w^i_l à partir des résidus :

$$w^i_b = \begin{cases} e^{-\lambda\left(\frac{r^i_l - r^i_{min}}{r^i_{max} - r^i_{min}}\right)^2} & \text{si } r^i_{max} \neq r^i_{min} \\ 1 & \text{sinon.} \end{cases} \tag{5.16}$$

où λ est un paramètre permettant d'ajuster la sélectivité du tirage.

Chaque tirage pondéré va permettre d'associer une classe $\mathcal{C}^i_{p_i}$ à chaque arête E_i parmi les k_i classes obtenues pour cette arête. Le tirage pondéré permet de favoriser parmi toutes les combinaisons possibles les candidats présentant les résidus les plus faibles et qui ont donc le plus de chance de correspondre véritablement à une arête. Plusieurs hypothèses sur la pose de la caméra peuvent donc être obtenues à partir des hypothèses déterminées au bas niveau. À chaque tirage, une minimisation non linéaire est effectuée selon l'équation (5.17) en utilisant les jeux de points de la classe sélectionnée lors du tirage :

$$S_{\text{MH}}({}^c\mathbf{M}_w) = \frac{1}{N_e} \sum_i \sum_{e'_{i,j,l} \in \mathcal{C}^i_{p_i}} \rho\left(d^2_\perp(E_i, e'_{i,j,l})\right) \tag{5.17}$$

où on rappelle que $E_i = L_i({}^c\mathbf{M}_w)$.

On obtient alors, pour chaque tirage pondéré, une hypothèse de pose issue des hypothèses bas niveau. La Figure 5.4 illustre le procédé.

Finalement, la pose retenue correspond à l'hypothèse pour laquelle la procédure d'optimisation a donné la plus petite erreur résiduelle de reprojection.

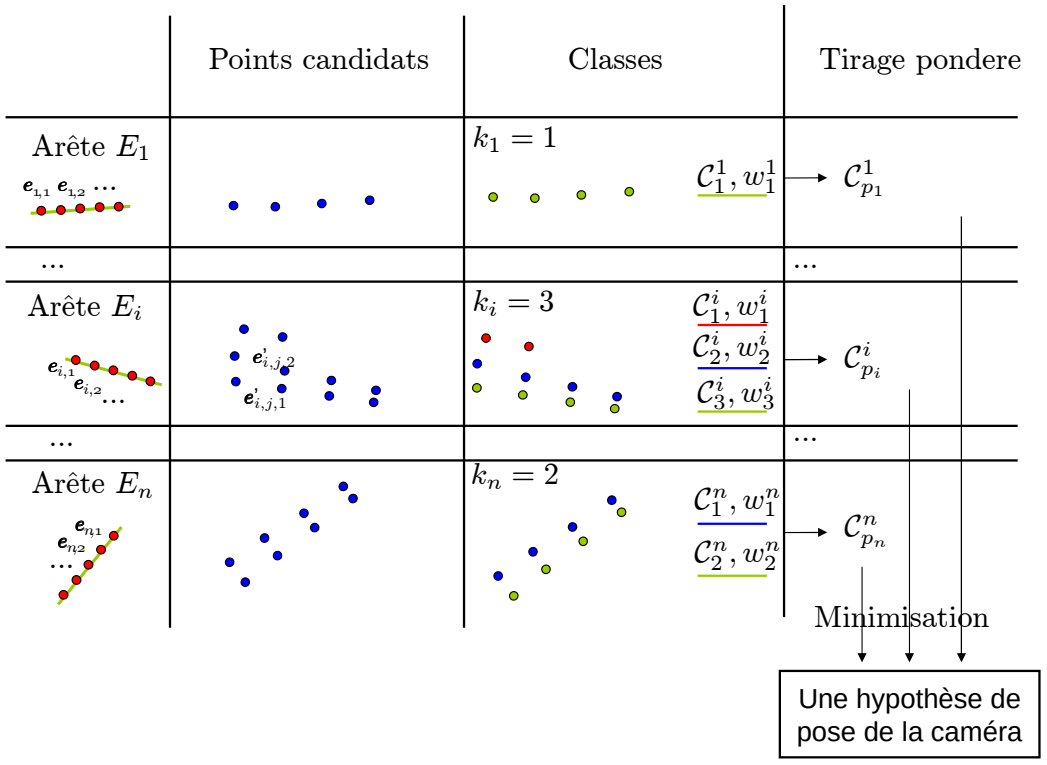


FIGURE 5.4 – *À partir des hypothèses de bas niveau, des classes de points sont extraites. Pour chaque arête projetée, un tirage pondéré est effectué parmi les différentes classes possibles. Après une étape de minimisation, les classes sélectionnées permettent de générer une hypothèse de pose de la caméra.*

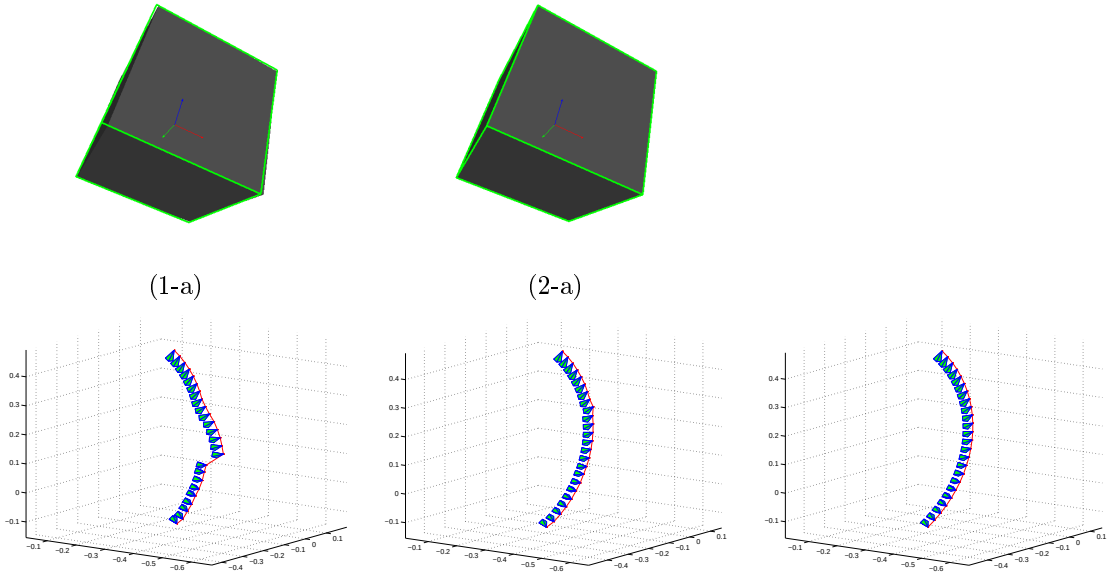
5.1.3.3 Résultats comparatifs

Pour illustrer l'apport de l'approche proposée en termes de robustesse, la Figure 5.5 présente un exemple de situation dans laquelle la prise en compte de plusieurs hypothèses permet d'éviter l'échec du suivi.

L'image choisie correspond à un extrait d'une séquence dans laquelle le suivi de pose échoue lorsqu'une seule hypothèse est considérée. Lorsque la procédure multi-hypothèses décrite dans cette section est utilisée, une seule droite candidate est détectée à partir de tous les contours projetés excepté celui correspondant à l'arête supérieure de la face arrière (voir Figure 5.5, hypothèses détectées). Pour cette arête deux candidates ont été détectés, qui conduisent à deux tirages distincts lors de l'étape (5.17). L'algorithme classique échoue en faisant la mauvaise mise en correspondance tandis que l'algorithme multi-hypothèses identifie la pose correcte (Figure 5.5-(2-b)).

On présente également les résultats obtenus sur une séquence simulée, pour laquelle le résultat du suivi peut être comparé à une vérité terrain. Les résultats comparés entre la méthode de suivi classique et notre approche multi-hypothèses sont présentés Figure 5.6.

FIGURE 5.5 – *Exemple de suivi multi-hypothèses. Sur la première image on a représenté l'ensemble des lignes correspondant aux classes détectées par l'algorithme de* k-mean*, ainsi que les points correspondants. Le niveau de gris utilisé pour tracer la ligne est fonction de sa vraisemblance. (1-a) et (2-a) montrent deux différents tirages obtenus à partir des lignes candidates, conduisant à deux poses distinctes (1-b) et (2-b). Dans le cas du premier tirage, l'arête supérieure de la face arrière a été mise en correspondance avec celle de la face avant, entrainant une perte du suivi. La prise en compte de plusieurs hypothèses permet d'améliorer la robustesse du suivi face à ce type de situation.*



(1-a) (2-a)

(1-b) Trajectoire (2-b) Trajectoire MH (2-c) Trajectoire Ref

FIGURE 5.6 – *Résultats comparatifs sur une séquence de simulation avec des auto-occultations de l'objet. L'algorithme classique (1-a), (1-b) échoue lorsqu'une nouvelle face apparaît. Lorsqu'on prend en compte plusieurs hypothèses (2-a), (2-b), l'objet est suivi tout au long de la séquence. La vérité terrain est représentée sur la figure (2-c).*

Dans l'algorithme classique le point de maximum de vraisemblance est sélectionné, et le suivi échoue lorsque deux contours sont très proches, au moment où une nouvelle face apparaît dans l'image (Figure 5.6-(a) et (c)). Dans le cas de l'approche multi-hypothèses, la pose associée au résidu le plus faible dans le processus d'optimisation est sélectionnée. L'objet est suivi avec succès tout au long de la séquence.

Cependant, le fait de sélectionner la pose donnant le plus faible résidu n'est pas toujours une solution satisfaisante car l'information représentée par les autres poses candidates est perdue. En outre, l'algorithme proposé nécessite toujours un déplacement inter-images assez faible pour converger. Dans le cadre d'une application sur un drone soumis à des déplacements importants, une étape de prédiction devient impérative. Dans la section suivante, nous considérons ainsi un autre cadre de résolution pour le problème du suivi de pose basé modèle : le filtrage particulaire.

5.2 Le suivi basé modèle par filtrage particulaire

Récemment, différents travaux ont proposé d'aborder le problème du calcul de pose basé modèle en utilisant un autre cadre de résolution fourni par le filtrage particulaire [Pupilli 06], [Klein 06], [Nuske 08]. La densité de probabilité associée à la pose ${}^{c}\mathbf{M}_{w}$ est représentée par un ensemble de particules. Chaque particule correspond à une pose possible. L'espace d'état considéré est ainsi l'ensemble $SE(3)$ des transformations rigides de l'espace.

Comme souligné section 5.1.1, $SE(3)$ n'est pas un espace vectoriel et les notions de distance, moyenne ou distribution gaussienne ne sont donc pas définies de ma-

nière naturelle. La question de la mise en place d'un filtre particulaire sur $SE(3)$ a cependant été traitée dans [Chiuso 00] et [Kwon 07]. Nous rappelons ici deux éléments nécessaires à la réalisation d'un filtre particulaire, à savoir comment propager les particules et comment calculer une moyenne sur $SE(3)$.

5.2.1 Modèle d'évolution

Le modèle d'évolution le plus simple consiste à considérer la position comme constante et représenter l'erreur de modélisation sous la forme d'un bruit gaussien. Le bruit gaussien est appliqué aux composantes du vecteur $\mathbf{v}$ qui paramètrent la pose dans l'algèbre $se(3)$, et la pose résultante est obtenue à l'aide de la bijection définie par l'équation (5.5).

Ainsi, [Klein 06] utilise le modèle de propagation :

$$\mathbf{x}_{pred} = \mathbf{M}_{\boldsymbol{\sigma}}.\mathbf{x} \tag{5.18}$$

où $\mathbf{M}_{\boldsymbol{\sigma}} = exp(\mathbf{v})$, $\mathbf{v} \sim \mathcal{N}_{\mathbf{0},\boldsymbol{\sigma}^2\mathbf{I_6}}$, $\mathbf{v} \in \mathbf{se(3)}$ et $\boldsymbol{\sigma}$ est le vecteur des covariances associées aux composantes de $\mathbf{v}$.

5.2.2 Moyenne dans SE(3)

Pour calculer l'estimée à partir de l'ensemble des particules, on utilise généralement la moyenne pondérée des particules. Cependant, l'addition n'étant pas une loi de composition interne à $SO(3)$ (et donc *a fortiori* dans $SE(3)$), la moyenne arithmétique $\overline{\mathbf{R}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{R}_i$ d'un ensemble de matrices de rotation n'est évidemment pas une rotation. Cependant, [Moakher 02] montre qu'une rotation moyenne peut être calculée dans $SO(3)$ en projetant la moyenne arithmétique sur l'espace $SO(3)$. Soit $\overline{\mathbf{R}} = \mathbf{U\Sigma V}$ la décomposition en valeurs singulières de $\overline{\mathbf{R}}$, la rotation moyenne $\mathbf{R}_m$ est alors donnée par :

$$\mathbf{R}_m = \begin{cases} \mathbf{VU}^\top & \text{si } det(\overline{\mathbf{R}}) > 0 \\ \mathbf{VHU}^\top & \text{sinon,} \end{cases} \tag{5.19}$$

où $\mathbf{H} = diag(1, 1, -1)$.

La moyenne des particules de poses est alors obtenue à partir de cette moyenne sur $SO(3)$ et de la moyenne arithmétique des translations de chaque particule.

5.2.3 Mesure de vraisemblance

La mesure de vraisemblance doit être calculée pour chaque particule et constitue l'étape critique du filtre particulaire en termes de temps de calcul. Le calcul de la fonction de distance définie dans l'équation (5.14) nécessite la projection du modèle, l'échantillonnage des contours projetés et l'exploration de la normale pour chaque particule, ce qui est relativement coûteux. Dans [Klein 06], l'étape de projection du modèle est réalisée pour chaque particule. Les particules sont alors évaluées en mesurant le rapport entre le nombre de pixels d des contours projetés qui sont suffisamment proches d'un contour effectif dans l'image et le nombre total v de

pixels échantillonnés sur les contours projetés. La vraisemblance d'une particule s est alors calculée selon l'équation :

$$p(\mathbf{z} \mid \mathbf{x} = s) = e^{\left(\lambda \frac{d}{v}\right)} \tag{5.20}$$

où λ est un paramètre permettant de régler la confiance attribuée à la mesure. Pour déterminer si un pixel est proche d'un contour de l'image, [Klein 06] fait appel à une carte de distance [Fabbri 08] qui associe à chaque pixel de l'image la distance et l'orientation du plus proche contour. La carte de distance est construite une seule fois pour chaque image en suivant les étapes suivantes :

- extraction des contours de l'image,
- seuillage,
- calcul de la carte de distance à partir de l'image des contours seuillée.

(Voir Figure 5.7).

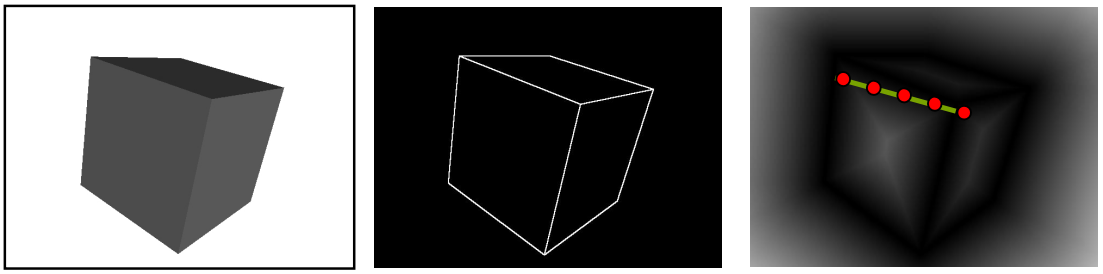


FIGURE 5.7 – *Une carte de distance est construite à partir de l'image des contours seuillée. Elle permet d'accéder directement à la distance du plus proche contour.*

Le compte des d pixels suffisamment proches d'un contour est alors déterminé simplement en utilisant un seuil sur la distance et sur l'orientation du contour. L'usage de la carte de distance permet de calculer la vraisemblance de façon efficace.

Dans nos travaux, nous utilisons directement la valeur moyenne de la distance sans passer par un seuillage binaire. Soient $\{\boldsymbol{l}_i\}_{i=1\ldots n}$ l'ensemble des pixels échantillonnés sur les contours projetés. Si d_i est la distance donnée par la carte de distance pour le pixel $\boldsymbol{l}_i$, c'est-à-dire la distance entre le pixel $\boldsymbol{l}_i$ et le contour le plus proche dans l'image, alors la distance moyenne est donnée par :

$$d(s) = \frac{1}{n} \sum_i d_i \tag{5.21}$$

On n'utilise pas ici d'information sur l'orientation des contours même si celle-ci pourrait dans certains cas améliorer le caractère discriminant de la fonction de distance. La Figure 5.9 montre l'allure de la fonction de distance d de l'équation (5.21) en fonction de mouvements de translation parallèles au plan de la fenêtre pour l'image de la Figure 5.8. La vraisemblance est alors déduite de cette fonction distance par l'équation :

$$p(\mathbf{z} \mid \mathbf{x} = s) = \begin{cases} e^{-\lambda\left(\frac{d(s)-d_{min}}{d_{max}-d_{min}}\right)^2} & \text{si } d_{max} \neq d_{min} \\ 1 & \text{sinon.} \end{cases} \tag{5.22}$$

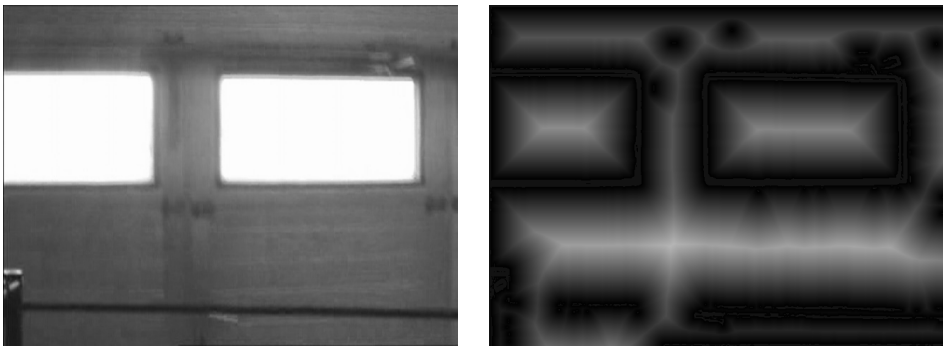


FIGURE 5.8 – *Exemple de carte de distance obtenue sur une image acquise depuis un drone.*

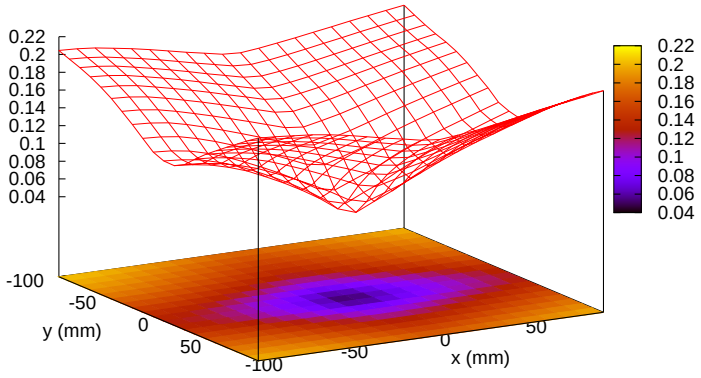


FIGURE 5.9 – *Fonction de distance obtenue à partir de l'image de la fenêtre ci-dessus pour des translations suivant x et y. L'origine correspond à la position réelle.*

5.2.4 Intégration de particules optimisées dans le cadre du filtrage particulaire

L'ensemble des particules du filtre particulaire constitue une discrétisation d'une partie de l'espace des transformations rigides $SE(3)$. Pour obtenir une estimation précise, un grand nombre de particules est alors nécessaire. Dans cette section on propose d'intégrer des hypothèses générées par le processus d'optimisation présenté dans la section 5.1.3 afin de guider le filtre vers les zones correspondant à des hypothèses pertinentes. L'approche proposée est inspirée des approches de filtrage particulaire hybrides [Sullivan 01] [Bray 04] [Shan 07] dans lesquelles les particules sont déplacées vers le maximum local de la fonction de vraisemblance le plus proche, par une optimisation locale. Dans notre cas, la méthode d'optimisation locale utilisée correspond à l'approche multi-hypothèses de la section 5.1.3. Pour réduire la complexité algorithmique, l'optimisation n'est pas appliquée à chaque particule mais à un sous-ensemble des « meilleures » particules prédites, c'est-à-dire aux particules dont la vraisemblance est supérieure à un certain pourcentage de la vraisemblance maximale.

Les N^* nouvelles particules résultant du processus d'optimisation sont alors ajoutées aux particules prédites par l'étape d'évolution classique. Cependant, ces nouvelles particules ne résultant pas d'un échantillonnage selon la densité de probabilité *a priori* du filtre $f_k(\mathbf{x}_k) = p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})$, elles ne peuvent pas être utilisées directement dans l'estimation. Pour pouvoir les intégrer dans le filtre particulaire on considère que ces nouvelles particules $\{(s_k^{*(i)})\}_{i=1..N^*}$ sont échantillonnées selon une fonction d'importance $g_k(\mathbf{x}_k)$ (voir section 1.3.4.3). Pour compenser le fait que les particules soient échantillonnées selon g et non f, le calcul des poids de chacune des particules fait alors intervenir un terme correctif f/g [Isard 98b].

Pour calculer les termes correctifs, les fonctions f et g sont approchées par des sommes de fonction gaussiennes centrées sur les particules :

$$f_k(\mathbf{x}_k) = \frac{1}{N}\sum_{i}^{N}\mathcal{N}(s_k{}^{(i)}, \Sigma)(\mathbf{x}_k) \tag{5.23}$$

$$g_k(\mathbf{x}_k) = \frac{N}{N+N^*}\left(\frac{1}{N}\sum_{i=1}^{N}\mathcal{N}(s_k{}^{(i)}, \Sigma)(\mathbf{x}_k) + \frac{1}{N^*}\sum_{i=1}^{N^*}\mathcal{N}(s_k^{*(i)}, \Sigma)(\mathbf{x}_k)\right) \tag{5.24}$$

où $\mathcal{N}(s, \Sigma)$ est la fonction gaussienne de dimension 6 de covariance Σ, centrée sur les coordonnées exponentielles canoniques de la pose s.

L'algorithme 9 résume la méthode proposée.

Algorithme de calcul de pose par filtrage particulaire hybride.

Initialisation : générer N particules $\left\{s_0^{(i)}\right\}_{i=1}^{N}$ et poser $\pi_0^{(i)} = \frac{1}{N}$.

Pour $k = 1, ..., T$:

1. Prédiction : générer N particules $s_k^{'(i)}$ selon le modèle de l'équation (5.18).
2. Calcul de la carte de distance et mesure de la distance (5.21) pour chaque particule prédite. La distance minimale d_{min} est mémorisée.
3. Optimisation (selon la méthode illustrée Figure 5.4) des meilleures particules vérifiant $d(s_k^{'(i)}) < 0.7\ d_{min}$. Un ensemble de particules optimisées $\left\{(s_k^{'*(i)}, \frac{1}{N^*})\right\}_{i=1..N^*}$ est obtenu.
4. Mesure de la distance (5.21) pour les particules optimisées.
5. Calcul des poids de l'ensemble $\left\{(s_k^{'(i)}, \frac{1}{N+N^*})\right\}_{i=1..N+N^*}$ des particules à partir de la distance et du terme correctif :
   $\pi_k^{(i)} \propto \frac{f_k(s_k^{'(i)})}{g_k(s_k^{'(i)})} p(\mathbf{z}_k \mid \mathbf{x}_k = s_k^{'(i)})$, avec $\sum_{i=1}^{N+N^*} \pi_k^{(i)} = 1$.
6. Rééchantillonnage : tirage aléatoire pondéré de N particules dans l'ensemble des $N + N^*$ particules.

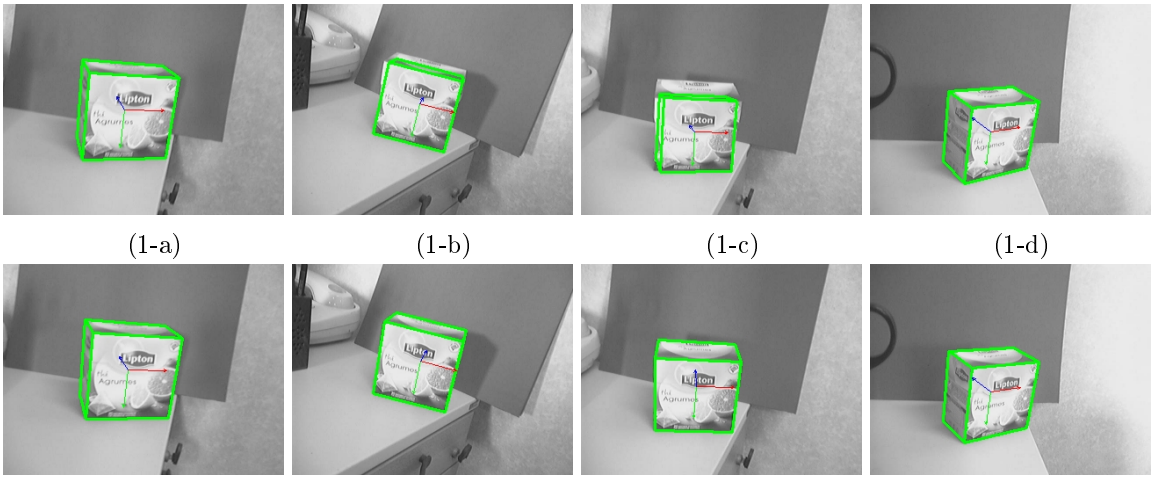
L'estimée est la moyenne pondérée des particules (voir section 5.2.2).

Algorithme 9 – Calcul de pose par filtrage particulaire hybride.

5.3 Résultats expérimentaux

L'algorithme proposé a été testé sur différentes séquences vidéos.

Comparatif des performances. Les Figures 5.10 et 5.11 présentent des extraits des séquences vidéos sur lesquelles l'approche proposée est comparée à une méthode nominale de recalage par optimisation simple [Comport 06]. Figure 5.10, l'estimation de pose échoue avec l'approche nominale à cause d'une mauvaise mise en correspondance des arêtes projetées. La pose est finalement retrouvée à la fin de la séquence (image de droite). L'approche que nous avons présentée dans la section précédente (voir l'algorithme 9), suit l'objet avec succès tout au long de la séquence.



(1-a) (1-b) (1-c) (1-d)

(2-a) (2-b) (2-c) (2-d)

FIGURE *5.10 – Résultats comparatifs pour le suivi d'une boîte. La première rangée montre les résultats obtenus pour une méthode d'optimisation classique. La deuxième rangée présente la pose obtenue avec l'approche proposée.*

Pour mettre en évidence l'amélioration apportée par notre approche dans le cadre applicatif qui nous intéresse, l'algorithme a également été testé sur différentes séquences acquises depuis des drones. La Figure 5.11 présente des extraits d'une séquence vidéo comportant des déplacements inter-images importants, des sorties de champ, et des changements d'illumination. La séquence montre une fenêtre, filmée depuis un drone en pilotage manuel. Le modèle 3D de la fenêtre étant connu, l'algorithme permet d'estimer la pose du drone par rapport à celle-ci.

Lorsque la sortie de champ est trop importante (Figure 5.11 1-c), l'approche nominale échoue. L'usage du filtre particulaire permet l'estimation de la pose tout au long de la séquence. Grâce à la procédure hybride qui utilise des particules optimisées, un faible nombre de particules est nécessaire. Dans cette séquence, seules 100 particules ont été utilisées.

La Figure 5.12 montre les résultats comparatifs obtenus pour une séquence structurée complexe comportant des ambiguïtés (rangées 3 et 5 notamment), du bruit (2ème rangée par exemple) et une information parfois limitée avec peu d'arêtes présentes dans l'image (4ème rangée). La 1ère colonne présente les résultats obtenus avec une approche nominale (optimisation déterministe simple). Sur la 2ème colonne un filtre particulaire utilisant 25 particules est appliqué, ce qui s'avère insuffisant pour suivre la pose de la caméra. Avec 200 particules, le filtre ne diverge plus mais la précision est approximative (3ème colonne). L'approche hybride (4ème colonne) permet une bonne estimation avec un nombre très réduit de particules (25 particules

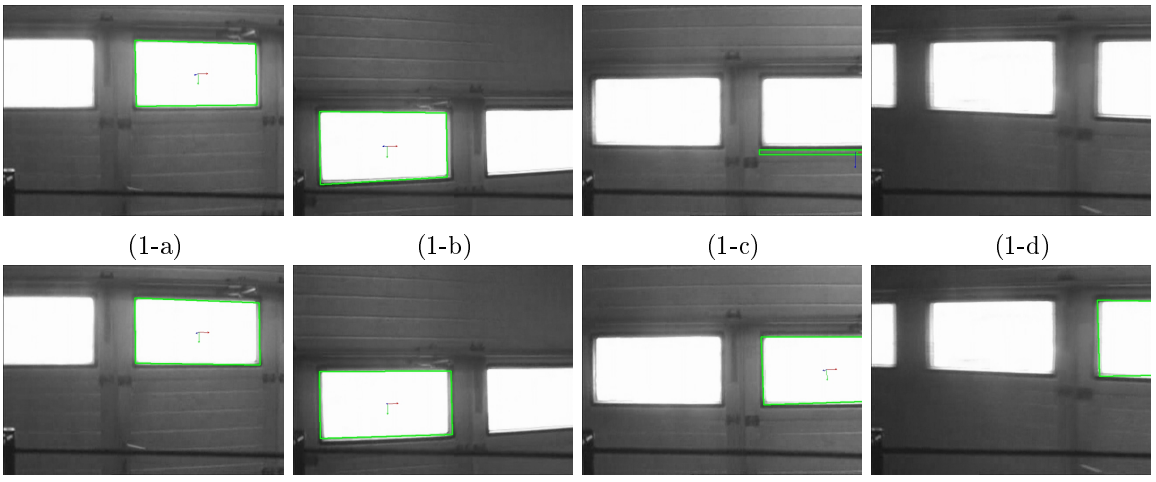


FIGURE 5.11 – *Résultats comparatifs pour le suivi de fenêtre. L'approche nominale (première rangée) échoue lorsque d'importantes sorties de champ se produisent (1-c). L'approche proposée est robuste et estime la pose durant toute la séquence.*

suffisent dans cet exemple).

Complexité et temps de calcul. Dans les approches proposées, plusieurs paramètres sont à prendre en compte dans l'évaluation de la complexité :

- la taille de l'image ;
- la taille du modèle 3D ;
- le nombre d'optimisations exécutées ;
- le nombre d'itérations par optimisation ;
- le nombre N de particules lorsqu'un filtrage particulaire est utilisé.

Pour une image et un modèle de tailles données, on peut donc comparer la complexité des approches utilisées en fonction des 3 derniers paramètres. Si M est le nombre moyen d'itérations nécessaires pour une optimisation simple, la complexité de cet algorithme est $C = O(M)$.

L'optimisation multi-hypothèses décrite dans la section 5.1.3 requiert le calcul de plusieurs optimisations lorsque des ambiguïtés sont rencontrées, ce qui multiplie d'autant la complexité algorithmique. En pratique le nombre K d'optimisations nécessaires pour obtenir une réelle amélioration de la robustesse ne dépasse pas 4 ou 5, et cette approche reste donc adaptée pour une application temps réel : $C = O(M)$.

Dans le cas du filtrage particulaire SIR classique (voir l'algorithme 3), la complexité dépend essentiellement du nombre de particules utilisées : $C = O(N)$. Cependant, lorsqu'un processus d'optimisation multi-hypothèses est appliqué à certaines particules la complexité devient $C = O(N \times M \times K)$. On note cependant que, d'une part le nombre N de particules nécessaires est nettement plus faible que dans le filtre particulaire classique, et d'autre part, le nombre moyen d'itérations nécessaires pour optimiser les particules est réduit grâce à la prédiction du filtre et au fait que la procédure ne soit appliquée que sur les meilleures particules.

Dans l'exemple de la Figure 5.12 le temps d'exécution de l'approche hybride est ainsi de l'ordre de 10 images par seconde, sans optimisation spécifique de l'implé-

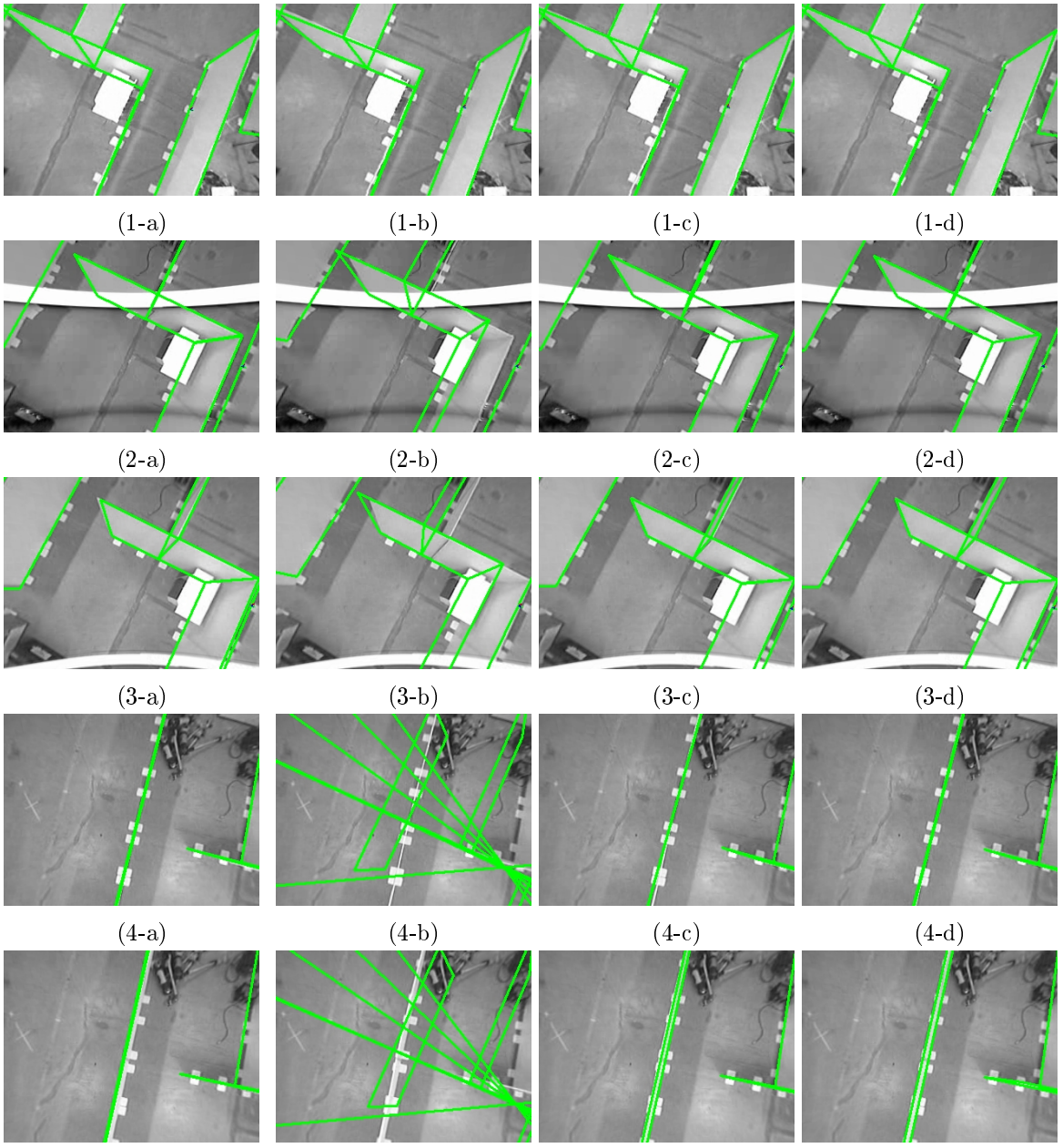


FIGURE *5.12 – Résultats comparatifs sur une séquence structurée complexe.*

mentation. Le filtre particulaire étant adapté à la parallélisation, l'approche peut donc être envisagée dans des applications temps-réel.

5.4 Conclusion

Dans ce chapitre nous avons considéré le problème du calcul de la pose relative entre une caméra et un environnement structuré dont un modèle est supposé connu. Après avoir dressé un aperçu des différentes approches existantes et des limitations qu'elles rencontrent en termes de robustesse, nous avons proposé une méthode d'optimisation multi-hypothèses. Cette approche permet de prendre en compte plusieurs hypothèses vraisemblables lors de la mise en correspondance de contours au bas niveau, et de traduire ces hypothèses en des poses possibles pour la caméra. Cette approche de suivi déterministe de la pose permet une plus grande robustesse face à des situations d'ambiguïté dans lesquelles une procédure d'optimisation classique peut être amenée à échouer.

Dans un deuxième temps, afin d'assurer une cohérence temporelle nous avons montré comment intégrer la procédure d'optimisation proposée dans le cadre du filtrage particulaire. Cette approche permet également de bénéficier à la fois de ces hypothèses issues d'une approche *bottom-up* ainsi que d'hypothèses *top-down* générées par le modèle d'évolution du filtre.

Les différentes approches ont été comparées sur diverses séquences présentant des caractéristiques contraignantes : grands déplacements inter-images, bruit, sorties de champ partielles, arêtes proches, etc. Notamment des séquences vidéos acquises depuis des drones pilotés manuellement ont mis en évidence les gains en performance de l'approche proposée par rapport aux approches déterministes ou bayésiennes classiques.

À l'issue de ce chapitre on considère qu'une estimation robuste de la pose 3D de la caméra par rapport à un environnement structuré est disponible. Le chapitre suivant montre comment cette information peut être utilisée pour asservir un drone en position.

Chapitre 6

Positionnement et navigation par points de passage

Dans ce chapitre nous considérons des tâches de positionnement d'un drone à partir de l'information de pose estimée par la vision. La vision est ici utilisée pour contrôler 4 degrés de liberté du drone, à savoir sa position dans l'espace tridimensionnel et son orientation en lacet. La commande du drone en position, présentée dans la section 6.1 nécessite une bonne estimation de la vitesse de translation de l'appareil. La section 6.2 montre comment cette information de vitesse est obtenue en fusionnant les données de vision avec des données inertielles. Enfin, l'approche proposée est validée expérimentalement section 6.3, avec à la fois une évaluation quantitative de la précision des estimations de position et de vitesse obtenues et une analyse de tâches de positionnement et de navigation par points de passage réalisées sur un drone quadrirotor.

6.1 Commande de positionnement pour un drone

Contrairement à la tâche considérée dans la première partie de ce mémoire, où l'information visuelle était une information de position relative estimée à un facteur d'échelle près, l'approche basée modèle présentée dans le chapitre 5 fournit une estimation de la pose 3D complète du drone par rapport à son environnement. Selon la typologie introduite dans la section 2.2.2.1, la tâche de positionnement que nous considérons dans cette section s'inscrit donc dans le cadre de l'asservissement visuel 3D.

On considère à nouveau un drone quadrirotor. La modélisation dynamique de ce type d'engin a été présentée section 4.1. La commande en lacet étant identique à celle proposée pour la tâche de poursuite (voir section 4.2.1), on se contente ici d'en rappeler le schéma de commande en Figure 6.1.

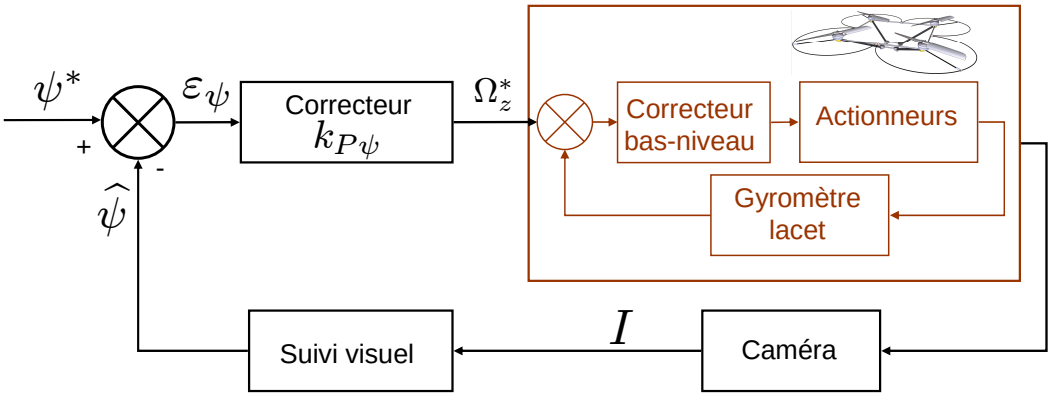


FIGURE 6.1 – *Contrôle du lacet.*

Commande de la translation. À la différence de la commande mise en place pour la tâche de poursuite (voir la section 4.2.2), la commande de la translation définie dans cette section prend en compte à la fois les déplacements de translation dans le plan $(x\ y)$ et l'altitude en z. La commande en translation que nous avons mise en place est également basée sur une commande hiérarchique, dans laquelle le correcteur de position envoie des commandes en vitesse à un correcteur de vitesse de translation. Le correcteur de vitesse envoie alors au drone des consignes d'assiette et de poussée T, supposées régulées par un correcteur interne à une fréquence élevée (voir Figure 6.2). L'assiette détermine le déplacement du drone dans le plan $(x\ y)$, la poussée permet de contrôler sa translation en z. Le modèle du drone proposé met en évidence un couplage entre ces différentes translations. En effet, si le drone est incliné (assiette non nulle) sa poussée n'étant pas verticale aura naturellement un effet à la fois sur son déplacement en translation et sur sa prise d'altitude. On suppose dans cette section que le drone évolue en régime quasi-stationnaire et que son assiette reste donc faible pendant ses déplacements. Cette hypothèse permet de considérer en première approximation que l'altitude du drone dépend uniquement de sa poussée. Nous montrerons dans la section expérimentale 6.3 de ce chapitre que cette hypothèse de vol quasi-stationnaire est vérifiée.

En reprenant les notations de la section 4.1, on désigne par $\mathbf{p} \in \mathbb{R}^3$ la position du centre de masse du drone par rapport au repère inertiel $\mathcal{R}_w$. La tâche de positionnement s'exprime ainsi sous la forme :

$$\boldsymbol{\varepsilon}_\mathbf{p} = \mathbf{p}^* - \mathbf{p} \tag{6.1}$$

où $\mathbf{p}^*$ est la consigne de position. L'utilisation d'un correcteur proportionnel détermine une consigne de vitesse de translation :

$$\mathbf{v}^* = k_P \boldsymbol{\varepsilon}_\mathbf{p} \tag{6.2}$$

k_P étant le gain proportionnel utilisé.

L'erreur en vitesse :

$$\boldsymbol{\varepsilon}_\mathbf{v} = \mathbf{v}^* - \mathbf{v} \tag{6.3}$$

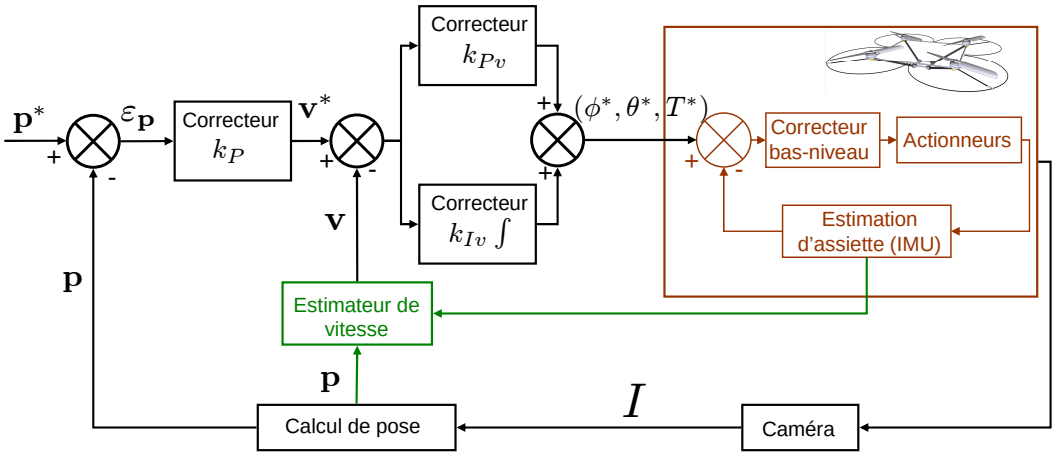


FIGURE 6.2 – *Contrôle du mouvement de translation.*

est régulée par un correcteur proportionnel intégral (voir Figure 6.2). Cette loi de commande nécessite une estimation de la vitesse de translation $\mathbf{v}$ du drone. La section suivante montre comment celle-ci est obtenue.

6.2 Estimation de la vitesse de translation

Afin d'obtenir une estimation de la vitesse de translation du drone, nécessaire à la stabilité de la commande, plusieurs options s'offrent à nous. Une possibilité consiste à utiliser un filtre dérivateur sur la position, comme nous l'avons fait pour la tâche de poursuite section 4.2.2. Cependant ce filtrage introduit un retard qui se répercute ensuite dans la commande. On souhaite ici obtenir une estimée plus fiable grâce à un modèle de prédiction de la vitesse.

Une approche possible consiste à inclure la vitesse dans l'état du filtre particulaire basé vision pour l'estimer en même temps que la pose. Quoique valide d'un point de vue théorique, nous avons écarté cette solution pour des raisons de temps de calcul. En effet, l'algorithme de suivi de pose présenté dans le chapitre 5 répond à nos attentes en termes de robustesse, mais en l'absence d'optimisation spécifique, il s'exécute à une fréquence encore faible sur le poste utilisé dans nos expériences. Ajouter des dimensions à l'espace d'état impliquerait une augmentation du nombre de particules nécessaires et donc une diminution non souhaitable de la fréquence d'exécution.

Pour valider le système proposé nous avons donc choisi d'utiliser une autre approche qui consiste à fusionner les données de vision et des données inertielles. On utilise pour cela un filtre de Kalman étendu (EKF, voir section 1.3.4.2).

L'état à estimer s'écrit $\mathbf{X} = (\mathbf{p},\ \mathbf{v},\ \mathbf{a})$ où $\mathbf{a}$ désigne l'accélération du drone.

Modèle de prédiction. Pour établir un modèle de prédiction de la vitesse du drone, nous reprenons ici la modélisation de la dynamique de translation présentée

dans la section 4.1 (voir équation (4.4)) :

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{v} \\ \dot{\mathbf{v}} = \mathbf{a} \\ m\dot{\mathbf{v}} = T\mathbf{R}\mathbf{e}_z + mg\mathbf{e}_z - f\mathbf{v}^2 \end{cases} \tag{6.4}$$

dans laquelle nous avons pris en compte une force de frottement $-f\mathbf{v}^2$ de coefficient f.

On définit alors un modèle d'évolution approché en faisant les hypothèses simplificatrices suivantes :

- les vitesses selon les axes x et y sont supposées découplées ;
- le coefficient de frottement f est supposé constant et indépendant de la direction du mouvement ;
- les angles de tangage et de roulis sont supposés petits, ce qui est raisonnable en régime quasi-stationnaire ;
- la poussée T est supposée quasi constante. Dans des applications en intérieur, en l'absence de vent, la poussée a, en effet, de faibles variations qui peuvent être négligées en première approximation.

Nous verrons par la suite (voir page 128) que ces approximations suffisent à obtenir une bonne précision dans l'estimation de vitesse, les erreurs de modélisation étant compensées par l'observation.

A partir des équations (6.4) et des hypothèses ci-dessus, le modèle de prédiction s'écrit de la façon suivante :

$$a_x{}^{(t+\delta t)} = \frac{T}{m}\phi^{(t-\tau)} - \text{sign}(v_x)\frac{f}{m}{v_x}^2 + n_{ax} \tag{6.5}$$

$$a_y{}^{(t+\delta t)} = \frac{T}{m}\theta^{(t-\tau)} - \text{sign}(v_y)\frac{f}{m}{v_y}^2 + n_{ay} \tag{6.6}$$

où τ représente le retard supposé constant. Les angles de tangage et de roulis θ et ϕ proviennent d'une estimation de la centrale inertielle.

Pour le déplacement vertical on utilise un modèle à accélération constante :

$$a_z{}^{(t+\delta t)} = a_z{}^{(t)} + n_{az} \tag{6.7}$$

n_{ax}, n_{ay}, n_{az} désignent les trois composantes d'un bruit blanc $\mathbf{n_a} = \mathcal{N}(0, \mathbf{Q_a})$. $\mathbf{Q_a}$ est la matrice de covariance associée au modèle d'accélération. La position et la vitesse sont alors déduites simplement par intégration :

$$\begin{cases} v^{(t+\delta t)} = v^{(t)} + a^{(t+\delta t)}\delta t \\ p^{(t+\delta t)} = p^{(t)} + v^{(t+\delta t)}\delta t. \end{cases} \tag{6.8}$$

Dans nos expériences les constantes τ, $\alpha = \frac{T}{m}$ et $\beta = \frac{f}{m}$ sont supposées connues. En pratique, les constantes que nous utilisons ont été estimées au CEA à l'aide d'un algorithme génétique, hors du cadre de cette thèse. Elles peuvent tout aussi bien être déterminées par une autre méthode d'estimation (moindres carrés robustes par exemple).

Les équations d'évolution (6.5) et (6.6) étant non linéaires, nous utilisons un filtre de Kalman étendu. La matrice Jacobienne $\mathbf{J_x}$ est donnée par :

$$\mathbf{J_x} = \begin{pmatrix} \mathbf{I} & \delta t\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \delta t\mathbf{I} \\ \mathbf{0} & -\text{sign}(v)2\beta va & \mathbf{0} \end{pmatrix} \tag{6.9}$$

La prédiction de la matrice de covariance de l'état $\mathbf{P}$ s'écrit alors :

$$\mathbf{P}_{(t+\delta t)|t} = \mathbf{J_x}\mathbf{P}_{t|t}{\mathbf{J_x}}^T + \mathbf{J_n}\mathbf{Q}{\mathbf{J_n}}^T \tag{6.10}$$

où

$$\mathbf{Q} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Q_a} \end{pmatrix} \tag{6.11}$$

$\mathbf{J_n}$ est la matrice des dérivées du modèle d'évolution en fonction des composantes de $\mathbf{n}$. (voir section 1.3.4.2)

Modèle d'observation. L'observation est donnée simplement par l'estimée $\hat{\mathbf{p}}$ fournie par l'algorithme de vision.

$$\mathbf{X}_{t+\delta t} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix}^\top \hat{\mathbf{p}} \tag{6.12}$$

6.3 Résultats expérimentaux

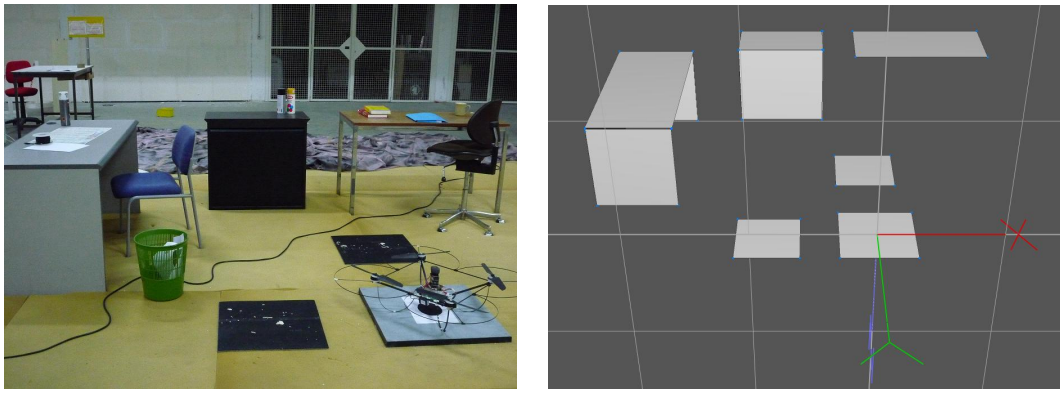
Dans cette section nous présentons les résultats expérimentaux obtenus sur un drone quadrirotor. Le système utilisé est similaire au système présenté section 4.3. Sur le poste au sol, le système complet (localisation, estimation de vitesse et calcul de commande) s'exécute à une fréquence de 20Hz. Le délai τ entre l'instant où une image est acquise par la caméra et l'instant où l'assiette désirée est calculée et atteinte par le véhicule est estimé et utilisé dans la prédiction de l'accélération (équations (6.5) et (6.6)). Dans les expériences menées : $\tau = 400$ms, soit une dizaine d'itérations de l'algorithme.

Pour la localisation, nous avons modélisé une scène constituée de différents meubles (voir Figure 6.3 a). La modélisation réalisée (Figure 6.3 b) est volontairement grossière afin d'évaluer la robustesse et la pertinence de l'approche dans des situations où seul un modèle approximatif de la scène est disponible.

Avant de présenter les résultats de tâches de positionnement, nous présentons une évaluation de la précision de la localisation et de l'estimation de vitesse de translation.

6.3.1 Évaluation de la précision

Pour évaluer la précision de la localisation obtenue par l'algorithme de suivi proposé, nous l'avons comparée à une *vérité terrain* obtenue à l'aide d'un dispositif de métrologie laser Leica [1] de grande précision dont dispose le CEA. Ce dispositif est



(a) (b)

FIGURE 6.3 – *Scène (a) et modèle 3D utilisé (b).*

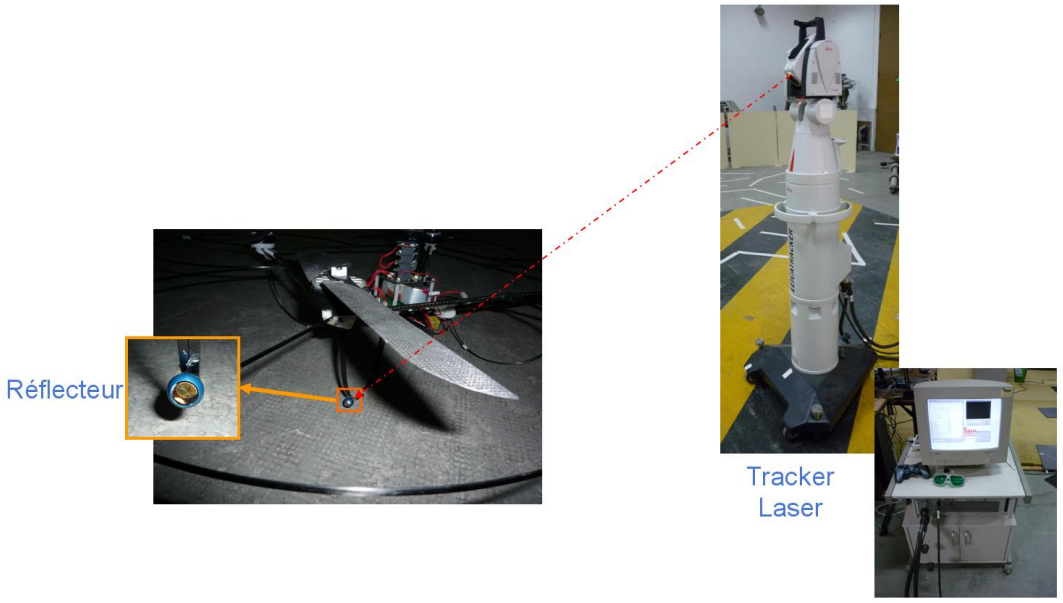


FIGURE 6.4 – *Dispositif d'utilisation du tracker laser Leica.*

illustré Figure 6.4. À l'aide d'un faiceau laser le système suit et mesure la position d'un réflecteur fixé sur le drone. Les mesures obtenues ont une précision de l'ordre du micron, et sont acquises à une fréquence de 1000Hz. La position étant mesurée en un seul point du drone la valeur de référence obtenue ne permet pas d'évaluer les rotations de l'appareil. Ce dispositif fournit cependant une valeur de référence fiable à laquelle comparer les mesures de position et de vitesse de translation estimées par notre système.

Précision de localisation. Les Figures 6.5 et 6.6 représentent les résultats comparatifs obtenus pour l'estimation de la position du drone évoluant en vol téléopéré dans la scène modélisée.

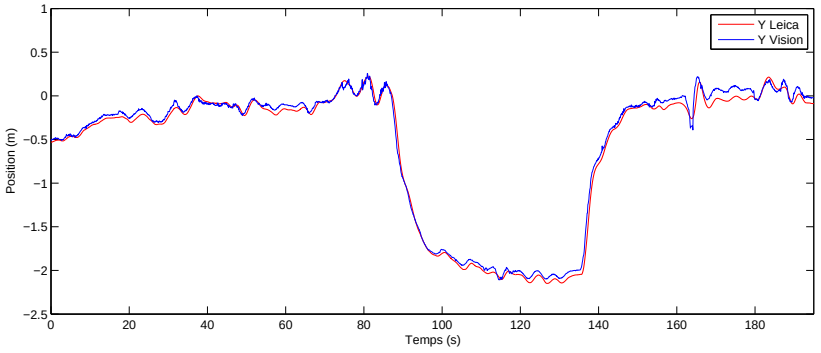


FIGURE 6.5 – *Position estimée par le suivi basé modèle (bleu) et valeur de référence (rouge).*

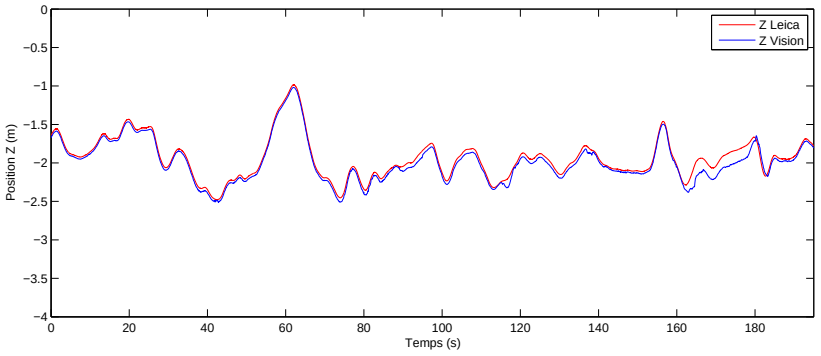


FIGURE 6.6 – *Position estimée par le suivi basé modèle (bleu) et valeur de référence (rouge).*

L'erreur moyenne de l'estimation de position pour la courbe de la Figure 6.5 est de 8mm avec un écart-type de 16cm. Pour l'estimation de l'altitude, Figure 6.6, on obtient une erreur moyenne de 4cm avec un écart-type de 15cm. À titre de comparaison, les méthodes de localisation par fusion de capteurs GPS et inertiels, destinées à la navigation en extérieur, atteignent une précision de l'ordre de quelques mètres.

1. http ://metrology.leica-geosystems.com/

De par la méthode de calcul de pose utilisée, la précision de la localisation est étroitement liée à la précision du modèle. Comme illustré sur la Figure 6.7, lorsque le modèle n'est pas exact, le recalage peut ne pas correspondre parfaitement à toutes les parties de modèle. On note cependant que l'algorithme permet une bonne localisation malgré certaines imprécisions de ce type dans le modèle de la scène.

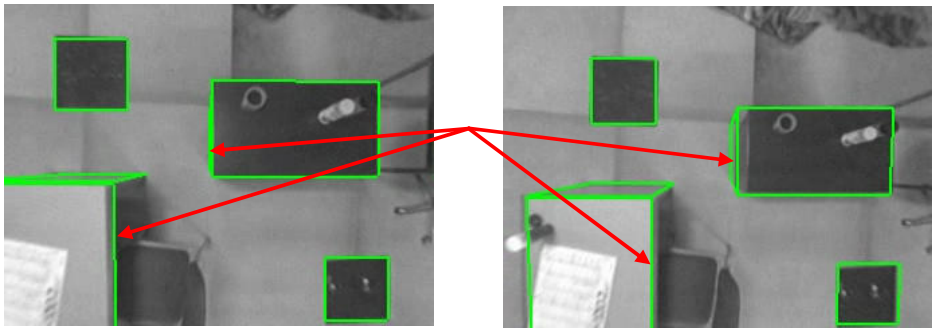


FIGURE 6.7 – *Exemple de recalage imparfait dû à l'imprécision du modèle 3D : le modèle ne coïncide pas à la fois avec le bureau et avec la commode dont la distance a été mesurée de façon approximative, mais la situation n'entraine pas l'échec du suivi.*

Précision de l'estimation de vitesse. Les Figures 6.8 et 6.9 montrent l'allure des vitesses obtenues par une simple différentiation de la position estimée par l'algorithme de suivi (en vert) et par le filtrage de Kalman présenté section 6.2 (en bleu).

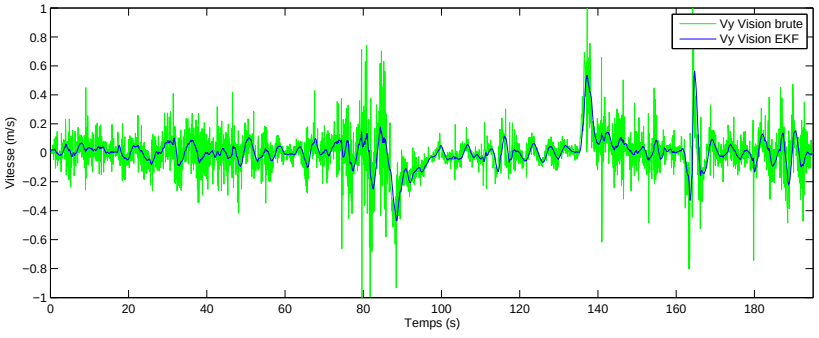


FIGURE 6.8 – *Vitesse sur l'axe x.*

Pour permettre une fréquence de traitement plus élevée l'algorithme de vision utilisé ici est l'approche multi-hypothèses sans filtrage particulaire décrite dans la section 5.1.3. On constate que la vitesse obtenue par un filtre dérivateur simple est inexploitable. Un simple filtrage passe-bas ne ferait qu'ajouter un retard néfaste à la commande et au vu des graphes obtenus l'EKF apparaît comme une solution adaptée.

Pour évaluer la précision de l'estimation de vitesse proposée, les Figures 6.10 et 6.11 montrent la comparaison entre la vitesse obtenue par la vision avec le filtrage

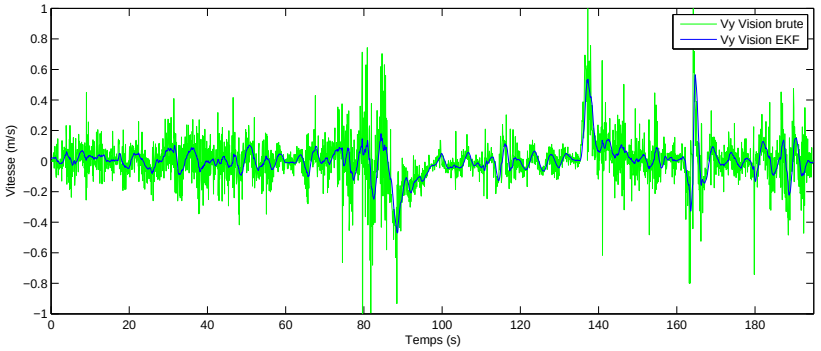


FIGURE *6.9 – Vitesse sur l'axe y.*

de Kalman (en bleu) et la vitesse de référence calculée en prenant la dérivée discrète de la mesure de position du laser.

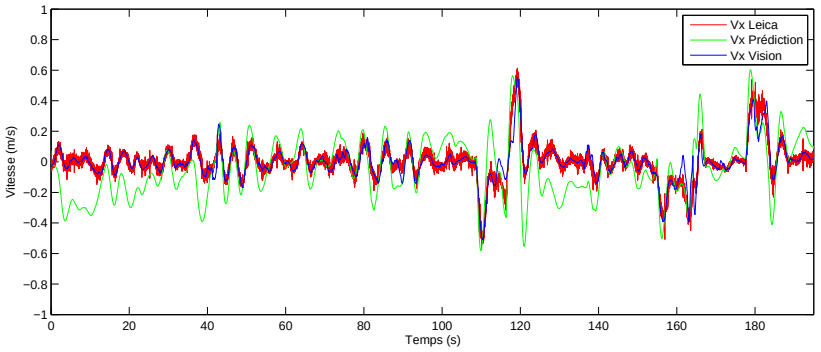


FIGURE *6.10 – Vitesse sur l'axe x.*

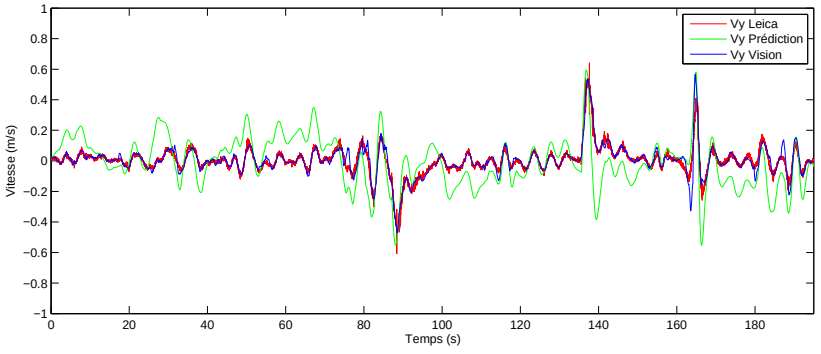


FIGURE *6.11 – Vitesse sur l'axe y.*

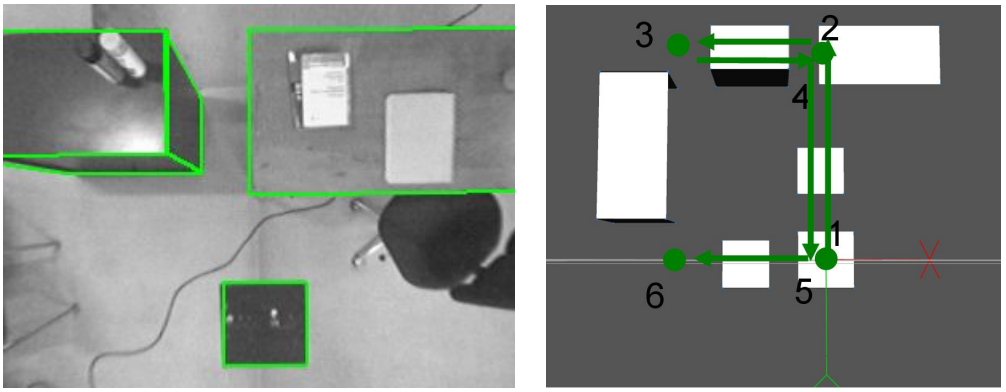
L'erreur moyenne de vitesse sur ces séquences est de 0.006m/s (resp. 0.005m/s) sur l'axe x (resp. y), avec des écarts-types de 0.08m/s (resp. 0.07m/s). Le tracé en vert des Figures 6.10 et 6.11 représente la prédiction seule, sans recalage par la vision. Bien qu'approximatif, le modèle utilisé rend compte de l'allure des variations de vitesse et son recalage avec les données de vision permet une bonne estimation de la vitesse de translation du drone.

6.3.2 Positionnement

Nous présentons à présent les expériences de positionnement effectuées.

L'initialisation de l'algorithme de suivi basé modèle n'étant pas considérée dans ces travaux, la position initiale est obtenue dans les expériences menées en détectant automatiquement une ellipse positionnée dans la scène à la position initiale du drone (voir Figure 6.3 a). La position, l'orientation et l'aire de l'ellipse détectée fournissent une estimation de la pose initiale du drone, qui permet d'initialiser l'algorithme de suivi. Le drone est alors capable de se localiser grâce au suivi basé modèle.

Une tâche de navigation par points de passage a été réalisée en utilisant la commande hiérarchique présentée dans la section précédente. L'objectif de la commande consistait à rejoindre successivement différentes positions définies dans le repère fixe $\mathcal{R}_w$. Le drone est d'abord stabilisé 2m au-dessus de la cible portant l'ellipse ($\mathbf{p} = (0, 0, -2)$). Puis les points de passage successifs sont définis par $(0, -2, -2)$, $(-2, -2, -2)$, $(0, -2, -2)$, $(0, 0, -2)$, $(-2, 0, -2)$ (voir Figure 6.12 b). Le lacet est régulé à 0 pendant toute la séquence de navigation.



(a) (b)

FIGURE *6.12 – Exemple de vue de la caméra embarquée avec la reprojection du modèle obtenue (a) et trajectoire demandée (b).*

Les Figures 6.13 et 6.14 montrent l'erreur de position (4.9) obtenue sur chaque axe. Les pics correspondent aux changements de consigne. À convergence, le drone se stabilise à la position demandée avec une erreur de l'ordre de 15cm sur les axes x et y, et jusqu'à 30cm sur l'axe z. La Figure 6.15 montre l'erreur de vitesse (4.11) obtenue. Les oscillations obtenues sur l'erreur de vitesse sont dûes principalement au problème de latence entre les différentes données (inertielles et images).

Les faibles vitesses observées confirment l'hypothèse de vol stationnaire énoncée dans la section 6.2.

La Figure 6.16 représente la trajectoire 3D correspondant à cette séquence telle qu'elle est obtenue par l'approche proposée (en bleu) et la vérité terrain acquise à l'aide du dispositif présenté Figure 6.4. La tâche de navigation est réalisée avec succès.

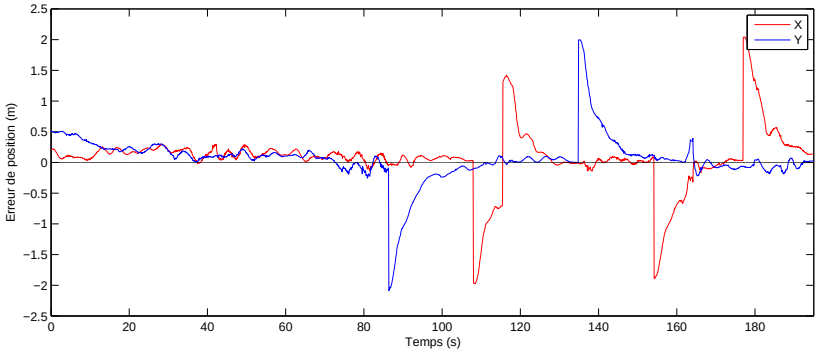


FIGURE *6.13 – Erreur de positionnement en x et y.*

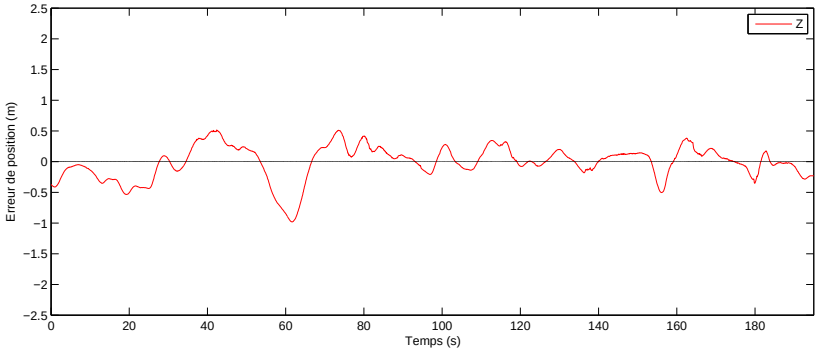


FIGURE *6.14 – Erreur de positionnement en altitude.*

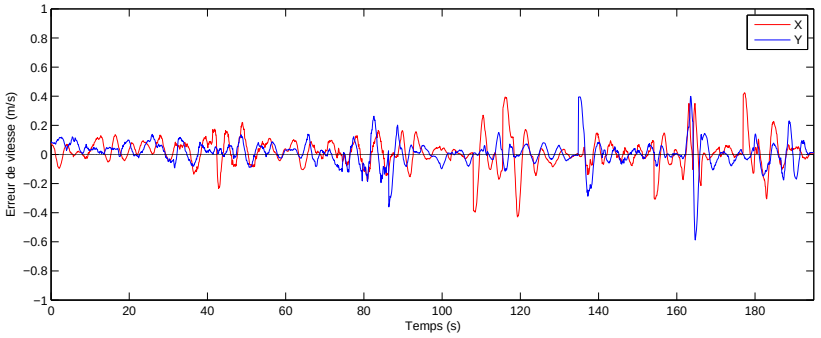


FIGURE *6.15 – Erreur de vitesse.*

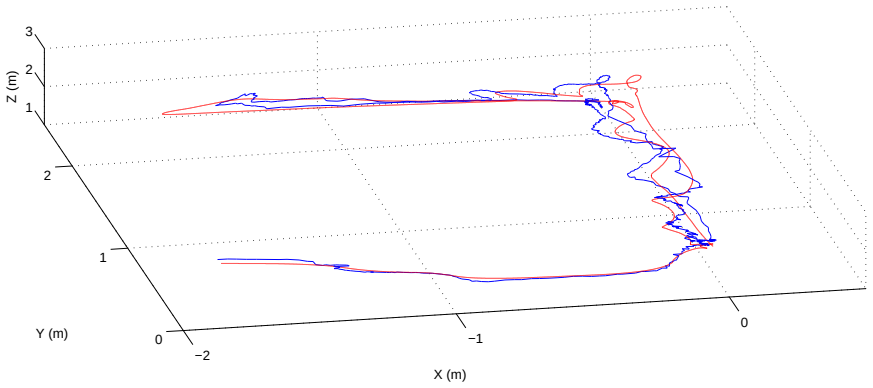


FIGURE *6.16 – Trajectoire du drone estimée par l'algorithme de vision (en bleu) et vérité terrain (en rouge).*

6.4 Conclusion

Dans ce chapitre nous avons montré comment utiliser les informations de pose reconstruites par vision afin d'effectuer des tâches de positionnement sur un drone quadrirotor. Après avoir présenté les lois de commande utilisées nous avons montré comment construire une estimée de la vitesse de translation du véhicule, nécessaire à la stabilité de la commande. La précision des estimations de position et de vitesse a été évaluée en comparant celles-ci aux mesures d'un système laser de haute précision. Cette évaluation a mis en évidence la qualité de la reconstruction obtenue. Enfin, nous avons présenté des résultats expérimentaux de positionnement 3D en effectuant des tâches de navigation par points de passage. Ces résultats démontrent ainsi la validité de l'approche proposée.

Conclusion et perspectives

L'objectif des travaux que nous avons réalisés dans cette thèse a consisté à proposer des solutions adaptées à la réalisation de tâches de « haut-niveau » par des mini-drones, à partir d'informations visuelles. Nous avons pour cela proposé des approches complètes, depuis l'extraction robuste d'informations visuelles jusqu'à la commande d'un drone à partir de ces informations. Nous avons également validé expérimentalement les approches proposées sur un drone quadrirotor développé par le CEA LIST.

Nous nous sommes intéressés en particulier à deux types de tâches. Dans un premier temps, nous avons considéré une tâche de poursuite d'un objet mobile dans un environnement inconnu. Dans ce cadre, l'extraction des informations visuelles doit permettre d'obtenir la position de l'objet dans l'image malgré des contraintes fortes : vitesse, déplacements importants, occultations éventuelles, bruit, etc. Pour y répondre nous avons proposé une représentation multi-noyaux de l'objet à suivre. Après avoir mis en évidence le caractère discriminant de ce mode de représentation, nous avons décrit des algorithmes de suivi de type optimisation déterministe et par filtrage particulaire utilisant ce descripteur pour estimer la position, la taille et l'orientation de l'objet dans l'image. Nous avons alors proposé une approche hybride entre ces approches, permettant de bénéficier des avantages de chacune de ces méthodes. En particulier les approches déterministes permettent un parcours continu de l'espace des paramètres permettant une grande précision. Le filtre particulaire échantillonne cet espace par un ensemble fini de particules mais bénéficie d'un modèle de prédiction et d'une plus grande robustesse notamment en cas d'occultation et de grands déplacements. Les qualités de l'approche hybride proposée ont été mises en évidence dans des expériences comparatives de suivi sur des séquences rapides.

Afin de prendre en compte de possibles échecs du suivi, nous avons également mis en place un système de détections des échecs et de réinitialisation automatique basé sur la technique du *Camshift*.

Une fois extraites les informations de position, orientation, et taille de l'objet dans l'image, nous avons proposé un schéma de commande (en translation et en lacet) permettant de réaliser la tâche de poursuite en elle-même. Le système complet

a été implémenté et testé sur un mini-drone quadrirotor dans des tâches de poursuite d'une voiture miniature. Notre approche a permis de faire face avec succès à des mouvements variés, en particulier des changements d'échelle importants et des occultations totales, sur des séquences allant jusqu'à 10 000 images environ.

Dans un deuxième temps nous avons considéré le cas d'un drone évoluant dans un environnement structuré dont un modèle peut être disponible - typiquement l'intérieur d'un bâtiment ou d'une installation industrielle. Dans ce contexte où le signal GPS fait généralement défaut, on souhaite utiliser la vision pour permettre la navigation ou le positionnement absolu du drone.

Nous avons pour cela proposé un algorithme d'estimation de pose robuste basé sur un processus d'optimisation multi-hypothèses intégré dans le cadre d'un filtrage particulaire. L'application d'une procédure d'optimisation multi-hypothèses sur les meilleures particules du filtre permet de guider celui-ci vers les régions pertinentes de l'espace d'état. L'algorithme qui en résulte a été testé sur différentes séquences acquises depuis un drone. Les résultats comparatifs par rapport aux approches classiques montrent les performances avantageuses de notre approche notamment en terme de robustesse aux ambiguïtés d'appariement des contours.

Dans ce cadre nous avons également considéré la commande du drone à partir des informations de pose. Pour des raisons de fréquence d'exécution nous avons choisi de valider notre approche en utilisant un retour de vitesse estimé en fusionnant les données visuelles et inertielles à l'aide d'un filtre de Kalman étendu, sans faire usage du filtre particulaire.

La précision des estimations de pose et de vitesse obtenues a été évaluée à l'aide d'une vérité terrain de haute précision, montrant des résultats très satisfaisants au regard de l'application envisagée, avec un écart-type de l'erreur de l'ordre de 15cm en translation. Enfin l'asservissement visuel a été mis en place, en considérant quatre degrés de liberté du drone à savoir sa position dans l'espace et son orientation en lacet. Des tâches de navigation ont été réalisées avec succès, validant l'approche complète proposée.

Perspectives

Si les approches proposées se sont révélées satisfaisantes dans des situations réelles, un certain nombre de questions n'ont pas pu être couvertes dans le cadre de cette thèse et ouvrent des perspectives pour des travaux ultérieurs.

Poursuite

Dans la tâche de poursuite que nous avons étudiée dans la première partie de nos travaux, nous avons considéré que le modèle de référence de l'objet suivi est constant tout au long de la séquence, il n'est réinitialisé qu'à la suite d'une détection automatique. Dans des séquences longues comportant des variations importantes de l'apparence de l'objet, du fond et des conditions d'illumination, il sera important de considérer la mise à jour automatique du modèle de référence. Une autre perspective

intéressante concernant le modèle de représentation pourrait consister à combiner des informations de couleur et de gradient.

En ce qui concerne la commande, l'estimation de la taille de l'objet dans l'image s'est avérée insuffisante pour contrôler l'altitude du drone. Sa fusion avec d'autres capteurs présents sur le drone (notamment le baromètre) pourrait améliorer cette partie de la commande.

D'autre part, nous avons ici considéré une tâche de poursuite unique. Il pourrait être intéressant également de considérer des contraintes supplémentaires dans l'exécution de cette tâche. On peut penser notamment à des contraintes d'évitement d'obstacles associées à la navigation du drone pendant la tâche de poursuite. De telles contraintes auraient alors un impact à la fois sur le suivi et sur la commande.

Positionnement

La deuxième partie de nos travaux a permis de mettre en place un système de navigation par la vision, basé sur un modèle simplifié de l'environnement. Les modèles que nous avons utilisés ont été construits par des mesures relativement grossières et le suivi s'est montré robuste à des erreurs de modélisation de l'ordre de quelques centimètres. Il serait cependant intéressant d'évaluer plus précisément à la fois la précision de modélisation requise par notre approche et la quantité d'information nécessaire dans le modèle.

L'utilisation d'un modèle soulève également la question de la gestion des environnements variables. En particulier, on peut se demander jusqu'à quel point les changements de l'environnement doivent être pris en compte comme des erreurs de modélisation auxquelles la robustesse de l'algorithme devra faire face, ou dans quelle mesure une approche adaptative par apprentissage sera préférable.

La validation expérimentale que nous avons réalisée a fait appel à une version simplifiée de notre algorithme de suivi dans la mesure où l'approche multi-hypothèses déterministe a été utilisée dans le cadre d'un filtre de Kalman étendu, sans filtrage particulaire pour avoir une fréquence d'exécution suffisamment élevée pour la commande. Si les résultats obtenus avec cet algorithme simplifié ont permis de montrer la validité de notre approche, il n'en serait pas moins intéressant de pouvoir évaluer plus précisément les apports engendrés par l'usage du filtrage particulaire dans la commande.

En outre, pour cette partie de nos travaux nous avons considéré que la navigation pouvait être initialisée en utilisant une cible connue au début de la mission. Cependant, cette approche ne permet pas de prendre en compte les échecs possibles du suivi, et un module de réinitialisation automatique serait alors à envisager.

Annexe A

Application du suivi multi-noyaux à l'asservissement basé moments

Avant de mettre en place une tâche de poursuite sur un drone à partir des informations visuelles basées sur l'approche multi-noyaux décrite dans le chapitre 3, nous avons utilisé ces informations pour asservir un robot cartésien à 6 degrés de liberté et de commande. Cette annexe présente les travaux que nous avons réalisés dans ce cadre.

A.1 Asservissement visuel

L'objectif de l'asservissement visuel mis en place est de minimiser l'erreur $\boldsymbol{\varepsilon}$ entre la valeur courante d'un ensemble de k primitives visuelles $\mathbf{s}$ et sa valeur désirée $\mathbf{s}^*$:

$$\boldsymbol{\varepsilon} = \mathbf{s} - \mathbf{s}^* \tag{A.1}$$

Soit $\mathbf{v} = (\boldsymbol{v}, \boldsymbol{\omega})$ la vitesse instantanée de la caméra, avec $\boldsymbol{v}$ la vitesse de translation de l'origine de la caméra et $\boldsymbol{\omega}$ la vitesse angulaire du repère attaché à la caméra. Dans le cas d'un objet mobile, la variation temporelle de l'erreur est donnée par :

$$\dot{\boldsymbol{\varepsilon}} = \dot{\mathbf{s}} = \mathbf{L}_\mathbf{s}\mathbf{v} + \frac{\partial \boldsymbol{\varepsilon}}{\partial \mathbf{t}} \tag{A.2}$$

où $\mathbf{L}_\mathbf{s} \in \mathbb{R}^{k\times 6}$ est la matrice d'interaction associée à $\mathbf{s}$. Le terme $\frac{\partial \boldsymbol{\varepsilon}}{\partial t}$ représente la variation temporelle de $\boldsymbol{\varepsilon}$ due au mouvement propre de l'objet.

A.2 Primitives visuelles et matrice d'interaction

À partir des coordonnées (x, y) du centre de gravité $\mathbf{c}$, de la taille S et de l'orientation ψ de l'objet obtenues par le suivi, on peut extraire quatre primitives visuelles.

Afin d'obtenir une matrice d'interaction bien conditionnée, les primitives suivantes ont été utilisées dans les expérimentations :

$$\mathbf{s} = (x_n, y_n, a_n, \psi) \tag{A.3}$$

où $a_n = \frac{1}{\sqrt{S}}, x_n = a_n x, y_n = a_n y$ et ψ est l'orientation estimée par l'algorithme de suivi (voir la section 3.2).

Les matrices d'interaction de ces primitives sont données dans [Tahri 05] à partir de l'utilisation des moments.

A.3 Estimation du mouvement propre de l'objet

Afin d'assurer la décroissance exponentielle de l'erreur $\boldsymbol{\varepsilon}$ (c'est-à-dire $\dot{\boldsymbol{\varepsilon}} = -\lambda\boldsymbol{\varepsilon}$), la loi de commande suivante est obtenue à partir de (A.2) :

$$\mathbf{v} = -\lambda\widehat{\mathbf{L}_\mathbf{s}^+}\boldsymbol{\varepsilon} - \widehat{\mathbf{L}_\mathbf{s}^+}\widehat{\frac{\partial\boldsymbol{\varepsilon}}{\partial\mathbf{t}}}. \tag{A.4}$$

Pour éviter les erreurs de traînage, une bonne estimation de $\frac{\partial\boldsymbol{\varepsilon}}{\partial t}$ est donc nécessaire. D'après (A.2) une première approximation de $\frac{\partial\boldsymbol{\varepsilon}}{\partial t}$ est :

$$\widehat{\frac{\partial\mathbf{e}}{\partial t}} = \widehat{\dot{\boldsymbol{\varepsilon}}} - \widehat{\mathbf{L}_\mathbf{s}}\widehat{\mathbf{v}} \tag{A.5}$$

où $\widehat{\dot{\mathbf{e}}}$ peut être approchée de manière discrète par :

$$\widehat{\dot{\boldsymbol{\varepsilon}}}_k = \frac{\boldsymbol{\varepsilon}_k - \boldsymbol{\varepsilon}_{k-1}}{\delta t}. \tag{A.6}$$

Pour les expérimentations, un filtrage de Kalman de $\widehat{\frac{\partial\boldsymbol{\varepsilon}}{\partial t}}$ a été implémenté, avec un modèle d'état à vitesse constante, en utilisant (A.5) et (A.6) pour la mesure (comme dans [Chaumette 93]). Cela permet de réduire l'erreur de traînage (voir Figure A.3) due au mouvement propre de l'objet.

A.4 Résultats expérimentaux

La méthode présentée ci-dessus a été testée pour le suivi d'un objet mobile (la voiture bleue dans la Figure A.1) en utilisant une caméra montée sur l'effecteur d'un robot à 6 degrés de liberté (Figure A.1 gauche). La position désirée est représentée par le rectangle rouge. Le rectangle vert indique la position courante de la voiture, telle qu'elle est estimée par l'algorithme de suivi. La voiture étant déplacée manuellement, il n'y a pas de vérité terrain. Cependant, les principales composantes du mouvement sont des translations le long de l'axe y, dans un sens, puis dans l'autre. De manière prévisible, on constate que le signe de la vitesse estimée de l'objet sur cet axe change lorsque la direction du déplacement change (voir Figure A.2). L'utilisation d'un estimateur de vitesse permet de conserver une erreur faible au cours

du déplacement. Le mouvement de l'objet étant estimé avec un modèle à vitesse constante, les variations brutales dans la vitesse de l'objet induisent des erreurs de traînage. En effet, les changements de vitesse qui apparaissent dans les images 30 et 37 (voir Figure A.2) entraînent une augmentation de l'erreur comme on peut le voir sur les Figures A.3 et A.4-c.

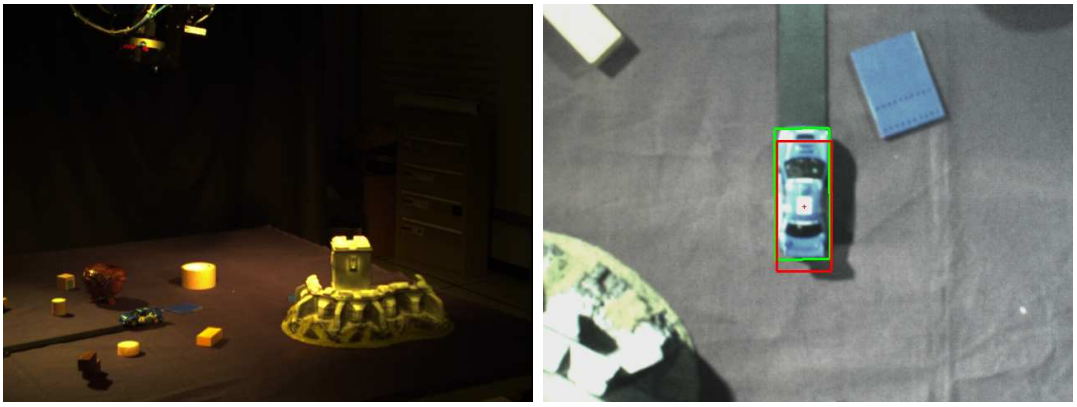


FIGURE *A.1 – Suivi d'objet mobile.*

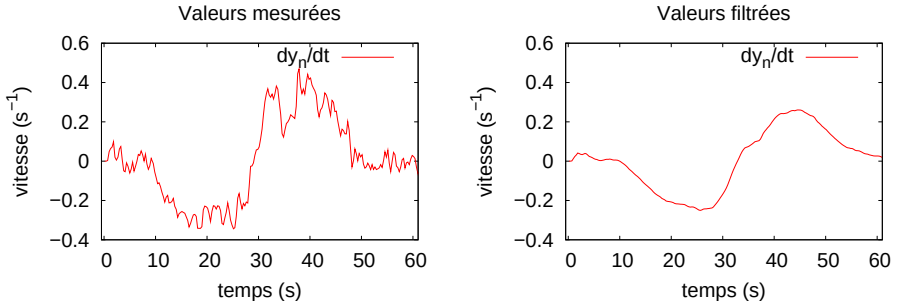


FIGURE *A.2 – Vitesses mesurée et estimée de la primitive visuelle y_n dues au mouvement propre de l'objet.*

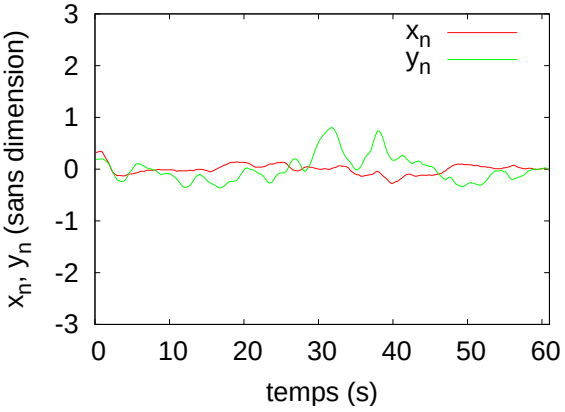
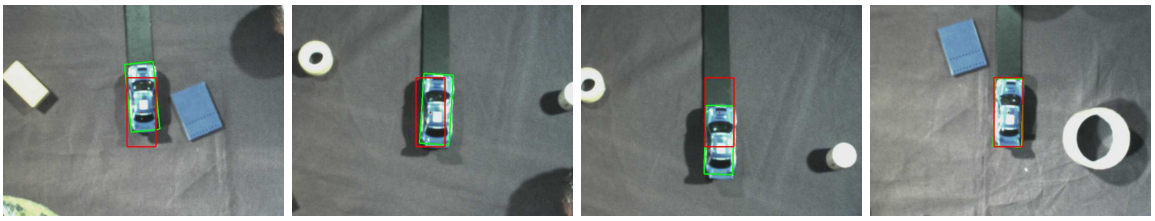


FIGURE *A.3 – Erreur pour les primitives x_n et y_n. La valeur désirée est zéro.*



(a) (b) (c) (d)

FIGURE *A.4 – Vue interne. Le rectangle rouge représente la position désirée, le rectangle vert indique la position courante fournie par la procédure de suivi. (a) et (b) montrent des exemples de vues avec des erreurs faibles en rotation et translations. (c) correspond au changement de direction, qui induit une forte erreur de traînage. (d) est la position finale. Bien que la cible semble verticale, il faut garder en tête que la caméra est contrôlée pour maintenir cette position dans l'image, malgré les mouvements de rotations qui ont été imposés à l'objet.*

Bibliographie

[Altug 02] E. Altug, J.P. Ostrowski, R. Mahony. – Control of a quadrotor helicopter using visual feedback. – *IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 72 – 77, 2002.

[Altug 03] E. Altug, J.P. Ostrowski, C.J. Taylor. – Quadrotor control using dual camera visual feedback. – *IEEE Int. Conf. on Robotics and Automation*, vol. 3, pp. 4294 – 4299, sept. 2003.

[Amidi 99] O. Amidi, T. Kanade, K. Fujita. – A visual odometer for autonomous helicopter flight. *Journal of Robotics and Autonomous Systems*, 28 :185 – 193, August 1999.

[Armstrong 95] M. Armstrong, A. Zisserman. – Robust object tracking. – *Proc. Asian Conference on Computer Vision*, vol. 1, pp. 58–61, 1995.

[Artieda 09] J. Artieda, J.and Sebastian, P. Campoy, J. Correa, I. Mondragón, C. Martínez, M. Olivares. – Visual 3-d slam from uavs. *Journal of Intelligent Robotic Systems*, 55 :299–321, 2009. – 10.1007/s10846-008-9304-8.

[Arulampalam 02] S. Arulampalam, S. Maskell, N. Gordon. – A tutorial on particle filters for on-line nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50 :174–188, 2002.

[Azrad 10] S. Azrad, F. Kendoul, K. Nonami. – Visual servoing of quadrotor micro-air vehicle using color-based tracking algorithm. *Journal of System Design and Dynamics*, 4(2) :255–268, 2010.

[Babu 07] R. V. Babu, P. Pérez, P. Bouthemy. – Robust tracking with motion estimation and local kernel-based color modeling. *Image Vision Comput.*, 25(8) :1205–1216, 2007.

[Bachta 10] W. Bachta, P. Renaud, E. Malis, K. Hashimoto, J. Gangloff. – Visual servoing for beating heart surgery. *Visual Servoing via Advanced Numerical Methods*, éd. par Graziano Chesi, Koichi Hashimoto, pp. 91–114. – Springer Berlin / Heidelberg, 2010.

[BarShalom 93] Y. Bar-Shalom, X.-R. Li. – *Estimation and Tracking, Principles, Techniques, and Software*. – Artech House, Boston, 1993.

[Benhimane 04] S. Benhimane, E. Malis. – Real-time image-based tracking of planes using efficient second-order minimization. – *IEEE Int.*

Conf. on Intelligent Robots and Systems, vol. 943-948, p. 1, Sendai, Japan, 2004.

[Bertrand 08] S. Bertrand, T. Hamel, H. Piet-Lahanier. – Stability analysis of an uav controller using singular perturbation theory. – *in Proceedings of the 17th IFAC World Congress*, pp. 5706–5711, 2008.

[Bhattacharyya 43] A. Bhattacharyya. – On a measure of divergence between two statistical populations defined by their probability distributions. *Bull. Calcutta Math Soc.*, 35 :99–109, 1943.

[Birchfield 98] S. Birchfield. – Elliptical head tracking using intensity gradients and color histograms. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 232–237, 1998.

[Birchfield 05] S. Birchfield, S. Rangarajan. – Spatiograms versus histograms for region-based tracking. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 1158–1163, Washington, DC, USA, 2005.

[Black 96] M. Black, P. Anandan. – The robust estimation of multiple motions : Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1) :75 – 104, 1996.

[Bouabdallah 05] S. Bouabdallah, R. Siegwart. – Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. – *IEEE Int. Conf. on Robotics and Automation*, pp. 2247 – 2252, apr. 2005.

[Bourquardez 09] O. Bourquardez, R. Mahony, N. Guenard, F. Chaumette, T. Hamel, L. Eck. – Image-based visual servo control of the translation kinematics of a quadrotor aerial vehicle. *IEEE Trans. on Robotics*, 25(3) :743 –749, jun. 2009.

[Bradski 98] G. Bradski. – *Computer Vision Face Tracking For Use in a Perceptual User Interface.* – Rapport de recherche, Intel Technology, 1998.

[Bray 04] M. Bray, E. Koller-Meier, L. Van Gool. – Smart particle filtering for 3d hand tracking. *IEEE International Conference on Automatic Face and Gesture Recognition*, 1 :675–680, 2004.

[Brown 71] D.C. Brown. – Close-range camera calibration. *Photogrammetric Engineering*, 4(2) :127–140, mars 1971.

[Brèthes 10] L. Brèthes, F. Lerasle, P. Danès, M. Fontmarty. – Particle filtering strategies for data fusion dedicated to visual tracking from a mobile robot. *Machine Vision and Applications*, 21 :427–448, 2010.

[Cha 08] S. Cha. – Taxonomy of nominal type histogram distance measures. – *MATH'08 : Proceedings of the American Conference on Applied Mathematics*, pp. 325–330, 2008.

[Chaumette 93] F. Chaumette, A. Santos. – Tracking a moving object by visual servoing. – *12th IFAC World Congress*, vol. 3, pp. 643–648, Sydney, Australia, July 1993.

[Chaumette 06] F. Chaumette, S. Hutchinson. – Visual servo control, part i : Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4) :82–90, December 2006.

[Chaumette 07] F. Chaumette, S. Hutchinson. – Visual servo control, part ii : Advanced approaches. *IEEE Robotics and Automation Magazine*, 14(1) :109–118, March 2007.

[Chen 01] H.-T. Chen, T.-L. Liu. – Trust-region methods for real-time tracking. *IEEE Int. Conf. on Computer Vision*, 2 :717, 2001.

[Chiuso 00] A. Chiuso, S. Soatto. – Monte-carlo filtering on lie groups. – *IEEE Conf. Decision and Control*, vol. 1, pp. 304–309, 2000.

[Collins 03] R.T. Collins. – Mean-shift blob tracking through scale space. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. II – 234–40, jun. 2003.

[Comaniciu 00] D. Comaniciu, V. Ramesh, P. Meer. – Real-time tracking of non-rigid objects using mean shift. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 142–149, 2000.

[Comaniciu 03] D. Comaniciu, V. Ramesh, P. Meer. – Kernel-based object tracking. – *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, 2003.

[Comport 06] A.I. Comport, E. Marchand, M. Pressigout, F. Chaumette. – Real-time markerless tracking for augmented reality : The virtual visual servoing framework. *IEEE Trans. on visualization and computer graphics*, 12(4) :615–28, July/August 2006.

[Courbon 08] J. Courbon, Y. Mezouar, P. Martinet. – Indoor navigation of a non-holonomic mobile robot using a visual memory. *Autonomous Robots*, 25 :253–266, 2008. – 10.1007/s10514-008-9093-8.

[Courbon 10] J. Courbon, Y. Mezouar, N. Guenard, P. Martinet. – Vision-based navigation of unmanned aerial vehicles. *Control Engineering Practice*, 18(7) :789 – 799, 2010. – Special Issue on Aerial Robotics.

[Crétual 01] A. Crétual, F. Chaumette. – Visual servoing based on image motion. *Int. Journal of Robotics Research*, 20(11) :857–877, novembre 2001.

[Dame 10] A. Dame, E. Marchand. – Accurate real-time tracking using mutual information. – *IEEE Int. Symposium on Mixed and Augmented Reality*, Seoul, Korea, October 2010.

[Dementhon 95] D.F. Dementhon, L.S. Davis. – Model-based object pose in 25 lines of code. *Int. Journal of Computer Vision*, 15 :123–141, 1995.

[Demonceaux 07] C. Demonceaux, P. Vasseur, C. Pegard. – UAV attitude computation by omnidirectional vision in urban environment. – *IEEE Int. Conf. on Robotics and Automation*, pp. 2017 –2022, apr. 2007.

[Doucet 01] N. Doucet, A.and De Freitas, N. Gordon (édité par). – *Sequential Monte Carlo methods in practice.* – Springer-Verlag, 2001.

[Dowson 08] N. Dowson, R. Bowden. – Mutual information for lucas-kanade tracking (milk) : An inverse compositional formulation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30(1) :180–185, janvier 2008.

[Drummond 02] R. Drummond, T. Cipolla. – Real-time visual tracking of complex structures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7) :932–946, 2002.

[DurrantWhyte 06] H. Durrant-Whyte, T. Bailey. – Simultaneous localization and mapping : part i. *IEEE Robotics Automation Magazine*, 13(2) :99 –110, jun. 2006.

[Fabbri 08] R. Fabbri, L. Da F. Costa, J. C. Torelli, O. M. Bruno. – 2d euclidean distance transform algorithms : A comparative survey. *ACM Comput. Surv.*, 40(1) :1–44, 2008.

[Fan 05] Z. Fan, Y. Wu, M. Yang. – Multiple collaborative kernel tracking. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 502 – 509 vol. 2, 20-25 2005.

[Fan 07] Z. Fan, M. Yang, Y. Wu. – Multiple collaborative kernel tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(7) :1268 –1273, july 2007.

[Faugeras 87] O. D. Faugeras, G. Toscani. – Camera calibration for 3d computer vision. – *International Workshop on Machine Vision and Machine Intelligence*, Tokyo, 1987.

[Faugeras 93] O.D. Faugeras. – *Three dimensional Computer Vision : A Geometric Viewpoint.* – MIT Press, 1993.

[Feddema 98] J.T. Feddema, R.W. Simon. – Cad-driven microassembly and visual servoing. – *IEEE Int. Conf. on Robotics and Automation*, vol. 2, pp. 1212–1219, Leuven, Belgique, mai 1998.

[Fischler 81] N. Fischler, R.C. Bolles. – Random sample consensus : A paradigm for model fitting with application to image analysis and automated cartography. *Communication of the ACM*, 24(6) :381–395, juin 1981.

[Gordon 93] N. J. Gordon, D. J. Salmond, A. F. M. Smith. – Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation. *Radar and Signal Processing, IEEE Proceedings F*, 140(2) :107–113, 1993.

[Grassia 98] F. S. Grassia. – Practical parameterization of rotations using the exponential map. *Journal of Graphics, GPU, and Game tools*, 3(3) :29–48, 1998.

[Guenard 06] N. Guenard, T. Hamel, L. Eck. – Control laws for the tele operation of an unmanned aerial vehicle known as an x4-flyer. – *IEEE Int. Conf. on Intelligent Robots and Systems*, pp. 3249 –3254, oct. 2006.

[Guenard 07] N. Guenard. – *Optimisation et implémentation de lois de commande embarquées pour la téléopération de micro drones aériens "X4-flyer"*. – PhD. Thesis, Université de Nice-Sophia Antipolis, 2007.

[Guenard 08] N. Guenard, T. Hamel, Mahony. – A practical visual servo control for an unmanned aerial vehicle. *IEEE Trans. on Robotics*, 24(2) :331–340, April 2008.

[Guskov 06] I. Guskov. – Kernel-based template alignment. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 610–617, Washington, DC, USA, 2006.

[Hager 98] G. Hager, P. Belhumeur. – Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(10) :1025–1039, 1998.

[Hager 04] G. Hager, M. Dewan, C. Stewart. – Multiple kernel tracking with ssd. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 1, pp. 790–797, 2004.

[Haralick 89] R. Haralick, H. Joo, C. Lee, X. Zhuang, V Vaidya, M. Kim. – Pose estimation from corresponding point data. *IEEE Trans on Systems, Man and Cybernetics*, 19(6) :1426–1445, novembre 1989.

[Haritaoglu 98] I. Haritaoglu, D. Harwood, L. S. Davis. – W4s : A real-time system for detecting and tracking people in $2\frac{1}{2}$d. *European Conf. on Computer Vision*, pp. 877–. – Springer Berlin / Heidelberg, 1998.

[Hartigan 75] J.A. Hartigan. – *clustering algorithms*. – John Wiley & Sons, 1975.

[Hartley 01] R. Hartley, A. Zisserman. – *Multiple View Geometry in Computer Vision*. – Cambridge University Press, 2001.

[Hérissé 10] B. Hérissé, T. Hamel, R. Mahony, F.-X. Russotto. – A terrain-following control approach for a vtol unmanned aerial vehicle using average optical flow. *Autonomous Robots*, pp. 1–19, 2010.

[Horaud 89] R. Horaud, B. Conio, O. Leboulleux, B. Lacolle. – An analytic solution for the perspective 4-points problem. *Computer Vision, Graphics and Image Processing*, 47(1) :33–44, juillet 1989.

[Horn 87] B. Horn. – *Robot Vision*. – MIT Press, Cambridge, 1987.

[Hrabar 03] S. Hrabar, G. Sukhatme. – Omnidirectional vision for an autonomous helicopter. – *IEEE Int. Conf. on Robotics and Automation*, pp. 558–563, 2003.

[Huber 81] P.-J. Huber. – *Robust Statistics*. – Wiler, New York, 1981.

[Irani 98a] M. Irani, P. Anandan. – Robust multi-sensor image alignment. – *IEEE Int. Conf. on Computer Vision*, p. 959, Washington, DC, USA, 1998.

[Irani 98b] M. Irani, P. Anandan. – A unified approach to moving object detection in 2d and 3d scenes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(6) :577 –589, jun. 1998.

[Isard 98a] M. Isard, A. Blake. – Condensation : conditional density propagation for visual tracking. *Int. Journal of Computer Vision*, 29(1) :5–28, 1998.

[Isard 98b] M. Isard, A. Blake. – Icondensation : Unifying low-level and high-level tracking in stochastic framework. – *European Conf. on Computer Vision*, vol. 1, pp. 893–908, 1998.

[Jepson 03] A.D. Jepson, D.J. Fleet, T.F. El-Maraghi. – Robust online appearance models for visual tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(10) :1296 – 1311, oct. 2003.

[Jeyakar 08] J. Jeyakar, R. Babu, K.R. Ramakrishnan. – Robust object tracking with background-weighted local kernels. *Computer Vision and Image Understanding*, 112(3) :296 – 309, 2008.

[Jurie 02] F. Jurie, M. Dhome. – Hyperplane approximation for template matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7) :996–1000, 2002.

[Kanazawa 93] Y. Kanazawa. – Hellinger distance and akaike's information criterion for the histogram. *Statistics & Probability Letters*, 17(4) :293 – 298, 1993.

[Kemp 05] C. Kemp, T. Drummond. – Dynamic measurement clustering to aid real time tracking. – *IEEE Int. Conf. on Computer Vision*, pp. 1500–1507, Washington, DC, USA, 2005.

[Klein 04] G. Klein, T. Drummond. – Tightly integrated sensor fusion for robust visual tracking. *Image and Vision Computing*, 22(10) :769–776, September 2004.

[Klein 06] G. Klein, D. Murray. – Full-3d edge tracking with a particle filter. – *British Machine Vision Conf.*, vol. 3, pp. 1119–1128, 2006.

[Klose 10] S. Klose, J. Wang, M. Achtelik, G. Panin, F. Holzapfel, A. Knoll. – Markerless, vision-assisted flight control of a quadrocopter. – *IEEE Int. Conf. on Intelligent Robots and Systems*, Taipei, Taiwan, October 2010.

[Kwon 07] J. Kwon, M. Choi, C. Chun, F.C. Park. – Particle filtering on the euclidean group. – *IEEE Int. Conf. on Robotics and Automation*, pp. 3552–3557, 2007.

[Lepetit 05] V. Lepetit, P. Fua. – Monocular model-based 3d tracking of rigid objects : A survey. – *Foundations and Trends in Computer Graphics and Vision*, pp. 1–89, 2005.

[Lots 01] J.-F. Lots, D. Lane, E. Trucco, F. Chaumette. – A 2-D visual servoing for underwater vehicle station keeping. – *IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, mai 2001.

[Lowe 91] D.G. Lowe. – Fitting parameterized three-dimensional models to images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13 :441–450, 1991.

[Lowe 92] D.G. Lowe. – Robust model-based motion tracking trough the integration of search and estimation. *Int. Journal of Computer Vision*, 8(2) :113–122, 1992.

[Lu 00] C.P. Lu, G.D. Hager, E. Mjolsness. – Fast and globally convergent pose estimation from video images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(6) :610–622, juin 2000.

[Lucas 81] B.D. Lucas, T. Kanade. – An iterative image registration technique with an application to stereo vision. – *Int. Joint Conf. on Artificial Intelligence, IJCAI'81*, pp. 674–679, 1981.

[MacQueen 67] J. B. MacQueen. – Some methods for classification and analysis of multivariate observations. – L. M. Le Cam, J. Neyman (édité par), *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297. University of California Press, 1967.

[Maggio 05a] E. Maggio, A. Cavallaro. – Hybrid particle filter and mean shift tracker with adaptive transition model. – *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 221–224, 2005.

[Maggio 05b] E. Maggio, A. Cavallaro. – Multi-part target representation for color tracking. – *Int. Conf. on Image Processing*, vol. 1, pp. I – 729–32, 11-14 2005.

[Maggio 07] E. Maggio, F. Smerladi, A. Cavallaro. – Adaptive multifeature tracking in a particle filtering framework. *IEEE Trans. on Circuits and Systems for Video Technology*, 17(10) :1348 –1359, oct. 2007.

[Maggio 09] E. Maggio, A. Cavallaro. – Accurate appearance-based bayesian tracking for maneuvering targets. *Computer Vision and Image Understanding*, 113(4) :544 – 555, 2009.

[Malis 06] E. Malis, E. Marchand. – Experiments with robust estimation techniques in real-time robot vision. – *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'06*, pp. 223–228, Beijing, China, October 2006.

[Malis 07] E. Malis, S. Benhimane. – Homography-based 2d visual tracking and servoing. *The Int. Journal of Robotics Research*, 26(7) :661–676, 2007.

[Marchand 02] E. Marchand, F. Chaumette. – Virtual visual servoing : A framework for real-time augmented reality. – G. Drettakis, H.-P. Seidel (édité par), *EUROGRAPHICS 2002 Conference Proceeding*, vol. 21(3) of *Computer Graphics Forum*, pp. 289–298, Saarebrücken, Germany, September 2002.

[Marchand 05] E. Marchand, F. Spindler, F. Chaumette. – Visp for visual servoing : A generic software platform with a wide class of robot control skills. *IEEE Robotics and Automation Magazine*, 12(4), dec 2005.

[Mariottini 07] G. L. Mariottini, G. Oriolo, D. Prattichizzo. – Image-based visual servoing for nonholonomic mobile robots using epipolar geometry. *IEEE Trans. on Robotics*, 23(1) :87 –100, feb. 2007.

[Masutani 94] Y. Masutani, M. Mikawa, N. Maru, F. Miyazaki. – Visual servoing for non-holonomic mobile robots. – *IEEE Int. Conf. on Intelligent Robots and Systems*, vol. 2, pp. 1133–1140, sep. 1994.

[Matthews 03] I. Matthews, T. Ishikawa, S. Baker. – The template update problem. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26 :810–815, 2003.

[Mebarki 10] R. Mebarki, A. Krupa, F. Chaumette. – 2d ultrasound probe complete guidance by visual servoing using image moments. *IEEE Trans. on Robotics*, 26(2) :296–306, April 2010.

[Megret 06] R. Megret, M. Mikram, Y. Berthoumieu. – Inverse composition for multi-kernel tracking. *Computer Vision, Graphics and Image Processing*, pp. 480–491. – Springer Berlin/Heidelberg, 2006.

[Mei 07] C. Mei, P. Rives. – Cartographie et navigation simultanée avec un capteur de vision, october 2007.

[Mejías 06] L. Mejías, S. Saripalli, P. Campoy, G. Sukhatme. – Visual servoing of an autonomous helicopter in urban areas using feature tracking. *Journal of Field Robotics*, 23, 2006.

[Mejías 07] L. Mejías, P. Campoy, I. Mondragón, P. Doherty. – Stereo visual system for autonomous air vehicle navigation. *IFAC Symposium on Intelligent Autonomous Vehicles*, 2007.

[Metni 06] N. Metni, J.-M. Pflimlin, T. Hamel, P. Souères. – Attitude and gyro bias estimation for a vtol uav. *Control Engineering Practice*, 14(12) :1511 – 1520, 2006.

[Metni 07] N. Metni, T. Hamel. – Visual Tracking Control of Aerial Robotic Systems with Adaptive Depth Estimation. *Int. Journal of Control, Automation, and Systems*, 5(1) :51–60, 2007.

[Michael 10] N. Michael, D. Mellinger, Q. Lindsey, V. Kumar. – The grasp multiple micro-uav testbed. *IEEE Robotics Automation Magazine*, 17(3) :56 –65, sep. 2010.

[Moakher 02] M. Moakher. – Means and averaging in the group of rotations. *SIAM journal on matrix analysis and applications*, 24 :1–16, 2002.

[Nummiaro 02] K. Nummiaro, E. Loller-Meier, L. Van Gool. – A color-based particle filter. – *European Conf. on Computer Vision*, pp. 53–60, 2002.

[Nummiaro 03] K. Nummiaro, E. Loller-Meier, L. Van Gool. – An adaptive color-based particle filter. *Image and Vision Computing*, 2003.

[Nuske 08] S. Nuske, J. Roberts, G. Wyeth. – Visual localisation in outdoor industrial building environments. – *IEEE Int. Conf. on Robotics and Automation*, pp. 544–550, 2008.

[Okuma 04] K. Okuma, A. Taleghani, O. De Freitas, J. Little, D. Lowe. – A boosted particle filter : Multitarget detection and tracking. – *European Conf. on Computer Vision*, pp. 28–39, 2004.

[Pérez 02] P. Pérez, C. Hue, J. Vermaak, M. Gangnet. – Color-based probabilistic tracking. – *European Conf. on Computer Vision*, pp. 661–675, May 2002.

[Pflimlin 06] J.-M. Pflimlin. – *Commande d'un minidrone à hélice carénée : de la stabilisation dans le vent à la navigation autonome.* – PhD. Thesis, Ecole Doctorale Systèmes de Toulouse, 2006.

[Phong 95] T.-Q. Phong, R. Horaud, A. Yassine, P.-D. Tao. – Object pose from a 2D to 3D point and line correspondance. *Int. Journal of Computer Vision*, 15(3) :225–243, juillet 1995.

[Pressigout 07] M. Pressigout, E. Marchand. – Real-time hybrid tracking using edge and texture information. *Int. Journal of Robotics Research, IJRR*, 26(7) :689–713, July 2007.

[Pupilli 06] M. Pupilli, A. Calway. – Real-time camera tracking using known 3d models and a particle filter. – *Int. Conf. on Pattern Recognition*, pp. 199–203, August 2006.

[Raletz 10] R. Raletz. – *Basic theory of the helicopter.* – Cépaduès Editions, 2010.

[Rives 97] P. Rives, J.-J. Borrelly. – Visual servoing techniques applied to an underwater vehicle. – *IEEE Int. Conf. on Robotics and Automation*, vol. 3, pp. 1851 –1856 vol.3, Albuquerque, USA, apr. 1997.

[Rives 02] P. Rives, J.R. Azinheira. – *Visual Auto-landing of an Autonomous Aircraft.* – Rapport de Recherche n4606, INRIA, 2002.

[Rives 04] P. Rives, J.R. Azinheira. – Linear structures following by an airship using vanishing point and horizon line in a visual servoing scheme. – *IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 255 – 260, apr. 2004.

[Rock 98] S.M. Rock, E.W. Frew, H. Jones, E.A. LeMaster, B.R. Woodley. – Combined cdgps and vision-based control of a small autonomous helicopter. – *American Control Conference*, vol. 2, pp. 694 –698, jun. 1998.

[Saripalli 03] S. Saripalli, J.F. Montgomery, G.S. Sukhatme. – Visually guided landing of an unmanned aerial vehicle. *IEEE Trans. on Robotics and Automation*, 19(3) :371 – 380, june 2003.

[Saripalli 07] S. Saripalli, G.S. Sukhatme. – Landing a helicopter on a moving target. – *IEEE Int. Conf. on Robotics and Automation*, pp. 2030 –2035, apr. 2007.

[Shakernia 99] O. Shakernia, Y. Ma, T. John Koo, S. Sastry. – Landing an unmanned air vehicle : Vision based motion estimation and nonlinear control. *Asian Journal of Control*, 1 :128–145, 1999.

[Shan 07] C. Shan, T. Tan, Y. Wei. – Real-time hand tracking using a mean shift embedded particle filter. *Pattern Recognition*, 40(7) :1958–1970, 2007.

[Sharp 01] C.S. Sharp, O. Shakernia, S.S. Sastry. – A vision system for landing an unmanned aerial vehicle. – *IEEE Int. Conf. on Robotics and Automation*, vol. 2, pp. 1720 – 1727, 2001.

[Shi 94] J. Shi, C. Tomasi. – Good features to track. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'94*, pp. 593–600, Seattle, Washington, juin 1994.

[Slama 80] C.C. Slama. – *Manual of Photogrammetry.* – American Society of Photogrammetry and Remote Sensing, Falls Church, Virginia, USA, 1980.

[Smith 92] A. F. M. Smith, A. E. Gelfand. – Bayesian statistics without tears : A sampling-resampling perspective. *The American Statistician*, 46(2) :84–88, May 1992.

[Stauffer 99] C. Stauffer, W.E.L. Grimson. – Adaptive background mixture models for real-time tracking. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 2, p. 252, 1999.

[Stein 97] G.P. Stein. – Lens distortion calibration using point correspondences. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 602–608, 1997.

[Sullivan 01] J. Sullivan, J. Rittscher. – Guiding random particles by deterministic search. – *IEEE Int. Conf. on Computer Vision*, vol. 1, pp. 323 –330, 2001.

[Tahri 05] O. Tahri, F. Chaumette. – Point-based and region-based image moments for visual servoing of planar objects. *IEEE Trans. on Robotics*, 21(6) :1116–1127, December 2005.

[Tamadazte 09] B. Tamadazte, N. Lefort-Piat, E. Marchand, S. Dembélé. – Microassembly of complex and solid 3d mems by 3d vision-based control. – *IEEE Int. Conf. on Intelligent Robots and Systems*, pp. 3284–3289, St Louis, USA, October 2009.

[Tomasi 91] C. Tomasi, T. Kanade. – *Detection and Tracking of Point Features.* – Rapport de Recherche nCMU-CS-91-132, Carnegie Mellon University Technical Report, avril 1991.

[Tsai 86] R.Y. Tsai. – An efficient and accurate camera calibration technique for 3D machine vision. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 364–374, Miami, Floride, juin 1986.

[Vacchetti 04] L. Vacchetti, V. Lepetit, P. Fua. – Combining edge and texture information for real-time accurate 3d camera tracking. – *IEEE Int. Symposium on Mixed and Augmented Reality*, pp. 48–57, 2004.

[Warner 83] F. W. Warner. – *Foundations of Differentiable Manifolds and Lie Groups*. – Springer-Verlag, 1983.

[Waslander 05] S.L. Waslander, G.M. Hoffmann, Jung Soon Jang, C.J. Tomlin. – Multi-agent quadrotor testbed control design : integral sliding mode vs. reinforcement learning. – *IEEE Int. Conf. on Intelligent Robots and Systems*, pp. 3712 – 3717, aug. 2005.

[Watanabe 10] Y. Watanabe, C. Lesire, P. Fabiani, M. Sanfourche, G. Le Besnerais. – The onera ressac unmanned autonomous helicopter : Visual air-to-ground target tracking in an urban environment. – *American Helicopter Society 66th Annual Forum (AHS 2010)*, 2010.

[Yilmaz 06] A. Yilmaz, O. Javed, M. Shah. – Object tracking : A survey. *ACM Comput. Surv.*, 38(4) :13, 2006.

[Zhang 99] H. Zhang, J.P. Ostrowski. – Visual servoing with dynamics : control of an unmanned blimp. – *IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 618 –623, 1999.

[Zhou 04] S. Zhou, R. Chellappa, B. Moghaddam. – Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Transactions on Image Processing*, 13 :1434–1456, 2004.

Résumé

Pour qu'un système robotisé puisse accomplir de façon autonome des fonctions en apparence simples, telles que se localiser ou se positionner par rapport à son environnement, il doit avant tout percevoir cet environnement. La perception visuelle obtenue à l'aide d'une caméra constitue à cet égard une source d'information particulièrement riche, largement utilisée en robotique.

Le travail présenté dans cette thèse concerne l'usage d'informations visuelles dans le contexte de la commande de mini-drones. En particulier deux types de tâches ont été considérées : une tâche de poursuite, dans laquelle un objet - une voiture - se déplace dans un environnement inconnu et l'on souhaite qu'un drone puisse suivre son mouvement, et une tâche de positionnement ou de navigation pour un drone évoluant dans un environnement structuré - intérieur de bâtiment - dans lequel le signal GPS n'est pas disponible.

Dans les deux cas, nous avons proposé des approches complètes, depuis l'extraction robuste d'informations visuelles jusqu'à la commande d'un drone à partir de ces informations. Des expériences mises en œuvre sur un mini-drone quadrirotor montrent la validité des approches proposées.

Mots-clefs : Vision par ordinateur, asservissement visuel, commande de drone

Abstract

For a robot to autonomously localise or position itself with respect to its environment, a key requirement is the perception of this environment. In this respect, the visual information provided by a camera is a particularly rich source of information, commonly used in robotics.

Our work deals with the use of visual information in the context of UAV control. More specifically, two tasks have been considered : first, a tracking task, in which a UAV has to follow a moving object - a car - in an unknown environment, second, a positioning task for a UAV navigating in a structured GPS-deprived environment.

In both cases, we propose full approaches, considering both the robust extraction of appropriate visual informations and the visual servoing of the UAV. The experiments performed on a small quadrotor UAV show the validity of our approaches.

Keywords : Computer vision, visual servoing, UAV control