

Université de Montréal

Estimation probabiliste du mouvement de caméra

par

Jamil Draréni

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en informatique

avril, 2003

© Jamil Draréni, 2003

Université de Montréal
Faculté des études supérieures

Ce mémoire intitulé :

Estimation probabiliste du mouvement de caméra

présenté par

Jamil Draréni

a été évalué par un jury composé des personnes suivantes:

Victor Ostromoukhov
président-rapporteur

Sébastien Roy
directeur de recherche

Neil Stewart
membre du jury

Mémoire accepté le _____

RÉSUMÉ

Ce mémoire présente une nouvelle méthode pour résoudre un problème de grande importance en vision par ordinateur : l'estimation du mouvement de caméra.

Le mouvement de caméra, ou encore *ego-mouvement* est défini comme étant le déplacement translationnel et rotationnel en trois dimensions qu'entreprend une caméra dans le temps.

Contrairement aux méthodes conventionnelles, celle qui est présentée dans ce mémoire utilise des concepts statistiques, considérés robustes, et ne requiert aucun suivi de points, de connaissance sur la scène ou de mise en correspondance explicite.

En effet, seules les variations d'intensités des pixels dans l'espace et le temps (dérivées spatio-temporelles) sont utilisées comme source d'information. Cette restriction permettra de considérer des scènes très complexes ou dans de faibles conditions d'éclairage.

Une fois le mouvement de caméra retrouvé, plusieurs applications sont alors envisageables. La plus évidente serait dans le domaine de la navigation dite passive où il est possible de retracer le chemin effectué par un robot mobile par le biais d'une caméra fixée sur ce dernier ou encore dans le domaine de la compression vidéo.

Mots clés : vision par ordinateur, analyse de mouvement, paramètres de caméra, flux optique.

ABSTRACT

This thesis presents a new approach for resolving a very important problem in Computer Vision, namely computing a camera ego-motion from a sequence of images.

The ego-motion is defined as the 3D rotational and translational displacement undertaken by a camera over time.

The method presented in this thesis is probabilistic, and thus very robust against noise, and does not require a prior knowledge of the scene structure nor tracking or feature matching.

Indeed, only the pixel's intensity variations (spatio-temporal derivatives) are required to perform such an estimation. This is an important feature because it allows handling of very difficult scenes such as a cluttered scene or a scene under bad lighting conditions.

Once the camera's egomotion is estimated, several applications can be envisaged. The most obvious ones are the so called *passive navigation* or video coding at low bit rate.

keywords : Computer vision, motion analysis, egomotion, camera parameters, optical flow.

TABLE DES MATIÈRES

Liste des Figures	iii
Chapitre 1 : Introduction	1
Chapitre 2 : Mouvement et flux optique	6
2.1 Généralité	6
2.2 Le problème d'ouverture	8
2.3 Flux Optique	9
2.4 Conclusion	18
Chapitre 3 : Caméras et paramètres de caméras	20
3.1 Modèle de camera	20
3.2 Paramètres internes	21
3.3 Paramètres externes	22
3.4 Mouvement de caméra	23
Chapitre 4 : Estimation du mouvement de caméra	25
4.1 Méthodes indirectes	26
4.2 Méthodes directes	29
4.3 Méthodes probabilistes	31
Chapitre 5 : (Article) Fast Probabilistic Estimation of Egomotion From Image Intensities	42
5.1 Introduction	42
5.2 Camera Motion Equations	45

5.3	Probabilistic Model of Camera Motion	50
5.4	Experiments and results	54
5.5	Conclusion	57
Chapitre 6 : Résultats supplémentaires		60
6.1	La résolution en temps réel	60
6.2	Les mouvements multiples	61
6.3	La minimisation d'énergie	62
Chapitre 7 : Discussion et conclusion		65
Références		67
Annexe A : Annexe		70
A.1	Angle de vue de la caméra	71
A.2	Minimisation de fonction d'énergie par la méthode de Powell	73

LISTE DES FIGURES

1.1	Exemple de mouvement difficile à estimer	5
2.1	Exemples d'heuristiques de correspondance.	9
2.2	Exemples d'ambiguïtés dûes au problème d'ouverture.	10
2.3	Autre exemple du probleme d'ouverture	11
2.4	Exemple de mouvements et de leur flux optique.	12
2.5	La contrainte de luminance (CBA).	14
3.1	Modèle de caméra sténopé.	21
4.1	Approche basée sur les points saillants.	26
4.2	Correspondance des points saillants.	27
4.3	distribution statistique de $\frac{6\pm 1}{3\pm 1}$	32
4.4	Distribution statistique de $\frac{0\pm 1}{0\pm 1}$	33
4.5	Distribution statistique de la puissance	34
4.6	Schéma de l'estimation probabiliste du mouvement de caméra	40
4.7	La contrainte de luminance (CBA).	41
5.1	Optical Flow	44
5.2	Random camera motions solved from normal flows	47
5.3	Constant Brightness assumption.	52
5.4	Distribution of v	52
5.5	Translation - Rotation ambiguity.	54
5.6	Translation - Rotation ambiguity Densities	55
5.7	Synthetic sequence	56

5.8	Translation scale factor	57
5.9	Corridor sequence and recovered trajectory of the camera	58
5.10	Sideways motion	59
5.11	Trajectory for Sideways sequence in low lighting conditions	59
6.1	Exemple de mouvement multiple	62
6.2	Courbe d'énergie pour un mouvement double	63
A.1	Angle de vue de la caméra	71
A.2	Minimisation 1D	75

à mes parents à qui je dois tout

REMERCIEMENTS

Je souhaite remercier tous ceux qui m'ont apporté le support si nécessaire à la création de ce mémoire, à commencer par mon directeur de recherche Dr Sébastien Roy, les membres du jury qui ont fourni un effort considérable à la lecture de ce mémoire, ma Fragola pour avoir été sans cesse à mes côtés, Lico pour ses encouragements, Isaac et Nawel pour leur soutien, ma nièce Laurence pour avoir fait de moi son *Da*, Mark pour toutes ses soirées qui m'ont fait oublier mes longues journées, et bien sûr le Chabeb inc pour tout le reste, sans oublier toutes les serveuses de la Brûlerie St-Denis qui s'occupaient de mes pauses-café.

Chapitre 1

INTRODUCTION

La recherche en vision par ordinateur s'applique à définir des algorithmes permettant la perception et la compréhension du monde physique (ex : structure, mouvements...) et ce à partir d'informations visuelles (images ou séquences d'images).

Le fait qu'une scène tridimensionnelle soit perçue à travers une ou plusieurs images 2D, rend la tâche d'analyse très ardue car il est impossible, dans la plupart des cas, de faire une correspondance directe entre un monde tridimensionnel et une image 2D. On a donc affaire à un problème mal posé.

En général, un système de vision complet se compose de trois étapes :

Extractions d'attributs : ceci est le plus souvent une opération de bas niveau tel que la couleur, la texture, le mouvement ou le filtrage.

Analyse des attributs : cette étape fournit des informations de plus haut niveau sur la scène comme la segmentation.

Interprétation de la scène : Cette étape permet de déduire ce qui se passe dans la scène.

Cette architecture, très générique certes, se retrouve dans tous les domaines d'applications de la vision par ordinateurs tel que la robotique, la compression vidéo, le suivi de mouvement, pour n'en citer que quelques uns.

Lorsque l'information visuelle varie au cours du temps, le mouvement, parmi tous les attributs bas-niveau, est sûrement le plus essentiel. Il apporte des informations qualitatives sur la structure tridimensionnelle de la scène, la trajectoire des objets ainsi que celle de la caméra.

Ce mémoire présente une nouvelle approche pour analyser et estimer le mouvement d'une caméra dans son univers tridimensionnel à partir d'une séquence vidéo. Ce mouvement est appelé *egomotion*.

Estimer ce mouvement revient à calculer le déplacement translationnel et rotationnel, entrepris par la caméra à un temps donné.

Les mouvements estimés peuvent être considérés instantanés ou discrets dépendamment de la magnitude apparente des vitesses. L'approche proposée dans ce mémoire traite les mouvements continus.

L'estimation de mouvement de caméra trouve sa principale application dans la robotique et la navigation automatisée. Il sera possible à un robot mobile, après estimation du mouvement, de déduire sa trajectoire sans avoir recours à des capteurs externes ou toute autre méthode invasive car la méthode présentée est purement passive.

D'autres applications sont envisageables dans le domaine de la compression vidéo où il est possible d'éliminer la redondance temporelle si le mouvement de caméra est connu, ce qui permet d'atteindre de plus hauts taux de compression.

L'abondance de la littérature dans ce domaine reflète l'importance de ce genre d'estimation. L'intérêt suscité par les chercheurs a donné naissance à une myriade d'algorithmes et de méthodes de résolution. Ces méthodes se divisent en deux grandes catégories : directes et indirectes.

La première catégorie regroupe les méthodes qui estiment le mouvement en se basant uniquement sur des mesures directes faites sur les images telles que les intensités des pixels, l'extraction de contours ou le calcul des dérivées spatio-temporelles.

La seconde catégorie par contre regroupe les méthodes qui estiment le mouvement en procédant à la résolution d'un sous-problème comme étape intermédiaire afin de résoudre le problème original. Parmi les sous-problèmes qu'un algorithme doit résoudre on retrouve souvent le flux optique (optical flow) et l'appariement des points d'intérêts (feature matching).

La méthode décrite dans ce mémoire est du type directe. Elle se base uniquement sur les changements d'intensités des pixels dans le temps (dérivées spatio-temporelles). La méthode présentée repose sur les mêmes hypothèses que celles du flux optique pour modéliser le mouvement et utilise un formalisme probabiliste pour résoudre le problème de l'ego-mouvement. L'usage de ce formalisme confère à la méthode une grande robustesse au bruit, typiquement associé aux conditions de faible éclairage, contrairement aux méthodes purement analytiques qui souffrent d'instabilités numériques dans de pareils cas.

Étant donné que la méthode proposée dans ce mémoire ne repose que sur le calcul des dérivées spatio-temporelles comme source d'informations il n'est donc pas question d'établir une correspondance entre des points saillants d'une scène (qui suppose la présence de points saillants distinguables), d'utiliser une corrélation pour trouver un déplacement 2D (qui ne fonctionne qu'en présence de forte textures) ou d'employer tout autre aspect qui émet des hypothèses sur la nature même de la scène.

Cette indépendance du contenu de la scène est importante car elle permet le traitement de scènes difficiles comme celle illustrée à la figure 1.1. Tout le long de cette séquence, la caméra tourne autour d'un amas de sphères regroupées dans un volume. La difficulté provient du fait que les boules ne possèdent pas de points saillants qui permettent d'effectuer un suivi (tracking). Même les contours des boules ne peuvent faire office de points saillants à cause des occlusions qui surviennent à tout moment de la séquence.

De plus l'absence de texture empêche d'utiliser les techniques de corrélation pour mesurer les déplacements entre les images.

Notre méthode est donc idéale pour ce genre de scènes empiriques où les autres méthodes conventionnelles échouent.

Organisation de la thèse

Ce mémoire se compose de trois volets principaux répartis sur six chapitres. Tout d'abord nous commencerons par introduire les concepts du mouvement perçu (flux optique), du mouvement de caméra (egomotion) ainsi qu'un aperçu de leurs estimation à travers une revue de littérature (chapitres 2, 3, 4).

Les détails des concepts proposés par ce travail seront présentés au chapitre 5 sous la forme d'un article qui a été soumis dans le cadre de la conférence scientifique ICCV'2003.

En plus des résultats présentés au chapitre 5, des résultats additionnels feront l'objet du sixième chapitre.

Finalement, nous conclurons sur le travail présenté au chapitre 7 avec une discussion sur les éventuelles extensions et perspectives de recherches.

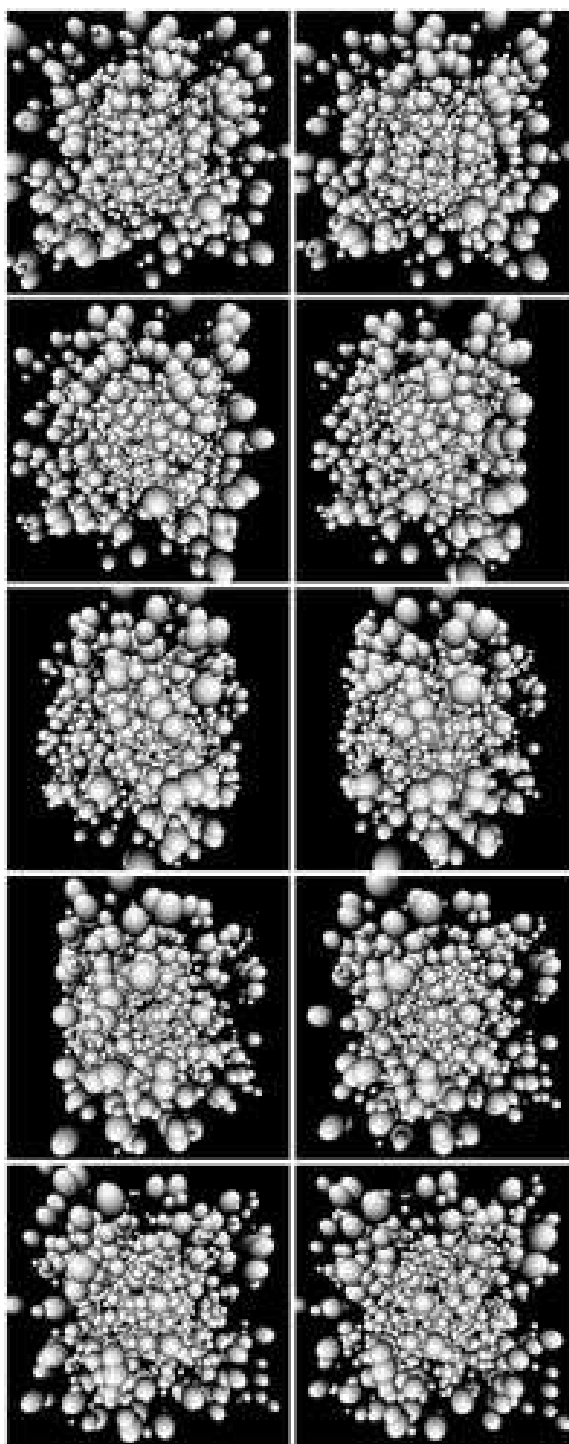


FIG. 1.1. Exemple de mouvement difficile à estimer. Dans cette scène où la caméra tourne autour d'un amas de sphères, il est difficile de retrouver le mouvement de caméra à cause des occlusions et de l'absence de points saillants (*feature points*). Les images évoluent de gauche à droite et de haut en bas.

Chapitre 2

MOUVEMENT ET FLUX OPTIQUE

2.1 Généralité

La recherche en vision par ordinateur vise à faire accomplir aux machines une tâche que notre système visuel accomplit presque sans effort¹. Cette tâche consiste à voir et à comprendre la structure du monde environnant. La vision 3D apportera donc aux machines une «*conscience*» du monde extérieur qui par la suite pourront accomplir des tâches plus «intelligentes».

La vision par ordinateur s'inspirant fortement de la vision humaine, il serait utile, dans un premier temps, de passer en revue quelques mécanismes de notre système visuel.

Un point 3D dans le monde se projette sur la rétine de l'oeil pour former un point rétinale ou point image. L'ensemble des points du monde (ou d'une scène) observés forment une image rétinale qui sera perçue et analysée par notre cortex visuel. Malheureusement une seule image rétinale n'apporte aucune information sur la profondeur des objets dans une scène. Ceci est dû au fait que la projection supprime la composante reliée à la profondeur².

Il faudra au moins deux images, prises de deux points de vue différents afin de reconstruire la profondeur. C'est pour cela que le cortex visuel utilise les informations rétinales issues des deux yeux pour reconstruire la scène en trois dimensions. Ce concept a d'ailleurs été largement exploité par la stéréoscopie, un des domaines phare de la vision 3D.

¹ moyennant quelques millions de neurones!

² Les détails de la projection seront vus dans le prochain chapitre.

Lorsqu'un observateur se déplace, il perçoit la scène à travers son image rétinale. Cette image rétinale subit des changements (ou des déformations) dans le temps qui sont reliés à la structure de la scène et au mouvement suivi par l'observateur.

L'étude de ce mouvement rétinale et de ses effets serait donc susceptible d'apporter une information sur la structure de la scène perçue ainsi que sur le mouvement de l'observateur. Il a même été démontré que le mouvement est l'un des aspects les plus importants de la perception visuelle chez les animaux et que chez certaines espèces, notamment les oiseaux, elle repose presque uniquement sur le déplacement des objets dans une scène. Dans un cadre général, le mouvement rétinale est dû à deux principaux mouvements : le mouvement des objets dans une scène et le mouvement de l'observateur lui-même.

Il est évident que les deux types de mouvements peuvent survenir en même temps et se combiner, voire même s'annuler. Cette situation arrive lorsque, par exemple, un observateur se déplace à la même vitesse (et dans la même direction) qu'un objet mobile. La vitesse de l'observateur combinée à celle de l'objet annule le mouvement rétinale de l'objet par rapport à l'observateur.

Afin de faire le parallèle avec la vision par ordinateur, on peut substituer notre observateur par un robot mobile équipé d'une caméra vidéo. L'image produite par la caméra sera alors substituée à l'image rétinale. Tout comme l'image rétinale réelle, le mouvement perçu sur l'image de la caméra peut être dû au mouvement de la caméra, au mouvement des objets dans la scène ou à une combinaison des deux. Le mouvement de caméra est désigné par *Egomotion* dans la littérature [26] et son estimation constitue le sujet principal de ce mémoire.

À ce stade-ci il est important de faire la distinction entre le mouvement réel des objets et leur mouvement rétinale. Lorsque un observateur immobile regarde un objet mobile, l'image de ce dernier se projette sur la rétine de l'observateur et sa déformation dans le temps crée un mouvement rétinale. Dans ce cas-ci, il est tout

à fait concevable d'équivaloir³ le mouvement de l'objet et son mouvement rétinale. Par contre, au moindre déplacement de l'oeil de l'observateur, l'équivalence entre le mouvement réel de l'objet et le mouvement rétinale n'est plus valide. Par exemple, lorsque un observateur balaye du regard une scène, il se produit un mouvement rétinale (dû au mouvement de l'oeil) même si la scène est complètement statique. A l'opposé, lorsqu'un observateur suit du regard un objet mobile, ce dernier semblera stationnaire car la combinaison du mouvement de l'oeil et celui de l'objet produit un mouvement rétinale nul.

2.2 Le problème d'ouverture

Le déplacement d'un objet donne l'impression d'un mouvement continu et ce grâce à la capacité du cortex visuel à établir une correspondance spatio-temporelle entre les objets dans deux images qui se succèdent dans le temps. Cette correspondance se base sur plusieurs heuristiques telles que : la minimisation des distances, la correspondance des formes ou encore l'appariement chromatique. Quelques unes de ces heuristiques sont illustrées à la figure 2.1.

Ces heuristiques sont robustes et valides pour la plupart des mouvements rencontrés dans la vie courante⁴. Cependant il existe une limitation avec la méthode des correspondances. Connu sous le nom de *problème d'ouverture*, ce problème est très important dans la perception et la compréhension du mouvement et constitue un handicap majeur pour la vision par ordinateur. Pour la vision humaine, la cause de ce problème ne vient pas du fait qu'on perçoive le monde à travers une petite ouverture (comme la désignation laisserait supposer) mais il est dû au fait que les cellules responsables de la perception du mouvement n'agissent que sur de petites portions du champ visuel, comme si elles «regardaient »la scène à travers une petite

³ à une projection près

⁴ à condition que le mouvement ne soit pas trop rapide.

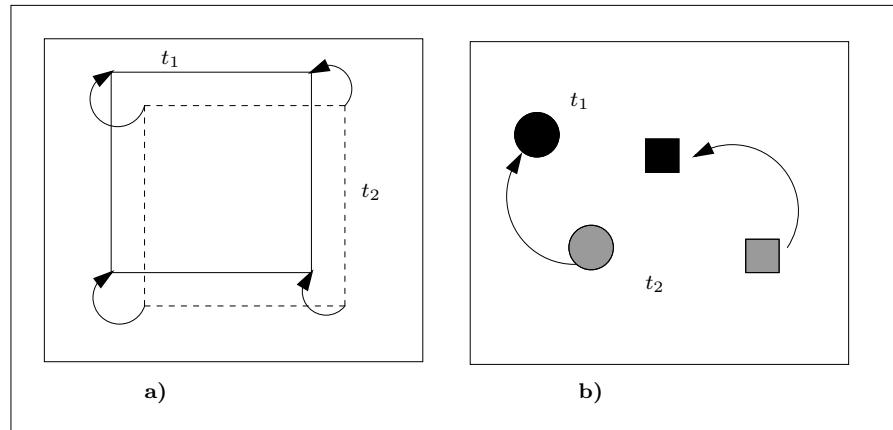


FIG. 2.1. Exemples d'heuristiques de correspondance. a) en minimisant les distances entre les coins du carré au temps t_2 et ceux au temps t_1 , l'heuristique suggère un mouvement en diagonal. b) en présence de plusieurs objets, il est plus facile de déduire le mouvement en faisant un appariement basé sur la forme des objets.

ouverture.

En vision 3D le problème d'ouverture est en fait un problème d'ambiguïté locale dans l'estimation de la direction et de la magnitude du mouvement lorsque un déplacement est perçu à travers une ouverture suffisamment petite pour que toute correspondance deviennent ambiguë tel qu'illustré sur les figures 2.2 et 2.3.

En présence de telles ambiguïtés, on ne peut calculer que la composante du déplacement normale au gradient.

2.3 Flux Optique

Lorsqu'un objet se déplace dans une scène ou qu'un observateur entreprend un mouvement, il induit un champ de mouvement rétinale appelé *Flux Optique*, une désignation due à James J. Gibson [14]. Le flux optique est un champs vectoriel où chaque vecteur est associé à un point de l'image. Quand un point de l'image se déplace dans le temps, son vecteur correspondant dans le flux optique indique la direction et la

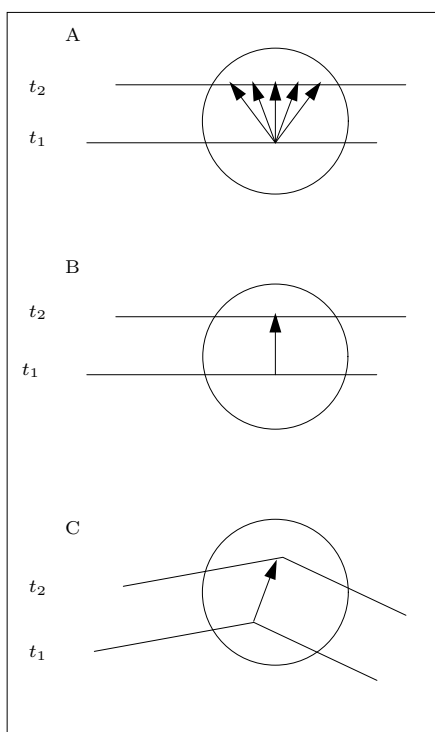


FIG. 2.2. Exemples d'ambiguïtés dûes au problème d'ouverture, le mouvement d'une droite perçue à travers une ouverture est ambigu car tout point au temps t_1 peut être apparié à n'importe quel point au temps t_2 (A). Le seul mouvement perçu est perpendiculaire (B). Lorsque suffisamment de structure est visible (ici un coin) une correspondance unique est établie, il est alors possible de reconnaître le vrai mouvement (C,D).

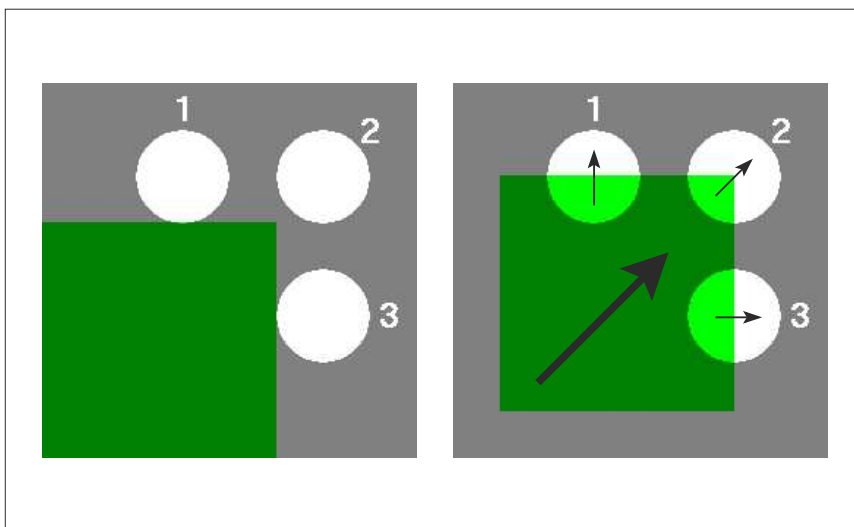


FIG. 2.3. Voici un autre exemple du problème d'ouverture où on voit un carré se déplacer en diagonal vis-à-vis de 3 ouvertures. L'interprétation du mouvement est différente selon l'ouverture considérée.

magnitude du déplacement. La figure 2.4 illustre quelques exemples de mouvements avec leur flux optique associé. Les vecteurs du flux optique peuvent être vus comme étant la projection sur le plan rétinale des vecteurs déplacement 3D de chaque point de la scène.

L'estimation du flux optique est fort utile car en plus d'indiquer le déplacement des objets dans une scène, il informe sur les mouvements globaux ou prépondérants qui sont dus au déplacement de l'observateur (egomotion).

Les déplacements d'un observateur (ou plutôt de la caméra) induisent des flux optiques caractéristiques, . Il est même possible de paramétrer et d'obtenir un modèle de flux optique pour un mouvement d'observateur donné. Ce concept est très utile car réciproquement il est possible de déduire le mouvement entrepris par un observateur ou une caméra en analysant le flux optique global qu'il induit.

Vu l'importance de l'analyse du mouvement, la résolution du flux optique a suscité un vif intérêt auprès des chercheurs en vision donnant lieu à une myriade de méthodes

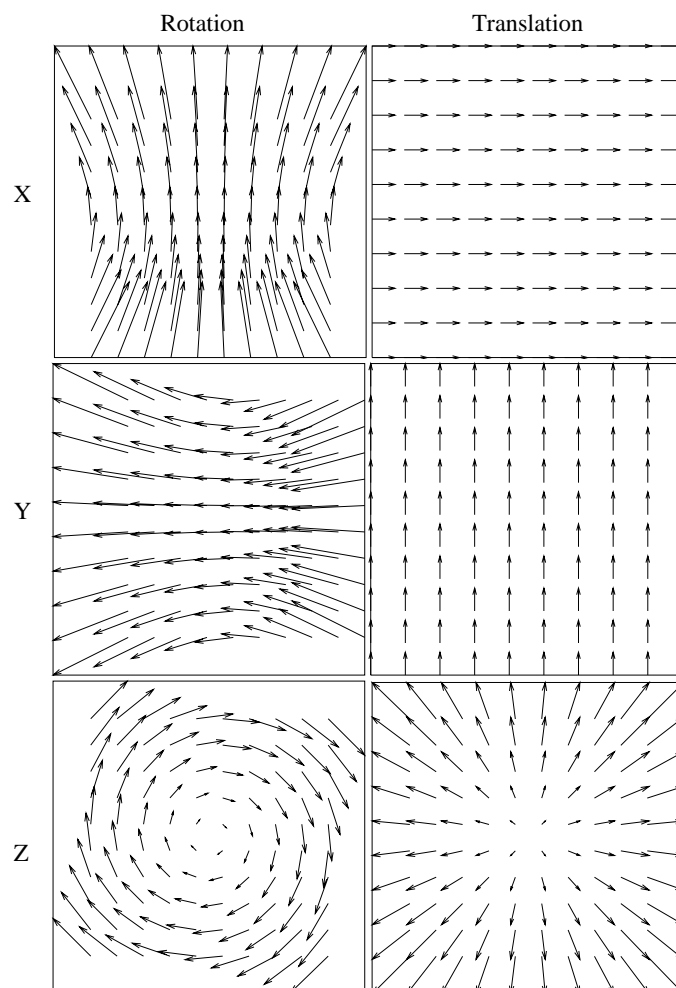


FIG. 2.4. Exemple de flux optiques engendrés par des mouvements de rotations ou de translations selon les trois principaux axes.

de résolution. Bien que ces méthodes de résolution soient très nombreuses, il est possible de les classer en quatre familles principales [6] :

- méthodes différentielles ;
- méthodes basées sur les régions ;
- méthodes basées sur l'énergie ;
- méthodes basées sur les changements de phases ;

Méthodes différentielles

Ces méthodes sont de loin les plus explorées et les plus utilisées. Ces méthodes utilisent les dérivées spatio-temporelles des images afin de calculer les vitesses. En supposant les conditions d'éclairage invariables, des objets opaques et mats, on peut affirmer que les changements d'intensité dans une image ne sont dûs qu'aux mouvements. Ceci implique que l'intensité $I(x, y, t)$ du point $p = [x, y]^T$ au temps t demeure constante même si p subit un déplacement apparent $(\Delta x, \Delta y)$, c.a.d :

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \delta t) \quad (2.1)$$

Étant donné que l'intensité de l'image est conservée dans le temps, $dI(x, y, t)/dt = 0$, l'équation 2.5 devient :

$$\dot{I}_x v_x + \dot{I}_y v_y + \dot{I}_t = 0 \quad (2.2)$$

Le gradient spatio-temporel de l'image $(\dot{I}_x, \dot{I}_y, \dot{I}_t)$ est la dérivée de l'intensité de l'image par rapport à ses 3 composantes : x, y et t . Les composantes de la vitesse du pixel (x, y) sont notées (v_x, v_y) .

L'équation (2.2) définit une droite qui impose une contrainte sur les vitesses permises pour un pixel sachant ses dérivées spatio-temporelles (voir figure 2.3).

Cette contrainte se nomme *contrainte d'intensité constante* (*Constraint Brightness assumption* ou *CBA* [25]).

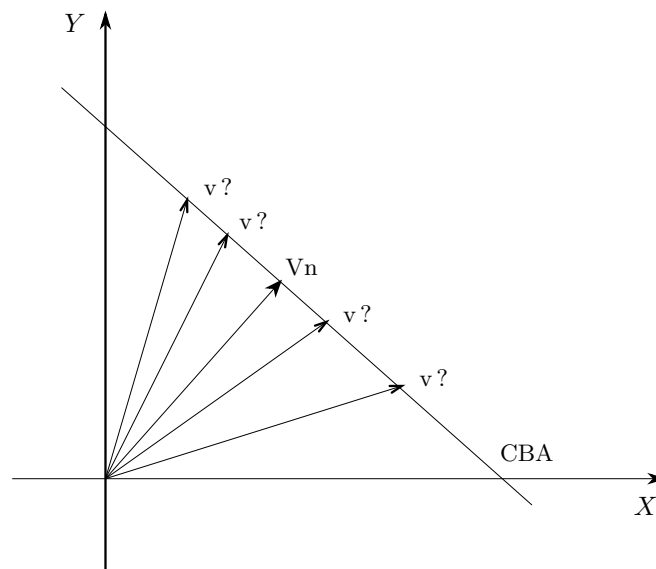


FIG. 2.5. La contrainte de luminance (CBA) est déterminée par les dérivées spatio-temporel et elle contraint les vitesses permises pour un pixel. Le vecteur normal V_n à la droite représente la composante de la vitesse dans la direction du gradient.

Il est évident que l'équation (2.2) ne peut déterminer complètement la vraie vitesse d'un pixel car elle représente en elle-même un système sous-déterminé. Dans la pratique, il est courant d'introduire au moins une contrainte de lissage (contrainte temporelle ou spatiale) à l'équation du flux optique afin de lever l'ambiguïté sur la mesure du mouvement et de résoudre le problème d'ouverture.

On pourrait, par exemple, supposer que les vitesses demeurent les mêmes dans un voisinage de 5×5 pixels tel que proposé par Lucas et Kanade [19]. Sachant que chaque pixel induit une équation similaire à l'équation (2.2), il ne restera plus qu'à trouver la vitesse associée à cette région selon la méthode des moindres carrés.

La taille du voisinage détermine le taux de lissage ou de régularité sur les vitesses. Un grand voisinage (ex : 7×7) permet de contrer les irrégularités dues au bruit tandis qu'un petit voisinage assure une meilleure estimation locale notamment au niveau des discontinuités telles que les contours.

Horn et Schunck (HS) [16] sont les premiers à avoir proposé une formulation qui remplace le problème mal posé par un problème d'optimisation convexe quadratique. Afin de résoudre l'équation du flux optique, HS introduisent une contrainte supplémentaire appelée *Contrainte de lissage*, qui postule que le champ de vitesse recherché est régulier. Cela veut dire que les vitesses ne varient pas beaucoup au sein d'un voisinage Ω . Mathématiquement, cette contrainte revient à minimiser pour toutes les vitesses v dans un voisinage Ω :

$$F_{lissage} = \int \int \left\| \frac{\nabla v}{\nabla(v_x, v_y)} \right\| \quad (2.3)$$

Le terme $\left\| \frac{\nabla v}{\nabla(v_x, v_y)} \right\|$ représente la magnitude du gradient du vecteur vitesse par rapport à ces composantes v_x et v_y pour un pixel du voisinage Ω .

HS introduisent [16] une fonction de coût dont le minimum satisfait à la fois la contrainte de lissage (2.3) et la contrainte de luminosité constante (2.2). Cette fonction se formule comme suit :

$$\int \int (v_x \cdot I_x + v_y \cdot I_y + I_t)^2 + \alpha^2 \left\| \frac{\nabla v}{\nabla(v_x, v_y)} \right\| \quad (2.4)$$

Le facteur α^2 contrôle l'influence du terme de lissage.

Méthodes basées sur les régions

Lorsque les déplacements considérés dans les images sont trop grands, typiquement plus que deux pixels par image, ou que le calcul des dérivées spatio-temporelles est sujet à des erreurs numériques, il convient de se tourner vers des méthodes dites à *régions*. Ces méthodes consistent à apparier (*matching*) chaque région R_i d'une image au temps t à une autre région R'_i au temps $t + 1$ et ce en trouvant le déplacement $d = (d_x, d_y)$ qui minimise une fonction d'erreur et maximise la similarité entre les deux régions. Une telle fonction d'erreur pourrait être une somme de corrélation croisée ou encore une somme des différences au carré (*SSD*) :

$$SSD(d = (d_x, d_y)) = \sum_{k=-n}^n \sum_{l=-n}^n [R_i(k, l) - R'_i(k - d_x, l - d_y)]^2 \quad (2.5)$$

$R_i(k, l)$ et $R'_i(k, l)$ désignent les intensités des pixels aux coordonnées (k, l) dans les régions R_i et R'_i respectivement.

Il est facile de voir que pour une déplacement d donnée, la fonction $SSD(d)$ vaut 0 lorsque les deux régions sont identiques et augmente en fonction des différences entre les régions.

Parmi les méthodes d'estimation du flux optique basées sur les régions on trouve celle de Anandan [22] qui fait partie des classiques.

Anandan utilise un modèle pyramidal hiérarchique afin d'estimer le mouvement. Le modèle pyramidal estime le mouvement à l'aide d'un raffinement successif (coarseto-fine). On commence par analyser la séquence à basse résolution afin de considérer les mouvements de grande magnitude. Après quoi, la résolution est raffinée de plus en plus afin de considérer les mouvements de plus petites magnitudes.

À chaque étape, les images subissent un filtrage gaussien 5×5 afin d'en réduire le bruit, les vitesses (elles-même) sont calculées de façon à minimiser la fonction $SSD(d)$ de l'équation (2.5) sur un voisinage de 3×3 .

En plus de minimiser la fonction $SSD(d)$, la solution doit minimiser un terme de lissage qui rappelle un peu celui employé par Horn et Schunk [16] afin d'imposer une régularité sur le champ de vitesses.

Méthodes basées sur l'énergie

Ces méthodes se basent sur les valeurs d'énergie que produisent certains filtres «orienté vitesse»[12, 5]. Ces méthodes sont aussi appelées *méthodes fréquentielles* car la réponse des filtres se fait dans le domaine fréquentiel.

Méthodes basées sur les changements de phase

La dernière catégorie des méthodes considérées est surtout appliquée dans le calcul des mosaïques. Le principe étant que deux images reliée par une homographie et dont le contenu est identique génèrent la même transformée de Fourier mais avec une phase translatée [17]. La translation de la phase est directement reliée à l'homographie qui relie les deux images.

Afin de calculer le flux optique avec cette méthode, il faudra donc considérer les images deux par deux et calculer le déphasage dans le domaine fréquentiel.

Malheureusement toutes les méthodes qui opèrent dans le domaine fréquentiel présentent une perte de localisation à cause de l'aspect global de la transformation de Fourier. Ces méthodes sont donc plus appropriées pour des mouvements globaux qui remplissent l'image que pour de petits mouvements qui sont très difficile à isoler.

Pour un plus large éventail de méthodes de calcul du flux optique, le lecteur intéressé est convié à lire le comparatif de Barron, Fleet et Beauchemin [6].

2.4 Conclusion

Un aperçu des différentes classes de méthodes concernant l'analyse du mouvement a été présenté.

Les méthodes basées sur la mise en correspondance de primitives (ou points saillants) consistent premièrement à extraire un ensemble de primitives 2D dans l'image (coins, segments de droites, contour, etc...). Ensuite la mise en correspondance temporelle de ces primitives est effectuée. Finalement les paramètres du mouvement sont calculés sous certaines conditions telles que le mouvement rigide. Ces contraintes se traduisent souvent par un système d'équations dont les inconnues sont les paramètres du mouvement.

Les primitives obtenues et leurs correspondances servent à résoudre ce système d'équations.

Les méthodes basées sur le flux optique reposent en premier lieu sur le calcul du flux optique 2D à partir des changements temporels des niveaux de gris dans l'image. Ce calcul est en général mené sur la totalité de l'image et produit un mouvement apparent dense.

Les autres méthodes opèrent dans des domaines fréquentielles. Le plus souvent on recherche des changements de phases ou bien des *pattern* bien définis afin d'estimer les déplacements.

Chaque famille d'approches présente des avantages et des inconvénients et leur choix dépend aussi du contexte et des données.

Si l'on retient le cadre «points saillants», encore faut-il les choisir, les extraire et les mettre en correspondance sur deux, voire même trois images ou plus, pour les suivre. Il a l'avantage pouvoir opérer sur des déplacements importants. Les formulations pour estimer de façon robuste le mouvement sont souvent basées sur la contrainte de rigidité du mouvement et des objets dans la scène.

Les méthodes reposant sur le mouvement apparent (flux optique) permettent, par

contre, d'opérer directement sur les images, mais exigent un mouvement lisse avec de faibles magnitudes. Une cadence élevée d'acquisition est souvent nécessaire. Ce dernier critère ne représente pas vraiment un obstacle car la méthode proposée par ce mémoire repose sur des acquisitions vidéo dont l'acquisition est cadencée à 30 images/sec.

Les méthodes fréquentielles sont attrayantes mais souffrent de précision et de temps de calcul à cause du passage spatial/fréquence.

En dépit de leurs différences, toutes ces méthodes suivent le même schéma de résolution, à savoir :

- Pré-filtrage (ou lissage) afin de réduire le bruit ;
- Extraction des mesures de bas niveau (ex : gradients, contours...);
- Intégration des mesures afin de produire un champ de vitesses. Cette étape implique souvent des techniques de lissage ;

Chapitre 3

CAMÉRAS ET PARAMÈTRES DE CAMÉRAS

En vision par ordinateur, les caméras constituent les principales sources d'images à travers lesquelles une scène est perçue et analysée. Il est donc impératif de définir le modèle de caméra utilisé pour la plus part des applications. Dans ce mémoire le modèle *sténopé* (*pinhole*) sera employé. Ce modèle de caméra perspective est celui qui s'apparente le mieux aux caméras réelles.

3.1 Modèle de camera

Dans sa forme la plus simple, la caméra se compose principalement d'un point focal O et d'un plan focal (ou plan image) sur lequel l'image se forme, tel qu'illustré à la figure 3.1. On appelle *longueur focale* ou simplement *focale*, la distance qui sépare le point focal O du plan focal. Cette distance, notée f , détermine directement l'angle de vue de la caméra : plus grande sera la focale, plus petit sera l'angle de vue (voir l'annexe A.1). Le point focal O représente l'origine du système de la caméra auquel on associe un système d'axe $(\vec{X}, \vec{Y}, \vec{Z})$. La caméra *regarde* dans la direction du vecteur \vec{Z} . Un point $P = [x, y, z]^T$ de la scène se projette sur le plan focal en $p' = (\frac{x'}{z}, \frac{y'}{z})$ exprimé dans le système de l'image défini par (O', \vec{X}', \vec{Y}') . Le point p' peut aussi être vu comme l'intersection de la droite \overline{OP} avec le plan focal.

De façon générale, un point $P = [x, y, z]^T$, exprimé dans le système de coordonnées du monde, est lié à son image $p' = [x', y']^T$ sur le plan focal par la relation suivante :

$$\begin{bmatrix} p' \\ 1 \end{bmatrix} = A \cdot \begin{bmatrix} P \\ 1 \end{bmatrix} \quad (3.1)$$

$$A = K \cdot R \cdot T \quad (3.2)$$

Les matrices R et T représentent la rotation et la translation 3D nécessaire pour aligner le système de coordonnées de la caméra avec celui du monde. Les éléments de la matrice $R \cdot T$ caractérisent donc les paramètres externes de la caméra. La matrice A représente l'opérateur de projection et caractérise les propriétés intrinsèques (ou paramètres internes) de la caméra (distance focale, centre de l'image,...).

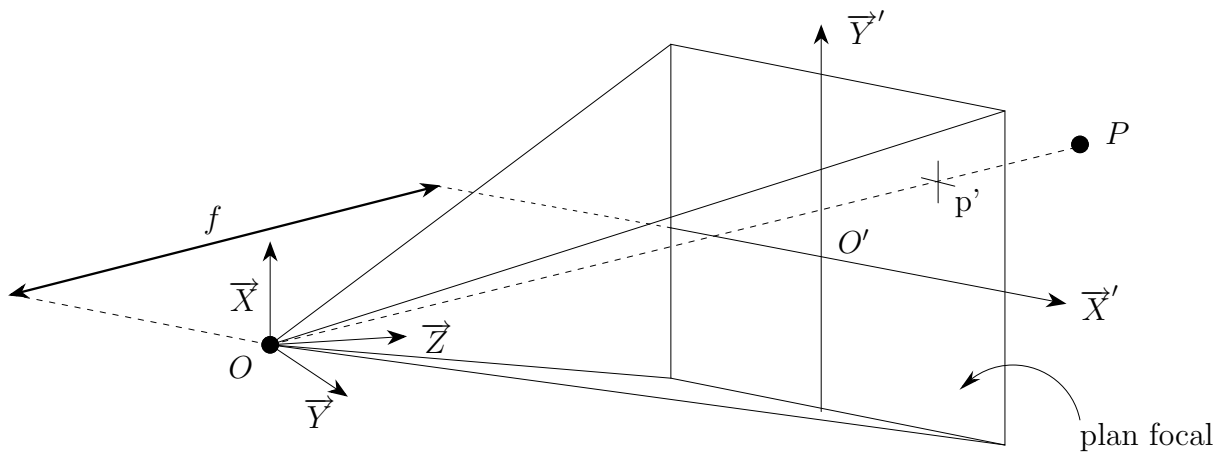


FIG. 3.1. **Modèle de caméra sténopé.** Le point 3D $P = [x, y, z]^T$ exprimé dans le système $(\vec{X}, \vec{Y}, \vec{Z})$ se projette sur le plan focal en $p' = [x', y']^T$ qui est exprimé dans le système de coordonnées de la caméra.

3.2 Paramètres internes

Étant donné que les paramètres internes se rattachent aux propriétés intrinsèques de la caméra, ils sont censés demeurer fixes lors du tournage d'une séquence vidéo, à l'exception des caméras munies de zoom dont la longueur focale peut varier avec l'ajustement du zoom.

L'estimation des paramètres internes peut souvent être faite une fois pour toute en laboratoire à l'aide d'objets de calibration dont la géométrie est connue. On ne s'attardera pas longtemps sur l'estimation des paramètres internes car elle ne constitue pas l'objet de ce mémoire mais surtout parce qu'elle représente à elle seule un vaste

domaine de recherche. Dans ce qui suit, on considérera les paramètres de la caméra connus, ceci permettra une interprétation géométrique valide à l'analyse du mouvement. Dans le cas contraire, il sera toujours possible d'apporter une interprétation qualitative du mouvement global observé.

3.3 Paramètres externes

Les paramètres externes définissent l'orientation de la caméra par rapport au repère du monde et se composent essentiellement de 3 paramètres translationnels (T_x , T_y et T_z) et de 3 rotationnels (ω_x , ω_y et ω_z) exprimés en radians. A l'opposé des paramètres internes, les paramètres externes changent avec les mouvements de caméra et leur estimation constitue le sujet de ce mémoire.

Il est possible de considérer la matrice 3×4 , A , comme matrice de projection au lieu de la séparer en matrice interne et en matrice externe. La relation entre le point $P = [x, y, z]^T$ et son image $p' = [x', y']^T$ s'exprimera par l'équation suivante :

$$\begin{bmatrix} p' \\ 1 \end{bmatrix} = \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \end{pmatrix} \cdot \begin{bmatrix} P \\ 1 \end{bmatrix} \quad (3.3)$$

Contrairement à l'équation (3.1) la matrice A n'est plus vue comme étant le produit de 3 matrices, ses entrées sont indépendantes et n'ont aucune relation entre elles. Pour trouver les 12 paramètres a_{ij} de la matrice A , il faudra disposer de 4 points 3D avec leurs images 2D pour constituer un système à 12 équations.

Le problème avec cette approche est qu'il faudra procéder à la calibration de la caméra à partir d'images dont la structure est inconnue, à l'aide de méthodes dites *d'auto-calibration*. En plus d'être gourmandes en temps de calcul, ces méthodes n'offrent pas autant de robustesse que la calibration en laboratoire.

3.4 Mouvement de caméra

Tout comme le choix de modèle de caméra, un modèle de mouvement doit être adopté pour les besoins de l'estimation du mouvement. Dans ce mémoire on ne considère que les mouvements de caméras instantanés. La vitesse 3D (ou le déplacement) V instantanée d'un point P dans le système de la caméra s'exprime comme suit [18] :

$$V = -T - \omega \times P \quad (3.4)$$

où $T = (T_x, T_y, T_z)$ représente le vecteur translation de la caméra et $\omega = (\omega_x, \omega_y, \omega_z)$, le vecteur rotation dont les composantes déterminent l'axe de la rotation et dont la magnitude représente l'angle de la rotation en radians.

Remarque : étant donné que l'unité de temps est discrète et unitaire, les termes *déplacement* et *vitesse* réfèrent à la même mesure.

La projection de la vitesse V sur le plan image, notée v , représente le déplacement perçu dans l'image et est obtenue en projetant d'abord le point P sur l'image afin d'obtenir le point image $p' = [x', y']^\top$ (dans le système de la caméra) :

$$p' = f \frac{P}{z} \quad (3.5)$$

Par la suite, en dérivant l'équation (3.5) on obtient le déplacement (ou vitesse) perçu du point P au pixel p' [28] :

$$v_x = \frac{T_z x' - T_x f}{z} - \omega_y f + \omega_z y' + \frac{\omega_x x' y'}{f} - \frac{\omega_y x'^2}{f} \quad (3.6)$$

$$v_y = \frac{T_z y' - T_y f}{z} - \omega_x f + \omega_z x' + \frac{\omega_y x' y'}{f} - \frac{\omega_x y'^2}{f} \quad (3.7)$$

Ou encore plus simplement :

$$v = M_\omega \omega + \frac{1}{z} M_T T \quad (3.8)$$

Avec :

$$M_\omega = \begin{pmatrix} x'y' & -(1+x'^2) & y' \\ (1+y'^2) & -x'y' & -x' \end{pmatrix}$$

$$M_T = \begin{pmatrix} -1 & 0 & x' \\ 0 & -1 & y' \end{pmatrix}$$

Nous rappelons que le point $P = [x, y, z]^\top$ se projette dans l'image au point $p' = [x', y']^\top$.

L'équation (3.8) associe à chaque point de l'image $p'_i = [x'_i, y'_i]^\top$ un déplacement, dans l'image, $v_i = [v_x, v_y]^\top$ suite à un mouvement de caméra $M = (\omega, T)$.

En supposant les vitesses v_i calculées et la profondeur des points z connue, une façon d'estimer M serait d'introduire les v_i dans l'équation (3.8) et constituer un système d'équations sur-déterminé qu'on pourrait résoudre au sens des moindres carrés. Un mouvement de caméra étant composé de 6 paramètres (3 translationnels et 3 rotationnels), en théorie 3 pixels suffiraient à la résolution de ce système d'équations. Dans la pratique la présence d'ambiguïté de mouvement, de bruit et d'instabilités numériques font que cette voie ne mène pas toujours à la bonne solution.

L'approche optée dans ce mémoire sera plus globale et moins sensible au bruit. Elle sera introduite dans le chapitre 4 puis détaillée dans le chapitre 5.

Chapitre 4

ESTIMATION DU MOUVEMENT DE CAMÉRA

Devant l'importance de l'estimation du mouvement 3D, la recherche en vision a donné lieu à une multitude de techniques pour estimer les paramètres du mouvement d'une caméra. Ces différentes approches dépendent du type d'images considérées : mouvements lents, mouvements rapides, images stéréo, etc. La composition de la scène joue aussi un rôle important dans le choix d'approches.

Le choix d'une approche se détermine aussi par le type d'acquisition employée. Les méthodes d'acquisitions d'images en vision se divisent en deux familles : *active* et *passive*. La première catégorie interagit avec la scène et se fait le plus souvent par le biais de lumière structurée [15]. Le principal avantage de cette technique réside dans le calcul simple et rapide de la profondeur d'une scène. Étant contrainte à un éclairage contrôlé, la lumière structurée se limite en général aux scènes intérieures et aux petits volumes de reconstruction.

L'acquisition passive se contente d'observer la scène par le biais de caméra vidéo conventionnelles. Elle n'apporte pas autant de facilités que la lumière structurée pour l'estimation de la profondeur, mais son champ d'action est plus large, tel que la navigation d'automobile.

Autant de combinaisons entre les techniques d'estimation de mouvement et d'acquisition ont donné lieu à une multitude d'approches pour résoudre le problème du *egomotion*. Ceci dit, il est possible de classer chacune de ses approches en deux catégories principales distinctes : directes et indirectes.

4.1 Méthodes indirectes

Les méthodes indirectes d'estimation du mouvement ne peuvent estimer les paramètres de caméra à partir des mesures sur les intensités des pixels car elles requièrent, des données supplémentaires qui ne sont pas immédiatement disponibles dans les images. Elles font donc appel à une étape intermédiaire ou à un pré-calcul qui lui aboutira à l'estimation des paramètres externes de la caméra. Cette étape intermédiaire peut être le calcul du flux optique, la détection de contours, la segmentation des images, etc.

4.1.1 Méthodes basées sur les points d'intérêts

Ces méthodes se basent principalement sur l'observation d'un petit nombre de points saillants (*image features*) dans une image et d'en faire un suivi le long de la séquence vidéo. Les points saillants qui sont le plus souvent considérés dans une image sont les coins, les droites, les régions, etc. La figure 4.1 montre les principales étapes d'estimation du mouvement de caméra basée sur les points saillants.

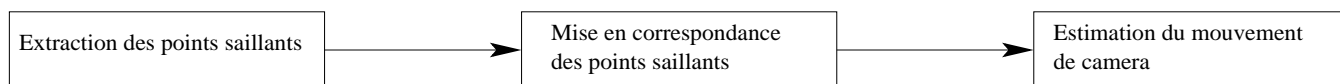


FIG. 4.1. Approche basée sur les points saillants.

À partir de deux images de la même scène prises à des instants différents, cette approche consiste à trouver des points saillants dans la première image et pour chaque point considéré, l'algorithme devra trouver une correspondance dans la seconde image de sorte que les deux points représentent la projection 2D du même point 3D dans la scène tel qu'illustré à la figure 4.1.1. Cette mise en correspondance requiert une très grande précision à cause de l'accumulation des erreurs. Le problème devient plus

ardu lorsque on considère des correspondances sur trois images ou plus. Ceci est dû en partie au fait que certains points saillants peuvent disparaître durant la séquence à cause des occlusions.

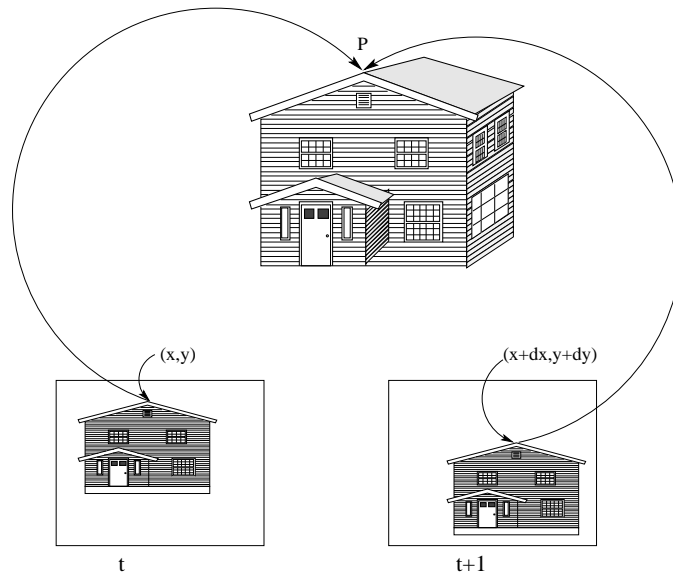


FIG. 4.2. **Correspondance des points saillants.** Pour mettre en correspondance le point (x, y) dans l'image t il faudra trouver le point $(x + d_x, y + d_y)$ dans l'image $t + 1$ de sorte à ce que ces deux points soient la projection du même point 3D P .

Estimation du mouvement pour la navigation passive

Dans [9], Bouguet et Perona proposent une méthode d'estimation de mouvement combinée à une estimation de la structure de la scène destinée à la navigation autonome de robots mobiles.

Cette méthode se compose essentiellement de 4 étapes :

- **Extraction et suivi des points saillants** : Dans cette étape des points saillants de l'image sont extraits et un suivi leur est appliqué. Le suivi des points d'intérêts repose sur la méthode proposée dans [20]. Le résultat de cette étape est une carte du flux observé dans l'image.

- **Première Estimation des paramètres :** Cette première estimation est faite à partir du flux de l'image calculé à l'image précédente. À cette étape-ci une ambiguïté apparaît entre la magnitude de la translation et la profondeur de la scène.
- **Estimation de la profondeur de la scène :**
- **Raffinement de l'estimation des paramètres :**

Les deux dernières étapes sont résolues par un filtrage de Kalman. Les auteurs ne fournissent pas de détails sur la résolution de l'un des principaux paramètres de l'ego-mouvement : l'estimation de la profondeur. Il est mentionné dans [9] qu'une première estimation de la profondeur est effectuée par triangulation (stéréoscopie) et que cette estimation est mise à jour à l'aide d'un filtre de Kalman.

Les filtres de Kalman apportent une mise à jour très satisfaisante dans l'estimation des paramètres à condition que ces derniers n'évoluent pas abruptement dans le temps. Cette supposition demeure valide dans le cas où la caméra ne se déplace pas à grande vitesse et si la scène demeure statique. Dans le cas où la scène comporte des objets mobiles cette méthode d'estimation pourrait échouer pour peu qu'un point d'intérêt (ou plusieurs) tombe sur un objet mobile. Dans ce cas-ci, la vitesse mesurée en ce point pourrait donner une fausse estimation du mouvement de caméra.

Un autre problème potentiel de cette méthode est la disparition des points d'intérêt dans la scène. Les auteurs ne traitent pas le cas où des points d'intérêt, qui évoluent dans le temps, disparaissent de la scène (donc de l'image). Dans de pareils cas, la méthode proposée ne mentionne pas si une nouvelle sélection de points saillants doit être faite.

faudra-t-il choisir de nouveaux points dans le cas où un autre choix est effectué
faudra-t-il refaire une estimation initiale de la profondeur.

Toutes ces lacunes font que l'estimation du mouvement de caméra par suivi de points saillants n'est pas vraiment adaptée à des scènes génériques car elles supposent la présence de textures assez riches pour permettre l'extraire des points saillants.

L'insertion d'une étape intermédiaire, dans ce cas-ci l'estimation du mouvement 2D, ne favorise pas non plus le temps de calcul.

Méthodes basées sur le flux optique

Les méthodes basées sur le flux optique procèdent toujours à la résolution du flux optique avant d'estimer le mouvement de caméra. Les déplacements ainsi obtenus par le biais du flux optique permettront d'établir des correspondances comme pour les méthodes basées sur les point saillants.

L'avantage (et l'inconvénient aussi) d'établir des correspondances par le flux optique est qu'il est possible de considérer de petits mouvements et même des déplacements au sous-pixel. Cette vertu du flux optique peut être exploitée afin de réduire les temps de calcul. Imaginons des images vidéo sources de taille standard, 720×480 pixels. Il est possible de réduire les dimensions au cinquième de la taille originale, par exemple, d'estimer les déplacements au sous-pixel près. Ces vitesses sont maintenant cinq fois plus petites et il faudra alors les multiplier par cinq pour avoir la magnitude originale.

Les limitations de cette approche sont liées aux limitations du calcul du flux optique (conditions d'éclairage, problème d'ouverture...) qui non seulement prennent du temps de calcul, mais représentent aussi une source potentielle d'erreurs cumulatives, nuisant ainsi à la qualité et à l'acuité de l'estimation du mouvement de caméra.

4.2 Méthodes directes

De façon générale, les méthodes d'estimation du mouvement de caméra dites directes ne requièrent pas de calculs intermédiaires ou de résolution de sous-problèmes pour estimer le mouvement de caméra. Elles exploitent directement les propriétés de bas niveau des éléments de l'image. Ces propriétés peuvent être par exemple les intensités des pixels ou bien les dérivées spatio-temporelles.

Estimation qualitative du mouvement

Dans [13], Fermuler et Aloimonos proposent de résoudre le problème du mouvement de caméra de façon qualitative. En partant du fait que chaque mouvement de caméra induit un flux de vitesses caractéristique, celui-ci étant vu comme une signature (voir fig 2.4), cette méthode tente de trouver les paramètres du mouvement en résolvant le problème inverse.

En analysant la disposition des vecteurs vitesses d'une image, les auteurs proposent de faire une recherche dans l'espace des mouvements de caméra afin de trouver les meilleurs paramètres qui justifient le champ de vitesse observé. Fermuler et Aloimonos exploitent plutôt l'influence du mouvement 3D sur le flux normal et non pas sur le flux optique complet. L'avantage de cette approche est que le flux normal peut être calculé sans ambiguïté car il est entièrement déterminé par les dérivées spatio-temporelles, contrairement au calcul du flux optique qui souffre du problème d'ouverture et requiert donc l'utilisation d'une contrainte supplémentaire. Après avoir calculé le flux normal, cette méthode trouve la direction de déplacement de la caméra (heading direction) ainsi que le centre de rotation par simple recherche spatiale. Ensuite, l'espace des mouvements de caméra est réduit en éliminant les mouvements impossibles et ce à travers un système de vote.

Le vote est effectué en plusieurs étapes classées par ordre croissant de restriction. D'une étape à une autre, des paramètres sont estimés grossièrement et leur contribution est éliminée du flux normal observé, le flux résiduel est par la suite injecté à l'étape suivante pour un raffinement successif.

L'idée de considérer les aspects qualitatifs du mouvement confère à cette méthode une indéniable robustesse. Par contre elle dépend de l'estimation de la direction du déplacement ainsi que de l'axe de rotation. La recherche spatiale requise pour les identifier, en plus de prendre du temps de calcul, est une source d'erreurs et d'imprécisions.

Les auteurs ne décrivent pas une méthode pour estimer la direction de déplacement

de la caméra. Elle est simplement définie comme un point dans l'image où le flux est nul (*focus of expansion*). Malheureusement, ce point n'est à l'intérieur de l'image que si la caméra avance dans la direction de son axe optique.

4.3 Méthodes probabilistes

La vision probabiliste

Avant d'entamer ce volet concernant l'estimation de mouvement probabiliste, il convient de lever le voile sur l'emploi et l'importance des probabilités en vision. L'idée principale derrière la vision probabiliste consiste à modéliser et à représenter explicitement l'incertitude dans l'estimation des paramètres au lieu de leur attribuer une valeur fixe.

Une incertitude est en réalité un intervalle d'erreurs. Cet intervalle qu'on suppose régi par une distribution de probabilité. A titre d'exemple, si une caméra fournit des intensités de pixels avec une erreur de 1% distribuée uniformément, il convient de considérer les intensités suivantes (200,150,70) avec leurs intervalles d'erreurs c.a.d : $(200 \pm 2, 150 \pm 1.5, 70 \pm 0.7)$.

Ajouter un intervalle d'erreurs sur des données indépendantes n'est pas d'une grande utilité. Par contre la moindre combinaison de données par le biais d'opérateurs (ex : gradient, filtre gaussien, etc...) complexifie la représentation de l'incertitude. Prenons l'exemple d'une simple division telle que $\frac{6}{3}$. Le résultat de cette opération est irréfutablement 2.

Mais qu'en serait-il si on avait omis de préciser que les opérands provenaient d'une mesure expérimentale avec une incertitude de ± 1 distribuée uniformément ? Il serait alors plus juste de considérer l'opération suivante $\frac{6 \pm 1}{3 \pm 1}$ au lieu de $\frac{6}{3}$.

Étant donné que les données de cette division suivent une loi de probabilité uniforme, on s'attend à ce que le résultat de la division suive une distribution de probabilité aussi. La figure 4.3 illustre la distribution suivie par ce résultat lorsque les

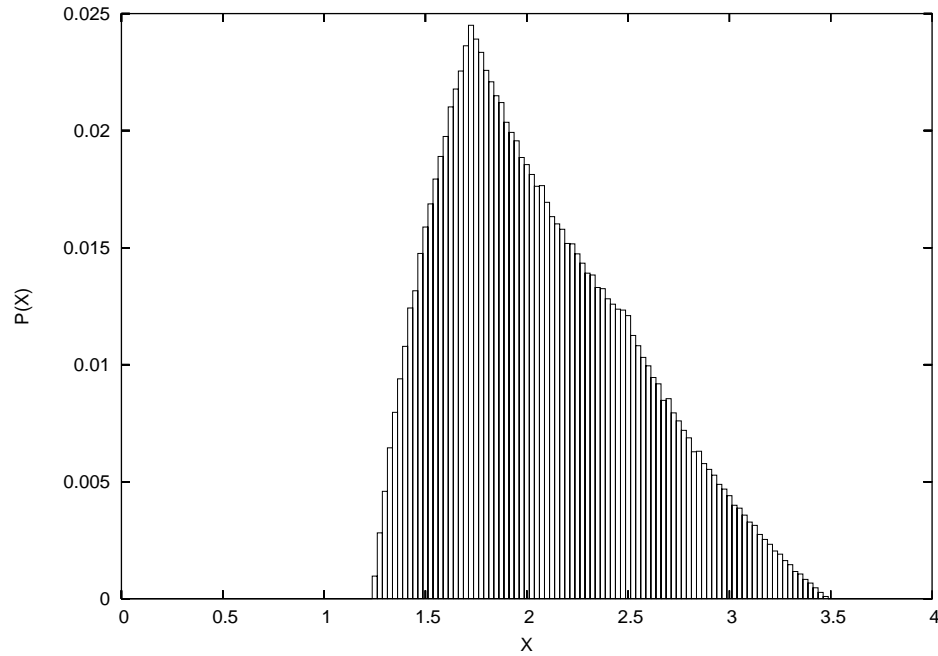


FIG. 4.3. distribution statistique de $\frac{6\pm 1}{3\pm 1}$

opérandes suivent une loi uniforme.

Le résultat d'une telle division est d'autant plus surprenant lorsque l'on considère un cas indéterminé tel que $\frac{0}{0}$ (voir Fig 4.4).

Si on voulait le premier résultat à la puissance du deuxième ($\frac{6\pm 1}{3\pm 1} \frac{0\pm 1}{0\pm 1}$) on peut considérer le résultat le plus probable c.a.d. 1.75 et on obtiendrait :

$$\frac{6 \pm 1}{3 \pm 1} \frac{0\pm 1}{0\pm 1} = 1.75^2 = 3.06 \quad (4.1)$$

En faisant ce choix on fausse la valeur de cette dernière opération car en réalité nous combinons deux distributions statistiques et non pas deux valeurs fixes. La figure 4.5 illustre la distribution statistique de ce dernier résultat.

Le principe qui découle de ces observations est qu'il faut retarder la prise de décisions ou le choix d'une solution intermédiaire le plus possible car on court le risque de faire un mauvais choix. Au contraire, on préférera garder toutes les possibilités afin

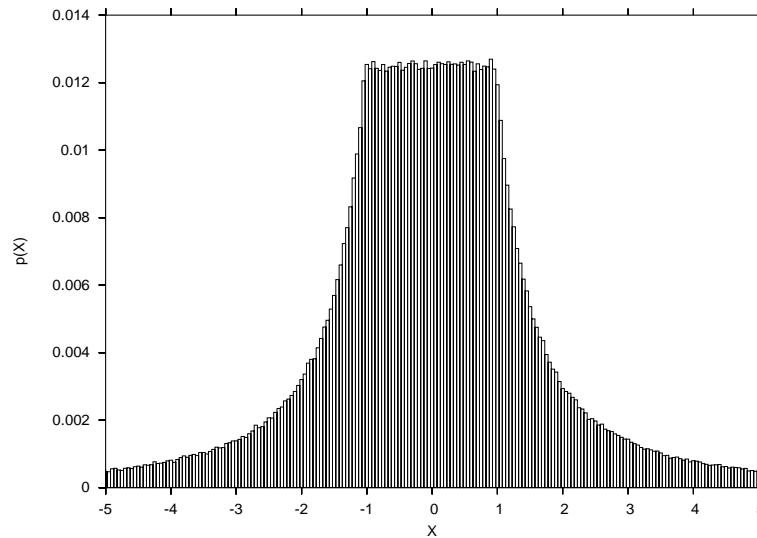


FIG. 4.4. Distribution statistique de $\frac{0 \pm 1}{0 \pm 1}$.

de les combiner entre elles et ne prendre de décisions qu'à la fin pour ne retenir que le résultat le plus probable.

Ce principe est à la base de notre estimation du mouvement de caméra.

L'estimation probabiliste du mouvement de caméra

Nous proposons une nouvelle méthode pour estimer le mouvement de caméra à partir d'une séquence d'images. Cette méthode est directe car elle n'utilise que les dérivées spatio-temporelles des images qui peuvent être calculées à partir des intensités.

Lorsqu'une caméra entreprend un mouvement, le changement d'intensité observé au niveau des pixels de l'image induit un gradient spatio-temporel. Nous avons vu au chapitre 2 que chaque gradient associé à un pixel p' contraint le vecteur vitesse (ou déplacement) de p' à intersecter une droite dans l'espace des vitesses. Cette contrainte, présentée au chapitre 2, est appelée contrainte de luminance constante (constraint brightness assumption).

Tout vecteur vitesse qui intersecte la droite de contrainte du pixel p' est un vecteur

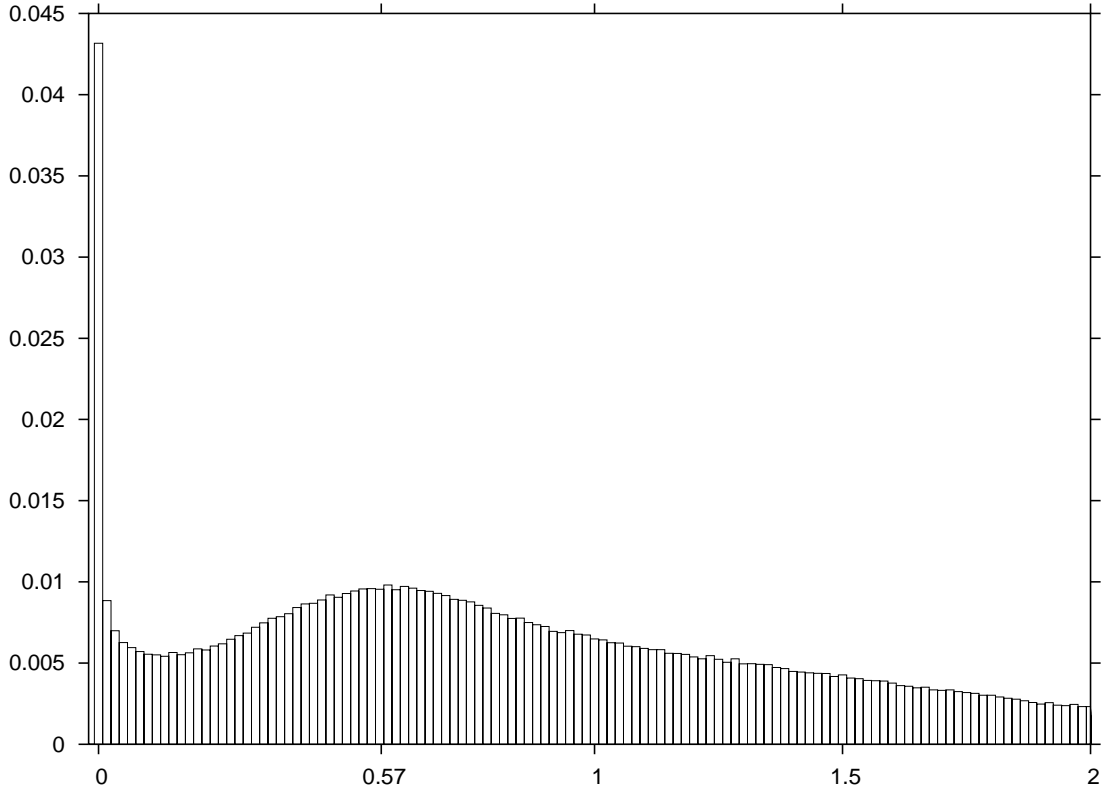


FIG. 4.5. Distribution statistique de $\frac{6 \pm 1}{3 \pm 1}^{\frac{0 \pm 1}{0 \pm 1}}$. Notez que 0 est le résultat le plus probable de cette opération.

vitesse valide pour p' (voir figure 4.3). Sans aucune autre contrainte de lissage ou information additionnelle sur le mouvement, il n'est pas possible de déterminer la vraie vitesse du pixel.

On supposera pour le moment que la vraie vitesse $v' = [v'_x, v'_y]^\top$ du pixel p' est connue.

La relation entre le vecteur vitesse v' observé en p' et le mouvement 3D de caméra $V = [\omega_x, \omega_y, \omega_z, T_x, T_y, T_z]^\top$ est expliquée par l'équation (3.9) et qu'on peut récrire comme suit :

$$v'_x = \omega_x(x'y') + \omega_y(-1 - x'^2) + \omega_z y' + T_x\left(\frac{-1}{z}\right) + T_y 0 + T_z\left(\frac{x'}{z}\right)$$

(4.2)

$$v'_y = \omega_x(1 + y'^2) + \omega_y(-x'y') + \omega_z(-x') + T_x 0 + T_y\left(\frac{-1}{z}\right) + T_z\left(\frac{y'}{z}\right)$$

ou encore plus simplement :

$$v'_x = N_1 \cdot (\omega_x, \omega_y, \omega_z, T_x, T_y, T_z)^\top$$

(4.3)

$$v'_y = N_2 \cdot (\omega_x, \omega_y, \omega_z, T_x, T_y, T_z)^\top$$

avec :

$$N1 = \left(x'y', -(1 + x'^2), y', \frac{-1}{z}, 0, \frac{x'}{z}\right)$$

$$N2 = \left((1 + y'^2), -x'y', -x', 0, \frac{-1}{z}, \frac{y'}{z}\right)$$

Dans l'espace 6D des mouvements de caméra (3 rotations et 3 translations), N_1 et N_2 peuvent être vus comme étant 2 normales 6D qui définissent un hyperplan 4D, H^4 . Cet hyperplan représente l'espace des solutions du système d'équation (4.3). Tout mouvement de caméra (représenté par un point 6D) qui appartient à H^4 est un mouvement valide pour le pixel p' car il vérifie l'équation (4.3) et donc ce mouvement est aussi consistant avec la contrainte de luminance imposée par le pixel p' .

Si chaque pixel p'_i définit un hyperplan H_i^4 , le vrai mouvement de caméra recherché doit être sur tous ces hyperplans H_i^4 . En théorie cela reviendrait à estimer l'intersection des hyperplans 4D, H_i^4 . Dans la pratique, le bruit fait de sorte qu'une unique intersection généralement n'existe pas. Le critère d'intersection doit donc être remplacé par une mesure plus flexible telle que la somme des distances aux hyper-plans. On définira donc le vrai mouvement de caméra $V = (\omega_x, \omega_y, \omega_z, T_x, T_y, T_z)^\top$ non pas comme le point d'intersection des hyperplans H_i^4 mais comme le point qui minimise la somme des distances par rapport aux H_i^4 :

$$V \in \mathfrak{R}^6, \min \sum_i d(H_i^4, V)$$

La fonction d détermine la distance entre un hyperplan et un point 6D.

Étant donné que la vraie vitesse 2D du pixel p' n'est pas connue, il faudra considérer tous les hyper-plans 4D issus de toutes les vitesses 2D possibles le long de la CBA. Deux choix se présentent afin d'accomplir une pareille intégration. Le premier étant de choisir au hasard un ensemble de vitesses le long de la CBA (uniformément par exemple) que définit le pixel p' . L'inconvénient de cette approche est qu'il faudra procéder à un grand nombre d'échantillonnages afin d'avoir un éventail complet de vitesses. Cette opération peut s'avérer coûteuse en terme de temps de calcul ainsi qu'en espace mémoire pour stocker les hyperplans 4D.

La deuxième approche consiste à intégrer l'hyperplan 4D, défini par la vitesse , sur toutes les vitesses possibles permises par la CBA. Cela revient à faire *glisser* cet hyper-plan le long de la droite CBA. Cette intégration donne naissance à un hyperplan 5D qui englobe chaque hyper-plan 4D associé à chacune des vitesses 2D le long de la CBA.

En d'autres termes, l'hyperplan 5D contient tous les points 6D qui représentent des mouvements de caméra valides et ce en considérant toutes les vitesses 2D consistantes avec le flux normal qu'on pourrait associé au pixel p .

Les détails de la génération de l'hyperplan 5D seront vus au prochain chapitre.

Remarque : Un hyperplan 4D définit les mouvements de caméra valides pour une seule vitesse 2D donnée alors qu'un hyper plan 5D englobe tous les mouvements de caméra valides pour toutes les vitesses que pourrait prendre un pixel le long de la CBA.

Étant donné que le vrai mouvement de caméra doit satisfaire à toutes les contraintes de luminance imposées par tous les pixels observés, il doit être sur tous les hyper-plans 5D engendrés par les gradients spatio-temporels. Cela signifie que la solution au

mouvement de caméra s’interprète géométriquement comme étant le point d’intersection de tous les hyperplans 5D. Dans la pratique, la présence du bruit fait qu’un tel point d’intersection n’existe pas. On définira donc la solution optimale comme étant le point 6D V tel que V minimise la somme des distances avec les hyperplans 5D H_i^5 :

$$V \in \mathbb{R}^6, \min \sum_i |H_i^5 \cdot V|$$

Le point V peut être calculé aisément par une minimisation numérique du genre descente de gradient ou plus directement par moindres carrés. Pour notre implantation, nous avons opté pour la méthode de Powell pour sa simplicité et sa rapidité, voir l’annexe A.2 pour plus de détails sur la minimisation.

Notre méthode d’estimation du mouvement de caméra est résumée à la figure 4.6.

Le principe de retarder la prise de décisions évoqué à la section précédente est formalisé à travers l’usage des hyperplans 5D. On aurait pu, dès le départ essayer de calculer une seule vitesse 2D par pixel le long de la droite CBA, ce qui aurait mené à une seule solution d’ego-mouvement par pixel mais au risque d’exclure des solutions potentielles, on a préféré considérer toutes les vitesses 2D qu’un pixel pourrait avoir ce qui a donné suite à un ensemble de mouvements de caméra possibles. Chaque ensemble de mouvements est caractérisé par un hyperplan 5D. On peut donc considérer cet ensemble d’hyperplan comme une sorte de «distribution» du mouvement de caméra.

Après avoir défini et modélisé les vitesses 2D, un paramètre crucial à l’estimation du mouvement doit être modélisé : la structure de la scène. On entend par structure de la scène, la profondeur z de chaque point de l’image dans le monde 3D. Cette composante est perdue après projection et il est en général très difficile de l’estimer de façon précise.

Pour estimer la vraie profondeur de la scène il faut se tourner vers des méthodes de reconstruction telles que *structure from motion* [7] ou *shape from shading* [27, 28] pour ne citer que ces deux là.

Ces méthodes représentent à elles seules un vaste domaine de recherche, elles ne seront pas considérées dans le cadre de notre estimation à cause des contraintes supplémentaires qu'elles imposent.

La stéréoscopie pourrait être vue comme une alternative aux méthodes de reconstruction ci-haut mentionnées. Cependant, l'usage de la stéréoscopie est peu commode dans notre cas car elle s'applique mieux sur des séquences binoculaires où on dénote de larges disparités alors qu'une séquence vidéo monoculaire comporte des déplacements de l'ordre du sous-pixel.

À défaut de considérer une reconstruction 3D complète qui nuirait à la rapidité de l'estimation du mouvement ainsi qu'à sa portée, nous présenterons une modélisation de la profondeur de la scène qui permettra une estimation complète du mouvement rotationnel et une estimation à un facteur d'échelle près de la composante translationnelle du mouvement sans exiger la connaissance exacte de la profondeur de chaque point.

Modélisation de la structure

Comme il en a été fait mention dans la section précédente, la structure (ou la profondeur) de la scène est un paramètre important dans l'estimation du mouvement de caméra. Son apparition explicite dans l'équation (3.8) en est la preuve.

Dans la pratique, à moins d'avoir recours à une méthode de reconstruction, la structure de la scène n'est pas disponible. Cependant, nous avons remplacé la profondeur z_i de chaque point p_i de la scène par la profondeur moyenne z_{moy} définie comme :

$$z_{moy} = \frac{1}{n} \sum_{i=0}^n z_i, \quad \text{avec } z_i \in [z_{min}, z_{max}]$$

z_{min} et z_{max} représentent respectivement la plus petite et la plus grande profondeur observées dans la scène.

Nous avons observé, sur nos données synthétiques, que les résultats obtenus en employant les vrais profondeurs des points étaient identiques à ceux obtenus en utilisant la profondeur moyenne de la scène. Et de plus, si la profondeur moyenne z_{moy} est remplacée par $z_{moy'}$, tel que $z_{moy'} = \alpha z_{moy}$, la magnitude de la composante translationnelle estimée est multipliée par ce même facteur α . Les résultats expérimentaux seront présentés au prochain chapitre.

Les premières conclusions qu'on peut tirer de ces résultats est que la rotation n'est nullement affectée par la magnitude de la profondeur moyenne de la scène. De plus, la variation de la profondeur moyenne affecte proportionnellement la translation : si la profondeur moyenne est sur-estimée, la magnitude de la translation sera aussi sur-estimée afin d'expliquer le mouvement perçu et vice-versa. La profondeur n'a donc pas autant d'impact qu'il n'en paraît à première vue dans la détermination du mouvement de caméra.

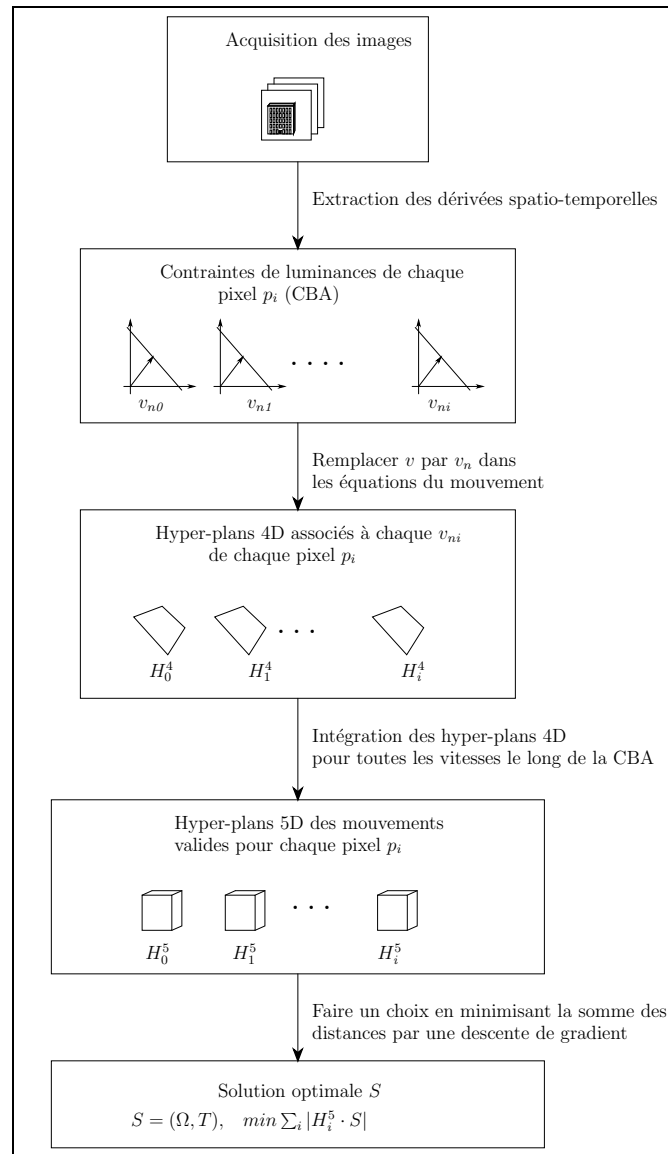


FIG. 4.6. Schéma de l'estimation probabiliste du mouvement de caméra.

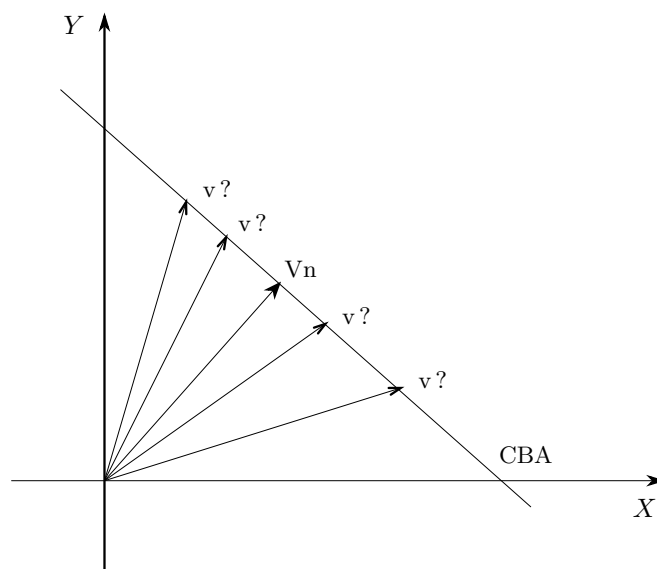


FIG. 4.7. La contrainte de luminance (CBA) est déterminée par les dérivées spatio-temporel et elle contraint les vitesses permises pour un pixel. Le vecteur normal V_n à la droite représente la composante de la vitesse dans la direction du gradient.

Chapitre 5

(ARTICLE) FAST PROBABILISTIC ESTIMATION OF EGOMOTION FROM IMAGE INTENSITIES

Cet article, publié par Sébastien Roy et Jamil Draréni, a été soumis dans le cadre de la conférence scientifique *International Conference on Computer Vision 2003* qui se tiendra à Beijing en Chine.

Il est présenté ici dans sa version originale.

Abstract

This paper proposes a real-time probabilistic solution to the problem of camera motion estimation in a video sequence. Instead of using explicit tracking of features, it only uses instantaneous image intensity variations without prior estimation of optical flow. We represent the camera motion as a probability density which is constructed from the individual motion densities, estimated from spatio-temporal derivatives, of each pixel of the image. The density is formed by accumulating the contribution of each pixel, making it very robust to local perturbations in the image. A fast algorithm is proposed and experimental results show how real-time motion estimation is possible directly from the image stream with good precision.

5.1 Introduction

Determining the egomotion of a moving camera relative to its environment from a sequence of images is very useful in passive navigation and in video coding. The computation of the egomotion can be handled by two different approaches.

The first approach starts by performing features correspondance or optical flow

computation to obtain the image velocities. After that, the equations relating these image velocities to the 3D motion parameters are solved yielding to an estimation of the camera motion parameters [11, 10]. Such methods can be time-consuming and unstable because they rely on results of ill-posed problems such as the estimation of the optical flow and the features correspondance problems [3].

The second approach consists of computing the motion parameters from the normal flow which is fully determined by the spatio-temporel derivatives of the image sequence. Because this approach only relies on image properties, it is called a “direct method” [13].

As most motion algorithms are meant for passive navigation, they usually make several assumptions on the scene structure (ex : vehicle on a planar road [24]) or on the motion itself (ex : constrained robotic motion [4]).

In this paper we propose a method to estimate the unconstrained motion parameters of a camera directly from normal flow without making any assumption on the scene geometry. Typical optical flow and associated normal flow are illustrated in Figure 5.1.

We propose to define a probability distribution for the instantaneous camera motion (egomotion) based on the estimation of normal flow [6]. The camera motion, represented by a rotation vector Ω and a translation vector T , is related directly to the spatio-temporal derivatives \dot{I} at an image point p obtained from the image sequence. Therefore, we must estimate the density

$$p(\Omega; T | \dot{I})$$

First, we define the camera motion equations and propose a transformation to use normal flow directly instead of optical flow. Second, we define a probabilistic model to relate spatio-temporal derivative and pixel motion estimation in the context of optical flow. After that, the probabilistic framework for camera motion estimation will be described, followed by experimental results and discussion.

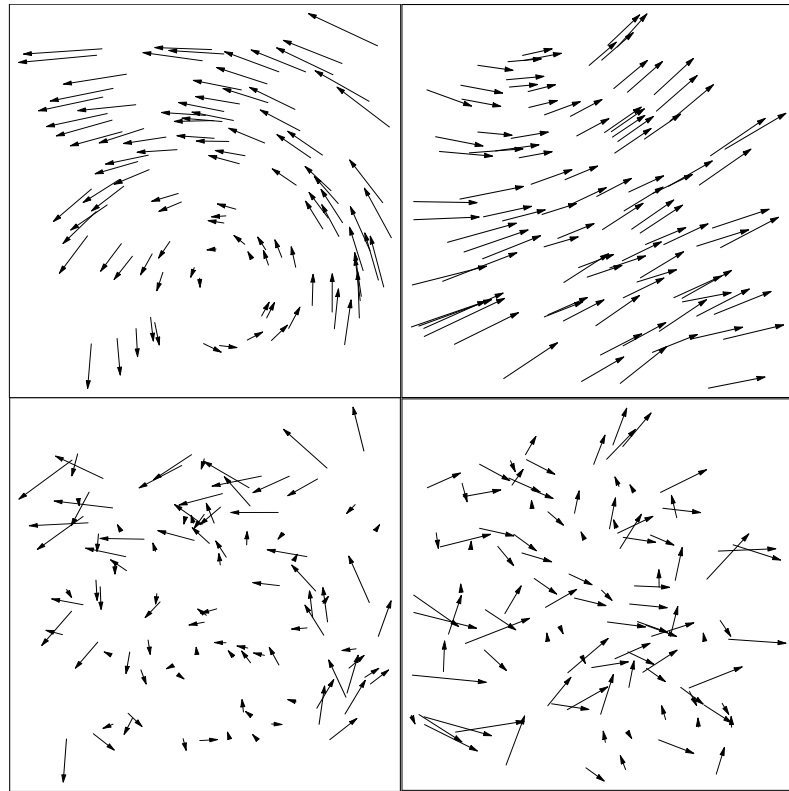


FIG. 5.1. Optical Flow. At top, full optical flow induced by typical camera motions. At bottom, typical normal flows observed for the same camera motions.

5.2 Camera Motion Equations

Before proceeding further, the relation between the camera motion and the observed pixel motion must be described.

The 3D motion V of a 3D point P (expressed in the camera coordinate system) under a camera motion (Ω, T) is

$$V = -T - \Omega \times P \quad (5.1)$$

where T is the camera translation and Ω is the rotation represented as a vector whose direction is the axis of rotation and the norm is the magnitude of the rotation [28].

The corresponding image motion v is obtained by first projecting point P onto the image to obtain the image point p (in camera coordinates)

$$p = \frac{P}{P_z} \quad (5.2)$$

where P_z is the z coordinate of point P . Deriving equation (5.2), we obtain the image motion

$$v = p' = \left(\frac{P}{P_z}\right)' = \frac{P_z P' - P P'_z}{P_z^2} = \frac{P_z V - P V_z}{P_z^2}$$

After replacing V from equation (5.1) we obtain the motion field v induced by the camera motion (Ω, T)

$$v = \mathbf{M}_\Omega \Omega + \frac{1}{P_z} \mathbf{M}_T T \quad (5.3)$$

where which features a translational part \mathbf{M}_T that depends on reverse scene structure $(\frac{1}{P_z})$ and a rotation part \mathbf{M}_Ω that is invariant to scene structure.

5.2.1 Getting rid of depth

In Eq. (5.3), if the depth P_z of each point is known then it is easy to build a linear equation system using a minimum of 3 points to solve for the 6 unknowns (Ω, T) of the camera motion :

$$v^i = M_\Omega^i \Omega + d^i M_T^i T$$

where d^i is the inverse depth $\frac{1}{F_z}$ of point i .

In practice, depth is usually unknown. However, we can change the system of equations to

$$v^i = M_\Omega^i \Omega + \bar{d} M_T^i T$$

where \bar{d} is the average scene disparity, defined as

$$\bar{d} = \frac{1}{N} \sum_{i=1}^N d^i = \frac{1}{N} \sum_{i=1}^N \frac{1}{Z^i}.$$

We observed empirically that the solution of this new system is the exact same motion as the original system. Figure 5.2 shows how varying the constant \bar{d} has no effect on the accuracy of rotation estimation (Fig. 5.2-A,B) or translation direction (Fig. 5.2C), as indicated by the flat error curves obtained. Also, the translation magnitude is scaled from the true magnitude in proportion to the ratio of \bar{d} to the true average scene disparity (Fig. 5.2D). The exact context where this *average scene disparity* property holds remains to be investigated.

If the exact average scene disparity \bar{d} is not known, then an arbitrary value can be used for \bar{d} and the resulting translation will be scaled by an arbitrary factor.

Even if the translation is estimated up to an unknown scale factor in each image along a sequence, there are many ways to relate translation magnitude temporally along a sequence. This is usually accomplished by making various assumptions regarding the motion or scene structure, such as rigid scene, constant velocity, planar motion, planar world, etc. [8]. In our case, we focus on instantaneous motion recovery only, leaving the translation scale determination to other subsequent algorithms. This reduces the need for specific constraints about the motion or the scene and makes the algorithm more general.

5.2.2 From optical flow to normal flow

Equation 5.3 relates the optical flow to camera motion. In practice, optical flow must be computed first. Since this is an ill-posed problem, the accuracy of flow field

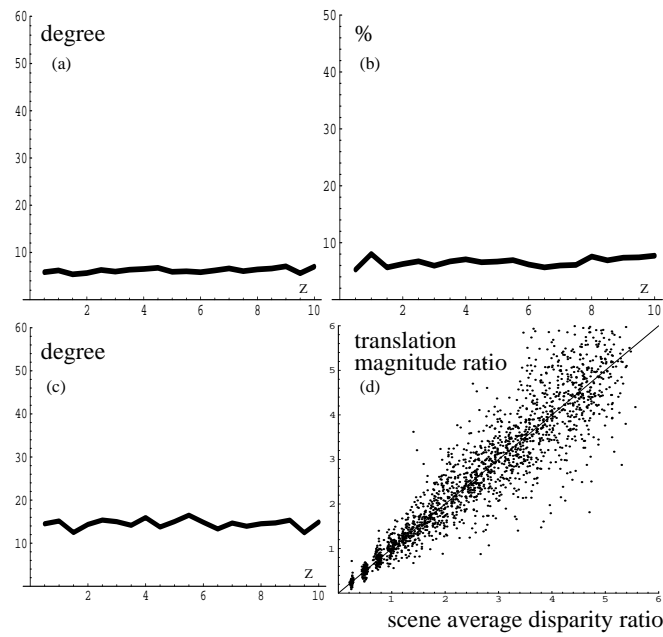


FIG. 5.2. Random camera motions were solved from normal flows while varying the average scene disparity value $\bar{d} = 1/Z$ from $1/0.5$ to $1/10$. A) rotation axis error, expressed as an angle with the true axis, B) rotation magnitude error, expressed as a fraction of true magnitude, C) Translation direction error, expressed as angle with true direction, and D) Ratios of computed translation magnitude to true magnitude compared with ratios of used \bar{d} to true scene average disparity.

is low and it requires a large amount of time to compute.

An alternative is to use the so called *normal flow*, which provides the motion component along the image-gradient direction. The normal flow field is fully determined by the image intensity derivatives and is thus fast to compute and more accurate than optical flow. We can convert Eq. (5.3) so it can work with normal flow.

The solution space of the system Eq. (5.3) is represented by a 4D hyperplane in the motion space that satisfies the camera motion equation. Any motion that falls onto that hyperplane is possible while any motion outside the hyperplane is not possible.

A 4D hyperplane is represented in the motion space with two normals : n_1 and n_2 . The position s of the hyperplane is set by using a sample solution

$$s = (0, 0, 0, -P_z v_x, -P_z v_y, 0).$$

Intuitively, this solution corresponds to explaining the image motion of a single pixel by a simple camera translation parallel to the image plane moving in the opposite direction of the observed pixel motion v .

The 4D hyperplane has two normals n_1 and n_2 which are directly extracted from Eq. 5.3 as

$$\begin{aligned} n_1 &= (p_x p_y, -(1 + p_x^2), p_y, -1/P_z, 0, p_x/P_z) \\ n_2 &= ((1 + p_y^2), -p_x p_y, -p_x, 0, -1/P_z, p_y/P_z) \end{aligned}$$

It then follows that any camera motion $c = (\Omega, T)$ consistent with a single pixel motion (p, v, P_z) is represented by the triple (s, n_1, n_2) and satisfies

$$(c - s) \cdot n_1 = 0 \quad \text{and} \quad (c - s) \cdot n_2 = 0.$$

From the constant brightness assumption (CBA) illustrated in Figure 5.3, we see that the optical flow v is constrained along a line defined by the normal flow v_n as

$$v = v_n + k v_n^\perp \quad (k \in -\infty \dots \infty)$$

Replacing the definition in s yields

$$s = s' + kn_3$$

where $s' = (0, 0, 0, -P_z v_{nx}, -P_z v_{ny}, 0)$ and $n_3 = (0, 0, 0, P_z v_{ny}, -P_z v_{nx}, 0)$. Considering all possible values of k is equivalent to add a dimension to the 4D hyperplane. The new 5D hyperplane has a single normal n_0 which must be a linear combination of n_1 and n_2 and orthogonal to n_3 . We have

$$n_0 \cdot n_3 = (\alpha n_1 + \beta n_2) \cdot n_3 = 0$$

This system is easy to solve and yields $(\alpha, \beta) = v_n$ so

$$n_0 = v_{nx}n_1 + v_{ny}n_2.$$

Now we can relate the normal flow to camera motion

$$n_0 \cdot (c - s') = 0$$

which can be expanded to

$$\begin{pmatrix} v_{nx}P_xP_y + v_{ny}(1 + P_y)^2 \\ -v_{nx}(1 + P_x^2) - v_{ny}P_xP_y \\ v_{nx}P_y - v_{ny}P_x \\ -v_{nx}/P_z \\ -v_{ny}/P_z \\ (v_{nx}P_x + v_{ny}P_y)/P_z \end{pmatrix}^\perp \begin{pmatrix} \Omega \\ T \end{pmatrix} = \|v_n\|^2 \quad (5.4)$$

We can now solve directly for camera motion from the normal flow. The scene disparity can be handled in the same way as before by substituting the average scene disparity \bar{d} for each individual disparity $1/P_z$ in Eq. 5.4.

5.3 Probabilistic Model of Camera Motion

The normal flow used in solving for camera motion is determined by the image intensity derivatives, which are known to be noisy [24].

If we consider the image derivatives as random variables with some known probability densities, then we can derive a probability density

$$p(\Omega; T|\dot{I})$$

of camera motions conditional on measurements of image derivatives \dot{I} at an image point p .

This distribution is broken in two terms, one expressing how the camera motions relate to normal flow, and one expressing how normal flow relates to image intensity derivatives. We have at each point p

$$p(\Omega; T|\dot{I}) = \int p(\Omega; T|v_n)p(v_n|\dot{I})dv_n \quad (5.5)$$

These two terms are described in the following subsections.

5.3.1 Camera Motion from normal flow

We define the probability distribution of the camera motion (Ω, T) for a given normal flow v_n at pixel p as

$$p(\Omega; T|v_n)$$

where the depth of point p is assumed to have been replaced by a constant as in section 5.2.1 so the depth P_z does not appear in the density.

In practice, the squared distance between a motion point $c = (\Omega, T)$ and the 5D hyperplane of Eq. 5.4 will be used to estimate the probability of motion. This distance has the form

$$D(c, p, v_n) = ((c - s') \cdot n_0)^2$$

Note that n_0 is assumed to be normalized.

The probability density is defined as

$$p(\Omega; T|v_n) \propto e^{-D((\Omega, T), p, v_n)^2}$$

5.3.2 Optical flow and image derivatives

As proposed in [24], we define the probability of a true spatio-temporal gradient \dot{I}^* from the measurements \dot{I} as

$$p(\dot{I}^*|\dot{I}) = N(\dot{I}, \Sigma) \quad (5.6)$$

which is a Gaussian distribution centered around the measurement and covariance Σ .

We easily model the distribution of normal image flow v_n from a given true derivative \dot{I}^* as the density

$$p(v_n|\dot{I}^*) = \delta\left(-\frac{\dot{I}_t^*}{\|\nabla\dot{I}^*\|^2}\nabla\dot{I}^*\right)$$

where $\delta(x)$ is the impulse density function. We can then define the distribution of normal flow for an intensity measurement \dot{I} as

$$\begin{aligned} p(v_n|\dot{I}) &= \int p(v_n|\dot{I}^*)p(\dot{I}^*|\dot{I})d\dot{I}^* \\ &= \int \delta\left(-\frac{\dot{I}_t^*}{\|\nabla\dot{I}^*\|^2}\nabla\dot{I}^*\right)N(\dot{I}, \Sigma)d\dot{I}^* \end{aligned} \quad (5.7)$$

Assuming that the full motion v lies within an interval $\pm k$ from the normal flow v_n on the CBA line, we can derive a density $p(v|\dot{I})$ for the full flow by using the relation

$$v = v_n + \lambda v_n^\perp$$

where λ is a random variable with uniform density $U[-k, k]$. Figure 5.4 illustrates the distributions of v for various levels of spatial texture. Notice how the density gracefully represents the increase of motion ambiguity as texture fades.

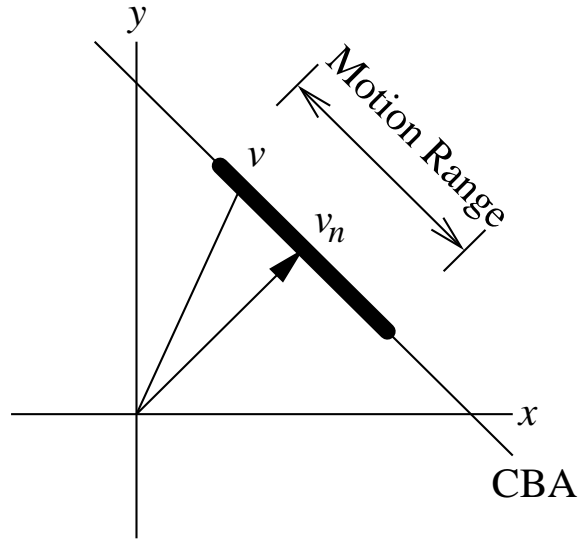


FIG. 5.3. Constant Brightness assumption. The normal flow v_n determines a line that defines the possible motions v . We restrict the motion within an interval around v_n .

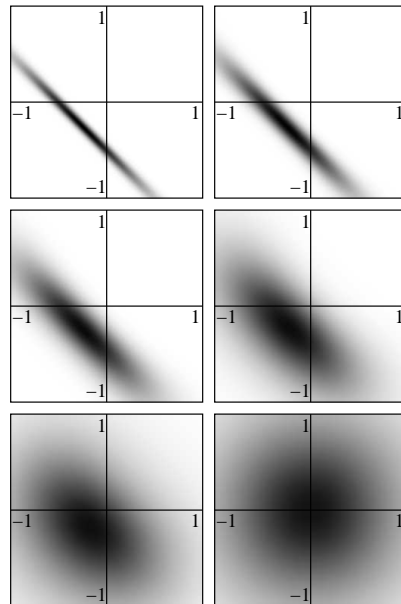


FIG. 5.4. Distribution of v . The spatio-temporal gradient is $\lambda(1, 1, \frac{1}{2})$, with spatial gradient magnitudes set to $\lambda = [16, 8, 4, 2, 1, 0]$ from top-left to bottom-right. The derivative noise is set to a fixed level $\sigma = (1, 1, 1)$.

5.3.3 Global density of camera motion

The probability of camera motion $p(\Omega; T)$ associated with multiple pixels is simply the sum of individual densities of each image point p_i

$$p(\Omega, T) = \sum_i p(\Omega; T | \dot{I}_i).$$

The sum is used in order to provide robustness against multiple simultaneous motions and errors induced by outliers. To find the most probable camera motion, we can simply maximize $p(\Omega; t)$ using a simple gradient descent. The function maximized is continuous and very smooth, making the search converging very quickly while generally avoiding local minima.

Since the density of camera motion cannot be derived analytically, it is built using samples. For each image point p , we compute the spatio-temporal gradient \dot{I} . A number of samples are generated around that gradient according to our model (see eq. 5.6). For each sample gradient \dot{I} , a hyperplane for (p, v_n) is obtained. The hyperplane samples are cumulated and generate the final density of camera motions.

In practice, Eq. 5.5 can be hard to represent directly. We use samples to approximate the density $p(v_n | \dot{I})$. We have also replaced the Gaussian model $N(\dot{I}, \Sigma)$ of Eq. 5.7 by an impulse density $\delta(\dot{I})$ to represent cases where no image noise is expected.

5.3.4 Rotation - Translation Ambiguity

It is well known that horizontal translation (T_x) is hard to distinguish from rotation around the Y axis ($\Omega = (0, \Omega_y, 0)$) [2]. This is illustrated in Figure 5.5. This fact should be reflected in the density of egomotion. To demonstrate this, we have computed the egomotion density for one image point locate in the middle of the image ($p = (0, 0)$) with a horizontal motion ($v = (1, 0)$). The density, shown on the left of Figure 5.6 is ambiguous. Translation and Rotation cancel can each other perfectly.

However, when the selected image point is located in the upper left corner ($p = (-1, -1)$) with the same horizontal motion, it is possible to resolve the ambiguity, as

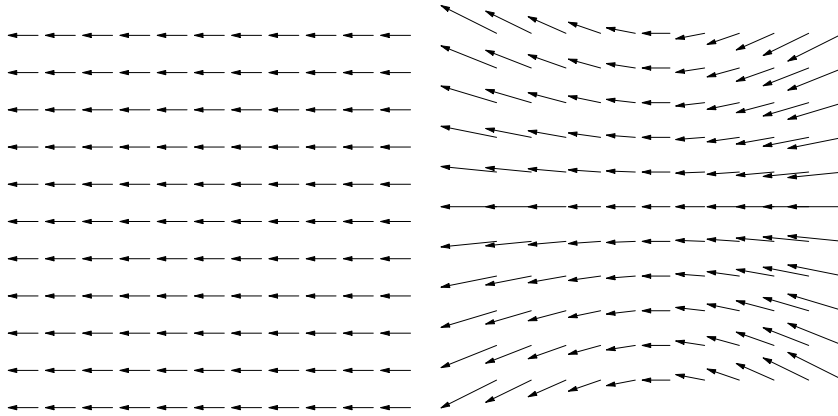


FIG. 5.5. Translation - Rotation ambiguity. Left) A motion field for horizontal translation. Right) A field for pure rotation around the y axis.

shown in the density on the right of Figure 5.6. The minimum in this case is clearly defined and yields the correct minimum.

5.4 Experiments and results

Our method was tested on synthetic and real image sequences. In all cases, we computed the full density of egomotion, where no *a priori* knowledge is available on the camera motion or scene observed.

In all the experiments presented, images were reduced to a size of 256x256 pixels or less. From the images, a small portion of the image pixels are randomly selected (typically 10%, or 400 points). With such low number of points, the running time is close to real time at 10 to 15 frames per second on a 1Ghz PC.

Also, no temporal smoothing of any kind is applied on the camera motion computed at each frame along a sequence. The trajectories presented are the simple accumulation of raw instantaneous motion data.

The 3D camera motion trajectory is illustrated with a vector depicting the z axis of the camera and a vector depicting the y axis.

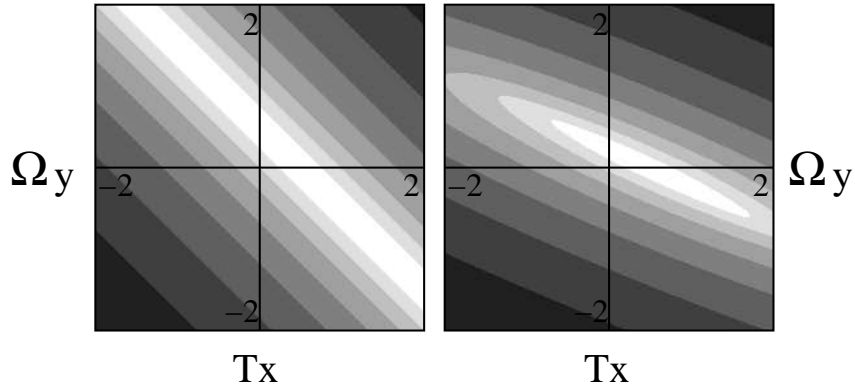


FIG. 5.6. Translation - Rotation ambiguity. Densities are shown for T_x and Ω_y . Left) Density for a point in the middle of the image. The minimum is ambiguous. Right) Density for a point in a corner of the image, removing the ambiguity.

5.4.1 Synthetic sequence

The synthetic sequence used for our tests is presented at the top of Figure 5.7. It features a large number of shaded spheres distributed randomly into a large cube. The camera undergoes a circular motion around the spheres, always looking toward the center. Notice that this sequence contains a large amount of occlusions and very smooth textures, thereby making it very difficult to track using feature point detection. It can be considered "cluttered" as described by Mann et al. [21].

The results, shown in Figure 5.7, show excellent trajectory recovery. In the middle result, the scene depth was known for each point. This allows the algorithm to compute the exact translation magnitude. However, it was slightly over estimated, which explains why the trajectory does overlap itself slightly.

The bottom result of Figure 5.7 was computed using a single depth value along the sequence. The trajectory is now slightly distorted, because the translation magnitude is over estimated when the distribution of sphere depth comes closer to the camera. This happens four times as the camera circles around the corner of the cube of spheres. In this case, we made the assumption that the depth distribution in the

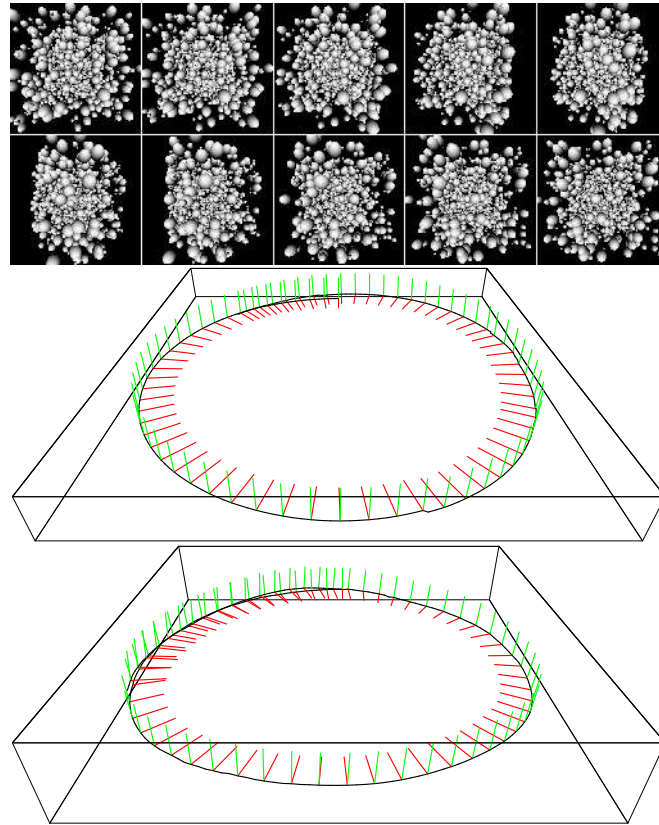


FIG. 5.7. Synthetic sequence. Top) The image sequence. Middle) recovered camera motion trajectory for known depth of scene. Bottom) recovered camera motion trajectory computed without knowledge of depth.

scene is fixed, which is not satisfied. A better assumption would have been to impose constant translational velocity.

Figure 5.8 illustrates on the left how the average depth of the scene varies as the camera circles around the scene. When a corner of the cube of spheres comes closer to the camera (at the diagonal lines) the average depth is closer to the camera. On the right of Figure 5.8, the impact of this average depth variation on the computed translation magnitudes is illustrated. When the average scene depth is over estimated, the translation magnitude is also over estimated.

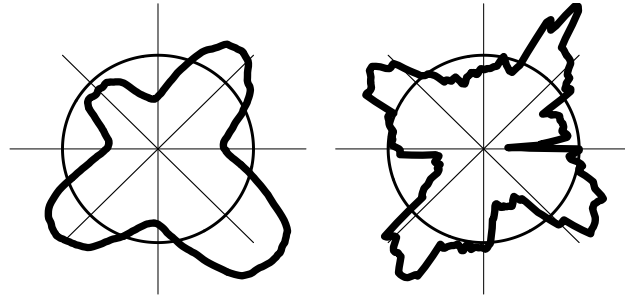


FIG. 5.8. Translation scale factor. Left) Average scene disparity as camera turns around. The 4 bumps near diagonals correspond to the scene getting closer to the camera. Right) Estimated translation magnitude for constant scene depth. The magnitude is over estimated when the scene depth is closer than assumed.

5.4.2 Real image sequences

Real image sequences were taken with a consumer video camera. The motion of the camera is only known approximately, since the sequences were taken handheld without a calibrated rig.

Figure 5.9 shows a forward motion sequence through a corridor. Notice the lack of texture and large amount of specularity on the floor. The results show excellent recovery of the forward trajectory.

Figure 5.10 illustrates a sideway motion sequence (along the x axis of the camera) taken under very low lighting conditions (the images are histogram equalized for illustration purposes). The result, shown in Figure 5.11, show excellent recovery of the sideway motion.

5.5 Conclusion

We presented a fast method to express the probability density of camera motion directly from image intensity derivatives, thereby removing any need for full optical flow estimation. Only the normal flow is used.

We proposed to use the average disparity of the scene for all points to alleviate

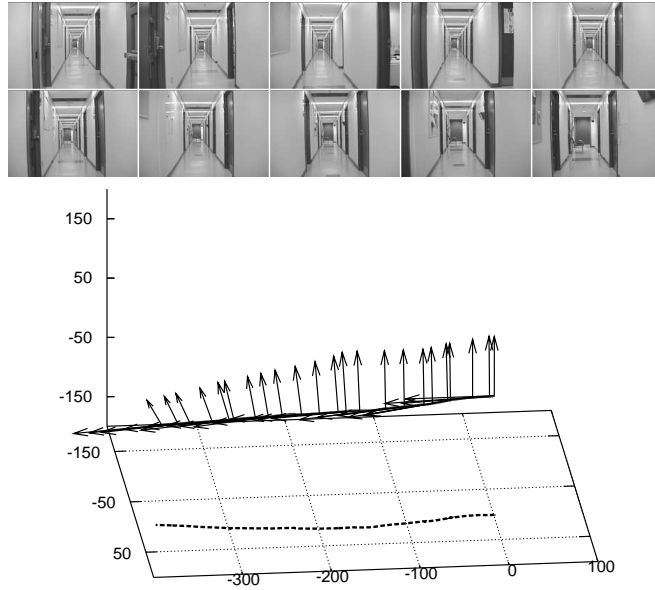


FIG. 5.9. Corridor sequence and recovered trajectory of the camera. The 3D trajectory is projected on the X-Z plane and runs along the Z axis.

the depth estimation problem. This eases considerably the process of estimating the camera translation. It also makes it easy to eventually express temporal constraints about the evolution of scene depth in time.

The probabilistic approach makes this method very robust to image noise and can naturally express camera motion ambiguity when these are effectively present in a sequence. In this framework, image sequences with low texture, bad illumination conditions, or large amount of occlusions can be handled without difficulty.



FIG. 5.10. Sideways motion. Camera is moving left under low lighting conditions. Notice the amount of pixel noise.

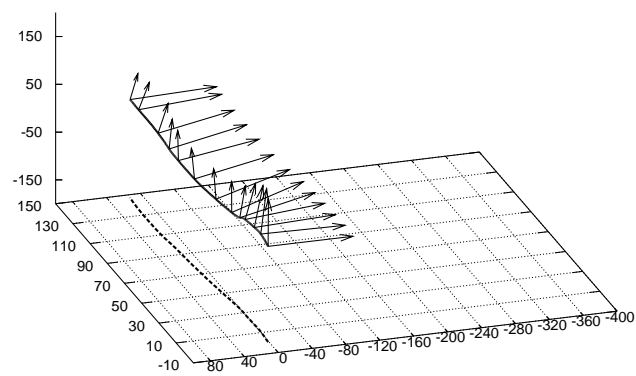


FIG. 5.11. Trajectory for Sideways sequence in low lighting conditions. The trajectory is projected on the X - Z plane and runs along the X axis.

Chapitre 6

RÉSULTATS SUPPLÉMENTAIRES

Dans ce chapitre nous ajoutons quelques résultats qui n'étaient pas disponibles lors de la rédaction de l'article qui a fait l'objet du chapitre précédent.

Les résultats supplémentaires s'articulent autour de trois aspects importants de l'estimation proposée : la résolution en temps réel, le choix de la technique minimisation d'énergie et la prise en compte des mouvements multiples.

6.1 La résolution en temps réel

Tel que mentionné au chapitre précédent, l'estimation statistique du mouvement repose sur un processus de vote. Ce processus étant fortement quantitatif, la solution sera déterminée selon la majorité. En général cette majorité se forme rapidement, avec l'utilisation d'un faible nombre d'échantillons, et cela se traduit par la formation d'un pic dans le tracé de la fonction d'énergie vis-à-vis de la solution. Cette propriété est fort utile quand on songe à une application en temps réel car l'utilisation d'un petit nombre d'échantillons pour procéder à un vote offre des temps de calcul très faibles.

Afin de mettre cette aspect en évidence, nous avons procédé à une série d'estimations de mouvement de caméra sur la séquence des boules utilisée au chapitre précédent en faisant varier le nombre de pixels pris en compte lors du processus de vote. Les résultats obtenus ont été reportés au tableau 6.1. Cette expérience démontre que pour une certaine tolérance à l'erreur, l'estimation du mouvement peut se faire dans des temps très courts et ce en employant qu'une petite fraction des points de la séquence (typiquement 1% des points de l'image).

Nb de points	% de points	ΔR	ΔT	Temps (sec)
65×10^3	100	0.2%	0.8%	0.869
33×10^3	50	0.2%	0.8%	0.429
16×10^3	25	0.2%	0.8%	0.292
6×10^3	10	0.6%	0.8%	0.251
650	1	1%	1.7%	0.052

TAB. 6.1. Erreurs sur le mouvement estimé par rapport au nombre de points utilisés.

Remarque : Dans tous nos tests nous ne sélectionnons que les pixels dont la magnitude des dérivées spatio-temporelles dépassent un certain seuil.

6.2 Les mouvements multiples

Jusqu'à présent nous n'avons considéré que des scénarios où une caméra se déplaçait dans une scène statique ou avec des objets se déplaçant à faible vitesse. Le mouvement observé est équivalent à celui qui serait observé dans un scénario réciproque, où une caméra fixe observe une scène mobile. Cependant, si deux ou plusieurs objets se déplacent dans des directions opposées, une méthode purement analytique suggérerait un mouvement de caméra nul à défaut de pouvoir estimer un ego-mouvement consistant avec les déplacements observés. A l'opposé, nous avons observé que notre méthode pouvait prendre en compte les scènes à mouvements multiples. En effet, comme on le verra dans l'expérience qui sera présentée, notre méthode suggérera autant d'ego-mouvement que de mouvements indépendants observés pour expliquer ces derniers.

Notre expérience a été effectuée sur une séquence réelle. Une caméra fixée sur un trépied qui observe une scène où d'abord une personne défile dans le champs de la caméra suivi d'une deuxième personne qui passe devant la caméra en sens inverse

(voir fig 6.2).



FIG. 6.1. Exemple de mouvement multiple

Nous avons tracé la courbe de la fonction d'énergie à deux moments de la séquence. Le premier correspond à un instant où la caméra ne voyait qu'une personne se déplacer, le deuxième tracé a été effectué à un moment où les deux personnes se déplaçaient dans le champ de vision de la caméra. Les résultats sont illustrés à la figure 6.2. On voit à travers le tracé de la fonction d'énergie, que la méthode intercepte bien les deux mouvements principaux dans la scène qui font culminer la fonction en deux endroits différents.

Cette propriété donne accès à un domaine très important qu'est l'analyse des mouvements prépondérants dans une scène. Cette technique est très utilisée dans la segmentation du mouvement où il s'agit de segmenter des objets indépendants en se basant sur leurs déplacements.

6.3 La minimisation d'énergie

Tel que cité au chapitre précédent, chaque pixel engendre un hyperplan 5D qui contient les mouvements de caméra valides pour ce pixel. Afin de retrouver le mouvement de caméra on peut penser à résoudre le système linéaire composé de toutes les équations engendrés par les pixels de l'image ou au lieu de cela, minimiser une fonction d'énergie, comme dans notre cas, à l'aide de la méthode de Powell. A cause de la présence d'hyperplan 5D aberrants, la solution de ce système d'équations présente des instabilités numériques. Ceci nous a motivé à employer la méthode de Powell au

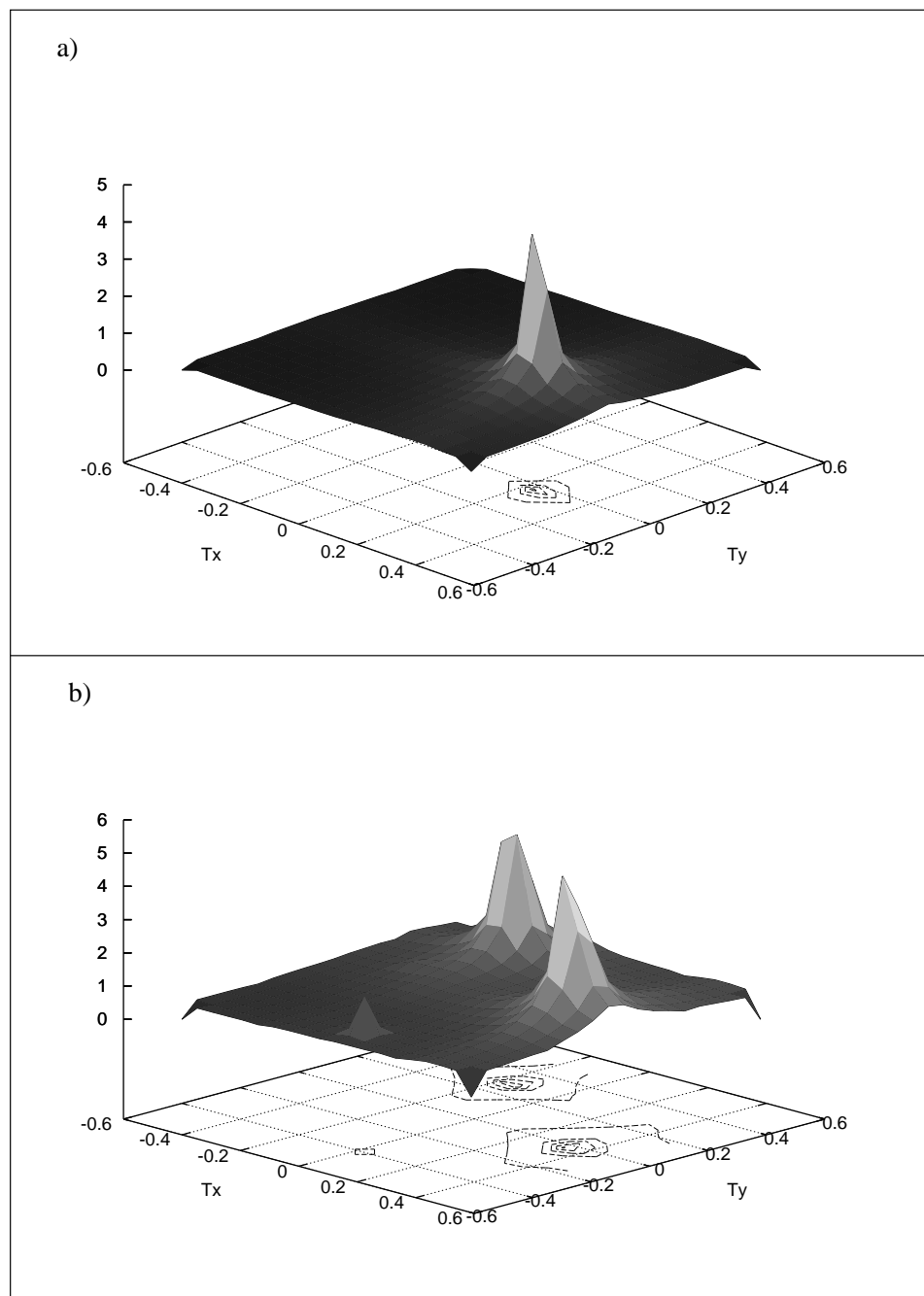


FIG. 6.2. Courbe d'énergie par rapport à la translation en x et en y. La courbe a) représente le tracé de la fonction d'énergie au début de la séquence quand la caméra ne voyait qu'une seule personne. La courbe b) représente l'évolution de l'énergie pour une image où les deux personnes défilent devant la caméra. Chaque pic est associé à un mouvement indépendant de la scène.

Sequence	Valeurs reelles	SVD	Powell
$A (Tx)$	-0.1	-0.094	-0.093
$B (Tz)$	0.1	0.15	0.10
$C (Tx, Ty, Ry)$	(0.6, 0, 1)	(0.68, 2.07, 0.17)	(0.645, 0.01, 0.94)

TAB. 6.2. Comparaisons entre les minimisations à l'aide de la méthode de Powell et de la résolution du système d'équation. Dans les séquences de tests, la caméra observe des boules placées au hasard, dans la séquence A la caméra effectue une translation en X, en B la caméra translate en Z et dans la séquence C la caméra contourne les boules en translatant selon son axe X et en tournant en même temps selon son axe Y

lieu de la résolution du système d'équations. Nous avons établi un test comparatif entre la minimisation d'énergie par la méthode de Powell et la résolution du système d'équations à l'aide d'une décomposition en valeurs singulières afin de justifier le choix de notre minimisation.

Nos tests ont été effectués pour 3 mouvements différents sur la même scène que celle illustrée à la figure 5.7. Les résultats sont illustrés au tableau 6.2.

On conclue, d'après les résultats précédents, que la résolution du système d'équations ne donne pas toujours de bons résultats ce qui n'est pas le cas avec la minimization d'énergie à l'aide de la méthode de Powell.

Chapitre 7

DISCUSSION ET CONCLUSION

Une nouvelle méthode directe pour estimation l'ego-mouvement a été présentée. Les principales caractéristiques de cette méthode sont les suivantes :

- **Aucun suivi de points saillants** : Cette méthode estime le mouvement de caméra en utilisant la variation d'intensités des pixels dans le temps comme seule source d'information et ne nécessite donc aucun suivi de points saillants.
- **Aucune hypothèse sur la nature de la scène** : Aucune connaissance *a priori* n'est requise sur la nature de la scène traitée telle que la présence de textures. Aucune restriction n'est imposée sur la nature du mouvement tel que le font les méthodes destinées à la navigation et qui supposent que le mouvement soit planaire. Cependant, le déplacement entre deux instants doit être de faible magnitude à cause de l'usage des dérivées spatio-temporelles qui suppose que l'intensité est préservée dans le temps.
- **Utilisation d'une mesure de profondeur relaxée** : La connaissance de la profondeur en tout point de la scène n'est pas nécessaire. Il est possible d'estimer le mouvement de caméra en utilisant simplement la moyenne des profondeurs de la scène comme connaissance sur la structure de la scène.
- **Utilisation de formalismes probabilistes** : Une solution robuste au bruit et aux perturbations locales grâce à l'utilisation explicite des densités de probabilité.
- **Rapidité de calcul** : Le fait que cette méthode repose sur un système de vote entraîne que la contribution de tous les pixels d'une image n'est pas obligatoire. Ceci permet de réduire le nombre d'échantillons pour l'analyse statistique et d'atteindre un taux de traitement de près de 15 images par seconde.

- **Prise en compte des mouvements multiples :** En présence de mouvements multiples, l'algorithme est garanti de converger vers un des mouvements observés dans une scène et ce même si les mouvements ont la même prépondérance.

Notre méthode a été testée sur une séquence vidéo synthétique démunie de textures et de points saillants. Ce genre de scène est très difficile à analyser avec des méthodes traditionnelles. L'analyse du mouvement a donné de très bons résultats avec de faibles erreurs et ce malgré la présence d'occlusions.

Nous avons entrepris des tests sur des séquences réelles pour lesquelles aucune connaissance sur la structure de la scène n'était disponible. Même en présence de bruit, notre méthode a pu fournir de bons résultats sur la nature du mouvement observé.

Nous croyons que cette nouvelle approche probabiliste peut être utilisée avec succès pour résoudre le problème du mouvement de caméra à partir de séquences d'images. De plus, nous croyons que cette nouvelle méthode se montrera supérieure aux méthodes plus conventionnelles qui reposent sur le calcul du flux optique, de la mise en correspondance ou à la résolution de tout autre sous-problème intermédiaire.

Certains développements futurs sont envisageables. Nous pensons qu'il serait très intéressant d'inclure à notre algorithme une estimation grossière de la profondeur qui permettrait d'approximer la profondeur moyenne de la scène. Cette approximation pourrait être combinée à une mise à jour constante à l'aide d'un filtre de Kalman pour accélérer le processus d'estimation de la structure.

De même une meilleure prise en compte des mouvements multiples pourrait être apportée afin de pouvoir extraire efficacement tous les mouvements observés dans une scène.

RÉFÉRENCES

- [1] Ansel Adams. *The Camera*. Little, Brown, Boston, 1980.
- [2] G. Adiv. Inherent ambiguities in recovering 3-d motion and structure from a noisy flow field. *Transaction in Pattern Analysis and Machine Intelligence*, pages 477–489, 1989.
- [3] J. K. Aggarwal et N. NandHakumar. On the computation of motion from sequences of images – a review. *Proceedings of the IEEE*, pages 917–935, 1988.
- [4] Y. Aloimonos et Z. Duric. Estimating the heading direction using normal flow. *International Journal of Computer Vision*, pages 33–56, 1995.
- [5] Knutsson H. Barman H., Haglund L. et Granlund G. Estimation of velocity, acceleration and disparity in time sequences. Dans *Proc. Ieee Workshop on Visual Workshop*, pages 44–51, 1991.
- [6] J. L. Barron, D. J. Fleet, et S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, pages 43–77, 1994.
- [7] Daphna Weinshall Boubakeur Boufama et Michael Werman. Shape from motion algorithms : a comparative analysis of scaled orthography and perspective. Dans *European Conference on Computer Vision, Stockholm, Sweden*, pages 199–204, May 1994.
- [8] J.-Y. Bouguet et P. Perona. Visual navigation using a single camera. Dans *International Conference on Computer Vision*, pages 645–653, 1995.
- [9] Jean-Yves Bouguet et Pietro Perona. Visual navigation using a single camera. Dans *International Conference of Computer Vision*, pages 645–652, 1995.
- [10] A. Branca, E. Stella, et A. Distanto. Mobile robot navigation using egomotion estimates. *International Conference on Intelligent Robots and Systems*, pages 533–537, 1997.

- [11] W. Burger et B. Bhanu. Estimating 3-d egomotion from perspective image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1040–1058, 1990.
- [12] Heeger D.J. Optical flow using spatiotemporal filters. *International Journal of Computer Vision*, 1 :279–302, 1988.
- [13] Cornelia Fermüller et Yiannis Aloimonos. Qualitative egomotion. *International Journal of Computer Vision*, pages 7–29, 1995.
- [14] J.J Gibson. *The perception of the visual world*. Houghton Mifflin, Boston, 1950.
- [15] S. Hall-Holt, O. ; Rusinkiewicz. Stripe boundary codes for real-time structured-light range scanning of moving objects. Dans *International Conference on Computer Vision*, pages 108–115, 2001.
- [16] K. Horn et B. Schunck. Determining optical flow. *Artificial intelligence*, pages 185–203, 1981.
- [17] Duncan J.H et Chou T.C. Temporal edges : The detection of motion and the computation of optical flow. Dans *International Conference of Computer Vision*, pages 374–382, 1988.
- [18] H.C. Longuet-Higgins et K. Prazdny. The interpretation of a moving retinal image. pages 385–397, 1980.
- [19] B.D. Lucas et T. Kanade. An iterative image registration technique with an application to stereo vision. Dans *DARPA IU Workshop*, pages 121–130, 1981.
- [20] B.D. Lucas et T. Kanade. An iterative image registration technique with an application to stereo vision. Dans *International Joint Conferences on Artificial Intelligence*, pages 674–679, 1981.
- [21] R. Mann et M. S. Langer. Optical snow and the aperture problem. Dans *International Conference On Pattern Recognition*, pages 264–267, 2002.
- [22] Anandan P. A computational framework and an algorithm for the measurement of visual motion. pages 283–310, 1989.

- [23] William H. Press, Brian P. Flannery, Saul A. Teukolsky, et William T. Vetterling. *Numerical Recipes : The Art of Scientific Computing*. Cambridge University Press, Cambridge (UK) and New York, 2nd édition, 1992.
- [24] Gideon P. Stein, Ofer Mano, et Amnon Shashua. A robust method for computing vehicle ego-motion. Dans *IEEE Intelligent Vehicles Symposium*, pages 362–368, 2000.
- [25] Sébastien Roy et Venu Govindu. MRF solutions for probabilistic optical flow formulations. Dans *International Conference On Pattern Recognition*, pages 1041–1047, 2000.
- [26] C. Silva et J. Santos-Victor. Robust egomotion estimation from the normal flow using search subspaces. *IEEE Trans. Pattern Analysis and Machine Intelligence*, pages 1026–1034, 1997.
- [27] A. James Stewart et Michael S. Langer. Towards accurate recovery of shape from shading under diffuse lighting. Dans *Conference on Computer Vision and Pattern Recognition*, pages 411–416, June 1996.
- [28] Emanuele Trucco et Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, New Jersey, 1998.

Annexe A

ANNEXE

A.1 Angle de vue de la caméra

L'angle de vue de la caméra est complètement défini par les paramètres internes de la caméra, c.a.d : le facteur d'échelle (les dimensions de l'image), la longueur focale et le centre de l'image [1]. Le plus souvent on caractérise l'angle de vue de la caméra comme étant l'angle de vue horizontal.

Tel que mentionné précédemment, les paramètres internes sont connus et mesurés en laboratoire à l'aide d'objets de calibration dont la géométrie est connue.

Si on prend une vue du dessus de la caméra, tel qu'illustré à la figure A.1, la tangente de la moitié de l'angle de vue est définie comme suit :

$$\tan\left(\frac{\alpha}{2}\right) = \frac{\text{largeur}}{2f}$$

Ce qui donne :

$$\alpha = 2 \arctan\left(\frac{\text{largeur}}{2f}\right)$$

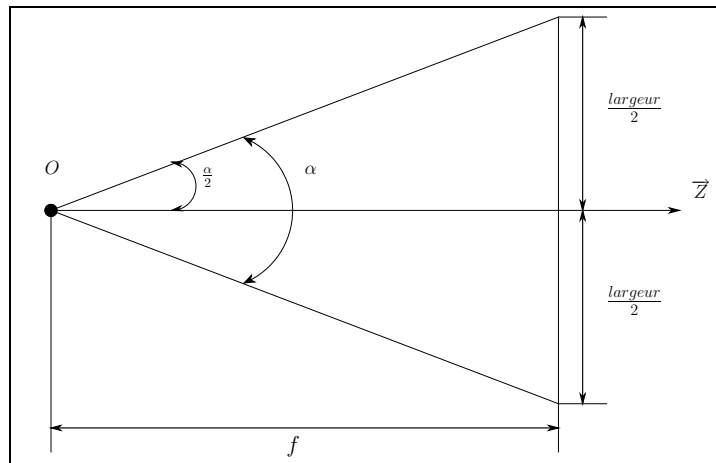


FIG. A.1. Illustration d'une vue de dessus de la caméra. Le point O et la longueur f désignent respectivement le point focal et la longueur focale, avec l'angle de vue, noté α , et la largeur de l'image ils constituent les paramètres internes de la caméra.

Parmi tous les paramètres internes ci-haut mentionnés, la longueur focale est celui qui influence le plus l'angle de vue de la caméra car dans la pratique il n'est pas possible de changer les autres paramètres sur les caméras. La longueur focale par contre peut varier à l'aide d'un zoom par exemple.

A.2 Minimisation de fonction d'énergie par la méthode de Powell

Il existe plusieurs méthodes numériques destinées à minimiser une fonction d'énergie. La plupart de ces méthodes reposent sur de simples routines de minimisation unidimensionnelle telles que la minimisation dichotomique linéaire ou quadratique. Ces simples routines sont destinées à minimiser une fonction 1D. Cependant afin de pouvoir prendre en compte les fonctions multidimensionnelles, ces méthodes numériques combinent ces routines à une judicieuse stratégie qui sélectionne l'ordre dans lequel chacune des dimensions sera sélectionnée afin de les minimiser successivement.

Il existe deux principales familles de méthodes de minimisation. La première famille suppose le gradient (ou la dérivée) de la fonction à minimiser est évaluable. Parmi ces méthodes on retrouve la descente de gradient à pas simple ou à pas adaptatif, la méthode de Newton-Raphson et la méthode de Levenberg-Marquardt pour n'en citer que quelques unes. L'inconvénient de ces méthodes est qu'elles reposent sur la connaissance analytique de la dérivée ou du moins d'une fonction qui évalue le gradient. Cette fonction n'est malheureusement pas toujours disponible et notre cas en est un exemple.

Afin d'y remédier, il faudra faire appel à une deuxième famille de méthodes de minimisation qui ne reposent pas sur l'évaluation du gradient. L'une des méthodes les plus utilisées et qui a été employée dans ce mémoire, est la minimisation de Powell.

La méthode de Powell repose sur une routine de minimisation 1D appelée *minimisation de Brent*. A cela sera combiné un sélecteur de dimension afin de traiter les fonctions à n -dimensions.

Nous allons présenter les principes de base de la minimisation de Powell. Pour plus de détails, l'ouvrage [23] constitue une référence dans le domaine du calcul numérique.

La minimisation de Brent

Cette méthode simple, minimise une fonction à une dimension par recherche dichotomique et se résume comme suit : partant d'un encadrement initial du minimum, on cherche à réduire l'intervalle d'encadrement jusqu'à ce que l'on puisse considérer que l'on a atteint un seuil de convergence.

Pour encadrer un zéro, deux valeurs suffisent, sous réserve que la fonction change de signe. Dans le cas d'un minimum local, trois valeurs a , b et c sont nécessaires, satisfaisant les conditions de la recherche dichotomique :

$$a < b < c, \quad f(b) < f(a), \quad f(b) < f(c)$$

Par continuité de la fonction f , on a alors un minimum entre a et c . On choisit alors un point intermédiaire x entre a et b , ou entre b et c .

Suivant la position de b par rapport à x , et en fonction de $f(b)$ par rapport à $f(x)$, on doit distinguer quatre cas de figures différents illustrés par la figure A.2.

On poursuit ainsi l'itération jusqu'à ce que l'on ait atteint le seuil de convergence souhaité, à savoir que la longueur de l'intervalle d'encadrement $|c - a|$ soit suffisamment petite.

Le problème qui subsiste maintenant est le choix du point intermédiaire x dans l'intervalle entre a et c . Brent suggère d'utiliser une interpolation parabolique.

L'interpolation parabolique consiste à calculer les paramètres d'une parabole passant par les points $(a, f(a))$, $(b, f(b))$ et $(c, f(c))$, le point intermédiaire x correspondant au minimum de cette parabole. On obtient par quelques calculs simples non détaillés ici la formulation suivante :

$$x = b - \frac{1}{2} \frac{(b-a)^2(f(b)-f(c)) - (b-c)^2(f(b)-f(a))}{(b-a)(f(b)-f(c)) - (b-c)(f(b)-f(a))}$$

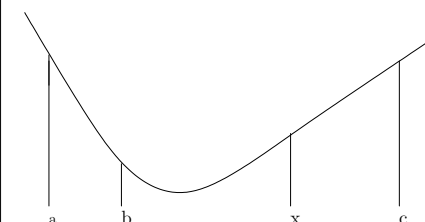
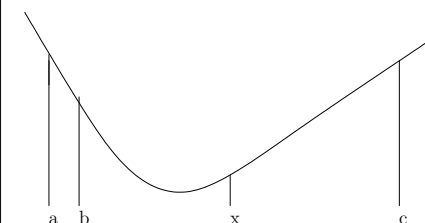
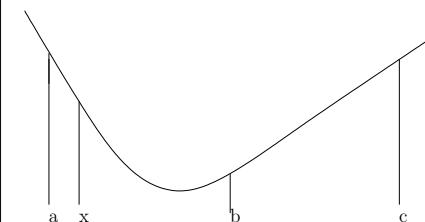
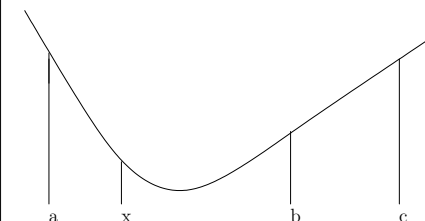
$x > b$	$f(x) > f(b)$		Intervalle courant a,b,c Nouvel intervalle a,b,x
	$f(x) < f(b)$		Intervalle courant a,b,c Nouvel intervalle b,x,c
$x < b$	$f(x) > f(b)$		Intervalle courant a,b,c Nouvel intervalle x,b,c
	$f(x) < f(b)$		Intervalle courant a,b,c Nouvel intervalle a,x,b

FIG. A.2. Les quatre cas de divisions possibles.

La minimisation à n -dimensions

Maintenant qu'on dispose d'une méthode pour minimiser une fonction 1D, il est possible de minimiser une fonction g à n dimensions connaissant :

- la fonction $g : \mathfrak{R} \rightarrow \mathfrak{R}$.
- un point $P_0 \in \mathfrak{R}^n$ donné.
- une direction $u \in \mathfrak{R}^n$ donnée.

On peut alors définir une fonction h à une variable unique tel que :

$$h(\lambda) = g(P_0 + \lambda u)$$

Dans ce cas, la fonction g représente une coupe de la fonction f , suivant la direction u et passant par le point P_0 . On peut alors minimiser cette fonction en utilisant la méthode de Brent décrite précédemment.

L'idée de base de la méthode de Powell repose alors sur l'itération suivante :

- on part d'un point P_0 et d'une direction donnée u_0 .
- par minimisation 1D suivant la direction u_0 , on obtient un point P_1 .
- à partir de P_1 on regarde ce qui se passe suivant u_1 .
- par itération, on crée ainsi P_2, u_2, \dots jusqu'à convergence de l'itération.

Si le gradient de la fonction g avait été calculable, on aurait pris les directions du gradient comme étant les directions u_i . Comme ce n'est pas le cas, les directions de base utilisées sont les vecteurs unitaires.

Au-delà de n itérations (on a minimisé suivant toutes les directions), on recommence le processus avec u_0 , et ce autant de fois que nécessaire.

