N° d'ordre : 2179

THÈSE

présentée pour obtenir

LE TITRE DE DOCTEUR
DE L'INSTITUT NATIONAL POLYTECHNIQUE DE TOULOUSE

École doctorale Informatique et Télécommunications
Spécialité Informatique de l'Image et du Langage

Par
Matthijs DOUZE

Estimation d'homographies inter-images
—
cas des mosaïques et du suivi en temps réel
—
applications en réalité augmentée

Soutenue le 10 décembre 2004 devant le jury composé de:

M	R. Mohr	*Président*
MM	M. Dhome	*Rapporteurs*
	M.-O. Berger	
MM	S. Van Huffel	*Examinateurs*
	A. Ayache	
	B. Thiesse	
	V. Charvillat	

Remerciements

Ce travail a été réalisé au Laboratoire d'Informatique et Mathématiques Appliquées de l'École Nationale Supérieure d'Électrotechnique, d'Électronique, d'Informatique et de Télécommunications de Toulouse, dépendant de l'Institut de Recherche en Informatique de Toulouse. Je remercie Alain Ayache, directeur de l'établissement et directeur de ma thèse pour m'avoir accordé sa confiance en me permettant d'y effectuer ma thèse et en me donnant ensuite la fonction d'Attaché Temporaire d'Enseignement et de Recherche.

Je suis sensible à l'honneur que m'a fait Monsieur Roger Mohr de présider le jury de cette thèse.

Je remercie Madame Marie-Odile Berger et Monsieur Michel Dhome d'avoir accepté la lourde tâche de rapporteurs et d'avoir consacré un temps précieux à l'examen de ce manuscrit. La qualité et la précision de leurs remarques m'ont permis de l'améliorer.

Je remercie Madame Sabine Van Huffel pour avoir accepté d'examiner cette thèse et de participer à ce jury.

Durant cette thèse j'ai été encadré par Bernard Thiesse et Vincent Charvillat. Leurs réponses à mes interrogations théoriques, leur motivation inépuisable et la complémentarité de leurs approches m'ont été très précieuses. Je leur adresse mes vifs remerciements.

Je tiens aussi à remercier tous les membres de l'équipe VPCAB pour le cadre favorable qu'ils ont su créer ces dernières années mais aussi pour les nombreux moments partagés en dehors de ce cadre qui ont su transformer des collègues en amis.

Estimation d'homographies inter-images

—

Cas des mosaïques et du suivi en temps réel

—

Applications en réalité augmentée

Matthijs Douze

Compilé le 9 décembre 2004 à 18:12

Table des matières

Chapitre 1

Introduction

1.1 Vu à la télévision

Dans la panoplie des effets spéciaux au cinéma ou à la télévision, un des plus classiques consiste à insérer un personnage réel dans un décor de synthèse. Les prévisions météorologiques à la télévision, où un présentateur désigne des régions sur une carte, en sont un exemple typique. Cette carte n'existe pas en réalité : le présentateur est filmé devant un arrière-plan (bleu). Celui-ci peut être détecté automatiquement grâce à sa couleur inhabituelle et remplacé par la carte que l'acteur semble montrer. Ce procédé, appelé *chroma-keying*, est intégré dans les chaînes de traitements vidéo analogiques depuis les années soixante-dix.

Cependant, si le présentateur se déplace et si un opérateur « consciencieux » ou non averti croit bien faire en braquant la caméra sur lui, la supercherie est dévoilée en plein jour : la carte en arrière-plan ne bouge pas tandis que le présentateur semble être en lévitation au-dessus du sol.

Si l'image en arrière-plan est remplacée par une image synthétisée –en temps réel– en cohérence avec les mouvements de la caméra, alors la position relative du présentateur et de ce plan est réaliste : le truquage devient indécelable. Cette synthèse cohérente nécessite de connaître une information minimale mais suffisante, caractéristique des mouvements de la caméra.

Disons, pour faire court, que le calcul de ladite information est au cœur de notre travail.

1.2 Les grandes lignes de notre approche

Au moins deux voies existent pour solutionner le problème précédent :

– à l'aide d'un système mécanico-électronico-informatique sophistiqué. Il peut s'agir d'un capteur couplé à la caméra, pour en déterminer la pose et la focale qui transmet ces paramètres avec le flux vidéo. Le système FlySPEC [LKF+02] est composé de 4 caméras fixes qui filment la scène en entier et d'une caméra orientable, montée sur un pied motorisé, commandée par un calculateur ;

– à l'aide des outils de vision par ordinateur. Il peut s'agir de détecter l'arrière-plan pour en déduire les paramètres de pose et de focale de la caméra. Matsunaga et Kanatani [MK01] marquent un fond bleu d'un motif aisément détectable pour faciliter la localisation précise de l'arrière-plan. Ceci fait, les paramètres de la caméra coulent de source.

Quant à nous, nous empruntons la seconde voie avec les deux particularités capitales suivantes :

– aucun arrière-plan particulier n'est ajouté à la scène observée. Autrement dit, la scène est ce qu'elle est (raisonnablement « texturée ») et les images de la séquence constituent des données suffisantes pour calculer les paramètres de la caméra ;
– la séquence d'images est traitée au rythme de la vidéo.

La tâche, trop compliquée dans le cas général, trouve une solution sous les deux hypothèses simplificatrices :

– le centre optique de la caméra est fixe par rapport à la scène. Les changements de direction de visée et de focale (zoom) ne sont soumis à aucune contrainte ;
– une image de référence (dite « panoramique[1] »), embrassant toute la scène susceptible d'être observée, est disponible et avec elle la position du centre optique de la caméra qui a permis de l'obtenir.

Remarques :

– en pratique, les deux contraintes précédentes sont à relativiser vis à vis de la distance caméra $\leftrightarrow$ scène ;
– l'image panoramique de référence est toute indiquée pour faire fonction d'arrière-plan.

1.2.1 Le contexte scientifique en quelques repères

Au cœur de notre travail se trouvent l'étude et l'exploitation de la géometrie qui lie plusieurs (≥ 2) images d'une même scène. Deux ouvrages (tout sauf obsolètes) concurrents mais complémentaires sur le sujet seront cités a de maintes reprises, tant ils constituent des références incontournables. Nous avons nommé « l'œuvre » de R. Hartley et A. Zisserman [HZ03] d'une part et celle de O. Faugeras, Q.T. Luong et T. Papadopoulo d'autre part [FLP01].

Longtemps (et toujours) utilisés en environnement robotique ces outils de vision par ordinateur ont trouvé un autre champ d'application avec l'entrée massive du « tout numérique » dans la vie de tout un chacun : nous voulons parler des effets spéciaux au cinéma, à la télévision,... En France, la conférence invitée de O.D. Faugeras lors du congrès RFIA 1998 [Fau98] a servi de détonateur pour beaucoup d'équipes de recherche : le groupe VPCAB de l'ENSEEIHT est l'une d'entre elles. Les mots « homographie », « mosaïque », « image interpolée », « vidéo », « réalité augmentée », introduits et illustrés par O.D Faugeras sont des clefs de notre travail.

En terme d'applications, notre sujet d'intérêt est à rapprocher de la génération d'une

[1] Au sens propre, l'adjectif « panoramique », appliqué à une image ou une vue signifie « qui permet de découvrir un vaste "paysage" » [Lar02] p740.

vue panoramique à partir d'une séquence vidéo ([IAB+96], [DM96], [BDH03]). Celle-ci est souvent utilisée sur des séquences vidéo d'événements sportifs. La vue panoramique représente alors le stade, les athlètes sont détectés sur les images et extraits. Les solutions proposées se passent d'image de référence mais il n'est pas question d'envisager le temps réel !

Le domaine bénéficiant de cette technique est le codage vidéo. La tendance actuelle consiste à coder des flux vidéo de plus en plus complexes sous forme d'objets vidéo distincts. Dans ce cadre, la séparation des personnages (ou autres objets situés au premier plan) de l'image panoramique constituant l'arrière-plan trouve pleinement sa place [PE02].

1.2.2 Notre scénario

La figure 1.1 présente la structure générale de notre système d'acquisition et de traitement (appelé Pan/Cam). Il est constitué de trois composants essentiels.

La Génération des données de référence. Nous captons une vue de la scène immobile, soit avec une caméra panoramique, soit à partir de plusieurs vues de caméras standard organisées en mosaïque (tâche (1) sur la figure 1.1). Certains calculs préparatoires au suivi sont faits hors ligne (« sans » contrainte de temps) sur l'image panoramique ;

Le suivi (la navigation) et la détection d'« intrus ». En phase d'« exploitation », des objets (au sens élargi englobant les êtres vivants et les choses inanimées) peuvent évoluer dans la scène filmée par une caméra vidéo numérique DV (*digital video*) dont le centre optique coïncide avec celui de la caméra panoramique. Le flux vidéo est **aligné** sur l'image de référence. Les objets occultants sont détectés et extraits par « soustraction ». Tout ceci au rythme de la vidéo bien entendu !

La composition du flux vidéo en sortie. Un flux vidéo est composé d'un nouvel arrière-plan, arbitraire, re-projeté sur la séquence en cours (4), et des intrus. Cette séquence enrichie (au sens de la réalité augmentée) est affichée en temps réel sur un écran de visualisation (5).

Ce scénario de base peut être amélioré de maintes manières ; en voici six que nous avons traitées avec des degrés d'achèvement inégaux.

Modification de l'arrière-plan (6). Il peut s'agir tout simplement d'une version filtrée de l'image de référence ou encore d'une image de synthèse. Dans un autre registre, cet arrière-plan étant fabriqué hors ligne, un concepteur CAO ou un artiste peuvent le peaufiner tout à loisir.

Modification des intrus (7). Un exemple de traitement ordinaire consiste à harmoniser leurs couleurs ou leur illumination en fonction de leur nouvel arrière-plan.

Insertion d'autres objets virtuels (8).

Changement du point de vue. La position relative du premier plan et de l'arrière-plan est connue. Une caméra virtuelle peut simuler un autre point de vue durant l'étape de visualisation.

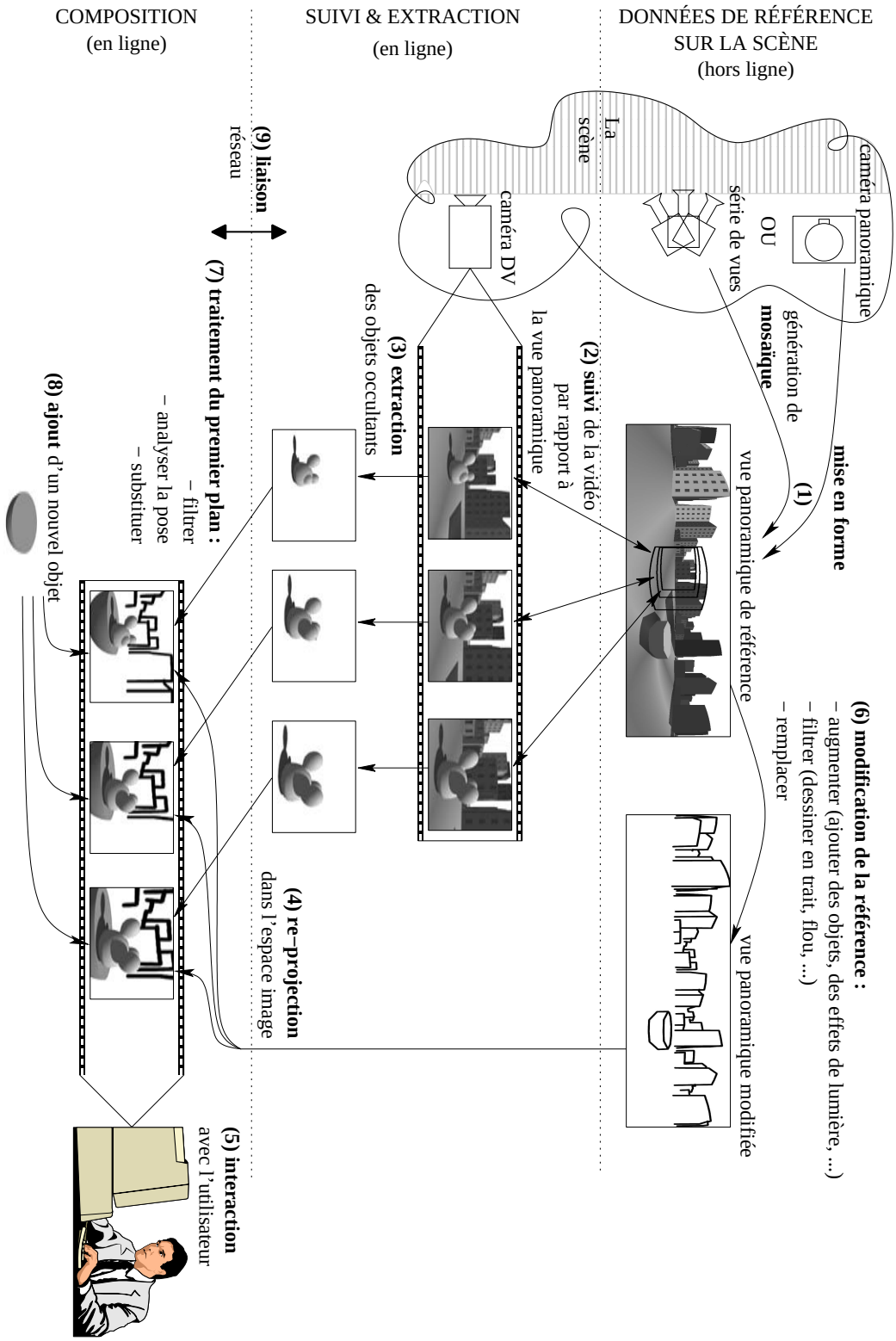


FIG. 1.1: La chaîne de traitements Pan/Cam, construite autour des algorithmes que nous avons développés.

Interaction entre Cam/Pan et l'utilisateur. Un « comportement » peut être associé à tout clic de l'utilisateur sur tout intrus ou sur tout objet de synthèse ajouté.

« Télé-exécution » à travers un réseau. Les opérations du suivi/détection des intrus et de composition peuvent être exécutées sur deux machines distantes, reliées par un réseau (9). Les données qui transitent par l'interface réseau sont préalablement compressées pour alimenter un multiplexeur. Cette conversion en flux de données peut être faite suivant la norme MPEG-4. Un codeur MPEG-4, exécuté sur le serveur, transforme la vue panoramique de l'arrière-plan et les images de l'intrus en flux sur le réseau. Un lecteur MPEG-4 quelconque, exécuté du côté client, décode ces flux et combine les images reconstruites pour les afficher sur l'écran de l'utilisateur. La « cerise sur le gâteau », c'est que le lecteur MPEG-4 peut aussi réagir aux actions de l'utilisateur, ce qui permet de réaliser à distance la tâche d'interaction ci-dessus mentionnée.

Ajoutons pour achever le tableau que ce type de scénario peut être tourné « chez soi » avec des accessoires que l'Homme Moderne utilise quotidiennement : une caméra DV, un ordinateur rapide et une connexion Internet à large bande passante.

1.3 Notre travail brossé en trois touches

Les tâches (5), (6), (7), (8) et (9) de la figure 1.1 dépendent de l'application ou de l'inspiration de l'utilisateur. Nous nous sommes focalisés sur celles qui mettent en jeu la géométrie et la photométrie des images multiples d'une même scène : (1), (2), (3) et (4).

1.3.1 Notre meilleur levier : l'homographie en dimension deux

Le mot-clé, **homographie**, est en dénominateur commun des tâches :
- génération de mosaïques (1) ;
- suivi de la vidéo par rapport à l'image panoramique (2) ;
- rectification de l'image panoramique pour la détection (3) ;
- rectification de l'image panoramique modifiée pour la visualisation (4).

Ajoutons à cela que les caméras panoramiques dédiées (maison) utilisées en amont de la génération de l'image de référence nécessitent un étalonnage préalable à la rectification de celle-ci.

Une homographie en dimension deux est une transformation ponctuelle qui permet d'établir une bijection entre une cible de l'espace 3D et une image, ou bien entre deux images d'une même cible quelconque (sous l'hypothèse d'un modèle de prise de vues est une projection centrale plane) dans au moins deux cas :
- la cible de l'espace 3D est assimilée à un plan ;
- le centre de projection d'une première prise de vue est confondu avec celui d'une seconde (sans contrainte sur la cible).

Concrètement, nous abordons l'estimation d'une homographie 2D sous trois angles différents, selon que :

– la zone de recouvrement entre deux images est inconnue a priori (donc éventuellement faible voire très faible). C'est le cas de la génération de mosaïques (figure 1.2) ;

FIG. 1.2: Régions mises en correspondance par une homographie dans deux images d'une scène, en vue d'en faire une mosaïque.

– la zone de recouvrement entre deux images est grande a priori (hypothèse de petites variations dans la géométrie scène ↔ caméra). C'est le cas de suivi ordinaire de surfaces (figure 1.3), lui-même composant du suivi dans une image panoramique (figure 1.4) ;

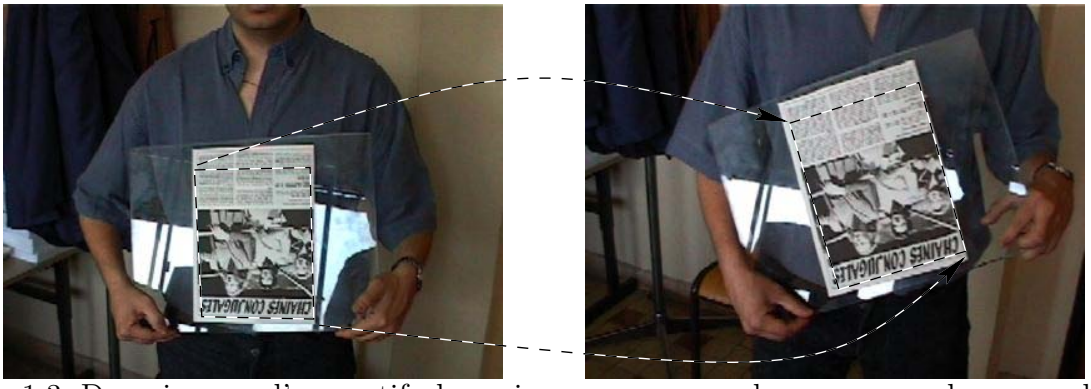


FIG. 1.3: Deux images d'un motif plan mises en correspondance par une homographie.

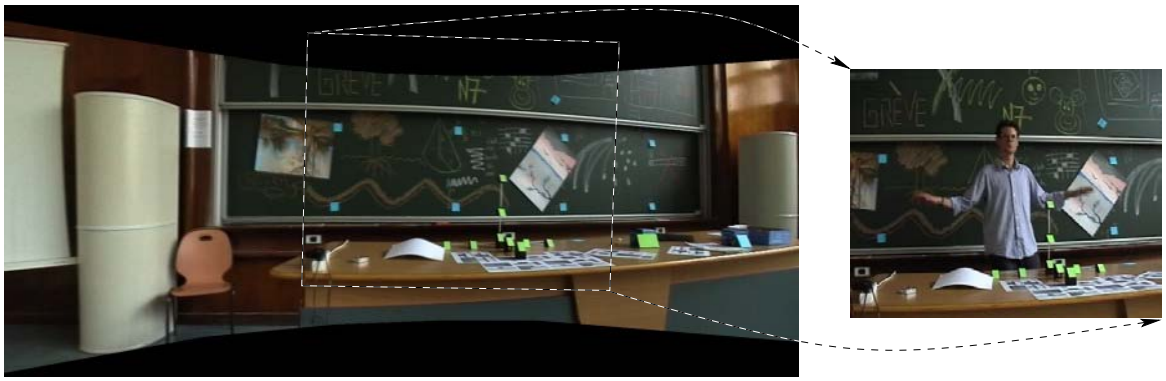


FIG. 1.4: Mise en correspondance, basée sur une homographie, d'une image de la séquence vidéo avec la vue panoramique de référence.

– la géométrie des « points d'intérêt » de la cible de l'espace 3D est parfaitement connue. C'est le cas de l'étalonnage géométrique d'une caméra panoramique (figure 1.5).

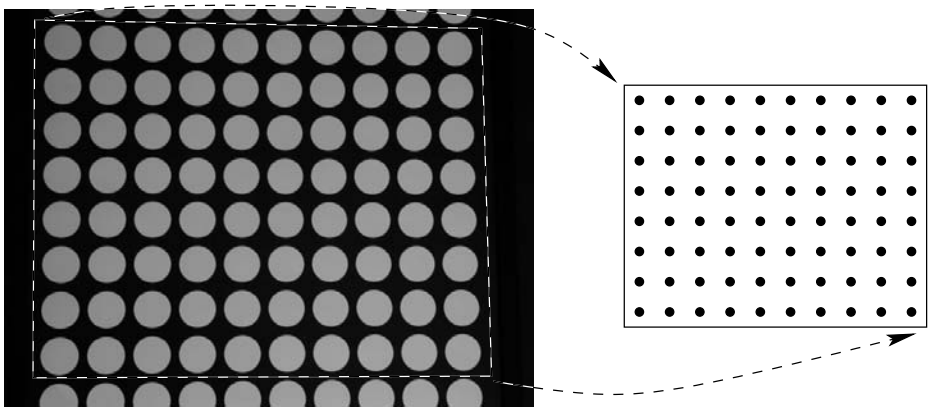


FIG. 1.5: Photo d'une mire d'étalonnage et vue schématique. Les centres des disques (3D) sont en bijection avec leurs images (2D) à travers une homographie.

1.3.2 Notre contribution

Sans conteste, notre concours[2] majeur (le point d'aboutissement) concerne la réalisation ♠
d'un système complet de « superposition » (nous parlerons d'alignement géométrique) appelé Cam/Pan, au rythme de la vidéo, des images acquises par une caméra DV sur une image panoramique de référence, fabriquée hors ligne par un algorithme « maison ». Cam/Pan est un assemblage réussi de plusieurs « briques » logicielles de base, elles-aussi maison : nous en mentionnons cinq.

[2]Nos contributions les plus originales sont signalées par une pique dans la marge.

1.3.2.1 Le calcul de l'homographie liant deux images « sans contrainte » de recouvrement

Une méthode d'échantillonnage aléatoire, LTTS, qui combine LTS et TLS améliore dans tous les cas l'estimation des homographies, ce qui permet la génération de **mosaïques** précises et universelles.

1.3.2.2 Le suivi d'un motif plan

À partir des fondements posés par Hager et Belhumeur, Jurie et Dhome, nous contrôlons la qualité de la superposition des points d'intérêt suivis (en vue d'estimer à nouveau une homographie) par minimisation d'un critère aux moindres carrés non linéaires : les résultats « se passeraient de commentaires ».

1.3.2.3 Le suivi d'un motif plan en présence d'objets occultants

Cette extension du suivi « ordinaire » repose sur la mise à jour du critère aux moindres carrés en y supprimant la contribution des points d'intérêt invisibles (masque d'occultation) : facile à faire sur les critères de Hager et Belhumeur (HB) et non linéaire (NL), mais jamais envisagé dans le cas de Jurie et Dhome (JD) ! Nous y remédions avec un algorithme hybride basé sur les techniques connues des spécialistes des moindres carrés linéaires sous les noms d'actualisation positive et/ou négative (*updating/downdating* en anglais).

1.3.2.4 Le suivi dans une image panoramique

Il s'agit de répondre à la question : « où se trouve l'image courante de la caméra DV (qui filme la scène) au sein de l'image panoramique de référence ? » Notre réponse est une adaptation du suivi ordinaire capable de traiter séparément des tuiles qui se chevauchent dans l'image panoramique. Un effet de bord (voulu) est la détection des objets ajoutés ou retirés. Ce procédé fait également gagner en robustesse.

1.3.2.5 La détection/extraction d'« intrus » dans une scène

Un double alignement, géométrique (cf. paragraphe précédent) et chromatique de l'image courante de la caméra DV et de l'image panoramique de référence ouvre le chemin à la détection (extraction si besoin est) des objets ajoutés ou retirés (intrus) par « soustraction ». Nous prenons la précaution de lisser le résultat en appliquant des opérateurs morphologiques bien choisis.

1.3.3 Plan du document

Cette thèse aborde le problème du calcul de l'homographie liant deux images dans les deux cas de figure que sont :

– la génération d'**Images Panoramiques** ;

– le **Suivi** d'un motif plan.

Parmi les nombreuses applications qui s'y rattachent, l'accent est plus particulièrement mis sur la **Réalité Augmentée**.

Le chapitre qui suit expose les différentes techniques d'**estimation de paramètres** omniprésentes du début à la fin de ce travail. Un tour d'horizon des principales méthodes d'estimation est fait : il se focalise sur les travaux menés en programmation mathématique (au sens le plus large) et en extrait (voire les « taille sur mesure ») les acquis les plus tangibles et les plus adaptés à la résolution de nos modèles sous-jacents aux problèmes de géométrie en vision par ordinateur.

Le chapitre 3 concerne la tâche (1) de la figure 1.1 : la génération de l'image de référence appellée **image panoramique**. Deux voies différentes permettent d'aboutir :

– l'utilisation de capteurs dédiés (caméra à balayage angulaire et caméra catadioptrique) conçus et réalisés en collaboration avec des collègues de laboratoire ;
– la combinaison de plusieurs images issues d'une caméra ordinaire pour former ce qui est connu dans la communauté sous le nom de **mosaïque**. Le calcul d'**homographies** entre paires d'images est un passage obligé.

Le chapitre 4 est tout entier consacré aux méthodes de **suivi d'un motif plan**. Deux situations correspondant au cas réel sont abordées :

– le cas le plus fréquent et le plus étudié où aucun effet de masquage ne se produit ;
– le cas où un objet opaque s'interpose entre le capteur et le motif suivi, provoquant ainsi une occultation. Il s'agit alors de garantir la fiabilité du suivi malgré les informations parasites.

Le chapitre 5 est celui des **applications**. L'accent est plus particulièrement mis sur quatre d'entre elles :

– la superposition (en temps réel ou non) d'une vidéo à une image panoramique de référence ;
– la détection, sur cette vidéo, d'objets (au sens le plus large) s'interposant au premier plan de **LA** scène située en arrière-plan ;
– le **codage vidéo** pour ce qui concerne la séparation fond $\leftrightarrow$ forme (ou arrière-plan $\leftrightarrow$ premier plan) ;
– la **réalité augmentée** sous les deux formes qui donnent son nom à cette application, à savoir l'insertion et la suppression d'objets de synthèse (ou pas) dans une vidéo.

En conclusion générale, sont synthétisées les contributions conceptuelles et pratiques significatives que nous avons apportées. Nous terminons en ouvrant des pistes en vue d'une exploitation plus exhaustive de la richesse intrinsèque des homographies inter-images.

Le chapitre 7 regroupe six appendices. Nous y détaillons divers points techniques : l'approximation d'une rotation par une translation, l'échantillonnage des images numériques, l'étalonnage d'une caméra à focale variable, l'utilisation d'un processeur dédié au traitement d'images, un algorithme de multiplication matricielle rapide. Un recensement les notations utilisées dans ce document y occupe la dernière place.

Chapitre 2

Estimation de paramètres

Ce chapitre traite de l'estimation de paramètres caractéristiques de systèmes[1] dont les entrées et les sorties sont mesurables. Nous nous plaçons dans le cas où :

– toutes ces quantités varient dans $\mathbb{R}$;

– le modèle mathématique du système est régulier (de classe $\mathcal{C}^2$ sur $\mathbb{R}$ au pire).

Nous insistons sur la nécessité de prendre en compte le type des erreurs rencontrées dans les mesures, ce qui permet d'atteindre l'objectif de robustesse. Nous examinons deux de ces types d'erreurs.

Nous présentons les méthodes les plus utilisées en vision par ordinateur en détaillant les améliorations et adaptations que nous y avons apportées.

2.1 État de l'art

Ce chapitre est en quelque sorte le cinquième « fruit » d'une démarche systématique, traduite en une méthode de travail, qui a été initialisée bientôt vingt ans en arrière au sein de l'équipe VPCAB de l'ENSEEIHT. La démarche en question consiste tout simplement à avoir comme premier réflexe de « faire appel » aux spécialistes (hommes de l'art) de programmation mathématique chaque fois qu'il s'agit de résoudre un nouveau modèle mathématique en vision par ordinateur.

Rappelons que les quatre autres retombées de cette démarche ont été présentées successivement dans les thèses de :

– Arslane Abi-Ayad ([AA89], annexe 2 : Outils Mathématiques d'Estimation de Paramètres) ;

– Fernando Martinez ([Mar93], § 4.6 : Minimisation d'un problème d'optimisation combinatoire) ;

– Yann Yvinec ([Yvi96], chapitre 5 : Méthodes de résolution) ;

– Vincent Charvillat ([Cha97], partie II : Méthodes de régression robuste dédiées à la vision) ;

[1] Système doit être compris comme « appareil ou dispositif formé d'éléments agencés et assurant une fonction déterminée » ([Lar02] p984)

L'« appel aux Spécialistes » se décline au quotidien de deux manières :
- fouiller dans la littérature appropriée ;
- soumettre (si besoin est) le problème concret, avec des données réelles, aux leaders dans le domaine concerné.

Voici une liste d'ouvrages, classés par thèmes, sur lesquels s'appuie directement ou indirectement notre étude :

Algèbre linéaire : [Var62], [Hou74], [GL91], [HJ85] ;

Optimisation, théorie, algorithmes, codes : [Lue69], [Cia85], [DS83], [Fle00], [GMW81] ;

Régression non linéaire avec le support des probabilités-statistiques : [Rat83], [Ful87], [SW89], [BW88], [ABC92] ;

Moindres carrés linéaires : [LH74], [Bjö96] ;

Moindres carrés totaux : [HV91], [Huf97], [SVH02] ;

Régression robuste : [Hub81], [RL87], [CW82], [CR88] ;

Optimisation combinatoire : [GL87], [LR03], [Gol89]

Parutions en vision par ordinateur : [Kan95], [Kan96], [HZ03], [Zha97], [Mee04].

Quant aux « collaborateurs », c'est l'occasion pour nous de remercier au nom du groupe VPCAB –au risque d'en oublier– tous ceux qui ont pris sur leur temps et nous ont permis (directement ou indirectement) de mieux aborder la résolution de nos modèles mathématiques. Nous voulons nommer, par ordre alphabétique :

> Stefan Van Aelst, Jesse Barlow, Åke Björck, Richard Byrd, Yeh-Ling Chen, Christophe Croux, Katrien Van Driessen, Lars Elden, Wayne Fuller, Fred Glover, Gene Golub, Mia Hubert, Sabine Van Huffel, Yvan Markowsky, Cleve Moler, Jorge Moré, Heinz Muhlenbein, Haesun Park, Janet Rogers, Peter Rousseeuw, Robert Schnabel, Arnold Stromberg et Alistair Watson.

2.2 Introduction – notations

Soit un système physique qui se prête à des mesures sur deux types de grandeurs à valeurs dans $\mathbb{R}$:
- les **covariables** ou « **régresseurs** » ou variables **explicatives** ou **indépendantes** ou **en entrée**. On peut les choisir (fixer) et le système réagit à leur valeur ;
- les **réponses** ou variables **expliquées** ou **dépendantes** ou **en sortie**. Le système impose leur valeur, mais on peut les mesurer.

Au système physique est associé un modèle mathématique paramétré.

Il s'agit de déterminer la valeur des **paramètres** à partir de mesures des variables –**observations**– réalisées au cours de plusieurs (n) expériences.

Formellement, nous notons :
- les variables en entrée $x \in \mathbb{R}^m$;
- les variables en sortie $y \in \mathbb{R}^q$;

– les paramètres $\beta \in \mathbb{R}^p$;
– le modèle

$$\begin{array}{rccc} f: & \mathbb{R}^m \times \mathbb{R}^p & \longrightarrow & \mathbb{R}^q \\ & (x, \beta) & \longmapsto & y \end{array}$$

Nous rassemblons toutes les informations utiles à la modélisation d'un système sous la forme synthétique graphique de la **boîte noire** :

$$x \in \mathbb{R}^m \rightarrow \boxed{\begin{array}{c} \beta \in \mathbb{R}^p \\ \hline m, n, p, q \end{array}} \rightarrow y = f(x, \beta) \in \mathbb{R}^q$$

À cette boîte noire, nous adjoignons le résultat des mesures issues de chaque expérience. Pour l'expérience i ($i \in [\![1, n]\!]$) ce sont :
– la valeur $\widetilde{x_i}$ de la variable x ;
– la valeur $\widetilde{y_i}$ de la variable y.

Remarque : ce procédé de calcul des β, faisant intervenir n observations des variables x et y est connu sous le nom d'**estimation de paramètres** par méthode indirecte.

2.3 Types de modèles de régression

Seuls les modèles nécessaires au traitement des problèmes de vision sous-jacents à nos objectifs sont mentionnés. On les appelle modèles de **régression** au sens où ils permettent une « réduction des données d'un phénomène complexe en vue de les représenter par une loi simplificatrice » ([Rob03] p2220).

2.3.1 Modèle linéaire

Dire que le modèle est linéaire est un abus de langage pour désigner le cas où f est affine en β. Il peut s'écrire sous la forme

$$f(x, \beta) = C(x)\beta + D(x)$$

avec

$$C : \mathbb{R}^m \longrightarrow \mathbb{R}^{q \times p}$$
$$D : \mathbb{R}^m \longrightarrow \mathbb{R}^q$$

2.3.2 Modèle bilinéaire

On appelle modèle **bilinéaire** un cas particulier du précédent pour lequel C et D sont aussi affines. Le modèle est en fait une fonction « bi-affine ».

2.3.3 Modèle non linéaire de classe $\mathcal{C}^2$ sur $\mathbb{R}$

La fonction f est quelconque, mais reste deux fois continûment dérivable en β et en x.

2.4 Types d'erreurs

La mesure des variables est sujette à des erreurs dont le processus d'estimation doit tenir compte. Ces erreurs peuvent apparaître sur les entrées et/ou les sorties.

Dans le domaine de la vision par calculateur ([Mee04], § 4.4), on se restreint le plus souvent à deux types d'erreurs : les erreurs additives gaussiennes et les aberrations.

2.4.1 Erreurs dans les sorties et les entrées

Si on peut fixer précisément les variables explicatives, alors les seules erreurs sont sur les mesures des variables expliquées. Dans le cas contraire (erreurs sur les mesures en entrée et en sortie), on parle de modèle EIV (*errors in variables* en anglais).

Notations ($i \in [\![1, n]\!]$ est l'indice de l'expérience) :

– erreurs sur les entrées : $\delta_i = [\delta_{i1} \ \cdots \ \delta_{im}]^\top$ perturbe la valeur exacte x_i

$$\widetilde{x_i} = x_i + \delta_i$$

– erreurs sur les sorties : $\varepsilon_i = [\varepsilon_{i1} \ \cdots \ \varepsilon_{iq}]^\top$ perturbe la valeur exacte y_i

$$\widetilde{y_i} = y_i + \varepsilon_i$$

2.4.2 Erreurs gaussiennes faibles

Dans ce modèle, les erreurs tant en entrée qu'en sortie sont additives, aléatoires et suivent des lois normales multidimensionnelles. Dans tous les cas, deux hypothèses d'indépendance (au sens des variables aléatoires qui les modélisent) sous-tendent ces modèles d'erreur :

– indépendance d'une expérience à l'autre ;
– indépendance entre toute erreur en entrée et toute erreur en sortie.

Sous ces hypothèses on distingue deux modèles :

– celui où

$$(\delta_i) \sim \mathcal{N}(0_m, \Sigma_i)$$

$$(\varepsilon_i) \sim \mathcal{N}(0_q, \Lambda_i)$$

$\Sigma_i \in \mathbb{R}^{m\times m}$ et $\Lambda_i \in \mathbb{R}^{q\times q}$ sont alors des matrices de variance-covariance (symétriques semi-définies positives) ;

– celui où $\Sigma_i = \mathrm{diag}(\sigma_{i1}, ..., \sigma_{im})$ et $\Lambda_i = \mathrm{diag}(\lambda_{i1}, ..., \lambda_{iq})$, c'est-à-dire que les erreurs ne sont pas corrélées d'une composante à l'autre des vecteurs.

Avoir des matrices de variance-covariance différentes selon les expériences implique qu'elles ne sont pas toutes équivalentes, par exemple parce que certaines sont moins « dignes de confiance » que d'autres. Dans notre cas, nous supposons que toutes les expériences sont équivalentes :

$$\Sigma_1 = \cdots = \Sigma_n = \Sigma \qquad \text{et} \qquad \Lambda_1 = \cdots = \Lambda_n = \Lambda$$

Les erreurs gaussiennes apparaissent souvent quand des facteurs multiples incontrôlés s'accumulent pour perturber faiblement la mesure.

2.4.3 Observations aberrantes

Si le système est un maillon d'une chaîne de traitements, il dépend des résultats des traitements précédents. Pour certaines expériences, les mesures utilisées pour l'estimation ne correspondent pas au modèle du système : ce sont des **aberrations** ou **aberrances** (*outlier* en anglais).

Cela arrive dans deux cas au moins (nous excluons le cas de mélange inopiné et malencontreux de mesures au cours de la collecte) :

- un prétraitement **discret** échoue et délivre des résultats dénués de tout sens ;
- les mesures correspondent à plusieurs (≥ 2) occurrences du même modèle paramétré (par exemple $2, 3, ..., d$ (inconnu) droites dans une image de contours).

Il s'agit alors de détecter les expériences contaminées et de ne pas en tenir compte.

L'intensité de l'erreur est caractérisée par la proportion (ε) d'expériences aberrantes.

Des observations sans aberration peuvent néanmoins être entachées d'autres types d'erreurs (par exemple additives gaussiennes).

2.5 Estimateurs et critères d'optimisation

L'estimation nécessite un nombre minimal d'expériences conduisant à un modèle mathématique associé « surdéterminé » ; par exemple $nq > p$ dans le cas linéaire.

À l'opposé, à amplitude d'erreur « constante », un plus grand nombre d'expériences permet de mieux résister aux erreurs.

À cause des erreurs et de la « surdétermination », l'égalité entre les sorties du modèle et celles du système ne peut être vérifiée qu'approximativement (on parle d'**écart**), même pour le vecteur de paramètres $\bar{\beta}$ exact, à savoir

$$\left\{ \begin{array}{rcl} y_i & = & f(x_i, \bar{\beta}) \\ \widetilde{y_i} - \varepsilon_i & = & f(\widetilde{x_i} - \delta_i, \bar{\beta}) \\ \widetilde{y_i} & \approx & f(\widetilde{x_i}, \bar{\beta}) \\ \widetilde{y_i} - \varepsilon_i & \approx & f(\widetilde{x_i} - \delta_i, \widehat{\beta}) \end{array} \right. \tag{2.1}$$

où $\widehat{\beta}$ est une estimation du vecteur de paramètres.

C'est à ce niveau que l'estimation par méthode indirecte intervient au travers d'un critère qui quantifie cet écart. Par suite, l'estimation $\widehat{\beta}$ retenue sera la solution d'un problème d'optimisation induit.

Le critère se présente sous la forme d'une fonction des mesures en entrée et en sortie ainsi que des paramètres :

$$\begin{array}{rccccc} K: & \mathbb{R}^{m\times n} & \times & \mathbb{R}^{q\times n} & \times\mathbb{R}^{p} & \longrightarrow & \mathbb{R}^{+} \\ & [x_1\ \cdots\ x_n]\,, & & [y_1\ \cdots\ y_n]\,, & \beta & \longmapsto & K([x_1\ \cdots\ x_n],[y_1\ \cdots\ y_n],\beta) \end{array}$$

si le critère est à minimiser, l'estimateur correspondant est :

$$\mathcal{E}\left(\begin{bmatrix}\widetilde{x_1} & \cdots & \widetilde{x_n}\\ \widetilde{y_1} & \cdots & \widetilde{y_n}\end{bmatrix}\right) = \underset{\beta}{\operatorname{argmin}}\ K([\widetilde{x_1}\ \cdots\ \widetilde{x_n}],[\widetilde{y_1}\ \cdots\ \widetilde{y_n}],\beta)$$

Les résidus. La plupart des critères que nous examinons par la suite sont construits comme une combinaison de **résidus**. Ils ont la forme suivante :

$$K([x_1\ \cdots\ x_n],[y_1\ \cdots\ y_n],\beta) = \underset{i\in[\![1,n]\!]}{\operatorname{comb}}\ K_{\mathrm{u}}(x_i,y_i,\beta)$$

où

- $\operatorname{comb}:\mathbb{R}^n\to\mathbb{R}$ est un opérateur de combinaison commutatif (le critère est indépendant de l'ordre des expériences), par exemple $\sum$ ou max ;
- $K_{\mathrm{u}}:\mathbb{R}^m\times\mathbb{R}^q\times\mathbb{R}^p\to\mathbb{R}^+$ est la fonction qui calcule le résidu pour une expérience donnée.

2.5.1 Propriétés des estimateurs

L'estimateur (et donc le critère) doit être adapté au type d'erreurs à traiter. Quand l'erreur est modélisée par une variable aléatoire, cette adaptation est déterminée par des propriétés statistiques.

2.5.1.1 Absence de biais

Soit un vecteur de paramètres β. Soit une variable aléatoire matricielle Ξ dont ξ est une réalisation. La matrice ξ représente les entrées et les sorties d'une série d'expériences sur un système paramétré par β :

$$\xi = \begin{bmatrix}\widetilde{x_1} & \cdots & \widetilde{x_n}\\ \widetilde{y_1} & \cdots & \widetilde{y_n}\end{bmatrix} \in \mathbb{R}^{(m+q)\times n} \tag{2.2}$$

L'estimateur $\mathcal{E}$ est **sans biais** ([Mee01] p16) si

$$E[\mathcal{E}(\Xi)] = \beta$$

C'est-à-dire qu'en moyenne sur un grand nombre de séries d'expériences, l'estimateur permet de trouver la bonne valeur des paramètres.

2.5.1.2 Maximum de vraisemblance

Soit une série d'expériences d'entrées $\widetilde{x_1}, ..., \widetilde{x_n}$ et de sorties $\widetilde{y_1}, ..., \widetilde{y_n}$. Pour un β donné, notons $P(\xi/\beta)$ la probabilité de ξ sachant β, où ξ est la matrice regroupant les mesures à la manière de l'équation (2.2). L'estimateur $\mathcal{E}$ atteint le **maximum de vraisemblance** ([DHS01] p86) si

$$\mathcal{E}(\xi) = \underset{\beta \in \mathbb{R}^p}{\operatorname{argmax}} \; P(\xi/\beta)$$

C'est-à-dire que, pour une série d'expériences donnée, l'estimateur trouve la valeur la plus probable des paramètres.

2.5.1.3 Robustesse

En présence d'aberrances, un estimateur est dit **robuste à une contamination** ε, si en modifiant arbitrairement une proportion ε d'expériences, il n'est pas possible d'accroître arbitrairement une composante du vecteur de paramètres β résultant. Il est dit **robuste** s'il est robuste à une contamination $\varepsilon > 0$.

2.5.2 Moindres carrés

Ces méthodes cherchent à rendre « aussi vraie que possible » l'approximation 2.1. Pour cela, elles calculent β qui minimise « conjointement »

$$\|\widetilde{y_i} - f(\widetilde{x_i}, \beta)\|^2 \qquad \forall i \in [\![1, n]\!]$$

Cette expression basique des résidus (§ 2.5.2.1) sera modifiée pour intégrer une pondération des mesures et prendre en compte les erreurs sur les entrées (§ 2.5.2.2, 2.5.2.3, 2.5.3).

Dans le cadre des moindres carrés, « conjointement » se traduit par : comb $= \sum$. Ceci permet de se replacer dans un contexte d'optimisation classique : les équations issues des différentes expériences sont traitées de manière identique.

Aucun des estimateurs présentés dans cette sous-section n'est robuste, car ils sont tous susceptibles de diverger quand une seule donnée prend une valeur arbitrairement grande.

2.5.2.1 Moindres carrés ordinaires (*Ordinary Least Squares*, OLS)

On parle aussi de « régression en distance verticale » (par analogie avec le cas 2D, figure 2.1). L'estimateur est :

$$\widehat{\beta_{\text{OLS}}} = \underset{\beta \in \mathbb{R}^p}{\operatorname{argmin}} \sum_{i=1}^{n} \|W(\widetilde{y_i} - f(\widetilde{x_i}, \beta))\|^2 \tag{2.3}$$

où $W \in \mathbb{R}^{q \times q}$ est une matrice de poids qui permet de compenser les différences d'ordre de grandeur des composantes des vecteurs.

Si :

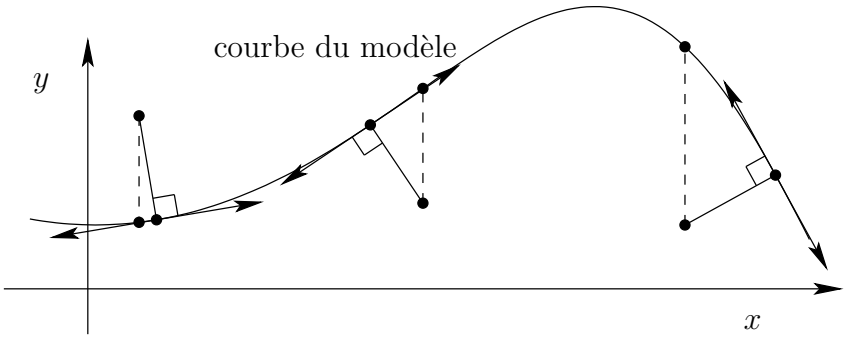


FIG. 2.1: Régression en 2D, $(q, m) = (1, 1)$. La régression consiste à rapprocher la courbe des points de données, c'est-à-dire réduire la somme des carrés des distances entre la courbe et les points. Dans le cas de la **distance verticale**, ce sont les longueurs des segments verticaux (en pointillés) qui sont prises en compte. Dans le cas de la **distance orthogonale** ce sont les segments (en trait plein) orthogonaux à la courbe.

- les sorties sont affectées d'erreurs additives gaussiennes ;
- les entrées sont sans erreur ;
- la matrice $W^2 = \Lambda^+$,

alors l'estimateur $\widehat{\beta_{\text{OLS}}}$ atteint le maximum de vraisemblance et il est sans biais.

Dans ce cas, le résidu est donné par $K_{\text{u}}(x, y, \beta) = \|W(y - f(x, \beta))\|^2$.

2.5.2.2 Moindres carrés orthogonaux (*Orthogonal Distance Regression*, ODR)

Les entrées et les sorties sont entachées d'erreurs. Il s'agit d'estimer « conjointement » β et les erreurs en entrée δ_i :

$$P_{\text{ODR}} \left\{ \begin{array}{l} \min \sum_{i=1}^{n} \|W(\widetilde{y_i} - f(\widetilde{x_i} + \delta_i, \beta))\|^2 + \|\Omega\delta_i\|^2 \\ \beta \in \mathbb{R}^p, \delta_i \in \mathbb{R}^m, i \in [\![1, n]\!] \end{array} \right. \tag{2.4}$$

où $W \in \mathbb{R}^{q\times q}$ et $\Omega \in \mathbb{R}^{m\times m}$ sont des matrices de poids.

On parle de « régression en distance orthogonale » (*Orthogonal Distance Regression*, ODR) par analogie avec le cas 2D (figure 2.1).

Si les erreurs sur les entrées et les sorties sont additives gaussiennes et si $W^2 = \Lambda^+$ et $\Omega^2 = \Sigma^+$, alors l'estimateur ODR atteint le maximum de vraisemblance.

Dans ce cas,

$$\begin{aligned} K([x_1 \ \cdots \ x_n], [y_1 \ \cdots \ y_n], \beta) &= \min_{\delta_i} \sum_{i=1}^{n} \|W(y_i - f(x_i + \delta_i, \beta))\|^2 + \|\Omega\delta_i\|^2 \\ &= \sum_{i=1}^{n} \min_{\delta} \|W(y_i - f(x_i + \delta, \beta))\|^2 + \|\Omega\delta\|^2 \end{aligned}$$

donc l'expression du résidu est

$$K_{\text{u}}(x, y, \beta) = \min_{\delta\in\mathbb{R}^m} \|W(y - f(x + \delta, \beta))\|^2 + \|\Omega\delta\|^2$$

2.5.2.3 Moindres carrés sur les données (*Data Least Squares*, DLS)

Si seules les entrées sont affectées, c'est un problème de moindres carrés sur les données. C'est un cas particulier du précédent où l'égalité entre les sorties mesurées et celles du modèle corrigé apparaît en contrainte :

$$P_{\mathrm{DLS}} \begin{cases} \min \sum_{i=1}^{n} \|\Omega\delta_i\|^2 \\ \beta \in \mathbb{R}^p, \delta_i \in \mathbb{R}^m, i \in [\![1, n]\!] \\ \widetilde{y}_i = f(\widetilde{x}_i + \delta_i, \beta) \end{cases} \tag{2.5}$$

où $\Omega \in \mathbb{R}^{m\times m}$ est une matrice de poids.

Le résidu $K_{\mathrm{u}}(x, y, \beta)$ est le minimum du problème

$$\begin{cases} \min \|\Omega\delta\|^2 \\ \delta \in \mathbb{R}^m \\ y = f(x + \delta, \beta) \end{cases}$$

2.5.3 Critères modérateurs (M-estimateurs)

Cette famille de critères est une généralisation de OLS.

$$\widehat{\beta_{\mathrm{ME}}} = \underset{\beta}{\operatorname{argmin}} \sum_i \rho(\|W(\widetilde{y}_i - f(\widetilde{x}_i, \beta))\|)$$

où

- la fonction de modération $\rho : \mathbb{R}^+ \longrightarrow \mathbb{R}^+$, est dérivable, croissante et $\rho(0) = 0$.
- $W \in \mathbb{R}^{q\times q}$ est une matrice de poids.

Le résidu pour une expérience est donné par

$$K_{\mathrm{u}}(x, y, \beta) = \rho(\|W(y - f(x, \beta))\|)$$

Une fonction ρ qui croît moins vite que le carré réduit l'influence sur le critère des expériences aberrantes. Le critère n'est pas en relation directe avec un type d'erreurs.

Citons quelques exemples de fonctions de modération.

- Dans le cas OLS, $\rho(x) = x^2$;
- Dans le cas LAD (*Least Absolute Deviation*), $\rho(x) = x$;
- Dans le cas de fonctions ρ bornées, une distinction est faite entre :
  - celles qui n'atteignent pas leur borne (*soft redescender*), à savoir

$$\lim_{x\to\infty} \rho(x) = v > 0$$

Un exemple typique est ([BJ98] § 4) :

$$\rho(x) = \frac{x^2}{\sigma^2 + x^2} \tag{2.6}$$

où σ détermine à partir de quelle amplitude une mesure est considérée comme aberrante (figure 2.2(a)) ;

– celles qui atteignent leurs bornes (*hard redescenders*) à savoir

$$\forall x > x_0, \rho(x) = v$$

Un exemple typique est la fonction de Tukey ([RL87], eq4.31) :

$$\rho(x) = \begin{cases} \frac{x^2}{2} - \frac{x^4}{2\sigma^2} + \frac{x^6}{6\sigma^4} & \text{si } x < \sigma \\ \frac{\sigma^2}{6} & \text{sinon} \end{cases} \tag{2.7}$$

σ où est un facteur d'échelle (figure 2.2(b)).

Remarque : Dans ces deux derniers cas, l'estimateur est robuste. En effet, l'influence d'une expérience donnée sur le critère est limitée à la borne v.

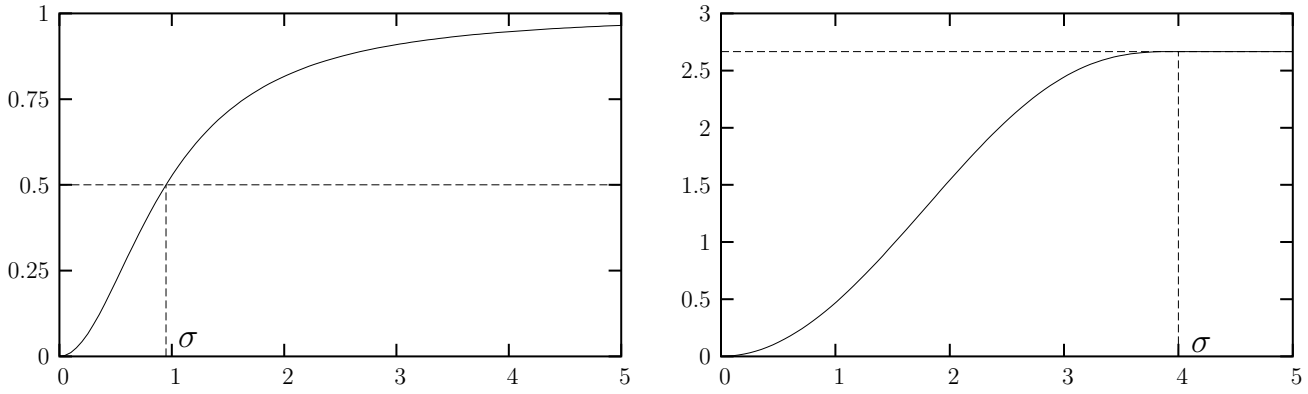


FIG. 2.2: Fonctions de modération ρ correspondant aux équations (2.6) et (2.7). Le coefficient σ agit comme un facteur d'échelle en abscisse.

2.5.4 Critères sélectifs

Les critères sélectifs cherchent à ignorer les expériences comportant des mesures aberrantes.

Un critère de minimisation K basé sur h (suffisamment grand mais pas nécessairement minimal) expériences :

– prend des valeurs faibles sur les échantillons non contaminés (sans aberrance) ;

– prend des valeurs élevées sur les échantillons contaminés.

Le critère sélectif coïncide avec le critère non robuste appliqué à un échantillon de h expériences. Le problème d'optimisation consiste à choisir l'échantillon et les paramètres qui minimisent ce critère, soit

$$P_{\text{sel}} \begin{cases} \min K([\widetilde{x_{i_1}} \ \cdots \ \widetilde{x_{i_h}}], [\widetilde{y_{i_1}} \ \cdots \ \widetilde{y_{i_h}}], \beta) \\ I = \{i_1, ..., i_h\} \in S_n^h, \beta \in \mathbb{R}^p \end{cases} \tag{2.8}$$

où S_n^h est l'ensemble des sous-ensembles de $[\![1, n]\!]$ de taille h.

Si le taux d'expériences aberrantes est ε et si $h < (1-\varepsilon)n$, alors l'estimateur est robuste.

On parle d'ajustement **sournois** quand des expériences aberrantes sont cohérentes entre elles et correspondent à un modèle dont les paramètres sont $\beta' \neq \beta$. Si plus de la moitié des expériences constituent un ajustement sournois (ce qui implique $\varepsilon > 1/2$), les estimateurs sélectifs renvoient β' au lieu de β.

Dans la suite nous présentons quelques estimateurs sélectifs classiques.

2.5.4.1 Moindre quantile des carrés (*Least Quantile of Squares*, LQS)

Cet estimateur dépend d'un coefficient $\alpha \in]0, 1]$ de sorte que $h = \lfloor \alpha n \rfloor$. L'opération de combinaison est comb = max et

$$K_{\mathrm{u}}(x, y, \beta) = \|y - f(x, \beta)\|$$

Il minimise donc le plus grand résidu sur une proportion α d'expériences choisie au mieux, les autres pouvant devenir arbitrairement grands.

2.5.4.2 Moindres carrés médians (*Least Median of Squares*, LMS)

Cet estimateur est un cas particulier du précédent où $\alpha = \frac{1}{2}$. Il minimise donc la médiane de la norme de tous les résidus.

2.5.4.3 Moindres carrés tronqués (*Least Trimmed Squares*, LTS)

Ici, $K_{\mathrm{u}}(x, y, \beta) = \|y - f(x, \beta)\|^2$, comb = $\sum$. C'est l'estimateur des moindres carrés ordinaires appliqué à un sous-ensemble d'expériences.

Si le taux de contamination ε est inconnu, l'estimateur robuste capable de résister au plus grand nombre d'ajustements sournois correspond à $h = \lfloor (n + p + 1)/2 \rfloor$.

2.6 Résolution

Les méthodes de résolution se classent en deux grandes familles :

les méthodes analytiques. Le problème peut se ramener à un nombre fini de calculs, la précision du résultat dépend de la précision de l'unité flottante qui les effectue. Dans le cadre de l'estimation, les problèmes « suffisamment linéaires » peuvent se ramener à des calculs matriciels ;

les méthodes itératives. Elles engendrent une suite de valeurs du paramètre inconnu, de manière à converger vers un minimum. La précision du calcul dépend en plus du nombre d'itérations pendant lesquelles l'algorithme « tourne ».

En général, quand une méthode analytique et une itérative existent pour résoudre un problème, la méthode analytique est préférable car elle est plus performante et ne pose pas de problème de convergence.

Les bibliothèques numériques auxquelles nous faisons référence (BLAS, LAPACK, MINPACK, ODRPACK, VANHUFFEL) sont disponibles dans la collection Netlib (`http://netlib.enseeiht.fr`)

2.6.1 Méthodes analytiques pour le cas linéaire unidimensionnel

Ces techniques permettent de faire l'estimation dans le cas de modèles linéaires ou bilinéaires et de critères moindres carrés.

2.6.1.1 Moindres carrés ordinaires

Si f est linéaire en β, le problème OLS (équation (2.3)) se ramène à :

$$\widehat{\beta_{\text{OLS}}} = \underset{\beta \in \mathbb{R}^p}{\operatorname{argmin}} \|A\beta - b\|^2 \tag{2.9}$$

où

$$A = \begin{bmatrix} WC(\widetilde{x_1}) \\ \vdots \\ WC(\widetilde{x_n}) \end{bmatrix} \in \mathbb{R}^{nq \times p} \text{ et } b = \begin{bmatrix} W(\widetilde{y_1} - D(\widetilde{x_1})) \\ \vdots \\ W(\widetilde{y_n} - D(\widetilde{x_n})) \end{bmatrix} \in \mathbb{R}^{nq}$$

Une méthode de résolution utilise la décomposition QR de A : $A = QR$ avec $Q \in \mathbb{R}^{nq \times nq}$ orthogonale et $R \in \mathbb{R}^{nq \times p}$ triangulaire supérieure. En décomposant les deux matrices :

$$R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \quad Q = \underset{p}{[Q_1} \quad \underset{nq-p}{Q_2]}$$

alors :

$$\|A\beta - b\|^2 = \|R\beta - Q^\top b\|^2 = \|Q_2^\top b\|^2 + \|R_1\beta - Q_1^\top b\|^2$$

Le premier membre est indépendant de β : on ne peut pas jouer dessus. Le second membre peut être annulé en résolvant en β le système triangulaire

$$R_1\beta = Q_1^\top b$$

Ceci montre qu'une relation linéaire lie $\widehat{\beta_{\text{OLS}}}$ et b si R_1 (donc A) est de rang plein. La matrice de cette relation est la pseudo-inverse de A :

$$A^+ = R_1^{-1} Q_1^\top$$

Cette technique de résolution est utilisée dans la fonction `dgels` de LAPACK.

2.6.1.2 Moindres carrés totaux

Le problème ODR (équation 2.4) admet une solution analytique dans le cadre des moindres carrés totaux (*Total Least Squares*, TLS, [HV91] § 2.2.3). C'est un cas restreint où :

- les dimensions vérifient $m = p$ et $q = 1$;
- le modèle f est bilinéaire de la forme

$$f(x, \beta) = x^\top \beta + q_0^\top \beta + q_1$$

  avec $q_0 \in \mathbb{R}^p$ et $q_1 \in \mathbb{R}$;
- la matrice de poids $\Omega = \omega \mathrm{Id}_m$.

Notons $W = [w]$ et rappelons l'équation 2.1 :

$$f(\widetilde{x}_i - \delta_i, \bar{\beta}) = \widetilde{y}_i - \varepsilon_i \qquad \forall i \in [\![1, n]\!]$$

En exploitant les linéarités et en synthétisant les égalités sosu forme matricielle :

$$\Bigg(\underbrace{\begin{bmatrix} \widetilde{x}_1^\top + q_0^\top \\ \vdots \\ \widetilde{x_n}^\top + q_0^\top \end{bmatrix}}_{A} + \underbrace{\begin{bmatrix} -\delta_1^\top \\ \vdots \\ -\delta_n^\top \end{bmatrix}}_{E} \Bigg) \bar{\beta} = \underbrace{\begin{bmatrix} \widetilde{y}_1 - q_1 \\ \vdots \\ \widetilde{y}_n - q_1 \end{bmatrix}}_{b} + \underbrace{\begin{bmatrix} -\varepsilon_1 \\ \vdots \\ -\varepsilon_n \end{bmatrix}}_{r}$$

Par suite, l'équation (2.4) s'écrit :

$$P_{\mathrm{TLS}} \begin{cases} \min \|[\omega E \quad wr]\|^2 \\ \beta \in \mathbb{R}^p, E \in \mathbb{R}^{n \times p}, r \in \mathbb{R}^n \\ (A + E)\beta = b + r \end{cases}$$

La résolution de (P_{TLS}) se base sur la décomposition en valeurs singulières (SVD) de $[A \quad b]$ (plus de détails dans § 2.6.2.2).

2.6.1.3 Moindres carrés sur les données

Sous les mêmes hypothèses que précédemment sur la forme du modèle et sur celle de la matrice de poids l'estimateur DLS (équation (2.5)) se ramène à :

$$P_{\mathrm{DLS}} \begin{cases} \min \|E\|^2 \\ \beta \in \mathbb{R}^p, E \in \mathbb{R}^{n \times p} \\ (A + E)\beta = b \end{cases}$$

Nous examinons sa résolution dans le § 2.6.2.3.

2.6.2 Méthodes analytiques pour le cas linéaire multidimensionnel

Les méthodes multidimensionnelles sont des extensions des méthodes de moindres carrés OLS, TLS et DLS linéaires. La résolution numérique se fait simultanément pour plusieurs seconds membres, chacun donnant un vecteur de paramètres estimés. D'une manière générale, plusieurs équations de la forme :

$$A\beta_i = b_i, \qquad i \in [\![1, k]\!]$$

où $A \in \mathbb{R}^{nq \times p}$ et $b_i \in \mathbb{R}^{nq}$, sont rassemblées en :

$$A[\beta_1 \cdots \beta_k] = [b_1 \cdots b_k]$$

Par rapport au problème « unidimensionnel » :
- le cas unidimensionnel est le cas particulier du multidimensionnel où $k = 1$;
- un problème multidimensionnel TLS et DLS n'est pas équivalent à k problèmes unidimensionnels où les vecteurs β_i sont estimés un par un. En effet, les erreurs estimées sur la matrice A des entrées sont communes.

Nous rencontrons ce genre de problème quand nous voulons estimer les colonnes d'une matrice. Dans cette section, nous notons :

$$N = nq \qquad X = [\beta_1 \cdots \beta_k] \in \mathbb{R}^{p \times k} \qquad B = [b_1 \cdots b_k] \in \mathbb{R}^{N \times k}$$

Il s'agit donc d'estimer X tel que :

$$AX \approx B$$

Notre contexte. Nous détaillons la résolution numérique des problèmes dans le cas où :
- la taille des matrices vérifie $N \gg p \gg k$;
- la matrice A est de rang plein ;
- la matrice A est composée d'entiers codés sur 1 (un) octet.

♠ Cette dernière propriété implique un calcul exact de $A^\top A$ (appendice 7.3). Ce calcul est aussi beaucoup plus rapide (de l'ordre de 20 fois) que tout autre calcul de complexité en $\mathcal{O}(Np^2)$.

En général, en résolvant des systèmes linéaires surdéterminés à partir de $A^\top A$, le risque est de perdre en précision. Dans notre cas, le problème est atténué car $A^\top A$ est calculé exactement et A est de rang plein.

2.6.2.1 OLS

Le problème OLS multidimensionnel s'écrit :

$$P_{\text{OLS}} \begin{cases} \min \|AX - B\|^2 \\ X \in \mathbb{R}^{p \times k} \end{cases}$$

Il peut être résolu indépendamment pour chaque colonne de X et B, ce qui mène à la solution :

$$\widehat{X_{\text{OLS}}} = A^+ B$$

Cependant, nous ne calculons pas A^+ à partir de la décomposition QR de A pour éviter sa complexité en $\mathcal{O}(Np^2)$ ([GL91] algo 5.2.5).

Les équations normales de Gauss, adaptées à ce cas multidimensionnel s'écrivent

$$A^\top A X = A^\top B$$

La matrice $A^\top A$ étant symétrique définie positive, nous résolvons le système à partir de sa décomposition de Cholesky : $A^\top A = R^\top R$, où $R \in \mathbb{R}^{p\times p}$ est triangulaire supérieure. L'algorithme 2.1 transcrit ces observations et particularités.

Signalons que dans nos conditions expérimentales, nous ne rencontrons pas de problème de précision.

$A' \leftarrow A^\top A$	`--` multiplication rapide
$B' \leftarrow A^\top B$	
$R \leftarrow$ factorisation de Cholesky de A'	
résoudre $RY = B'$ en Y	
résoudre $R^\top X = Y$ en X	
Fonctions utilisées :	
– factorisation de Cholesky (`dpotrf` de LAPACK) ;	
– résolution d'un système d'équations triangulaire (`dtrsm` de BLAS).	

Algorithme 2.1: Résolution de $\min \|AX - B\|^2$ en X.

2.6.2.2 TLS

Le problème TLS multidimensionnel s'écrit :

$$P_{\text{TLS}} \begin{cases} \min \|[\omega E \quad wR]\|^2 \\ X \in \mathbb{R}^{p\times k}, E \in \mathbb{R}^{N\times p}, R \in \mathbb{R}^{N\times k} \\ (A+E)X = B + R \end{cases} \tag{2.10}$$

Golub et Van Loan ont résolu ce problème ([GL91] théorème 12.3.1). Leur solution repose sur le fait que $\text{Im}(B+R) \subseteq \text{Im}(A+E)$. En raisonnant sur le rang de $[A+E \quad B+R]$, ils trouvent une borne inférieure sur la valeur du critère, atteinte pour une valeur $(\bar{E}, \bar{R})$. La solution $\widehat{X_{\text{TLS}}}$ s'en déduit.

Cette solution utilise la décomposition SVD de :

$$C = [\omega A \quad wB] = U\Sigma V^\top$$

avec

$$\Sigma = \text{diag}(\sigma_1, ..., \sigma_{p+k}) \qquad \text{et} \qquad V = \begin{matrix} \\ p \\ k \end{matrix} \begin{matrix} \begin{matrix} p & k \end{matrix} \\ \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \end{matrix}$$

La solution TLS est :

$$\widehat{X_{\text{TLS}}} = -\frac{\omega}{w} V_{12} V_{22}^{-1}$$

Elle est unique quand les valeurs singulières σ_p et σ_{p+1} sont distinctes.

♠ **Calcul de la SVD de C.** Nous n'avons besoin que de V. Nous l'obtenons à partir des vecteurs propres de $C^\top C$:

$$C^\top C = \begin{bmatrix} \omega^2 A^\top A & \omega w A^\top B \\ \omega w (A^\top B)^\top & w^2 B^\top B \end{bmatrix} = V \Sigma^2 V^\top$$

Le calcul de $C^\top C$ fait intervenir $A^\top A$ comme seul calcul de complexité en $\mathcal{O}(Np^2)$.

La matrice $C^\top C$ est symétrique, donc ses valeurs propres sont réelles. La relation

$$C^\top C V = V \Sigma^2$$

montre que les colonnes de V sont les vecteurs propres unitaires de $C^\top C$ (au signe près). De même, les valeurs propres de $C^\top C$ sont, par ordre croissant, $\sigma_{p+k}^2, ..., \sigma_1^2$.

Nous n'avons besoin que des k dernières colonnes de V, c'est-à-dire des vecteurs propres de $C^\top C$ correspondant aux k plus petites valeurs propres. L'algorithme 2.2 transcrit ces observations.

Cet algorithme a une précision acceptable si les valeurs singulières σ_p et σ_{p+1} de C sont suffisamment distinctes.

$A' \leftarrow A^\top A$ -- multiplication rapide
$C' \leftarrow \begin{bmatrix} \omega^2 A' & \omega w A^\top B \\ \times & w^2 B^\top B \end{bmatrix}$ -- par symétrie, inutile de remplir le bloc $\times$
$V \leftarrow k$ premiers vecteurs propres de C'
-- conventionnellement, les valeurs propres sont par ordre croissant
$V_{12} \leftarrow V_{1:p,1:k}$
$V_{22} \leftarrow V_{p+1:p+k,1:k}$
résoudre $V_{22}^\top Y = V_{12}^\top$ en Y
$X \leftarrow -(\omega/w) Y^\top$

Fonctions utilisées :
– extraction de vecteurs propres sélectionnés (`dsyevr` de LAPACK) ;
– résolution d'un système linéaire carré (`dgesv` de LAPACK).

Algorithme 2.2: Résolution du problème (P_{TLS}) (équation (2.10)) en X.

♠ **Cas où ω et w sont très différents.** Nous avons développé un algorithme spécifique pour le cas où :

– la matrice $[A \quad B]$ est relativement bien conditionnée (le rapport entre ses valeurs singulières extrêmes est inférieur à 10^4) ;

– le coefficient ω est beaucoup plus grand que w ($\omega/w > 10^2$).

En effet, le calcul des valeurs et vecteurs propres de $C^\top C$ peut s'avérer imprécis parce que son spectre est trop large ([Bjö96] § 2.6.1).

Notre solution consiste à décomposer C. Notons

$$D = \mathrm{diag}(\underbrace{\omega, ..., \omega}_{p}, \underbrace{w, ..., w}_{k}) \qquad \text{et} \qquad C' = [A \quad B]$$

alors

$$C = C'D$$

Nous calculons la décomposition QR : $C' = QR$. L'intérêt est double :

– la décomposition QR de C s'en déduit, à savoir $C = C'D = Q(RD)$ (RD est triangulaire) ;
– seule la matrice R est utilisée. Elle peut être obtenue en effectuant la décomposition de Cholesky de $C'^\top C$, qui est relativement bien conditionnée.

Nous calculons ensuite la décomposition SVD : $RD = U_\mathrm{R}\Sigma_\mathrm{R}V_\mathrm{R}^\top$. Elle résiste mieux au mauvais conditionnement parce qu'elle est n'est pas effectuée sur la matrice élevée au carré.

Le sous-espace du noyau de R engendré par les dernières colonnes de V_R est appelé **sous-espace singulier** correspondant aux plus petites valeurs singulières. Van Huffel *et al.* [HV87] ont développé une méthode de calcul d'une SVD partielle : seules ces colonnes de V_R sont formées. L'algorithme 2.3 transcrit ces particularités.

$A' \leftarrow A^\top A$ -- multiplication rapide
$C' \leftarrow \begin{bmatrix} A' & A^\top B \\ \times & B^\top B \end{bmatrix}$
$R \leftarrow$ décomposition de Cholesky de C'
$R \leftarrow R \times \mathrm{diag}(\omega, ..., \omega, w, ..., w)$
$V \leftarrow k$ vecteurs du noyau de R correspondant à ses k plus petites valeurs singulières.
$V_{12} \leftarrow V_{1:p,1:k}$
$V_{22} \leftarrow V_{p+1:p+k,1:k}$
résoudre $V_{22}^\top Y = V_{12}^\top$ en Y
$X \leftarrow -(\omega/w)Y^\top$

Fonctions utilisées :
– décomposition de Cholesky (`dpotrf` de LAPACK) ;
– calcul de SVD partielle (`psvd` de VANHUFFEL) ;
– résolution d'un système linéaire carré (`dgesv` de LAPACK).

Algorithme 2.3: Résolution du problème (P_TLS) (équation (2.10)) en X, pour ω et w très différents ($\omega/w > 10^2$).

En fait, n'importe quelle base du sous-espace singulier associé aux plus petites valeurs propres suffit pour résoudre le problème TLS. Une telle base peut être obtenue à partir d'une décomposition moins coûteuse que la SVD : les familles de décompositions ULV et URV peuvent être utilisées [HZ93].

2.6.2.3 DLS

Le problème DLS multidimensionnel s'écrit :

$$P_{\text{DLS}} \begin{cases} \min \|E\|^2 \\ X \in \mathbb{R}^{p\times k}, E \in \mathbb{R}^{N\times p} \\ (A+E)X = B \end{cases} \tag{2.11}$$

♠ **Résolution.** La démarche est inspirée de la résolution du problème TLS multidimensionnel ([GL91] théorème 12.3.1).

Nouvelle formulation. Soit la factorisation QR :

$$B = Q \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \begin{matrix} k \\ N-k \end{matrix}$$

En notant :

$$Q = \underset{k}{[Q_1} \quad \underset{N-k}{Q_2]} \qquad Q^\top A = \begin{bmatrix} Q_1^\top A \\ Q_2^\top A \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \begin{matrix} k \\ N-k \end{matrix} \qquad Q^\top E = \begin{bmatrix} E_1 \\ E_2 \end{bmatrix} \begin{matrix} k \\ N-k \end{matrix}$$

il vient :

$$\|E\|^2 = \|Q^\top E\|^2 = \|E_1\|^2 + \|E_2\|^2$$

$$\begin{aligned} (A_1+E_1)X &= R_1 && (2.12) \\ (A_2+E_2)X &= 0_{N-k,k} && (2.13) \end{aligned}$$

Nous supposons que A_1 et R_1 sont de rang plein. Le principe de la résolution est de minimiser $\|E_2\|^2$ sous la contrainte de l'équation 2.13, puis de montrer que nous pouvons poser $E_1 = 0_{k,k}$ dans l'équation 2.12.

Approximation de A_2 par une matrice de rang inférieur. Soit la factorisation SVD :

$$A_2 = \underset{p}{[U_1} \quad \underset{N-k-p}{U_2]} \begin{bmatrix} \Sigma_1 \\ 0_{N-k-p,p} \end{bmatrix} V^\top = U_1 \Sigma_1 V^\top$$

avec $\Sigma_1 = \text{diag}(\sigma_1, ..., \sigma_p)$.

La matrice R_1 étant de rang plein, X est également de rang plein k (équation (2.12)). Ainsi :

$$\begin{aligned} \text{Im}(X) &\subseteq (\text{Ker}(A_2+E_2))^\perp \\ k &\leq p - \text{rang}(A_2+E_2) \\ \text{rang}(A_2+E_2) &\leq p-k \end{aligned}$$

En approchant A_2 par une matrice $A_2 + E_2$ de rang $j < p$, on a ([HJ85] théorème 7.4.51) :

$$\|E_2\|^2 \geq \sum_{i=j+1}^{p} \sigma_i^2$$

Puisque $\text{rang}(A_2 + E_2) \leq p - k$, la norme de E_2 admet une borne inférieure :

$$\|E_2\|^2 \geq \min_{j \leq p-k} \sum_{i=j+1}^{p} \sigma_i^2 = \sum_{i=p-k+1}^{p} \sigma_i^2$$

qui est atteinte pour

$$E_2 = -U_1 \text{diag}(0, ..., 0, \sigma_{p-k+1}, ..., \sigma_p) V^\top$$

Remontée. Fixons la valeur de E en posant $E_1 = 0$ et construisons X compatible avec cette valeur. Notons

$$\underset{p-k}{U_1} = [\underset{p-k}{U_{11}} \quad \underset{k}{U_{12}}] \qquad \underset{p-k}{V} = [\underset{p-k}{V_1} \quad \underset{k}{V_2}]$$

Alors

$$A_2 + E_2 = U_{11} \text{diag}(\sigma_1, ..., \sigma_{p-k}, 0, ..., 0) V_1^\top$$

L'équation (2.13) devient $V_1^\top X = 0$. En la combinant avec l'équation (2.12) :

$$\begin{bmatrix} V_1^\top \\ A_1 \end{bmatrix} X = \begin{bmatrix} 0 \\ R_1 \end{bmatrix}$$

La matrice du système est carrée et inversible. La résolution du système repose sur l'observation :

$$\text{Im}(X) \subseteq (\text{Ker}(V_1^\top))^\perp = (\text{Im}(V_1))^\perp = \text{Im}(V_2)$$

qui garantit l'existence d'une matrice $Y \in \mathbb{R}^{p \times p}$ telle que $X = V_2 Y$, d'où :

$$\begin{aligned} A_1 V_2 Y &= R_1 \\ Y &= (A_1 V_2)^{-1} R_1 \\ \widehat{X_{\text{DLS}}} = X &= V_2 (A_1 V_2)^{-1} R_1 \end{aligned} \tag{2.14}$$

Implémentation. Dans la solution du problème DLS (équation (2.14)), nous évitons le calcul de $Q_2 \in \mathbb{R}^{N \times (N-k)}$ et $A_2 \in \mathbb{R}^{(N-k) \times p}$. ♠

Calcul de V_2. Seule la matrice V_2 de la décomposition SVD $A_2 = U_1 \Sigma_1 [V_1 \quad V_2]^\top$ est nécessaire. En conséquence, il est suffisant de connaître

$$C = A_2^\top A_2$$

que nous obtenons à moindres frais par la relation

$$C = A^\top Q_2 Q_2^\top A = A^\top (\text{Id}_N - Q_1 Q_1^\top) A = A^\top A - A_1^\top A_1$$

Les colonnes de V sont alors les vecteurs propres de C et la matrice V_2 rassemble les vecteurs propres correspondant aux plus petites valeurs propres.

Calcul de A_1. En sortie de la factorisation QR, Q est représentée sous forme d'un produit de matrices de Householder :

$$Q = H_1 \cdots H_k = \prod_{i=1}^{k} (Id_N - \tau_i v_i v_i^\top) \qquad \text{où } v_i = [\underbrace{0 \cdots 0}_{i-1} \quad 1 \; \underbrace{\mu_i^\top}_{N-i}]^\top$$

En général, cette représentation facilite le calcul de produits avec d'autres matrices, mais dans notre cas, pour accumuler $A_1 = Q_1^\top A$, elle est malcommode.

Nous calculons donc explicitement la matrice Q_1. En définissant les matrices $(M_i)_{i=1..k}$ satisfaisant à :

$$H_{k-i+1} \cdots H_k \begin{bmatrix} \mathrm{Id}_k \\ 0_{N-k,k} \end{bmatrix} = \begin{bmatrix} \mathrm{Id}_{k-i} & \\ & M_i \end{bmatrix}_{\substack{N-k+i \\ i}} \in \mathbb{R}^{N \times k}$$

il vient $Q_1 = M_k$, calculée à partir du schéma de récurrence :

$$M_1 = \begin{bmatrix} 1 - \tau_k \\ -\tau_k \mu_k \end{bmatrix} \qquad \text{et} \qquad M_{i+1} = \begin{bmatrix} 1 - \tau_{k-i} & -\tau_{k-i} \mu_{k-i}^\top M_i \\ -\tau_{k-i} \mu_{k-i} & (\mathrm{Id} - \tau_{k-i} \mu_{k-i} \mu_{k-i}^\top) M_i \end{bmatrix} \tag{2.15}$$

Algorithme. Les opérations qui permettent de trouver X sont présentées dans l'algorithme 2.4. Le problème de précision dans les calculs n'affecte pas cet algorithme car C a un spectre relativement étroit (au plus deux fois plus étendu que celui de A).

$(Q_1, R_1) \longleftarrow$ décomposition QR partielle de B
$A_1 \longleftarrow Q_1^\top A$
$C \leftarrow A^\top A - A_1^\top A_1$ -- multiplication rapide
$V_2 \leftarrow k$ premiers vecteurs propres de C
$K \leftarrow A_1 V_2$
résolution de $KY = R_1$ en Y
$X \longleftarrow V_2 Y$

Fonctions utilisées :
– décomposition QR (`dgeqrf` de LAPACK) avec calcul explicite de Q_1 (équation (2.15)) ;
– extraction de vecteurs propres sélectionnés (`dsyevr` de LAPACK) ;
– résolution d'un système linéaire carré (`dgesv` de LAPACK).

Algorithme 2.4: Résolution du problème DLS (P_{DLS}) (équation (2.11)) en X.

2.6.3 Méthodes itératives pour le cas non linéaire

Les méthodes analytiques permettent de faire l'estimation pour des modèles particuliers (la classe la plus importante est celle des modèles linéaires et bilinéaires). Pour résoudre les problèmes précédents dans le cas général dérivable, les méthodes d'optimisation itératives s'imposent.

Principe. Les méthodes itératives que nous examinons ici partent d'une estimation initiale β_0 de la solution. Elles construisent une suite $(\beta_k)_k$ d'estimations de β convergeant vers la solution. À chaque pas, elles utilisent une approximation affine, de validité locale du modèle, basée sur la valeur de la fonction f et sa matrice jacobienne

$$\frac{\partial f(x,\beta)}{\partial \beta} \qquad \text{et, dans le cas ODR} \qquad \frac{\partial f(x+\delta,\beta)}{\partial \delta}$$

L'estimation initiale β_0 doit être pertinente, sous peine de converger vers un optimum local au lieu de l'optimum global recherché.

L'algorithme peut cesser d'itérer pour une des raisons suivantes :
- il a atteint un nombre limite d'itérations ;
- la matrice du système linéaire (dépendant les matrices jacobiennes de f) associé au β_k courant n'est pas de rang plein (à la précision machine près) ;
- la suite d'estimations devient stationnaire ;
- le critère ne peut plus être amélioré (à la précision machine près), en particulier s'il est nul.

Les deux premiers cas sont des échecs, les deux derniers des succès.

2.6.3.1 Moindres carrés verticaux

L'optimisation du critère OLS dans le cas non linéaire utilise une approximation linéaire du modèle.

Notons r la fonction de β qui définit le vecteur de résidus :

$$r : \begin{array}{ccc} \mathbb{R}^p & \longrightarrow & \mathbb{R}^{nq} \\ \beta & \longmapsto & \begin{bmatrix} W(f(\widetilde{x_1},\beta)-\widetilde{y_1}) \\ \vdots \\ W(f(\widetilde{x_n},\beta)-\widetilde{y_n}) \end{bmatrix} \end{array}$$

La relation avec le critère est :

$$\|r(\beta)\|^2 = K([\widetilde{x_1}\ \cdots\ \widetilde{x_n}],[\widetilde{y_1}\ \cdots\ \widetilde{y_n}],\beta)$$

La solution au sens OLS (équation (2.3)) s'écrit donc :

$$\widehat{\beta_{\text{OLS}}} = \underset{\beta\in\mathbb{R}^p}{\operatorname{argmin}} \|r(\beta)\|^2 \tag{2.16}$$

Gauss-Newton. La suite $(\beta_k)_{k\in\mathbb{N}}$ est construite pour qu'à l'étape $(k+1)$, β_{k+1} soit dans le « voisinage » de β_k, c'est-à-dire que le **pas**

$$\Delta = \beta_{k+1} - \beta_k$$

est petit. Un développement de Taylor-Young à l'ordre 1 de r permet d'écrire :

$$r(\beta_k+\Delta) \approx r(\beta_k) + r'(\beta_k)\Delta$$

Cette approximation peut être utilisée pour solutionner l'équation (2.16) :

$$\min_{\Delta} \|r(\beta_k + \Delta)\|^2 \approx \min_{\Delta} \|r(\beta_k) + r'(\beta_k)\Delta\|^2$$

C'est un problème de moindres carrés linéaires ayant pour solution :

$$\widehat{\Delta_{\mathrm{GN}}} = -r'(\beta_k)^+ r(\beta_k)$$

Ceci définit le schéma itératif de Gauss-Newton. La matrice jacobienne de r s'exprime sans difficulté à partir de la matrice jacobienne de f par rapport à β pour les différentes expériences.

Amélioration. L'inconvénient de l'approche précédente est qu'elle est trop confiante : la valeur estimée Δ_{GN} peut « tomber » en dehors du domaine de validité de l'approximation linéaire de r.

Les pistes d'amélioration consistent à :

- s'assurer que le pas d'estimation reste à l'intérieur d'une **région de confiance** dans laquelle l'approximation est considérée comme fiable ;
- changer la matrice de la relation linéaire qui lie le résidu courant au pas à effectuer :

  $$\widehat{\Delta} = -Ar(\beta_k)$$

  où A est une matrice « bien choisie » pour donner une meilleure approximation du pas que $r'(\beta_k)^+$.

Levenberg-Marquardt. L'algorithme de Levenberg-Marquardt ([Lev44] eq6 p165, [Mar63] p434 eq10, eq11, eq12, [DS83] eq10.2.15, [Mor78] p106 eq2.3) met en œuvre ce canevas. La région de confiance est une boule de centre β_k et de rayon ρ_k, mis à jour à chaque itération. Le pas $\widehat{\Delta_{\mathrm{LM}}}$ est contraint à ne pas sortir de cette boule.

Cette contrainte est garantie par une modification de la matrice A en fonction la région dans laquelle se trouve $\widehat{\Delta_{\mathrm{GN}}}$:

- si $\|\widehat{\Delta_{\mathrm{GN}}}\| \leq \rho_k$,

  $$A = r'(\beta_k)^+$$

  c'est l'estimation de Gauss-Newton ;
- sinon,

  $$A = (r'(\beta_k)^\top r'(\beta_k) + \alpha \mathrm{Id}_p)^{-1} r'(\beta_k)^\top$$

  où α satisfait à $\|\widehat{\Delta_{\mathrm{LM}}}\| = \rho_k$. Cette valeur existe (résultat de la théorie de Lagrange-Karush-Kuhn-Tucker) mais son calcul nécessite la résolution algorithmique d'une équation non linéaire[2].

[2] La présentation de Levenberg ne « peut pas » s'appuyer sur les résultats d'optimisation avec contraintes d'inégalité établis par Karush en 1939 [Kar39]... dans un mémoire de maîtrise... et « approfondis » ensuite par Kuhn et Tucker [HT51]. Par contre, Marquardt qui aurait pu le faire ne le fait pas explicitement... et avoue avoir ignoré le travail de Levenberg !

À chaque pas d'itération, les résidus sont évalués. Deux cas se présentent :
- si $\|r(\beta_{k+1})\| < \|r(\beta_k)\|$, l'approximation est fiable, et l'algorithme étend la région de confiance ($\rho_{k+1} > \rho_k$) ;
- sinon l'approximation n'est pas fiable et l'algorithme restreint la région de confiance ($\rho_{k+1} < \rho_k$). L'estimation « repart » de β_k.

Nous utilisons la mise en œuvre de l'algorithme (calcul de ρ_k et α) de MINPACK [MGH80] (fonction `lmder`).

2.6.3.2 Régression en distance orthogonale

Le problème (P_{ODR}) peut être considéré comme un problème de moindres carrés pour lequel :
- les paramètres à estimer sont la juxtaposition des paramètres initiaux β et les erreurs en entrée $(\delta_i)_{i=1..n}$:
$$\beta' = [\beta^\top \ \delta_1^\top \ \cdots \ \delta_n^\top \ ]^\top \in \mathbb{R}^{p+nm}$$
- les résidus à traiter sont :
$$r(\beta') = \begin{bmatrix} W(\widetilde{y_1} - f(\widetilde{x_1} + \delta_1, \beta)) \\ \vdots \\ W(\widetilde{y_n} - f(\widetilde{x_n} + \delta_n, \beta)) \\ \Omega\delta_1 \\ \vdots \\ \Omega\delta_n \end{bmatrix} \in \mathbb{R}^{nq+nm}$$

La matrice jacobienne qui en résulte se présente sous la forme générique (en utilisant les notations de [BBS87] p1057, reprises dans [Bjö02] eq10 p228)
$$\mathcal{J} = \begin{matrix} \\ nq \\ nm \end{matrix} \begin{matrix} \begin{bmatrix} J & V \\ 0 & D \end{bmatrix} \end{matrix} \begin{matrix} \\ \\ \end{matrix} \\ \begin{matrix} p & nm \end{matrix}$$
où :
- le bloc J se déduit de $\frac{\partial f(x,\beta)}{\partial \beta}$ aux points $(\widetilde{x_i}, \beta)$;
- la matrice V est diagonale par blocs et se déduit de $\frac{\partial f(x+\delta,\beta)}{\partial \delta}$;
- la matrice D est constituée de n blocs diagonaux Ω.

Résolution. Pour résoudre ce problème, on peut utiliser la méthode de Levenberg-Marquardt. Le cœur d'une itération de l'algorithme LM est la décomposition QR de $\mathcal{J}$. Si on ne tient pas compte du caractère structuré de cette matrice, la complexité de ce calcul est en $\mathcal{O}((nq + nm)(p + nm)^2)$, ce qui est prohibitif.

Plusieurs propositions ont été faites pour remédier à cette difficulté :
- Schwetlick et Tiller [ST85] « contournent le problème » en n'implémentant le principe des régions de confiance de LM que pour les paramètres β, et non pour les erreurs $(\delta_i)_{i=1..n}$;

- Boggs *et al.* [BBS87] se ramènent au calcul de la décomposition QR d'une matrice de la forme MJ, où M est diagonale. Leur « levier » est la formule d'inversion de Sherman-Morrison-Woodbury ;
- Hartley et Zisserman ([HZ01] p571) traitent des problèmes moindres carrés plus généraux (que celui sous-jacent a (P_{ODR})) pour lesquels $r_i(\beta')$ dépend uniquement de β et δ_i. Ils ignorent donc :
  - la présence du bloc de zéros dans $\mathcal{J}$;
  - la structure diagonale par blocs de D.

  Leur algorithme « Levenberg-Marquardt creux » évite néanmoins une complexité en $\mathcal{O}(n^2)$, mais celle-ci demeure en $\mathcal{O}(nqp^2)+\mathcal{O}(nmp^2)+\mathcal{O}(nm^2(q+m))+\cdots$. De plus, il demande à être adapté afin d'implémenter complètement le principe des régions de confiance. Trop général pour traiter le cas (P_{ODR}), cet algorithme peut trouver son utilité dans des cas plus spécifiques.
- Björck [Bjö02] fait une proposition concurrente a celle de Boggs *et al.* Ici la clef est la décomposition QR d'une matrice de Hessenberg « inférieure » $[D^\top \quad v'^\top]^\top$. C'est tout simplement dommage qu'il n'existe à ce jour aucune implémentation de ce « bel » algorithme... Avis aux amateurs !

En pratique, nous utilisons la bibliothèque ODRPACK (fonction `dodrc`) qui exploite l'optimisation de Boggs *et al.* pour résoudre le problème efficacement [BBRS92].

2.6.3.3 Moindres carrés repondérés

Dans le cas d'un modèle linéaire, un algorithme itératif peut faire la M-estimation (§ 2.5.3). Le principe est de résoudre une série de problèmes de moindres carrés linéaires en changeant uniquement la matrice de poids ([Mee01] p71)

La justification de cette « stratégie » trouve son origine dans l'équation d'Euler, à savoir la solution β satisfait à :

$$\frac{\partial}{\partial\beta}\sum_{i=1}^{n}\rho(\|W(f(\widetilde{x}_i,\beta)-\widetilde{y}_i)\|)=0_{1,p}$$

donc :

$$\sum_{i=1}^{n}\omega_i(\beta)r_i(\beta)^\top r_i'(\beta)=0_{1,p}$$

où :

- le résidu vectoriel $r_i(\beta)=W(f(\widetilde{x}_i,\beta)-\widetilde{y}_i)$;
- le poids

$$\omega_i(\beta)=\frac{\rho'(\|r_i(\beta)\|)}{\|r_i(\beta)\|}$$

En exploitant la linéarité,

$$r_i(\beta)=W(C(\widetilde{x}_i)\beta+D(\widetilde{x}_i)-\widetilde{y}_i)=A_i\beta-b_i$$

où

$$\left\{ \begin{array}{rcl} A_i & = & WC(\widetilde{x}_i) \\ b_i & = & W(D(\widetilde{x}_i) - \widetilde{y}_i) \end{array} \right.$$

la condition nécessaire devient

$$\sum_{i=1}^{n} \omega_i(\beta) A_i^\top A_i \beta = \sum_{i=1}^{n} \omega_i(\beta) A_i^\top b_i$$

C'est cette équation qui justifie que dans la méthode des moindres carrés repondérés, à l'itération k, β_k est la solution du système linéaire suivant :

$$\underbrace{\left(\sum_{i=1}^{n} \omega_i(\beta_{k-1}) A_i^\top A_i \right)}_{A} \beta_k = \sum_{i=1}^{n} \omega_i(\beta_{k-1}) A_i^\top b_i$$

La résolution peut se faire à l'aide d'une factorisation de Cholesky de la matrice A, symétrique définie positive.

Fonction de modération adaptative. L'algorithme peut être adapté au cas où la fonction de modération ρ dépend d'un facteur d'échelle (équations (2.6) et (2.7)). L'adaptation consiste à faire évoluer le facteur σ au fil des itérations ([BJ98] § 4) :

- au début, σ est élevé. La solution courante est loin de l'optimum donc le résidu ne permet pas de distinguer clairement les mesures aberrantes ;
- à la fin, σ est petit. La solution est proche, donc la distinction entre mesures légitimes et aberrantes devient plus nette.

Convergence. La convergence du processus est garantie si ρ est convexe. Dans ce cas, ρ n'est pas bornée, donc le critère ne peut pas être robuste ([Hub81], p103).

2.6.4 Méthodes d'échantillonnage aléatoire

Dans le cas d'un critère sélectif (§ 2.5.4), la résolution passe par celle d'un critère non robuste, conformément à la procédure suivante (algorithme 2.5) :

- engendrer **tous** les sous-ensembles ;
- faire l'estimation de β pour chacun d'eux ;
- retenir celui qui fournit le critère le plus faible.

Cette méthode est presque toujours impraticable à cause de l'explosion combinatoire ($\mathrm{Card}(S_n^h) = C_n^h$). Les algorithmes se limitent donc à un certain nombre de sous-ensembles tirés aléatoirement (de l'ordre de 300 à 5000) [RL87].

Plusieurs stratégies permettent d'améliorer cette approche.

$v_{\min} \longleftarrow \infty$
Pour $I \in S_n^h$ **faire**
  $(\beta, v) \longleftarrow$`minimiser`$(I)$
  Si $v < v_{\min}$ **alors** -- retient le meilleur β
    $\widehat{\beta} \longleftarrow \beta$; $v_{\min} \longleftarrow v$
  finsi
finpour
Le résultat est $\widehat{\beta}$.

Fonction utilisée :
`minimiser`$(\{i_1, ..., i_h\})$ implémente l'estimateur $\mathcal{E}$ basé sur le critère non robuste. Il renvoie le couple (β, v) où

$$\beta = \mathcal{E}\left(\begin{bmatrix} \widetilde{x_{i_1}} & \cdots & \widetilde{x_{i_h}} \\ \widetilde{y_{i_1}} & \cdots & \widetilde{y_{i_h}} \end{bmatrix}\right)$$

et v est la valeur du critère en β ;

Algorithme 2.5: Calcul exhaustif d'un critère sélectif ((P_{sel}), équation (2.8)).

2.6.4.1 L'étape de concentration (*C-step*)

Après un tirage aléatoire et l'estimation non robuste correspondante, la fonction K_{u} permet de calculer les résidus associés à chaque expérience. Les expériences correspondant aux résidus les plus faibles constituent un **échantillon concentré** sur lequel l'algorithme 2.6 refait l'estimation non robuste [RD99].

Le couple d'opérations symétriques `meilleures_expériences`/`minimiser` constitue l'étape de concentration (c'est une technique de nuées dynamiques). Le nombre d'expériences utilisées (h_1 et h_2) peut être différent selon l'étape de l'algorithme. Un bon plan :

- faire l'estimation initale sur le minimum d'expériences (h_1 faible) pour éviter la contamination ;
- utiliser un maximum d'expériences (h_2 élevé) aux étapes suivantes (moins susceptibles de contamination) pour résister aux erreurs faibles.

En itérant cette opération de concentration on assure la convergence dans le cas LTS car :

- le nombre d'échantillons est fini ;
- le critère est décroissant.

Ce n'est pas le cas avec LQS (et LMS).

FAST-LTS. L'algorithme FAST-LTS proposé par Rousseeuw et Van Driessen [RD99][3] est une mise en œuvre de ce canevas où :

- la dimension de sortie est $q = 1$;
- l'estimateur de base est OLS ;
- la taille de l'échantillon h_1 est minimale ($h_1 = p$ en général) ;

[3]Ceci est une version appliquée à $n < 600$, sinon un étage supplémentaire de sélection d'échantillons est nécessaire. FAST-LTS est disponible à `http://www.agoras.ua.ac.be/Robustn.htm`.

$B \longleftarrow \{\}$ -- les meilleurs β trouvés et les valeurs de critère associés
Pour i **de** 1 **à** $i_{\max}$ -- traitement de nombreux échantillons aléatoires
  $I \longleftarrow$`échantillon_aléatoire` (n, h_1)
  $(\beta, v) \longleftarrow$`minimiser`$(I)$
  Pour j **de** 1 **à** $j_{\max}$ **faire** -- quelques concentrations
    $I \longleftarrow$`meilleures_expériences`(β, h_2)
    $(\beta, v) \longleftarrow$`minimiser`$(I)$
  finpour
  `ajouter_si_meilleur`$(B, k_{\max}, (\beta, v))$ -- ne retient que les $k_{\max}$ meilleurs
finpour
$v_{\min} \longleftarrow \infty$
Pour (β, v) **dans** B **faire** -- retraitement des meilleures estimations
  Répéter -- concentrations jusqu'à convergence
    $\beta_{\text{prec}} \longleftarrow \beta$
    $I \longleftarrow$ `meilleures_expériences`(β, h_2)
    $(\beta, v) \longleftarrow$`minimiser`$(I)$
  jusqu'à $\beta = \beta_{\text{prec}}$
  Si $v < v_{\min}$ **alors** -- retient le meilleur β
    $\widehat{\beta} \longleftarrow \beta$; $v_{\min} \longleftarrow v$
  finsi
finpour
Le résultat est $\widehat{\beta}$.

Fonctions utilisées :
- `échantillon_aléatoire`(n, h) renvoie un élément $\{i_1, ..., i_h\}$ de S_n^h tiré aléatoirement ;
- `minimiser`$(\{i_1, ..., i_h\})$ implémente l'estimateur $\mathcal{E}$ basé sur le critère non robuste. Il renvoie le couple (β, v) où
$$\beta = \mathcal{E}\left(\begin{bmatrix} \widetilde{x_{i_1}} & \cdots & \widetilde{x_{i_h}} \\ \widetilde{y_{i_1}} & \cdots & \widetilde{y_{i_h}} \end{bmatrix}\right)$$
et v est la valeur du critère en ce β ;
- `meilleures_expériences`(β, h) renvoie l'ensemble des numéros des h expériences pour lesquelles le critère unitaire $K_{\text{u}}(\beta, \widetilde{x}_i, \widetilde{y}_i)$ est le plus faible ;
- `ajouter_si_meilleur`$(B, k, (\beta, v))$ ajoute l'élément (β, v) à B en s'assurant que la taille de l'ensemble résultant ne dépasse pas k, à savoir :
  Si Card$(B) < k$ **alors**
    $B \leftarrow B \cup \{(\beta, v)\}$ -- ajouter (β, v) inconditionnellement
  sinon
    $(\beta', v') \longleftarrow \underset{(\beta', v') \in B}{\text{argmax}}\ v'$ -- β' est la pire estimation de l'ensemble
    Si $v < v'$ **alors** -- si β meilleur que β'
      $B \leftarrow (B \setminus \{(\beta', v')\}) \cup \{(\beta, v)\}$ -- remplacer (β', v') par (β, v)
    finsi
  finsi

Algorithme 2.6: Optimisation approchée d'un critère sélectif (problème (P_{sel}), équation (2.8)) par échantillonnage aléatoire avec étapes de concentration.

– au maximum, $h_2 = \lfloor \frac{p+n+1}{2} \rfloor$;
– $i_{\max} = 500$, $j_{\max} = 2$ et $k_{\max} = 10$ sont fixés empiriquement.

♠ **FAST-LTTS.** Nous avons adapté FAST-LTS à nos besoins. L'algorithme résultant, FAST-LTTS (*Fast Least Total Trimmed Squares*), apporte les modifications suivantes :
– éventuellement, $q \neq 1$;
– l'estimateur de base est TLS ;
– la taille minimale pour l'échantillon initial devient $h_1 = \lceil p/q \rceil$.
Le schéma itératif « répéter... jusqu'à » est assuré de converger avec LTTS pour les raisons citées à propos de LTS concernant les opérations de concentration.

2.6.4.2 Tirage aléatoire guidé par l'application

Le tirage aléatoire peut être rendu plus pertinent en choisissant les échantillons en fonction du contexte. En particulier, quand les données (en entrée/sortie) de chaque expérience révèlent des « poches d'accumulation », il peut être intéressant de ne tirer que des sous-ensembles répartis uniformément dans ces espaces de données. C'est une technique de **compartimentage** [CC03].

Par rapport au critère, cette technique peut avoir deux conséquences :
– pour un nombre de tirages donné, elle permet d'atteindre une valeur plus faible du critère (et donc d'améliorer l'estimation) ;
– *de facto*, le nombre d'échantillons susceptibles d'être tirés devient strictement inférieur à C_n^h, conduisant au problème $P'_{\text{sel}} \neq P_{\text{sel}}$.

2.7 Mise en œuvre

La mise en œuvre d'un estimateur requiert de connaître le type d'erreurs. On suppose souvent en première approximation qu'on est dans le cas OLS parce que la résolution est simple et efficace. Cependant, la recherche de la précision ne peut pas faire l'impasse sur la prise en compte des erreurs réellement rencontrées.

2.7.1 Bibliothèques

Les codes associés aux algorithmes de résolution itératifs se présentent généralement sous forme de bibliothèques qui nécessitent de fournir :
– une estimation initale de la solution ;
– une fonction qui calcule la valeur du critère pour un paramètre β donné ;
– une fonction qui calcule la dérivée du critère en un point par rapport à β (à défaut, la bibliothèque peut utiliser des différences finies).
Si l'estimation initiale est trop éloignée de la solution, la suite de solutions risque de converger vers un minimum local (inintéressant) du critère.

2.7.2 Estimateurs stochastiques

La performance des estimateurs stochastiques est très sensible aux choix d'implémentation. Ainsi, les algorithmes FAST-LTS et FAST-LTTS sont une combinaison de tirages aléatoires et de concentrations appliquées dans un ordre qui, empiriquement, donne un résultat acceptable compatible avec les contraintes temporelles.

2.7.3 Validation

Pour valider une implémentation, il est très utile d'avoir des données avec une « vérité terrain » (par exemple des données de synthèse) sur lesquelles l'estimateur peut être évalué et amélioré.

Nous ne nous sommes pas souciés des cas dégénérés, en particulier des matrices dont le rang n'est pas maximal. Lors de la mise en œuvre, ces cas se manifestent par des « `NaN` » ou « `Inf` » ou par des erreurs renvoyées par les fonctions des bibliothèques numériques. Ces symptômes révèlent :

– soit des données inutilisables (par exemple une image uniformément noire) ;
– soit une erreur d'implémentation.

Pendant la phase d'optimisation temporelle de la résolution d'un problème (§ 2.6.2), il est indispensable de comparer les résultats de l'algorithme avec une version (plus lente) éprouvée.

2.7.4 Interfaces entre programmes

Les bibliothèques mentionnées sont écrites en FORTRAN. L'interface avec le programme principal (en C, C++, MATLAB voire JAVA) nécessite le développement de « ponts » pour échanger les données et exécuter les appels de sous-routines. Ces « ponts » suivent les règles d'édition de liens sous Unix : nous les écrivons donc naturellement en C.

2.8 Conclusion

Nous avons détaillé l'optique avec laquelle nous abordons les problèmes d'estimation de paramètres, à savoir :

– analyse et modélisation du système ;
– choix d'une technique d'optimisation compatible ;
– implémentation à base de bibliothèques numériques éprouvées.

Nous avons aussi exposé les types de modèles et d'erreurs que nous rencontrerons dans la suite, avec les modélisations et les algorithmes associés.

Il existe beaucoup d'autres problèmes d'estimation en vision par ordinateur. Parmi les aspects non traités, citions :

– d'autres types de modèles et d'erreurs ;

- les modèles **implicites**. Le système n'a pas d'entrées et de sorties distinctes, seulement des variables qui vérifient un ensemble d'équations. Ce modèle est équivalent à un modèle explicite où les sorties sont nulles et sans erreurs ;
- la **sélection de modèle** ([Kan04], § 2.1). Le système peut être décrit par plusieurs modèles ordonnés. Chaque modèle est plus détaillé et dépend de plus de paramètres que le précédent. Il s'agit donc de sélectionner le modèle le plus pertinent et conjointement d'estimer ses paramètres en fonction des mesures. Des critères appropriés permettent d'évaluer l'adéquation d'un modèle à des mesures ;
- l'estimation conjointe des paramètres et de la loi de distribution de l'erreur. En particulier, dans le cas gaussien, la matrice de variance-covariance peut être estimée (***scale estimation*** [Hub81]) ;
- le **calcul progressif** (par degrés). Les expériences étant effectuées séquentiellement, comment estimer au mieux les paramètres avec les données disponibles ? Ce problème peut être résolu par une méthode de Kalman ([WB01] § 4) ;
- l'**estimation de la localisation** (*location estimation*). C'est plus élémentaire encore que le modèle linéaire ; il s'agit de trouver les coordonnées d'un point dans des données bruitées (par exemple une cible sur une image radar ou un maximum de concentration dans un nuage de points). Parmi les techniques de résolution, citons le *Mean Shift* ([Mee01], p50) ;
- les techniques d'**exploration de l'espace**. Elles permettent de trouver une solution dans les cas où le modèle n'est pas assez régulier (voire discret) ou s'il y a un fort risque de « tomber » dans un minimum local. Ces techniques, itératives, améliorent la (ou les) solution(s) courante(s) par deux types d'évolution : l'**intensification** (amélioration du critère par exploration locale) et la **diversification** (perturbation de la solution pour atteindre le bassin d'attraction d'un autre minimum local). Parmi ces techniques, citons le **recuit simulé**, la **recherche tabou**, la **recherche par dispersion** (*scatter search*) et les **algorithmes génétiques** ([Cha97] § 7.3).
- etc.

Chapitre 3

Images panoramiques

Précautions oratoires. En accord sans réserve avec le mot de la fin du prologue de Faugeras ([BK01] pVI), la référence incontournable sur le sujet des « image panoramiques et leur exploitation en vision artificielle » demeure bien l'ouvrage édité par Benosman et Sing Bing Kang [BK01].

Dans ce cadre-là notre modeste contribution se situe à trois niveaux :

- conception et réalisation d'un capteur dédié, appelé caméra panoramique à balayage angulaire (§ 3.3.2) ;
- mise en œuvre d'une caméra catadioptrique du commerce ;
- construction d'une image panoramique comme mosaïque d'images traditionnelles.

3.1 Des mots et ce qu'il y a derrière

Image panoramique. Intuitivement et conformément à la définition du dictionnaire ([Lar02] p740), c'est une image qui permet de découvrir un vaste paysage. Nous sous-entendons que les caméras ordinaires (classiques, traditionnelles) ne sont pas aptes à délivrer des images panoramiques.

Champ de vision d'un système optique. C'est la portion de l'espace 3D qu'embrasse le système optique. Nous réservons l'adjectif « visuel » pour le cas où le capteur est l'œil !

Système catadioptrique. C'est un objectif photographique composé d'un miroir au moins et d'éventuelles lentilles.

Objectif photographique à centre(s) optique(s) (OPCO). Un système optique (objectif photographique) est un (OPCO) s'il existe au moins un point de l'espace par lequel passent, réellement ou virtuellement, tous les rayons lumineux. Un tel point est appelé point de vue effectif.

Remarque. La projection centrale (plane ou non) modélise un objectif photographique a centre(s) optique(s). Dans ce cas il existe un seul point de vue effectif : le centre de projection !

3.2 Classification des caméras équipées d'un (OPCO)

3.2.1 Une définition mathématique du champ de vision

Appelons :
- O le point de vue effectif d'un capteur (C) intégrant un (OPCO) ;
- $S(O,1)$ la sphère unité centrée en O de l'espace affine tridimensionnel $\mathcal{E}_3$;
- M un point quelconque de $\mathcal{E}_3$;
- $m = S(O,1) \cap [OM]$. (OmM) est le **rayon de projection** associé à M.

3.2.2 Une proposition de classification

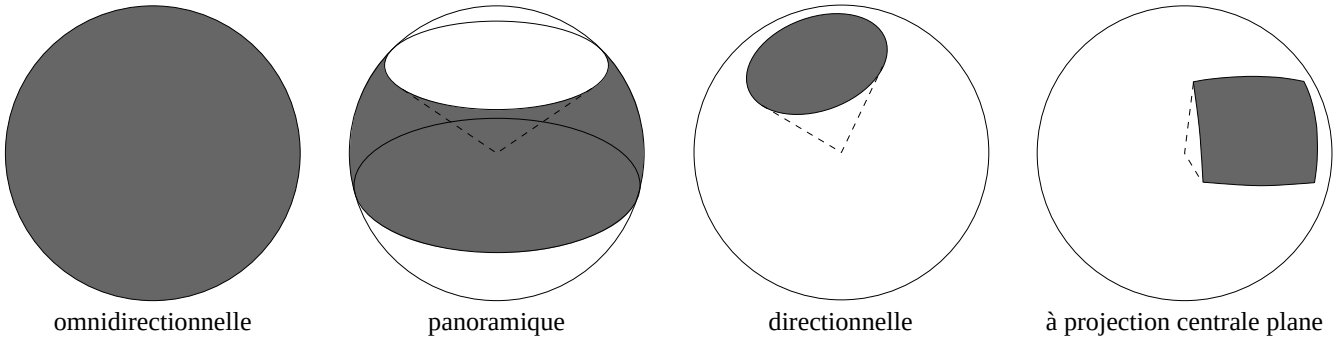


FIG. 3.1: Région s la sphère unité $\mathcal{S}(O,1)$, caractéristique du champ de vision.

Nous, avec Pajdla *et al.* ([PSc01] p74-75 § 5.2) caractérisons le champ de vision du capteur (C) par la surface de la région s sur la sphère $S(O,1)$ où :

$$s = \{m \in S(O,1) / \exists M \text{visible (ayant une image) depuis (C)}\}$$

Une caméra est (figure 3.1) :

directionnelle si s est incluse au sens strict dans un hémisphère de $S(O,1)$;

panoramique si s contient un grand cercle. L'appellation se justifie en imaginant que l'image peut être obtenue à partir d'un **panoramique**, à savoir un mouvement de rotation d'une caméra autour d'un axe passant par son centre optique ;

omnidirectionnelle si $s = S(O,1)$. Une telle caméra est difficile (impossible ?) à réaliser, ne serait-ce que parce que le capteur lui-même occulte une partie de l'espace 3D.

Remarque. Une caméra ordinaire à **projection centrale plane** est une caméra directionnelle par construction car :

- s est un hémisphère si et seulement si la surface de projection est un plan de $\mathcal{E}_3$ en entier ;
- l'image effective est toujours une infime partie du plan de projection : les CCD sont carrés ou rectangulaires avec une diagonale de quelques (au plus) centimètres pour une distance focale du même ordre.

3.3 Les caméras panoramiques

Une caméra ordinaire ne peut pas fournir une image panoramique « digne de ce nom » à cause des limites intrinsèques de son champ de vision. Les caméras panoramiques visent à étendre celui-ci.

3.3.1 Les caméras panoramiques catadioptriques

3.3.1.1 Principe

Une caméra panoramique catadioptrique est un capteur d'images qui combine un miroir, caractérisé par un seul point de vue effectif O, et une caméra ordinaire ([PSc01] §5.2 p75). L'objectif photographique de la caméra ordinaire est centré, à savoir composé de lentilles (et de miroirs éventuels) admettant un axe de symétrie commun appelé axe optique.

Le point de vue effectif intrinsèque au miroir appartient a l'axe optique de la caméra (figure 3.2).

3.3.1.2 Les « bons » miroirs

Sous l'hypothèse d'un objectif de caméra ordinaire à projection centrale plane (de centre de projection L), Baker et Nayar ([BN01] eq4.16 et 4.17) démontrent que les seuls miroirs satisfaisant au principe précédent ont une surface de type quadrique de révolution (avec deux restrictions) : autour de l'axe optique de la caméra ordinaire. La condition précédente est nécessaire mais pas suffisante pour obtenir au final un capteur d'images « digne de ce nom », comme le montre le bilan détaillé qui suit.

Cas où la quadrique est un plan ([BN01] § 4.2.3.1). Celui-ci est parallèle au plan image passant par le milieu de OL. Les rayons lumineux convergent virtuellement en O. Géométriquement, cette combinaison est équivalente à une projection centrale plane et ne permet pas d'augmenter le champ de vision.

Cas où la quadrique est un cône ([BN01] § 4.2.3.2). Celui-ci a son sommet en O qui doit être confondu avec L. Les rayons lumineux convergent réellement en O. Le champ de vision se réduit au cône !

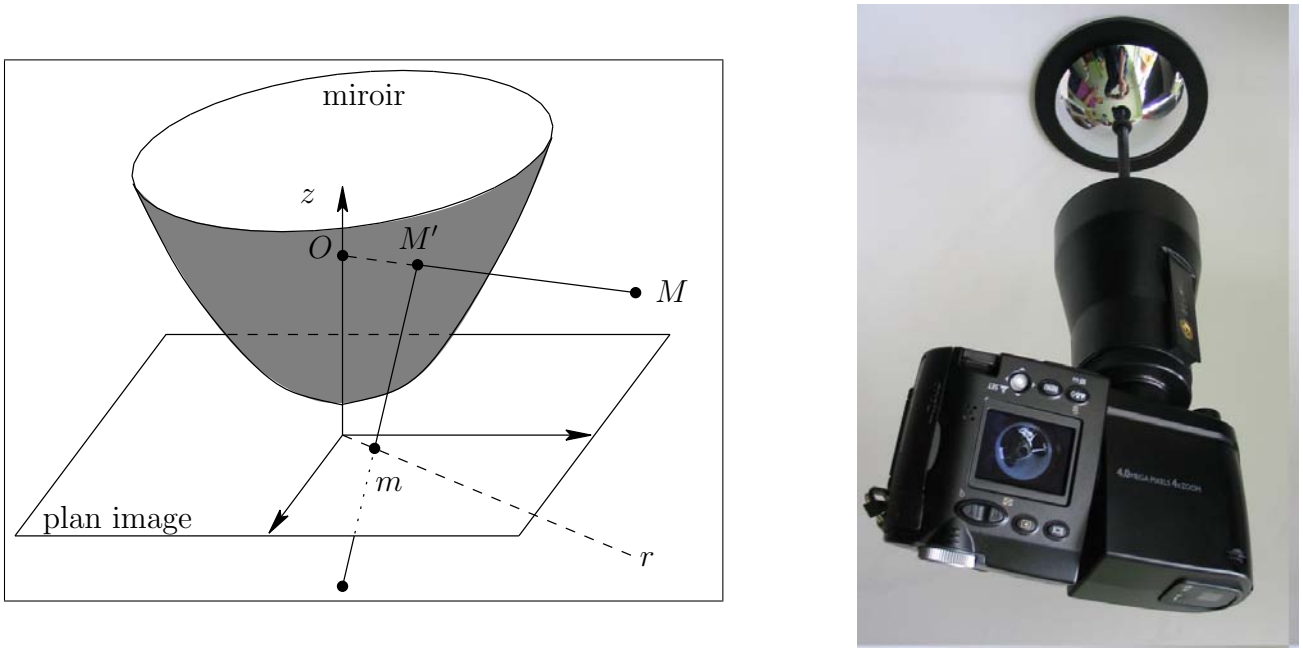


FIG. 3.2: La géométrie d'une caméra catadioptrique et le modèle que nous utilisons (OneShot360 de Panosmart).

Cas où la quadrique est un cylindre de révolution. C'est mathématiquement impossible ! Physiquement le cylindre rencontrerait la caméra ordinaire...

Cas où la quadrique est une sphère ([BN01] § 4.2.3.3). Celle-ci est centrée en O qui doit être confondu avec L. Les rayons lumineux convergent virtuellement en O. Le champ de vision se réduit au point O !

Cas où la quadrique est un ellipsoïde de révolution ([BN01] § 4.2.3.4). Celui-ci admet O et L pour foyers. Les rayons lumineux convergent réellement en O, d'où la nécessité de choisir un miroir concave obtenu en tronquant l'ellipsoïde par un plan passant par O. Le champ de vision est « réduit » à un hémisphère !

Cas où la quadrique est un hyperboloïde à deux nappes ([BN01] § 4.2.3.5). Celui-ci admet O et L pour foyers. Seule la nappe contenant L est conservée. Les rayons lumineux convergent virtuellement en O.

Cas où la quadrique est un hyperboloïde à une nappe. C'est mathématiquement impossible ! Physiquement les foyers seraient situés dans un plan orthogonal a OL !

Cas où la quadrique est un paraboloïde de révolution ([BN01] § 4.2.4). C'est mathématiquement impossible ! MAIS, MAIS, si on traite le cas limite où OL devient infini,

alors la projection centrale peut être approchée par une projection plane parallèle à OL et... les rayons lumineux convergent virtuellement en O qui est le foyer du paraboloïde. On obtient un miroir aux propriétés analogues a celles de l'hyperboloïde à deux nappes. La combinaison avec la caméra est tributaire du modèle de projection parallèle à OL.

Bilan. Les miroirs les plus adaptés à étendre le champ de vision d'une caméra ordinaire correspondent :
- à l'hyperboloïde à deux nappes ;
- au paraboloïde de révolution.

3.3.1.3 Notations

- L'espace affine $\mathcal{E}_3$ est rapporté à un repère orthonormé direct $(O, \vec{i}, \vec{j}, \vec{k})$ d'axes (Ox), (Oy), (Oz).
- O, l'origine du repère $(O, \vec{i}, \vec{j}, \vec{k})$ est l'unique point de vue effectif.
- Un point M admet x, y, z respectivement pour abscisse, ordonnée et cote.
- $M'(x', y', z')$ est à l'intersection de OM et du miroir.
- o est le point principal P de la caméra ordinaire, à savoir l'intersection de l'axe optique et du plan image.
- Le plan image de la caméra ordinaire, considéré comme un espace affine, est rapporté à un repère orthonormé $(o, \vec{i}, \vec{j})$ d'axes (ou), (ov), obtenu par translation du vecteur $\overrightarrow{Oo}$ du repère $(O, \vec{i}, \vec{j})$.
- Étant donné un point M de $\mathcal{E}_3$, le plan (OoM) considéré comme un espace affine est rapporté au repère orthonormé $(O, \vec{I}, \vec{k})$ d'axes $(OX) = (OoM) \cap (Oxy)$ et (Oz).

Dans ce contexte, nous nous proposons d'écrire les équations de la transformation T_{p} qui à un point $M(x, y, z)$ de $(\mathcal{E}_3)$ fait correspondre le point $m(u, v)$ sur les caméras panoramiques de type hyperbolique et parabolique.

3.3.1.4 Les équations de la caméra panoramique hyperbolique

Appelons $(\mathcal{H})$ la trace sur le plan (OLM) du miroir hyperbolique : $(\mathcal{H})$ est une hyperbole. Nous procédons en trois temps.

Équation de $(\mathcal{H})$. Soient (figure 3.3) :
- L le centre de projection (centre optique) de la caméra ordinaire modélisée comme une projection centrale plane ;
- $F_1 = O$ et $F_2 = L$ les deux foyers de $(\mathcal{H})$;
- S_1 et S_2 les deux sommets de $(\mathcal{H})$.

Posons :

$$\left\{ \begin{array}{rcll} S_1S_2 & = & 2a > 0 & \\ F_1F_2 & = & 2e > 0 & (a < e.\ e \text{ est l'excentricité linéaire et } e/a \text{ l'excentricité}) \\ b^2 & = & e^2 - a^2 & (b > 0) \end{array} \right.$$

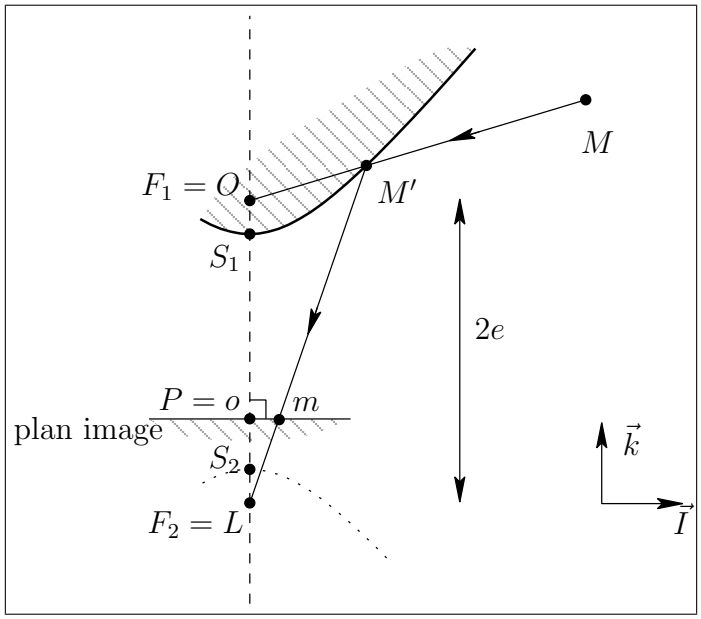


FIG. 3.3: Caméra catadioptrique hyperbolique.

Alors l'équation cartésienne de $(\mathcal{H})$ dans le repère (OXz) s'écrit ([RDO01] p227) :

$$\frac{(z+e)^2}{a^2} - \frac{X^2}{b^2} = 1$$

Calcul de $M'(x', y', z')$. Il suffit de déterminer $\lambda > 0$ satisfaisant à $\overrightarrow{OM'} = \lambda \overrightarrow{OM}$ ($\Longleftrightarrow$ $[x' \quad y' \quad z']^\top = \lambda [x \quad y \quad z]^\top$).

Dans le repère (OXz), λ vérifie aussi :

$$[X' \quad z']^\top = \lambda [X \quad z]^\top \qquad \text{où } X = \sqrt{x^2+y^2} \text{ et } X' = \sqrt{x'^2+y'^2}$$

En écrivant que $M' \in (\mathcal{H})$, on obtient l'équation du second degré en λ :

$$(b^2z^2 - a^2X^2)\lambda^2 + 2eb^2z\lambda + b^4 = 0$$

qui admet pour solutions :

$$\lambda_\varepsilon = \frac{b^2(-ze + \varepsilon a\sqrt{x^2+y^2+z^2})}{b^2z^2 - a^2(x^2+y^2)} \qquad \text{où } \varepsilon = \pm 1$$

Les contraintes physiques, à savoir :

– $\lambda > 0$;

– M ne peut pas être à l'intérieur du miroir (la région hachurée) ;

– M' ne peut pas être sur le second arc de l'hyperbole (courbe en pointillés) ;

conduisent à $\lambda = \lambda_{-1}$.

Calcul de m, obtenu par projection centrale plane de M' sur le plan image de la caméra ordinaire. Posons $OL = 1$ ($= f$, la distance focale), alors il suffit d'écrire que L, m et M' sont alignés (en tenant compte du parallélisme entre le plan image et le plan (Oxy)) pour obtenir :

$$\begin{cases} u = \dfrac{x'}{z' + 2e} = \dfrac{\lambda x}{\lambda z + 2e} \\ v = \dfrac{y'}{z' + 2e} = \dfrac{\lambda y}{\lambda z + 2e} \end{cases}$$

3.3.1.5 Les équations de la caméra panoramique parabolique

Applelons $(\mathcal{P})$ la trace sur le plan (OPM) du miroir parabolique : $(\mathcal{P})$ est une parabole. Nous procédons comme pour le cas hyperbolique.

Équation de $(\mathcal{P})$. Soient (figure 3.4(a)) :
- $F = O$ le foyer de $(\mathcal{P})$;
- S le sommet de $(\mathcal{P})$.

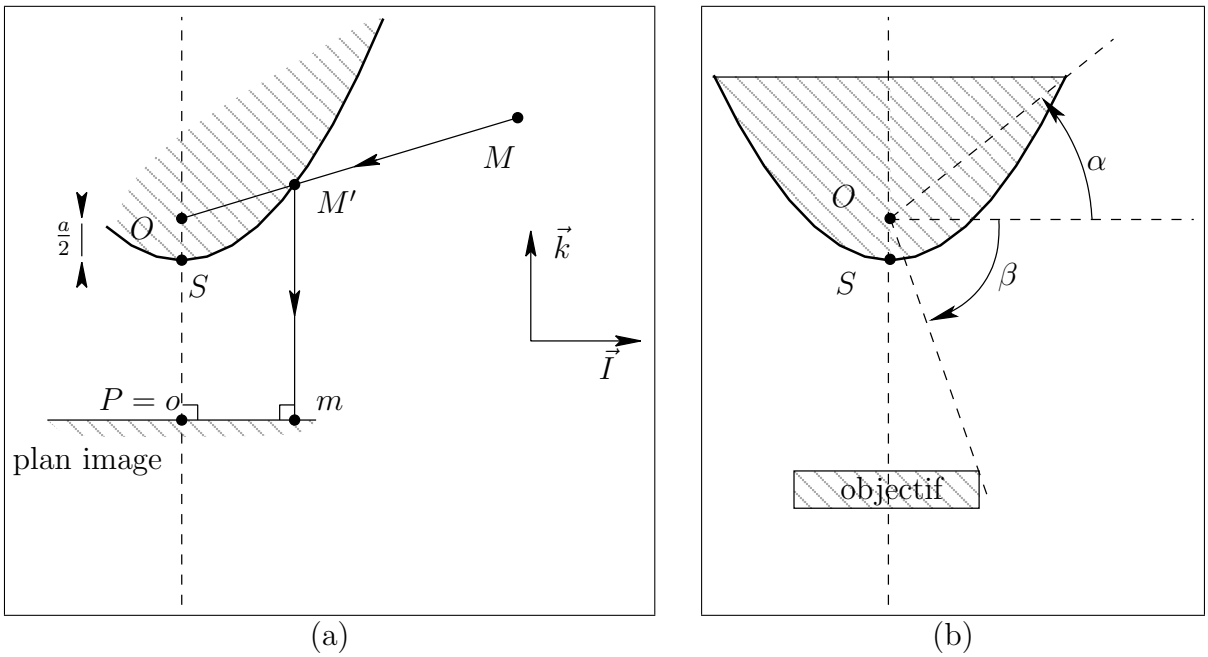


FIG. 3.4: Caméra catadioptrique parabolique : géométrie et champ de vision.

Posons $OS = \frac{a}{2} > 0$ (a est le paramètre de $(\mathcal{P})$ = distance du foyer à la directrice). Alors l'équation cartésienne de $(\mathcal{P})$ dans le repère OXz s'écrit ([RDO01] p223) :

$$z = \frac{X^2}{2a} - \frac{a}{2}$$

Calcul de $M'(x', y', z')$. Comme dans le cas de $(\mathcal{H})$, on aboutit à une équation du second degré en λ :

$$\lambda X^2 - 2\lambda a z - a^2 = 0$$

qui admet pour solutions :

$$\lambda_\varepsilon = \frac{a(z + \varepsilon\sqrt{x^2 + y^2 + z^2})}{x^2 + y^2} \qquad \text{où } \varepsilon = \pm 1$$

Les contraintes physiques, à savoir :
- $\lambda > 0$;
- M ne peut pas être à l'intérieur du miroir (la région hachurée) ;

conduisent à $\lambda = \lambda_{+1}$.

Calcul de m, obtenu par projection orthogonale plane de M' sur le plan image de la caméra ordinaire.

$$\begin{cases} u = x' = \lambda x \\ v = y' = \lambda y \end{cases}$$

3.3.1.6 La caméra panoramique catadioptrique de nos expériences

Nous utilisons :
- le modèle du commerce Panosmart OneShot360 (`http://panosmart.com`) comme objectif catadioptrique. Il se compose :
  - d'un miroir parabolique ;
  - d'une optique (des lentilles) réalisant la projection parallèle.

  Ces deux composants sont reliés par une tige rigide cylindrique d'axe de révolution confondu avec l'axe optique de la caméra ordinaire ;
- d'un appareil photo numérique Nikon Coolpix 4500 de 4 mégapixel comme caméra ordinaire.

L'objectif catadioptrique est vissé sur l'appareil photo.

3.3.1.7 À propos des coordonnées mesurées en pixels sur le plan image

Soit $(o', \vec{i'}, \vec{j'})$ le repère orthonormé d'axes $(o'u')$, $(o'v')$ habituellement utilisé pour une image numérique, à savoir :
- o' est le point en haut et a gauche ;
- $\vec{i'}$ est dans la direction des lignes de l'image ;
- $\vec{j'}$ est dans la direction des colonnes de l'image ;

Les coordonnées (u', v') d'un point m, mesurées en pixels, s'obtiennent à partir de (u, v) par une application affine de la forme :

$$\begin{bmatrix} u' \\ v' \end{bmatrix} = A \begin{bmatrix} u \\ v \end{bmatrix} + b$$

Nous avons effectué un étalonnage du système de prise de vues en supposant que :

- $\vec{i}$ et $\vec{i'}$ (resp. $\vec{j}$ et $\vec{j'}$) sont colinéaires ;
- la caméra ordinaire n'a pas de distorsion (ou qu'elles sont corrigées par ailleurs, voir appendice 7.4) ;
- les pixels sont carrés de surface e^2 ;
- le miroir et la caméra ordinaire sont parfaitement alignés au sens où l'axe de révolution de celui-là se confond avec l'axe optique de celle-ci ;
- le miroir est parfait mais a subi une troncature (intersection avec un plan parallèle à (Oxy), figure 3.4(b) où $\alpha = 15°$).

Dans ces conditions, $A = e\mathrm{Id}_2$ et b contient les coordonnées en pixels du point principal.

L'étalonnage consiste à détecter l'image du bord du miroir sur l'image panoramique : c'est un cercle centré au point principal $P(u'_0, v'_0)$, de rayon r_M satisfaisant à :

$$b = [u_0 \quad v_0]^\top$$

$$r_\mathrm{M} = \frac{ea}{\cos(\alpha)}(\sin(\alpha) + 1)$$

La valeur de a, exprimée en pixels, s'en déduit, ce qui est suffisant pour calculer l'équation de projection directement en coordonnées pixel.

3.3.2 Une caméra panoramique maison

3.3.2.1 Principe

- L'espace affine $\mathcal{E}_3$ est rapporté à un repère orthonormé direct $(O, \vec{i}, \vec{j}, \vec{k})$ d'axes (Ox), (Oy), (Oz).
- Une caméra à projection centrale plane (nous dirons Cam1D) équipée d'un CCD unidimensionnel tourne (angle θ direct mesuré sur Oxy à partir de Ox) autour de l'axe vertical (Oz).
- Au cours de la prise de vue (figure 3.5) :
  - la ligne image de Cam1D reste verticale (parallèle à (Oz)), décrivant ainsi un cylindre de révolution d'axe (Oz) ;
  - le centre optique L de Cam1D, coïncide invariablement avec O.
- Pour $\theta = 0, \Delta_\theta, ..., j\Delta_\theta, ...$, Cam1D capte une image 1D de la scène qui devient la colonne $0, 1, ..., j, ...$ de l'image panoramique.

3.3.2.2 Géométrie

Le point $M(x, y, z)$ est visible seulement lorsque Cam1D vise dans sa direction, c'est-à-dire quand

$$\theta = (\vec{i}, \overrightarrow{OM'})$$

où H est la projection orthogonale de M sur (Oxy).

Dans cette situation le système de coordonnées cylindriques (semi-polaires), dans le repère $(O, \vec{i}, \vec{j}, \vec{k})$ ([RDO01] p193) de son image m est $(\rho = LP = f, \theta, z = v)$. En posant

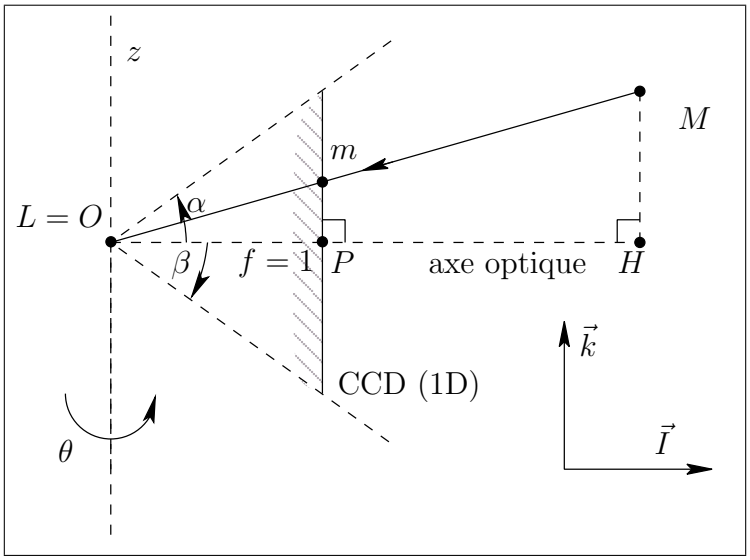


FIG. 3.5: Géométrie de notre caméra maison.

$f = 1$, il vient

$$v = \frac{z}{\sqrt{x^2 + y^2}}$$

Cette acquisition engendre une ligne verticale sur l'image, pour laquelle

$$u = a\theta + b$$

où $(a, b) \in \mathbb{R}^2$ et $\theta \in [0, 2\pi[$.

Si la scène est immobile, ce système équivaut à une projection centrale de centre O sur le cylindre d'axe de révolution Oz engendré par la ligne image de Cam1D.

3.3.2.3 Le prototype

♠ Philippe Puech et Jean Conter, membres de notre laboratoire, ont construit un prototype de caméra sur ce modèle (figure 3.6).

- θ varie dans l'intervalle $[0°, 380°]$ (20° de recouvrement).
- La résolution du CCD (1D) est de 2500 pixels.
- $(\widehat{\overrightarrow{OP}, \overrightarrow{OM}}) \in [-30°, 30°]$
- La résolution horizontale dépend de la vitesse de rotation, qui est programmable : typiquement, une image panoramique de 3500×2500 pixels se construit en 70 secondes (soit 50 lignes par seconde en moyenne).
- La luminosité de l'image peut être ajustée de deux manières :
  - via un diaphragme manuel ;
  - via une table de correspondance programmable qui ramène les mesures de luminosité quantifiées sur 2^{12} niveaux par couleur primaire (caractéristique intrinsèque du

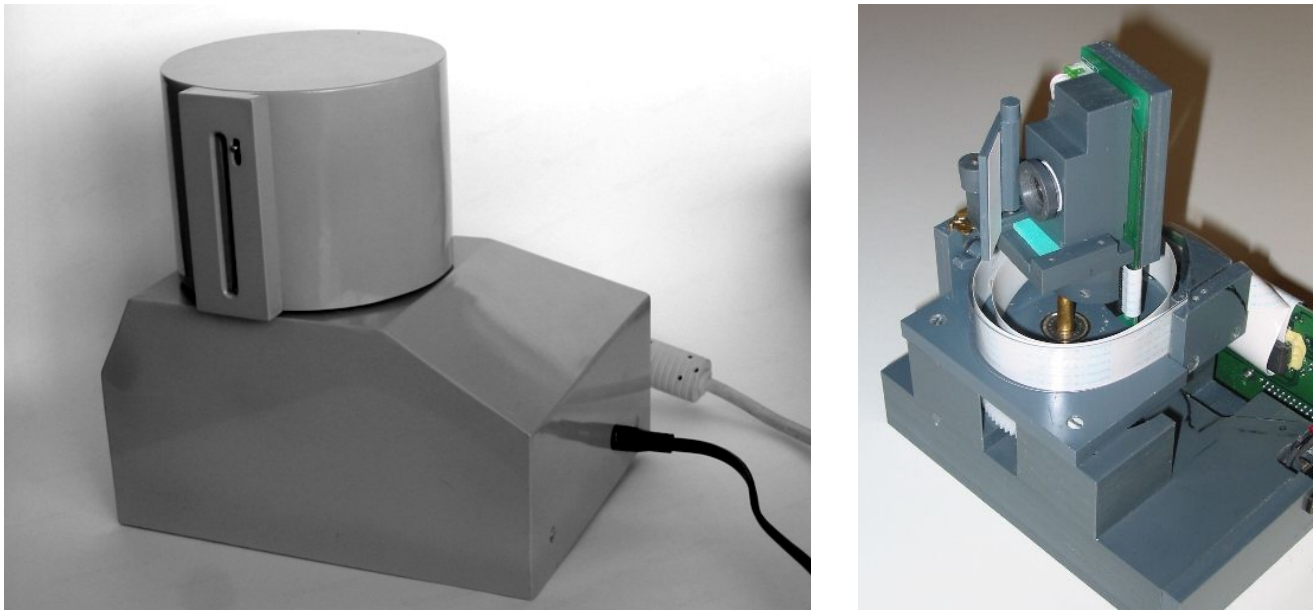


FIG. 3.6: Notre caméra à balayage angulaire : vues extérieure et intérieure.

convertisseur Analogique/Numérique) à 256 niveaux (le choix de la transformation est libre !)

3.3.3 Mise en correspondance des images

Deux images de caméras à projection centrale plane de même centre optique peuvent être mises en correspondance à l'aide d'une homographie 2D (§ 3.4.2.1). Il en va autrement des caméras présentées en § 3.3.

La bijection T qui fait passer des coordonnées 2D sur l'image panoramique aux coordonnées 2D sur l'image issue d'une caméra ordinaire ayant pour centre optique l'unique point de vue de la caméra panoramique (figure 3.7) se décompose en

$$T = T_3 \circ T_2 \circ T_1$$

où

$$T_1 : \begin{array}{ccc} (\mathcal{E}_2, (o_\mathrm{p}, \vec{i_\mathrm{p}}, \vec{j_\mathrm{p}})) & \longrightarrow & (S_\mathrm{p}, (O, \vec{i}, \vec{j}, \vec{k})) \\ (m_\mathrm{p}, (u_\mathrm{p}, v_\mathrm{p})) & \longmapsto & (M', (x', y', z')) \end{array}$$

Le point $m_\mathrm{p}(u_\mathrm{p}, v_\mathrm{p})$ correspond à $m(u, v)$ dans les § 3.3.1.3 et 3.3.2.2

$$T_2 : \begin{array}{ccc} (S_\mathrm{p}, (O, \vec{i}, \vec{j}, \vec{k})) & \longrightarrow & (S_\mathrm{o}, (O, \vec{i}, \vec{j}, \vec{k})) \\ (M', (x', y', z')) & \longmapsto & (m_\mathrm{o}, (x_\mathrm{o}, y_\mathrm{o}, z_\mathrm{o})) \end{array}$$

$$T_3 : \begin{array}{ccc} (S_o, (O, \vec{i}, \vec{j}, \vec{k})) & \longrightarrow & (\mathcal{E}_2, (o_o, \vec{i_o}, \vec{j_o}) \\ (M_o, (x_o, y_o, z_o)) & \longmapsto & (m_o, (u_o, v_o)) \end{array}$$

S_p est la surface soit du miroir (caméra catadioptrique) soit du CCD (1D) (caméra maison).

S_o est la surface du CCD de la caméra ordinaire.

Les caractéristiques géométriques intrinsèques des caméras panoramiques étudiées (catadioptrique et panoramique maison) induisent les propriétés suivantes :

- $T_3 \circ T_2$ est une projection centrale plane ordinaire ; elle s'exprime par une relation linéaire en coordonnées homogènes ([HZ01] eq5.7 p142) ;
- T_1^{-1} est la restriction d'une projection centrale plane à une surface connexe (hyperboloïde, paraboloïde ou cylindre) donc bijective ;
- T_2 est dans la même situation que T_1^{-1}.

Il reste à préciser T_1 pour les caméras étudiées. Reportons-nous pour cela aux figures 3.3, 3.4 et 3.5.

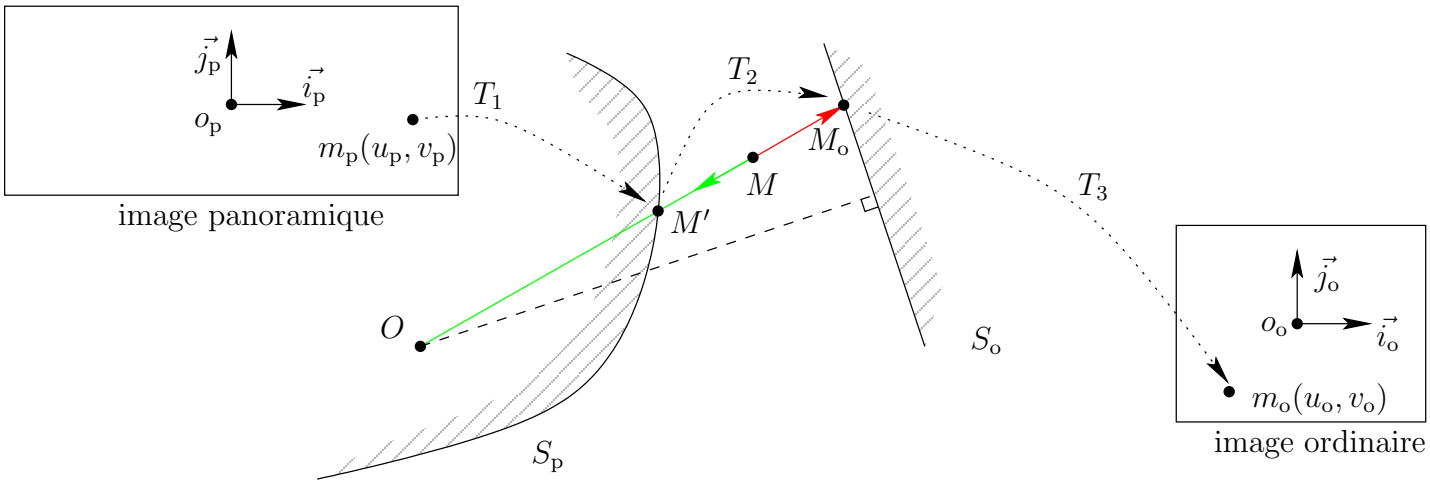


FIG. 3.7: Le rayon de projection OM coupe la surface S_p de la caméra panoramique en M'. Le point correspondant sur le plan image d'une caméra ordinaire est M_o.

3.3.3.1 Caméra panoramique hyperbolique

En :

- écrivant l'équation de l'hyperbole $(\mathcal{H})$ dans le repère (LXz) ;
- cherchant λ satisfaisant à $\overrightarrow{LM'} = \lambda \overrightarrow{Lm_p}$;
- tenant compte du fait que m_p est à la cote $f = 1$ dans ce repère,

nous trouvons aisément

$$\lambda = \lambda_{+1} = \frac{b^2(e + a\sqrt{u_p^2 + v_p^2 + 1})}{b^2 - a^2(u_p^2 + v_p^2)}$$

Par suite, du fait que $\overrightarrow{OM'} = \overrightarrow{OL} + \overrightarrow{LM'}$,

$$T_1(m_\mathrm{p}(u_\mathrm{p}, v_\mathrm{p})) = M' \begin{cases} x' = \lambda u_\mathrm{p} \\ y' = \lambda v_\mathrm{p} \\ z' = -2e + \lambda \end{cases}$$

3.3.3.2 Caméra panoramique parabolique

Les équations du § 3.3.1.5 s'inversent de manière évidente, d'où :

$$T_1(m_\mathrm{p}(u_\mathrm{p}, v_\mathrm{p})) = M' \begin{cases} x' = u_\mathrm{p} \\ y' = v_\mathrm{p} \\ z' = \frac{u_\mathrm{p}^2 + v_\mathrm{p}^2 - a^2}{2a} \end{cases}$$

3.3.3.3 Notre caméra panoramique maison

Il suffit de convertir le système de coordonnées cylindriques en coordonnées cartésiennes :

$$T_1(m_\mathrm{p}(u_\mathrm{p}, v_\mathrm{p})) = M' \begin{cases} x' = \cos((u_\mathrm{p} - b)/a) \\ y' = \sin((u_\mathrm{p} - b)/a) \\ z' = v_\mathrm{p} \end{cases}$$

3.3.3.4 Re-projection

Re-projeter l'image panoramique sur le plan image de la caméra ordinaire consiste à calculer l'image qui serait formée par cette caméra si son centre optique coïncidait avec l'unique point de vue du capteur panoramique.

Nous notons l'image panoramique I_ps (source) et l'image re-projetée I_p (cible). Ce sont des fonctions $\mathbb{R}^2 \to \mathbb{R}$ (resp. $\mathbb{R}^2 \to \mathbb{R}^3$) si les images sont en niveaux de gris (resp. en couleurs)[1].

La relation qui lie les deux images est :

$$I_\mathrm{ps}(\mathbf{m}) = I_\mathrm{p}((T_3 \circ T_2 \circ T_1)(\mathbf{m}))$$

La re-projection consiste à calculer I_p à l'aide de la formule inverse :

$$I_\mathrm{p}(\mathbf{m}) = I_\mathrm{ps}((T_1^{-1} \circ T_2^{-1} \circ T_3^{-1})(\mathbf{m}))$$

Dans cette expression :

[1] Une image peut être vue de trois façons :
- l'image théorique est une fonction $I : \mathbb{R}^2 \to \mathbb{R}$;
- l'image numérique mesurée est une fonction $\widetilde{I} : [\![0, l-1]\!] \times [\![0, h-1]\!] \to [\![0, 255]\!]$. Nous explicitons la relation entre I et $\widetilde{I}$ dans l'appendice 7.2.1 ;
- l'image $\mathcal{I}$, dans le langage courant, est la « représentation d'un être ou d'une chose par les arts graphiques ou plastiques, la photographie, le film, etc. » ([Lar02] p528).

Nous confondons les trois notations quand il n'y a pas d'ambiguïté.

– $T_1^{-1} \circ T_2^{-1} = T_\mathrm{p}$;
– T_3^{-1} est une application affine.

L'expression doit être convertie en coordonnées pixel sur les deux images. L'implémentation est décrite dans l'appendice 7.2.3 utilisant l'échantillonnage brut.

3.3.4 Discussion

Les figures 3.8 et 3.9 présentent des images panoramiques engendrées par les deux systèmes. Les images sont re-projetées sur les faces d'un cube d'axe vertical Oz. Sur la figure 3.8, les faces sont déployées en patron ; sur la figure 3.9, seules sont montrées les quatre faces du cube visibles sur l'image panoramique.

Les deux techniques fournissent une image de la scène à laquelle il manque deux calottes sphériques d'axe Oz (une au-dessous et l'autre au-dessus de la caméra). Pour la caméra catadioptrique, $\alpha = 15°$ et $\beta = 75°$ (figure 3.4) ; ces angles sont de $\alpha = \beta = 30°$ pour notre caméra maison.

La résolution (en pixels/stéradian) du système à balayage angulaire est plus uniforme que celle du système catadioptrique ; sur ce dernier elle devient très faible près du centre de l'image (objets proches de la verticale).

La caméra catadioptrique construit l'image panoramique en une seule prise. Elle peut donc, par exemple, faire des séquences panoramiques au rythme de la vidéo. À l'opposé, pendant la prise de vue de notre caméra maison, la scène et la caméra doivent rester immobiles. L'image fournie par cette caméra peut cependant atteindre de très hautes résolutions parce que la lenteur de l'acquisition autorise l'usage d'un capteur CCD (1D) de grande taille (en termes de pixels).

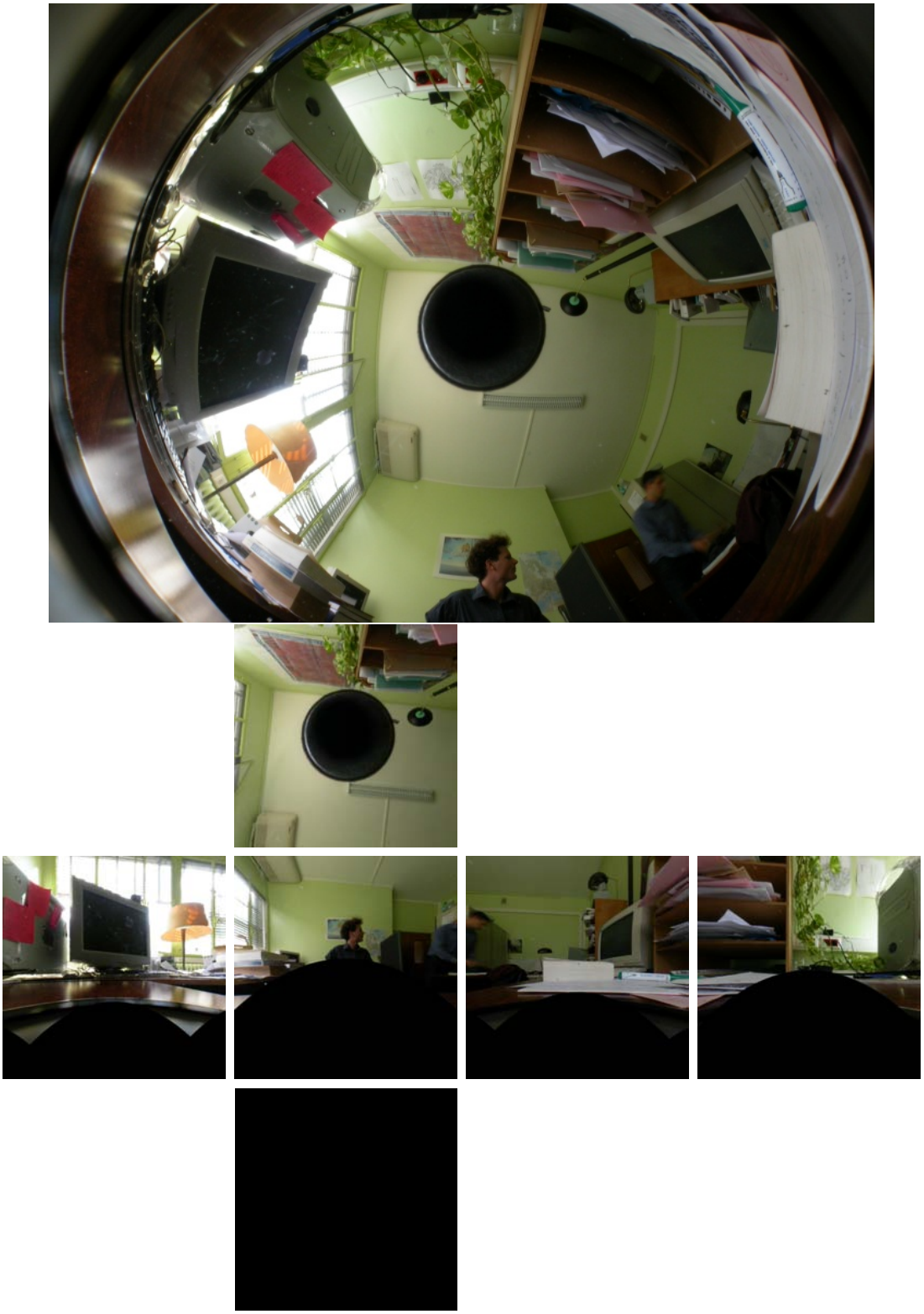


FIG. 3.8: Exemple d'image engendrée par la caméra catadioptrique parabolique et sa projection sur les faces d'un cube (disposées comme un patron).

FIG. 3.9: Exemple d'image engendrée par notre caméra maison et quatre images rectifiées correspondantes. Les lignes droites de la scène apparaissent droites, la partie invisible sur l'image panoramique est en noir.

3.4 Les mosaïques

L'autre manière d'obtenir une image panoramique est de faire une **mosaïque d'images** de plusieurs vues en projection centrale plane de la scène.

Les mosaïques réduisent l'information redondante présente dans une série d'images d'une scène. Des photos de la surface d'une planète peuvent être composées en mosaïque pour produire une carte complète. Dans notre cas, la mosaïque est le moyen d'obtenir un panorama « à la maison » sans matériel spécifique.

Notre contribution touche :

- **la robustesse de la mise en correspondance.** Une estimation initiale grossière est obtenue par appariement de régions (indices visuels grossiers) reposant sur des critères colorimétriques. Celle-ci simplifie les étapes ultérieures de mise en correspondance d'indices visuels plus fins en restreignant le domaine de recherche ;
- **La précision du modèle paramétré.** Nous utilisons un estimateur adapté au type d'erreurs rencontré.

3.4.1 État de l'art

Plusieurs aspects de la génération automatique de mosaïques ont été abordés.

3.4.1.1 Estimation des correspondances inter-images

Fondamentalement, la correspondance entre deux images I et I' doit être estimée. Cette estimation a été abordée sous plusieurs angles.

Cas où les seuls indices visuels sont les niveaux de gris. Une image de différence de niveaux de gris $\Delta I = I - I' \circ T_\theta$ est formée et il s'agit de calculer le modèle paramétré qui minimise $\|\Delta I\|$. Le mécanisme multirésolution est « incontournable » [SS97, SHK98].

Cas où les indices visuels sont plus élaborés (points d'intérêt,...). Deux sous-problèmes se posent dans cette situation ([SMB00, BL02, SZ02]) :

- quels indices ? les plus invariants, les plus faciles à détecter, les plus adaptés à une localisation précise,...
- quels critères pour apparier ces indices ? une mesure de ressemblance est « incontournable » et cruciale (§ 3.4.3.1) : la qualité de l'estimation en dépend directement.

Certains auteurs posent simultanément le problème de l'appariement et de l'estimation [TD03].

La démarche de Kanazawa et Kanatani [KK04]. Ils développent une approche par approximations successives. Cependant, au lieu de construire une hiérarchie de résolutions (pyramide), ils se basent sur une **hiérarchie de modèles** : translation, similitude, transformation affine, homographie ([KK04] § 3). À chaque étage, ils estiment les paramètres du modèle courant par une mise en correspondance de points, utilisant les résultats de l'étage

précédent comme solution initiale. Les estimations sont faites à partir d'un critère LMS ([KK04] appendice C) adapté pour être plus significatif dans ce contexte.

3.4.1.2 Paramètres de la transformation

Dans le cas d'images panoramiques, la transformation peut être exprimée comme une homographie générale ou sous forme d'une fonction dépendant d'un ensemble pertinent de paramètres physiques (typiquement trois angles de rotation et la distance focale).

3.4.1.3 Estimation des relations de voisinage entre images

Quand les images sont traitées en séquence, chaque image est située par rapport à la précédente. Un problème de topologie apparaît quand la caméra a fait un tour complet : les erreurs accumulées des mises en correspondance par couples engendrent un « trou » (biais de recalage) inacceptable entre la première et la dernière image.

Shum et Szeliski [SS97] suggèrent de distribuer cette erreur (sur les angles de rotation et les distances focales) entre les correspondances de paires calculées.

Le problème est plus aigu quand les images sont prises en plusieurs passes à différentes inclinaisons ou quand elles sont dans un ordre aléatoire. Dans le premier cas, un **graphe de voisinage** peut être mis à jour itérativement d'après les résultats d'une mise en correspondance à plusieurs images de référence [SHK98]. Dans le second cas, un graphe de voisinage initial peut être construit en fonction de statistiques sur le nombre d'éléments similaires entre chaque paire d'images [SZ02].

Enfin, quand on a une approximation des paramètres de la transformation pour chaque image, ils peuvent être ré-estimés simultanément (**ajustement de faisceaux**).

3.4.1.4 Visualisation

Un autre aspect est la visualisation de l'image panoramique. Les images peuvent être projetées sur un plan (dans le système de coordonnées de l'une d'elles), sur un cylindre ou sur une sphère. La question qui se pose est : comment combiner les images sur les régions où plusieurs d'entre elles se superposent ?

Dans l'idéal, elles devraient être exactement semblables, de sorte qu'une simple **moyenne de niveaux de gris** (ou de couleurs) suffise. En réalité, procéder ainsi introduit des « fantômes » aux endroits où les modèles géométrique ou photométrique ne sont pas vérifiés. Ceci peut arriver dans différentes situations : le modèle de projection centrale plane est inexact (distorsions non compensées), des objets apparaissent ou disparaissent entre les images, la luminosité change (localement ou globalement).

Les imprécisions locales peuvent être compensées en déplaçant légèrement et en ajustant la luminosité de petits blocs (de l'ordre de 32×32 pixels) pour les faire paraître plus cohérents entre les images [SS97, UES01].

À plus grande échelle, les couleurs de deux images peuvent être combinées avec des pondérations variant continûment. Brown et Lowe [BL03] proposent de synthétiser les approches globale et locale en une méthode **multi-bande** : les basses fréquences spatiales des

images sont combinées avec des pondérations progressives, mais ils utilisent des transitions brusques pour les hautes fréquences.

Les gros objets occultants peuvent être identifiés et supprimés à l'aide d'un graphe de correspondance qui les identifie entre les images [UES01].

3.4.1.5 Logiciels du commerce

En somme, la génération de mosaïques est une opération déjà largement explorée. De nombreux produits commerciaux sont disponibles. Pour les « amateurs », les appareils photo Canon sont livrés avec PhotoStitch. Pour les professionnels, RealViz Stitcher permet d'aligner finement les images, avec éventuellement une aide « manuelle ». Ces deux logiciels nécessitent une intervention de l'utilisateur pour indiquer la position approximative des images. Ils fournissent soit une image panoramique soit un fichier QuickTime VR.

3.4.1.6 Ce que nous en retenons

Dans notre travail, nous nous sommes surtout intéressés à l'estimation de la correspondance entre deux images.

Nous procédons par approximations successives, au niveau du modèle, à savoir :

- une translation à partir d'indices visuels grossiers (des régions) ;
- une homographie à partir d'indices visuels précis (points d'intérêt).

Nous délaissons volontairement la piste d'un appariement dense car :

- l'hypothèse est celle d'une transformation globale inter-images, à savoir une homographie ;
- le but est le calcul des huit paramètres de cette homographie.

3.4.2 Modélisation

3.4.2.1 Modèle homographique

Nous nous plaçons dans les conditions théoriques suivantes :

- la caméra suit un modèle de projection centrale plane ;
- le centre optique est invariant ;
- la scène observée est figée (ce qui implique l'absence d'occultations temporaires).

La caméra peut changer de direction de visée et de distance focale.

Nous avons plusieurs vues d'une scène. Nous traitons les images séquentiellement, par paires I et I' (figure 3.10, laquelle satisfait imparfaitement la troisième condition théorique). Un point M de la scène est vu sur les images I et I'. Nous cherchons la fonction T qui transforme le point $\mathbf{m}(x, y)$ image de M sur I en $\mathbf{m}'(x', y')$, sa projection sur I'.

FIG. 3.10: Images à combiner pour former une mosaïque (avec un personnage occultant).

Cette transformation est une **homographie** ([HZ01], p194) : il existe $\theta = [h_1 \cdots h_9]^\top$ tel que, pour tout M,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = T\left(\theta, \begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} \dfrac{h_1 x + h_2 y + h_3}{h_7 x + h_8 y + h_9} \\ \\ \dfrac{h_4 x + h_5 y + h_6}{h_7 x + h_8 y + h_9} \end{bmatrix}$$

La **matrice d'homographie**

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix}$$

est inversible. L'homographie dépend de 9 paramètres, mais ils sont définis à un facteur près, le nombre de degrés de liberté est donc 8. Le vecteur de paramètres θ doit être estimé.

3.4.2.2 Relâchement des hypothèses

En pratique, les hypothèses ci-dessus ne sont vérifiées qu'approximativement. Ainsi :

1. Des distorsions géométriques (radiales, voire tangentielles) sont quasiment inévitables sur tout objectif photographique. Si elles sont trop importantes, les bords des images sont inexploitables. Les distorsions qui peuvent s'exprimer comme une transformation 2D appliquée à une image parfaite peuvent être étalonnées et corrigées (appendice 7.4).

2. Un déplacement du centre optique peut dévoiler sur l'image I' des zones occultées sur I et réciproquement (dans le cas d'une scène non-plane) : certains parlent de « parallaxe ». Ceci a des conséquences sur la précision du calcul de T et sur la crédibilité des images résultantes dès que ces zones sont de surface tangible (typiquement un pixel). Cela dépend de l'amplitude du déplacement, de la distance caméra-scène, de la profondeur de champ de la scène. Si la scène est approximativement polyédrique, il serait envisageable, à la manière de M. Lhuillier [Lhu00] (triangulations de vues conjointes), de calculer des homographies locales. Dans tous les autres cas, la seule ressource est de traiter le couple d'images comme un stéréogramme classique.
3. Une occultation temporaire (par exemple le personnage sur la figure 3.10), localisée dans la zone de recouvrement, peut avoir des conséquences analogues à celles mentionnées au point 2.

3.4.3 Résolution

3.4.3.1 La mise en correspondance

L'estimation de θ repose sur la **mise en correspondance**, qui consite à :

1. « Désigner » sur I des **indices visuels** qui sont :
   - **présents** sur l'autre image (propriété d'invariance) ;
   - **localisés**, c'est-à-dire qu'on peut les réduire aux coordonnées d'un point.
2. Retrouver ces indices visuels sur l'image I' (apparier). Deux stratégies sont envisageables :
   - soit chercher sur I' les endroits les « plus ressemblants » aux indices visuels de I, par une méthode de corrélation ;
   - soit détecter les indices visuels sur I' et apparier les indices visuels des deux images.
3. Former l'ensemble des couples de points $(\mathbf{m}, \mathbf{m}')$ où $\mathbf{m}$ représente un indice visuel sur I et $\mathbf{m}'$ un sur I'.

Par la suite, on trouve la transformation qui permet « au mieux » de passer de $\mathbf{m}$ à $\mathbf{m}'$, pour tous les couples de points mis en correspondance.

3.4.3.2 Résolution en étapes

Nous proposons une estimation des paramètres θ en trois temps :

1. estimer de façon robuste un vecteur $\vec{t}$ de translation, comme première approximation de la transformation T ;
2. estimer une première homographie à partir d'un jeu de couples de points d'intérêt $\{(\mathbf{m}_i, \mathbf{m}'_j)_{i,j}\}$ appariés en tenant compte de $\vec{t}$;
3. itérer l'estimation de θ à partir de jeux $\{(\mathbf{m}_k, \mathbf{m}'_l)_{k,l}\}$ améliorés ;

$(T, S, L) \leftarrow$ décomposition de I dans l'espace de couleurs teinte, saturation, luminosité
$E \leftarrow$ `disque`(10) -- élément structurant : un disque de 10 pixels de rayon
$V \leftarrow$ `gradient_morphologique`(E, V) -- gradients morphologiques
$S \leftarrow$ `gradient_morphologique`(E, S)
$H \leftarrow$`gradient_morphologique_circulaire`(E, H)
$M \leftarrow \max(H, S, V)$ -- maximum pixel à pixel
$v \leftarrow$ `quantile`$(M, 1/2)$
$R \leftarrow$ `seuillage`(M, v) -- l'ensemble des pixels **en dessous** du seuil v
supprimer les sous-ensembles connexes de R de moins de 100 pixels
supprimer les sous-ensembles connexes de $\bar{R}$ de moins de 100 pixels
-- les régions résultantes sont les sous-ensembles connexes de R

Fonctions utilisées :
- `gradient_morphologique`(E, I) : gradient morphologique ([SM94] p28) de I par l'élément structurant E. C'est la dilatation de I par E moins (pixel à pixel) l'érosion de I par E,
- `gradient_morphologique_circulaire`(E, I) : idem, mais la différence est calculée modulo 256 (qui est notre niveau de gris maximum) pour que les quantités circulaires (comme les angles ou les teintes) soient comparées de manière pertinente,
- `quantile`(I, p) trouve le seuil tel que qu'une proportion p de pixels soit en-dessous du seuil (utilise l'histogramme de I).

Algorithme 3.1: Extraction de régions de couleur uniforme d'une image en couleurs I.

3.4.4 Étape 1 : la translation

Ici, nous apparions des régions pour estimer une translation. Cette approche repose sur les considérations suivantes : ♠

- Si entre la prise de la vue I et la vue I', la caméra effectue une rotation autour de son centre optique d'un angle α petit et si le « champ visuel » est réduit, alors le modèle de translation constitue une approximation exploitable de la transformation T restreinte aux sous-images utiles de I et I' (appendice 7.1).
- Une translation étant définie par seulement deux paramètres, un seul couple d'indices visuels homologues suffit en théorie. En pratique, un tel résultat ne serait correct qu'au voisinage de ces deux indices, d'où la nécessité d'une recherche globale et robuste.
- Le vecteur de translation $\vec{t}$ est utilisé en **schéma de prédiction** de l'homologue $\mathbf{m}'$ sur I' de tout point d'intérêt $\mathbf{m}$ de I : une technique de corrélation sur des **voisinages restreints** affine le résultat.

3.4.4.1 Extraction des régions

Les régions sont détectées à l'aide d'un gradient morphologique (algorithme 3.1). Le critère de segmentation[2] est l'uniformité de l'image sur la région. La segmentation n'est pas faite pour que les régions correspondent nécessairement des objets dans la scène. C'est cependant le cas sur des images de bâtiments et autres structures artificielles comprenant de nombreuses surfaces unies.

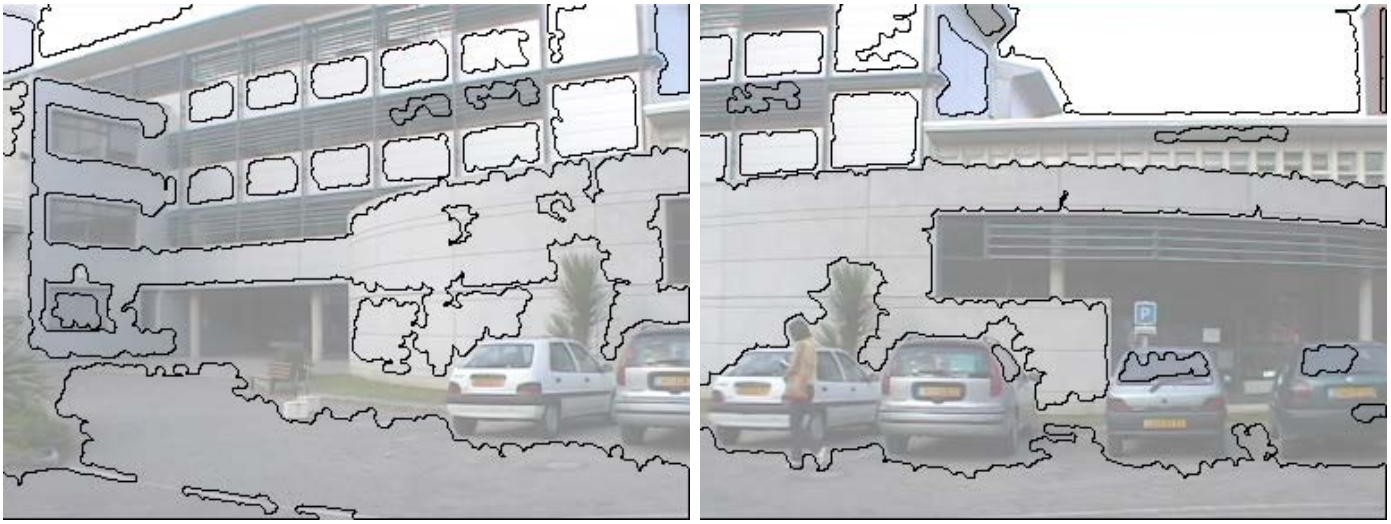


FIG. 3.11: Régions détectées sur les images de la figure 3.10.

La figure 3.11 montre les régions résultantes. Elles ne correspondent pas nécessairement à ce qu'un œil humain aurait sélectionné, mais l'estimation est suffisamment robuste pour s'en accommoder. Nous appelons $\{R_i\}_i$ et $\{R'_j\}_j$ l'ensemble des régions de I et I'.

3.4.4.2 Mise en correspondance

À chaque région R est associé un vecteur d'attributs $a(R)$, caractérisant :

sa forme : taille, dimensions de la boîte englobante, moments centrés d'ordre deux ;

ses niveaux de gris : moyenne et écart-type, vecteur gradient moyen ;

sa couleur : teinte et saturation moyennes.

À chaque paire de régions (R_i, R'_j) est associée une distance de Mahalanobis d_{ij} définie par :

$$d_{ij} = \sqrt{(a(R_i) - a(R'_j))^T \Lambda^{-1} (a(R_i) - a(R'_j))}$$

[2]Nous entendons « segmentation » au sens économique : « classification d'individus en groupes homogènes sur la base de certains critères » [Rob03]. Ici, les individus sont des pixels et les groupes des régions.

où Λ est la matrice de variance-covariance des composantes des vecteurs d'attributs, permettant de prendre en compte les ordres de grandeur des composantes, ainsi que d'éventuelles corrélations.

Une distance faible indique que les régions correspondent probablement au même objet dans la scène. Le coefficient $1/d_{ij}$ est donc un « **taux de confiance** » accordé à cette paire. Nous éliminons du processus les paires candidates dont le taux de confiance est inférieur à un certain seuil.

Chaque paire de régions suggère aussi un vecteur de translation : c'est la translation qui superpose aux mieux les deux régions :

- si les deux régions ne touchent pas le bord de l'image, ce vecteur est la différence entre leurs centres de gravité :
$$\vec{t_{ij}} = \overrightarrow{g_i g'_j}$$
où g_i (resp. g'_j) désigne le centre de gravité de R_i (resp. R'_j) ;
- sinon, nous le déduisons des boîtes englobantes. Les composantes x_t et y_t de $\vec{t_{ij}}$ sont déterminées séparément. En notant, pour $z = x$ ou y :
  - $[z_\mathrm{min}, z_\mathrm{max}]$ les bornes de la boîte englobante de R_i dans la dimension z ;
  - $[z'_\mathrm{min}, z'_\mathrm{max}]$ les bornes de R'_j ;
  - $[0, z_\mathrm{im}]$ les bornes de l'image (z_im est la largeur ou la hauteur) ;
  - $\tau_\mathrm{min} = (z_\mathrm{min} = 0$ ou $z'_\mathrm{min} = 0)$ indique si une des régions « touche » le bord inférieur de l'image ;
  - $\tau_\mathrm{max} = (z_\mathrm{max} = z_\mathrm{im}$ ou $z'_\mathrm{min} = z_\mathrm{im})$ idem pour le bord supérieur ;

  nous déterminons z_t en fonction de la bordure la plus fiable :

$$z_\mathrm{t} = \begin{cases} \frac{z'_\mathrm{max} - z_\mathrm{max}}{2} + \frac{z'_\mathrm{min} - z_\mathrm{min}}{2} & \text{si } (\text{non } \tau_\mathrm{min}) \text{ et } (\text{non } \tau_\mathrm{max}) \\ z'_\mathrm{max} - z_\mathrm{max} & \text{si } \tau_\mathrm{min} \text{ et } (\text{non } \tau_\mathrm{max}) \\ z'_\mathrm{min} - z_\mathrm{min} & \text{si } (\text{non } \tau_\mathrm{min}) \text{ et } \tau_\mathrm{max} \\ \text{indéfini} & \text{sinon} \end{cases}$$

3.4.4.3 Estimation de la translation

Les points pondérés. Le vecteur translation et le « taux de confiance » associé peuvent être considérés comme un **point pondéré**

$$(\vec{t_{ij}}, 1/d_{ij})$$

Les points dont le poids est élevé sont pour la plupart près de la « bonne » translation[3].

L'estimation. Ainsi, trouver cette translation se traduit par un problème de localisation dans l'espace des paramètres (on parle parfois de **transformée de Hough généralisée** [BL02]). La figure 3.12 montre l'ensemble de points pondérés correspondant aux régions de la figure 3.10.

[3] L'expression « bonne » translation est abusive parce que le modèle de translation est une approximation. Il est donc difficile de quantifier la qualité d'une translation sans recourir au modèle exact.

Une autre interprétation est que l'ensemble des points pondérés situés dans l'espace des paramètres est une approximation (dicrète, empirique) de la vraisemblance (probabilité d'avoir les données sachant les paramètres). Dans ce cas, l'estimation revient à calculer mesurer les modes de la fonction de densité de probabilité qui approche la vraisemblance. Une méthode de calcul classique pour ces modes est la procédure de *mean shift* ([Mee01] p50).

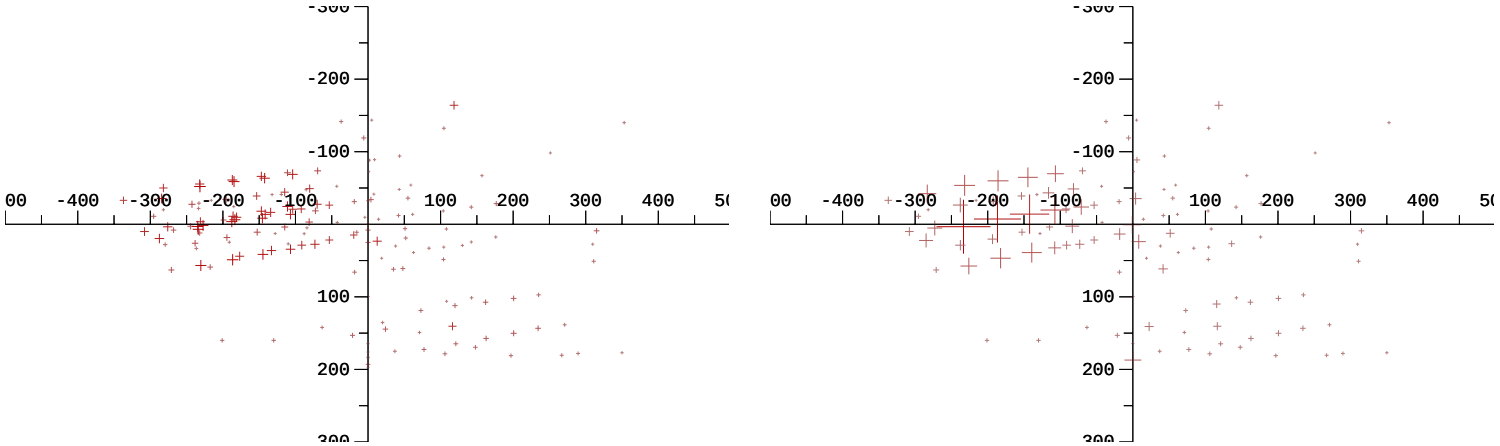


FIG. 3.12: Ensemble des translations (= points pondérés), et version agglomérée. Le poids de chaque point est indiqué par la taille de la croix. Le premier candidat renvoyé par l'algorithme est $(-233, 3)$.

Nous avons choisi de simplifier l'ensemble par plusieurs opérations d'« agglomération » qui remplacent des points proches les uns des autres par leur barycentre (algorithme 3.2).

Résultat. En sortie, le tableau S donne des propositions de translations, distinctes, par ordre de vraisemblance décroissant. Le plus souvent, la proposition $\vec{t} = S_0$ est la « bonne », mais il est bien évident que rien ne nous oblige à conserver une seule translation candidate [TD03].

Pour notre exemple, la translation obtenue donne la figure 3.13.

3.4.5 Étape 2 : l'homographie

Nous utilisons la translation comme point de départ pour estimer l'homographie. Cette estimation est basée sur une mise en correspondance de points en plusieurs étapes (algorithme 3.3).

3.4.5.1 Les points d'intérêt

Les points détectés doivent être de bons indices visuels pour l'appariement. Ils doivent également être équitablement répartis sur l'image.

Classer le tableau S par poids décroissants
-- pour que les plus « lourds » soient traités en priorité
Pour r **de** 5 **à** 20 **par** 5 **faire** -- rayons en pixels[4]
  $i \leftarrow 0$
  Répéter
    $n \leftarrow$ longueur(S) ; $j \leftarrow i+1$
    Répéter -- trouver un point S_j distant de moins de r de S_i
      $j \leftarrow j+1$
    jusqu'à $j = n$ ou $\|S_j - S_i\| < r$ -- trouvé !
    Si $j < n$ **alors** -- agglomérer S_i et S_j
      $p_{\text{new}} \leftarrow$ barycentre de (S_j, S_i)
      supprimer S_j et S_i
      insérer p_{new} à S en conservant l'ordre des poids (interclassement)
      $i \leftarrow$ index de p_{new}
    sinon
      $i \leftarrow i+1$
    finsi
  jusqu'à $i = n$ -- tous les points restants sont distants de plus de r pixels.
finpour

Algorithme 3.2: Simplification par agglomération de points voisins d'un champ de points pondérés S.

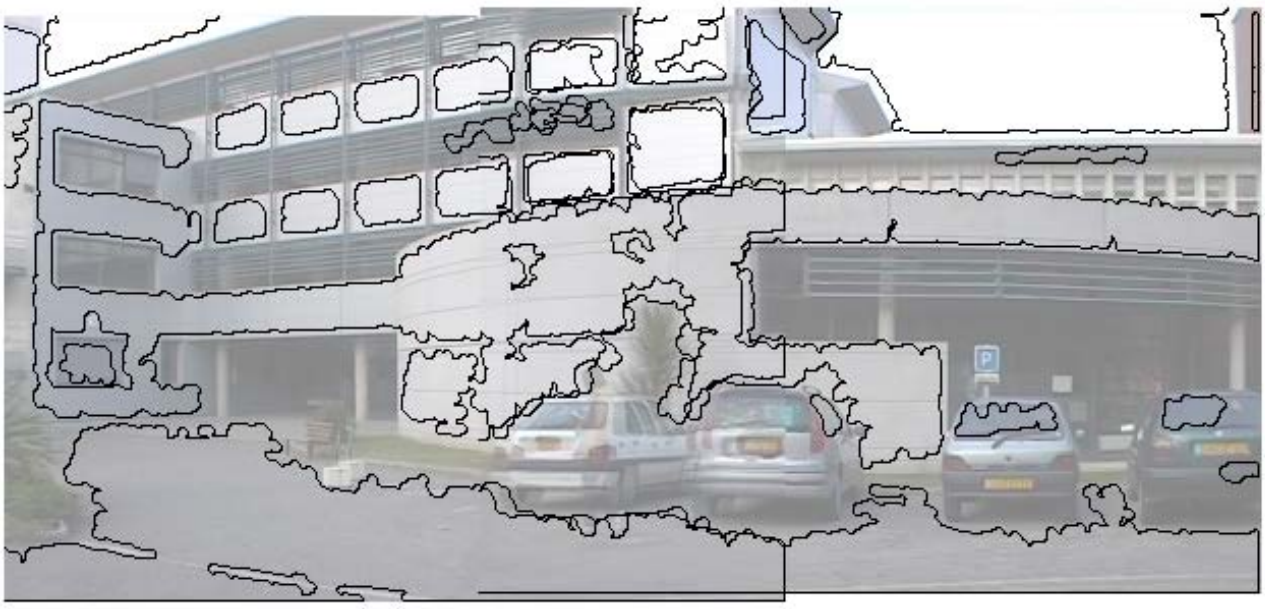


FIG. 3.13: Image construite à partir d'un modèle de translation.

Détecter les points d'intérêt $\mathbf{m}_i$ dans l'image I
Trouver sur I' le correspondant $\mathbf{m}'_i$ de $\mathbf{m}_i$ en s'appuyant sur $\vec{t}$
$\widehat{\theta} \leftarrow$ estimation robuste de θ avec FAST-LTTS appliqué à un modèle linéaire
-- Les appariements aberrants sont éliminés.
Répéter
$\widehat{\theta'} \leftarrow$ estimation de θ avec ODRPACK appliqué à un modèle non linéaire
-- θ sert de solution initiale
utiliser $\widehat{\theta'}$ pour améliorer les appariements $(\mathbf{m}_i, \mathbf{m}'_i)$
$\widehat{\theta} \leftarrow \widehat{\theta'}$
jusqu'à stabilité des appariements

Algorithme 3.3: Algorithme d'estimation robuste de l'homographie θ.

L'étude faite par Schmid *et al.* [SMB00] est convaincante sur la pertinence des détecteurs fondés sur la fonction d'autocorrélation de l'image, introduits par Moravec. Parmi eux le détecteur de Harris (et plus particulièrement la version améliorée) est le plus efficace au sens des critères du § 3.4.3.1. Les maxima de cette fonction indiquent une forte variation locale et isotrope des niveaux de gris (on parle souvent de « détecteur de coins »).

Adaptation. Nous sélectionnons les pixels maximisant la fonction de Harris ([SMB00], p156) sous deux contraintes visant à améliorer leur répartition :

- le point sélectionné doit être un maximum local (deux points voisins ne peuvent pas être sélectionnés) ;
- l'image est divisée en pavés. Nous appliquons un quota sur le nombre de points sélectionnés dans chaque pavé (ceci évite que tous les points sélectionnés s'accumulent dans un coin de l'image).

Les points d'intérêt sont détectés sur l'image I uniquement.

3.4.5.2 Mise en correspondance des points

Pour le point d'intérêt $\mathbf{m}$, la translation approximative suggère que son correspondant est en $\widetilde{\mathbf{m}} = \mathbf{m} + \vec{t}$ sur I'. Donc, si $\widetilde{\mathbf{m}}$ est à l'intérieur de I', nous cherchons le point correspondant sur un disque (de rayon 10 pixels) de centre $\widetilde{\mathbf{m}}$. Nous choisissons le point sur ce disque où la corrélation croisée centrée et normalisée entre voisinages 5×5 est la plus forte (figure 3.14).

Cette mesure est invariante aux changements affines de luminosité. Pour résister aux déformations dues à l'homographie, il faudrait adapter la forme de la fenêtre de corrélation comme le font Lotti et Giraudon [JLL94], Lan [Lan97] et Kanazawa et Kanatani [KK04]. Ce n'est pas crucial dans notre cas, parce que la fenêtre est relativement petite.

À l'issue de cette étape, nous avons un ensemble de paires de points $\{(\mathbf{m}_i, \mathbf{m}'_i), i \in [\![1, n]\!]\}$ à partir duquel l'homographie doit être estimée.

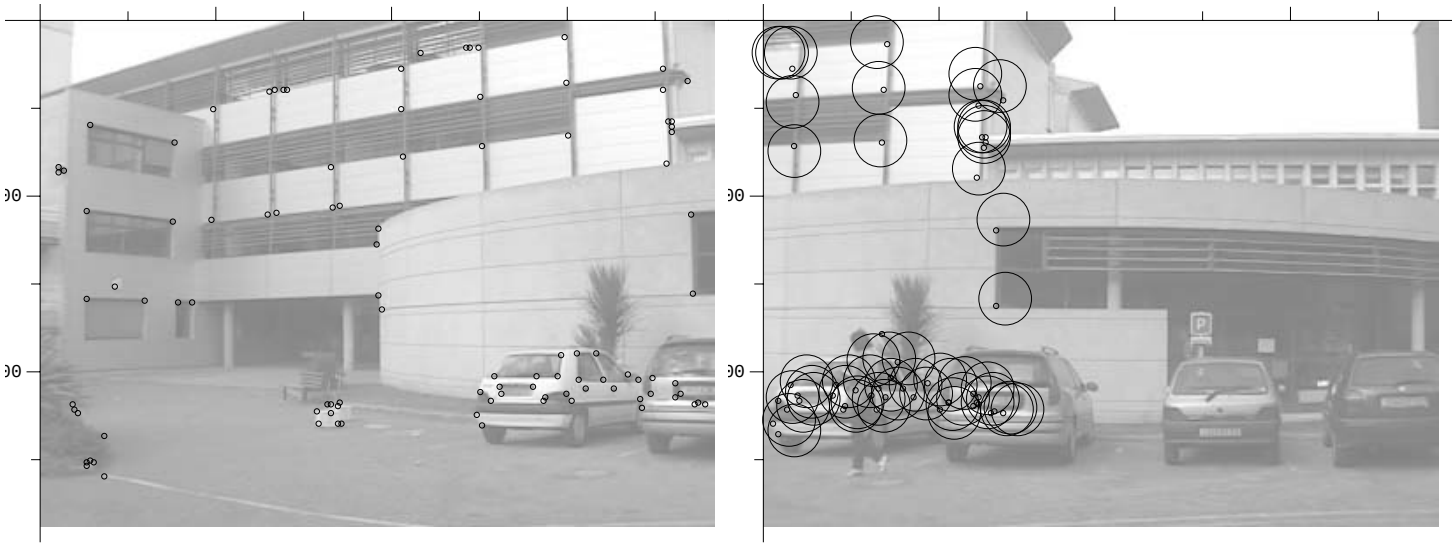


FIG. 3.14: Points d'intérêt sur la première image. Disque de recherche et points correspondants sur la seconde.

3.4.5.3 Estimation robuste de l'homographie

L'estimation robuste des paramètres θ repose sur un modèle linéaire en θ. Il s'écrit sous la forme de la « boîte noire » (§ 2.2) suivante :

$$X = \begin{bmatrix} x \\ y \\ x' \\ y' \end{bmatrix} \in \mathbb{R}^4 \rightarrow \boxed{\begin{array}{c} \theta \in \mathbb{R}^9 \\ \hline m = 4; n; p = 9; q = 2 \end{array}} \rightarrow Y = C(X)\theta = 0_2 \in \mathbb{R}^2$$

où C est

$$C([x \quad y \quad x' \quad y']^\top) = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -xx' & -yx' & -x' \\ 0 & 0 & 0 & x & y & 1 & -xy' & -yy' & -y' \end{bmatrix} \tag{3.1}$$

Le critère. Cette formulation mène à un critère de « distance algébrique » ([HZ01] p76) que nous minimisons avec un critère sélectif :

$$P_{\text{DA}} \left\{ \begin{array}{l} \min \sum_{i \in \mathcal{I}} \|C([x_i \quad y_i \quad x'_i \quad y'_i]^\top)\theta\|^2 \\ \mathcal{I} \in S_n^h, \theta \in \mathcal{S}_9 \end{array} \right.$$

où $\mathcal{S}_9$ est la sphère unité de dimension 9.

Cette formulation implicite est équivalente à un estimateur TLS ([Wat82], [GR70], [GL80], [HV91] p45 eq2.33 et 2.34) restreint à un sous-ensemble d'expériences. Nous utilisons donc l'algorithme FAST-LTTS pour la résoudre (§ 2.6.4.1).

Les erreurs. La formulation LTTS sous-jacente à (P_{DA}) suppose que la matrice

$$\begin{bmatrix} C([x_1 & y_1 & x'_1 & y'_1]^\top) \\ & \vdots & \\ C([x_n & y_n & x'_n & y'_n]^\top) \end{bmatrix} \in \mathbb{R}^{2n\times 9}$$

est affectée des erreurs de mesure suivantes :

- pour un nombre limité d'expériences (donc de lignes), des aberrations ;
- pour les autres, des erreurs gaussiennes indépendantes et de même écart-type sur les entrées.

Cette seconde proposition est contredite par l'expression de C (équation (3.1)). En effet :

- les valeurs à 0 et à 1 sont sans erreur ;
- l'erreur sur une coordonnée x n'a aucune raison d'être du même ordre de grandeur que l'erreur sur un produit de coordonnées xy.

La première objection peut être prise en compte partiellement en reformulant (P_{DA}) sous forme d'un problème **LS/TLS mixte** [HV91] : les deux colonnes ne contenant que valeurs à 1 ou 0 sont alors considérées comme sans erreur.

Il est impossible de traiter de la même manière les autres blocs à 0 et les différences d'ordre de grandeur entre coefficients : c'est un problème **TLS structuré**. Ceci nous contraint à basculer dans un modèle non linéaire.

L'apport de cette étape est une estimation $\widehat{\theta}$ et un ensemble de paires de points sans aberration I_{LTTS}. Dans la suite, nous ne retenons que les paires de cet ensemble, ce qui nous permet d'utiliser un modèle non robuste ODR.

3.4.5.4 Estimation précise

Nous affinons le résultat de l'estimateur robuste en intégrant un modèle d'erreur plus réaliste. ♠

Nous formulons le problème d'estimation non linéaire par la boîte noire suivante issue de la « distance géométrique » ([HZ01] p77) :

$$X = \begin{bmatrix} x \\ y \end{bmatrix} \in \mathbb{R}^2 \rightarrow \boxed{\begin{array}{c} \theta = [h_1 \cdots h_8]^\top \in \mathbb{R}^p \\ \hline m = 2; n; \\ p = 8; q = 2 \end{array}} \rightarrow Y = \begin{bmatrix} x' \\ y' \end{bmatrix} = T(\theta, X) \in \mathbb{R}^2$$

Nous considérons que la localisation des points sur les deux images (les entrées et les sorties) est sujette à un bruit gaussien. L'estimateur approprié est donc la régression en distance orthogonale (§ 2.5.2.2).

L'intensité du bruit n'a pas de raison d'être différente d'une image à l'autre, donc nous n'utilisons pas de pondération. Le problème correspondant est :

$$P_{\mathrm{DG}} \left\{ \begin{array}{l} \min \sum_{i=1}^{n} \|T(\theta, \mathbf{m}_i + \delta_i) - \mathbf{m}'_i\|^2 + \|\delta_i\|^2 \\ \theta \in \mathbb{R}^8, \delta_i \in \mathbb{R}^2, i = 1..n \end{array} \right.$$

Nous retrouvons ici le critère de Hartley et Zisserman ([HZ01], 3.8 et 3.16). On sait que sous les hypothèses de normalité et d'indépendance des erreurs commises sur les $\mathbf{m}_i$ et $\mathbf{m}'_i$, la solution $\widehat{\theta}$ vérifie le principe du maximum de vraisemblance de Gauss-Fisher.

La figure 3.15 montre l'image panoramique résultante.

FIG. 3.15: Mosaïque construite avec notre méthode. L'image I' est re-projetée sur I en utilisant l'homographie (§ 3.4.6.5). Sur la région commune, la couleur des pixels est la moyenne des couleurs correspondantes. Le personnage occultant apparaît donc comme un « fantôme »

3.4.6 Expérimentation

Nous avons fait des expériences sur des données de synthèse et réelles.

Dans les paragraphes suivants, nous évaluons la robustesse du processus. Nous traitons une série d'images de synthèse (figure 3.16). La caméra virtuelle a un angle de vue horizontal de 60° et un angle de rotation horizontal de $\alpha = 20°$ sépare deux images successives. Les images sont *a priori* idéales pour l'appariement : elles sont sans distorsion géométrique ou photométrique.

Comme la transformation exacte T_{e} est connue, elle sert de « réalité terrain » par rapport à laquelle nous évaluons le résultat de l'algorithme T_{a}. Nous utilisons comme

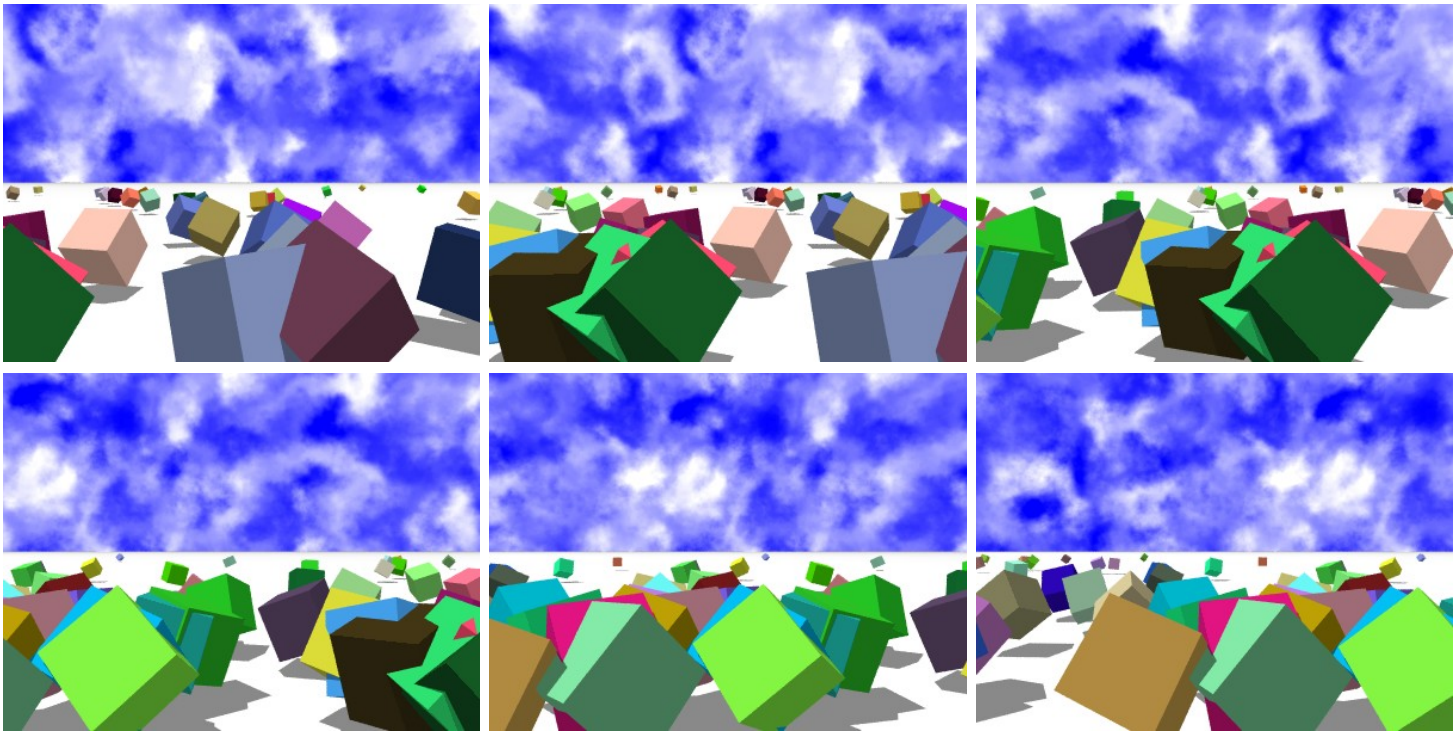


FIG. 3.16: Séquence d'images de synthèse servant à évaluer la robustesse de l'appariement.

critère d'erreur E la distance moyenne de $T_e(\mathbf{m})$ à $T_a(\mathbf{m})$ sur les points $\mathbf{m}$ de la région connue où les images se recouvrent.

3.4.6.1 Bruit gaussien

Nous ajoutons un bruit gaussien (moyenne 0, écart-type σ) à tous les pixels des images (aux trois couleurs primaires). Le graphe de la figure 3.17(a) montre comment E évolue en fonction de σ (les couleurs sont discrétisées entre 0 et 255). Pour un niveau de bruit raisonnable, l'algorithme résiste bien.

3.4.6.2 Bruit JPEG

Quand les images sont stockées sur un appareil photo numérique ou diffusées sur Internet, la source de bruit dominante est due à la compression JPEG/JFIF. Le graphe de la figure 3.17(b) montre l'erreur en fonction du facteur de qualité JPEG (une valeur de 100 correspond à un taux de compression de 1/4.8, $50 \rightarrow 1/30$, $0 \rightarrow 1/180$). On voit que, pour des qualités courantes (qualité > 50), le résultat reste correct.

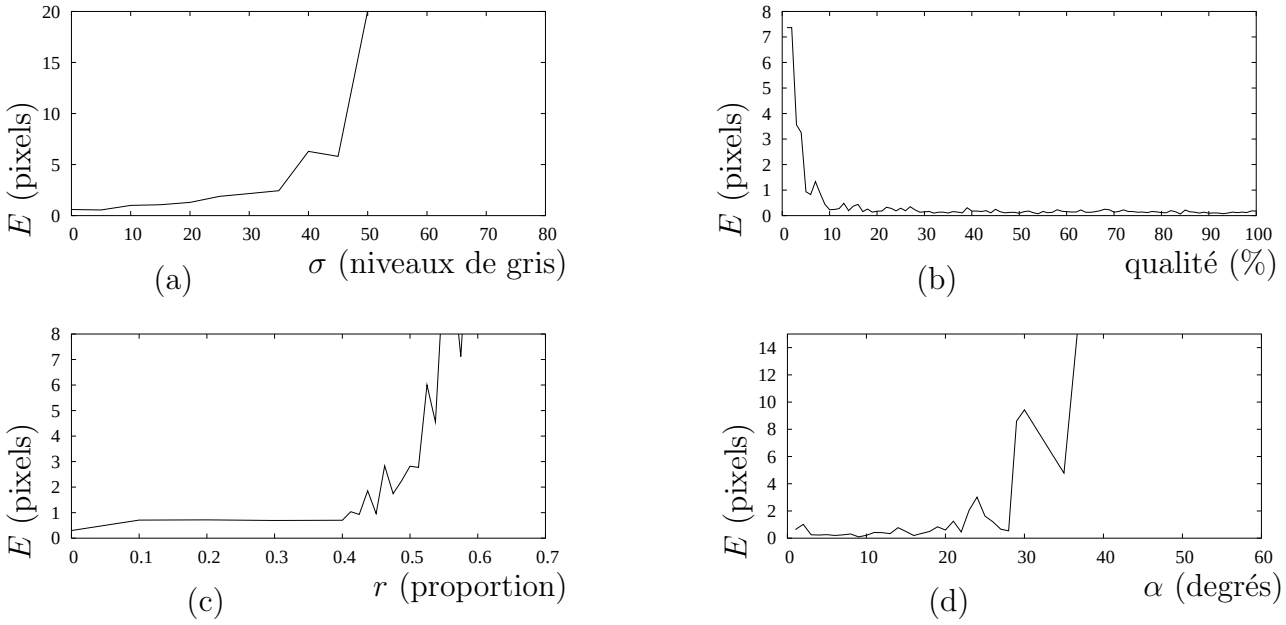


FIG. 3.17: Erreur de mise en correspondance en fonction de (a) : l'intensité du bruit sur les niveaux de gris, (b) : la qualité de compression JPEG, (c) : la proportion de points mal appariés, (d) : l'angle de rotation entre les deux images.

3.4.6.3 Erreurs d'appariement

L'estimation de l'homographie s'appuie sur des paires de points (m, m'). Nous ajoutons un bruit uniforme d'amplitude 30 pixels à une proportion r de ces points (ils deviennent donc aberrants).

Nous avons évalué l'erreur E en fonction du taux de contamination r (figure 3.17(c)). Jusqu'à 40% de points peuvent être contaminés sans perturber le processus.

3.4.6.4 Angle de rotation

Nous faisons varier l'angle de rotation α entre deux images de 0° à 60°. 20° correspond à un recouvrement de 2/3, 40° → 1/3, 60° → 0.

La figure 3.17(d) montre que l'algorithme est utile jusqu'à un angle de 35° (recouvrement de 40%).

3.4.6.5 Re-projection

Pour visualiser les mosaïques, nous re-projetons l'image I' dans l'image I, et nous combinons les deux images par une moyenne. Nous considérons les images en couleurs I et I', de dimensions $l \times h$ pixels, comme des fonctions :

$$I, I' : [0, l] \times [0, h] \rightarrow \mathbb{R}^3$$

Le vecteur de sortie représente les composantes rouge, verte et bleue de l'image en ce point.

L'image panoramique re-projetée est alors définie par :

$$I_{\mathrm{p}}(\mathbf{m}) = \begin{cases} \frac{1}{2}(I(\mathbf{m}) + I'(T(\theta, \mathbf{m})) & \text{si } \mathbf{m} \in [0,l] \times [0,h] \text{ et } T(\theta,\mathbf{m}) \in [0,l] \times [0,h] \\ I(\mathbf{m}) & \text{si } \mathbf{m} \in [0,l] \times [0,h] \\ I'(T(\theta,\mathbf{m})) & \text{si } T(\theta,\mathbf{m}) \in [0,l] \times [0,h] \\ v_{\text{blanc}} & \text{sinon} \end{cases}$$

Le triplet v_{blanc} représente du blanc. Ceci se généralise aux images panoramiques de plus de deux images en :

- composant les homographies pour ramener les images dans un référentiel commun ;
- faisant, en chaque point, la moyenne sur les composantes de couleurs des images présentes.

En pratique, les images sont définies de manière discrète, la re-projection s'effectue donc à la manière décrite au § 7.2.3.

3.4.6.6 Images réelles

Nous avons testé l'algorithme sur différents types de prises de vue avec divers types de caméras (table 3.1). Les figures 3.18 et 3.19 montrent les images panoramiques résultantes.

séquence	taille	nombre d'images	source	distorsions	bruit
SYNTHÈSE	640 × 480	5	lanceur de rayons	nulles	nul
INRIA	512 × 342	5	appareil photo analogique	fortes	dû à la compression
N7-A et N7-B	384 × 288	4 et 5	caméra DV	faibles	changement de luminosité
WEBCAM	352 × 288	26	*webcam*	fortes	fort
KEBLE	752 × 552	13	caméra DV (?)	faibles	flou

TAB. 3.1: Résumé des propriétés des séquences que nous avons traitées.

La série INRIA a été traitée par Jérôme Blanc[5] dans sa thèse ([Bla98] p158). Elle est intéressante parce qu'elle mélange des éléments naturels (la montagne et le ciel) et d'autres artificiels (les bâtiments, lampadaire) aux contours plus rectilignes.

Les deux séries N7 ont été prises avec une caméra vidéo numérique et sous-échantillonnées (pour supprimer l'entrelacement). Elles sont faciles à traiter tant que les différences de luminosité entre images ne sont pas trop grandes.

La série WEBCAM a été obtenue avec une caméra bas de gamme. L'approximation du système optique par un trou d'épingle n'est pas satisfaisante. L'image est fortement bruitée parce qu'elle est comprimée pour passer sur le bus USB. En revanche, les images sont en grand nombre et décalées de 10 à 20 pixels seulement, ce qui facilite l'appariement.

[5]Nous sommes reconnaissants à Jérôme Blanc pour avoir rendues publiques ces images sur le site web de MOVI, `http://www.inrialpes.fr/movi`.

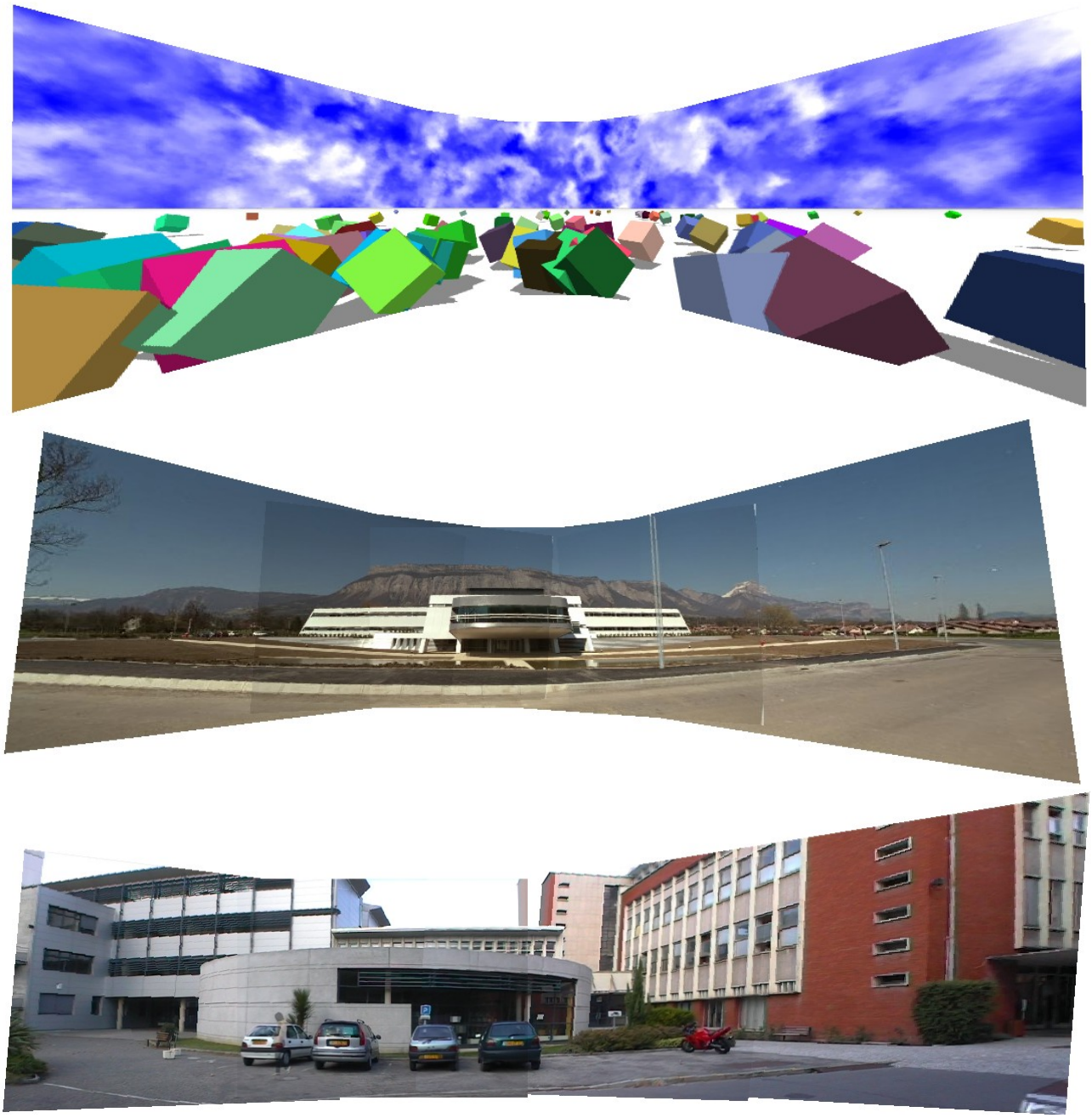


FIG. 3.18: Exemples d'images panoramiques réussies. De haut en bas : séquences SYNTHÈSE, INRIA et N7-A

FIG. 3.19: Exemples d'images panoramiques réussies (suite). Séquences N7-B, WEBCAM et KEBLE.

La série KEBLE a été traitée par Hartley et Zisserman[6] dans leur ouvrage « *Multiple View Geometry in Computer Vision* » ([HZ03], fig 8.9 p206). Elle est relativement facile à traiter car les images se recouvrent fortement (nous n'utilisons qu'une image sur deux) et contiennent des indices visuels faciles à détecter.

Échec. Généralement, quand la translation estimée est correcte, trouver l'homographie ne pose pas de problème. Trois raisons peuvent faire échouer l'estimation de la translation :

- toutes les zones de l'image sont trop texturées ;
- l'image n'est pas assez structurée, elle ne comporte pas de régions distinctes) ;
- les zones de recouvrement de l'image sont de luminosités très différentes. Ceci peut arriver quand la caméra corrige la luminosité automatiquement (§ 3.5.1).

Pour résoudre ce problème, une correction peut être appliqué aux couleurs pour les rendre cohérentes en couleurs. Cette correction peut être une simple transformation affine (avec saturation) ou une correction de « gamma ».

L'algorithme d'estimation de la translation diagnostique un échec quand aucun point n'est clairement supérieur dans l'ensemble de points pondérés (§ 3.4.4).

La figure 3.20 en présente un exemple : la caméra numérique s'est efforcée de régler elle-même son gain pour que toutes les images aient la même luminosité globale, ce qui fait échouer la détection des régions.

FIG. 3.20: Exemple de séquence d'images sur laquelle l'appariement échoue.

3.5 Le résultat

3.5.1 Le problème d'éclairage

Les images panoramiques souffrent d'un problème d'éclairage dans les scènes avec une lumière directionnelle dominante. C'est le cas pour un paysage ensoleillé ou une pièce éclairée par la lumière du jour venant d'une fenêtre. La luminosité du capteur (dépendant de l'ouverture, du temps d'exposition et du gain électronique) peut être :

[6]Nous leur sommes reconnaissants pour nous avoir fourni leurs images de Keble College (Oxford).

constante : la discrétisation de la mesure de niveau (de gris ou de couleur) sur $2^8 = 256$ valeurs (plus rarement, sur 12 ou 16 bits) devient difficile. Quand le capteur vise une orientation proche de la lumière directionnelle, il est ébloui : la valeur renvoyée par le capteur est la valeur maximale représentable (255 le plus souvent). Quand le capteur vise une zone sombre de la scène, il rencontre un problème de **dynamique** : les couleurs sombres sont représentées par peu de valeurs différentes. Ces problèmes sont visibles dans les images des figures 3.9 et 3.8 ;

variable : elle s'adapte à la luminosité dans la direction visée pour atténuer l'éblouissement dans les parties claires tout en ayant une dynamique suffisante dans les parties sombres. Ce procédé peut être mis en œuvre lors de prises de vue pour faire une mosaïque, plus difficilement sur des caméras panoramiques. Il pose un problème lors de la mise en correspondance des images à l'aide de contraintes photométriques : les parties correspondantes n'ont pas les mêmes couleurs (voir figure 3.19, séquence N7-B). Il requiert aussi des artifices d'affichage pour engendrer une mosaïque convaincante en évitant les transitions brusques entres images.

Une solution proposée dans le cadre de la mesure de lumière ([Gib04] p26) s'inspire des techniques de **bracketing** utilisées par les photographes. Elle consiste à faire plusieurs prises de vues panoramiques avec des réglages de luminosité variables. Il suffit ensuite de choisir l'image la plus pertinente, selon la luminosité de la région d'intérêt.

3.5.2 Comparaison des deux méthodes

Par rapport à l'approche à base de caméras dédiées, construire une image panoramique avec une mosaïque a deux avantages :

- on peut le faire sans matériel particulier. En fait, la caméra vidéo du suivi peut être utilisée pour préparer l'image panoramique, quoique la résolution supérieure d'un appareil photo numérique soit un avantage ;
- la résolution du résultat est relativement uniforme. Elle peut être accrue dans une direction donnée en ajoutant simplement une image de détail à l'ensemble d'images utilisées en entrée.

En revanche, la création automatique de la mosaïque échoue quelquefois, l'utilisateur doit alors intervenir manuellement pour obtenir le résultat.

3.5.3 Format du résultat

La sortie des deux systèmes de création d'images panoramiques se compose de :

- une image ou une série d'images ;
- un moyen (une fonction et des paramètres) pour re-projeter ces images selon une projection centrale plane.

Représentation naïve. Le plus simple pour faire une image compatible avec un modèle homographique est donc de choisir une direction et de re-projeter les images sur l'image

d'une caméra virtuelle qui pointe dans cette direction (§ 3.3.3.4, § 3.4.6.5). Le problème est qu'une caméra à projection centrale plane est directionnelle. Son angle de vue est limité à 180°, avec des distorsions inacceptables à plus de 120°.

Représentation complexe. Deux méthodes permettent de résoudre ce problème. Les deux sont présentes dans QuickTime VR, le standard de fait pour la visualisation de panoramas :

- les images sont laissées telles quelles, en leur associant les données nécessaires pour les échantillonner. Dans QuickTime VR, l'image panoramique est stocké en projection cylindrique, avec les angles de vue horizontal et vertical. Pour l'affichage, il est projeté sur un plan qu'il reste à afficher avec une transformation homographique (une des fonctions de base des cartes vidéo 3D) ;
- **plusieurs** images panoramiques sont stockées. La scène complète peut être représentée par sa projection sur les facettes d'un polyèdre. Ceci nécessite de faire un compromis sur le nombre de faces : beaucoup de faces rendent la densité des pixels plus uniforme, mais un grand nombre d'arêtes rendent plus visible les artefacts de jointure. Le polyèdre choisi dans QuickTime VR est le cube. Cette forme a l'avantage d'avoir des faces rectangulaires, ce qui évite d'avoir à manipuler des images triangulaires ou pentagonales.

Nous utilisons cette seconde solution, quatre faces d'un cube représentent le cylindre visible (figure 3.21).

FIG. 3.21: Le cube dont quatre des faces (figure 3.9) représentent le panorama.

Chapitre 4

Suivi d'un motif plan

Comme nous l'avons précisé en introduction générale (§ 1.3.1), notre étude traite le problème du suivi dans deux contextes de prise de vues différents : le cas des **motifs plans** et le cas d'une **image panoramique de référence**. Le premier cas est classique ; une littérature abondante y est consacrée. Le second cas est celui qui nous intéresse pour les applications. Les deux cas ont en commun la transformation homographique en dimension deux (2D) qui lie deux images quelconques provenant des prises de vues correspondantes. Ceci explique cela, à savoir un couplage très fort entre les algorithmes développés pour solutionner les deux types de suivi.

Le motif plan qui fait fonction de cible 3D peut :
- rester totalement visible durant toute la séquence (**suivi sans occultation**) ;
- être masqué en partie par des objets occultants (**suivi avec occultations**).

4.1 Suivi sans occultation

Cette étude est organisée conformément au plan suivant :
- état de l'art ;
- modélisation du problème ;
- résolution du modèle ;
- expérimentation.

4.1.1 État de l'art

Seules les techniques jugées applicables à notre problème sont visitées.

4.1.1.1 Le problème

La région cible. Le suivi de régions vise à localiser une région en 2D sur les images d'une séquence vidéo. Cette cible 2D peut être :
- définie arbitrairement. Dans la plupart des formats de codage vidéo, une partition de l'image est réalisée à base de sous-images carrées (d'une taille de 8×8 à 64×64

pixels) : les « macroblocs ». Lors du traitement de l'image courante, ce sont les paramètres de la transformation associée à chaque macrobloc qui sont codés ;
- définie sur la première image de la séquence par l'utilisateur ou un algorithme *ad hoc*. La cible 2D peut être décrite soit :
  - par un ensemble de points pouvant lui-même être épars [JD02a, WBC03] ou dense [DM96, CRM00],
  - par des contours [IB96, KWT88] ;
- détectée dynamiquement sur chaque image de la séquence. En pratique, il s'agit de :
  - fenêtres entourant des **points d'intérêt**, [BDH03, VGBS03]
  - « taches » (*blobs*), définis comme des extremums locaux sur l'image traitée par un filtre passe-bas [MJ02].

En ce qui nous concerne, nous sommes dans le second cas, le motif 3D est un objet plan, constituant intégrant de la scène. La cible 2D est un ensemble épars $\mathcal{R}$ de points sur la première image. Cette cible 2D subit des transformations d'une image à l'autre de la séquence.

La séquence d'images. Nous la notons $(I_t)_{t\in\mathbb{N}}$. L'image I_t, en tant qu'objet mathématique est une application $I_t : \mathbb{R}^2 \longrightarrow \mathbb{R}$. Sauf mention spéciale, nous ne traitons que des images en niveaux de gris.

Dans la pratique, une distinction s'opère selon que les traitements s'exécutent en temps différé ou en temps réel.

Cas du temps différé. La séquence vidéo est entièrement disponible et le suivi peut alors se faire en accédant aléatoirement aux images. C'est le cas par exemple dans [Far01] où la vidéo est considérée comme un volume (2D+t) de voxels colorés et les régions comme des ensembles connexes de ces voxels.

Cas du temps réel. À l'image I_t, la cible 2D a été localisée sur l'image I_{t-1}. Par nécessité, le suivi se fait **par degrés**, à savoir en traitant successivement les images I_1,...,I_t.

4.1.1.2 Les hypothèses

Le suivi n'est possible qu'en faisant des hypothèses sur l'évolution de la cible 2D.

Petits changements. Si l'objet bouge trop vite dans la scène ou si son aspect change beaucoup entre deux images, même une personne experte pourrait le « perdre de vue ».

L'**hypothèse de petits changements** sous entend une évolution « continue » de la région cible.

Photométrie. L'hypothèse photométrique concerne l'aspect de la région.

Quelquefois, elle édicte que la répartition (l'histogramme) des couleurs de la région ne change pas [CRM00, CL03].

Le plus souvent, elle impose qu'entre deux images (une **image de référence** et l'image courante), la couleur de la projection d'un point de la cible 3D ne change pas. Cette **hypothèse de conservation des couleurs** (ou **des niveaux de gris)** existe en deux versions (nous notons $\mathbf{m}$ la projection du point sur l'image de référence, et $\mathbf{m}'$ sur l'image courante) :

- la version **faible** ([SA96] eq 1, [ST94] eq 1, [SS97] eq 5, [BA96] eq 1, [BJ98] eq 3, [ECT98] p490, [BAHH92] p240, [BDH03] eq 7, [SS01] eq13.5 p231) est à la base de l'équation fondamentale du flot optique ([Hor86], p282) : l'image de référence est I_{t-1}. Les points correspondants $\mathbf{m}$ et $\mathbf{m}'$ vérifient :

  $$I_{t-1}(\mathbf{m}) \approx I_t(\mathbf{m}')$$

  L'approximation signifie que l'hypothèse ne peut pas être utilisée entre deux images distantes l'une de l'autre. Ceci permet une évolution de l'aspect de la cible (due, par exemple, à des changements de luminosité). Par l'hypothèse des petits changements, cette évolution doit cependant rester lente ;
- la version **forte** ([HB98] eq 1, [JD02b] p60, [MJD03] eq6) : l'image de référence est I_0. La cible doit garder le même aspect pendant toute la séquence vidéo. Pour les points correspondants :

  $$I_0(\mathbf{m}) = I_t(\mathbf{m}')$$

Géométrie. L'hypothèse géométrique décrit les déplacements de chaque point de la cible 2D au cours de la séquence vidéo. Par l'hypothèse des petits changements, ces déplacements sont limités en amplitude.

Une transformation, que nous appelons **mouvement**[1], modélise les déplacements de tous les points de la région cible. Cette fonction notée T dépend d'un vecteur de paramètres $\theta_t \in \mathbb{R}^p$ qui évoluent au cours de la séquence. Le **modèle de mouvement** s'écrit :

$$\mathbf{m}' = T(\theta_t, \mathbf{m})$$

Nous distinguons trois classes de modèles.

Linéaire en θ. La fonction T satisfait à $T(\theta, \mathbf{m}) = T_1(\mathbf{m})\theta + T_0(\mathbf{m})$. Ce modèle englobe les translations, les similitudes, les applications « bi-affines », ainsi que les applications quadratiques en $\mathbf{m}$.

Non linéaire en θ. Les exemples les plus courants sont les homographies à huit paramètres et les modèles qui traduisent la genèse des images, obtenues par projection centrale sur un plan, donc dépendants de six paramètres extrinsèques (pose) et quatre au moins paramètres intrinsèques, supposés invariants.

[1] Par convention, le terme « mouvement » englobe à la fois les **déplacements** de la région (de son centre de gravité par exemple) et ses **déformations**.

Combiné. Les modèles de mouvement combinés ont pour point commun de ne pas dépendre d'un nombre fixé de paramètres. Ainsi, pour représenter un mouvement plus complexe, il suffit de rajouter des degrés de liberté au modèle. En voici deux exemples :

- dans le suivi de maillage [GOM01], on effectue une partition de l'image en triangles. Le mouvement de chaque triangle est affine et déterminé par le déplacement de ses sommets (des nœuds du maillage). Le vecteur de paramètres du mouvement est donc la juxtaposition des vecteurs translation de tous ces nœuds : $p = 2 \times$ (nombre de nœuds) ;
- dans [BYJF97, FBJ98], on dispose d'une série $(V_i)_{i=1..p}$ de « champs de mouvement » élémentaires. Le champ V_i est un tableau qui associe à un point un vecteur : $V_i(\mathbf{m}) \in \mathbb{R}^2$. Le mouvement de la région est décrit à l'aide d'une combinaison linéaire de ces champs, dont le vecteur de paramètres contient les coefficients de pondération :

$$T(\theta, \mathbf{m}) = \mathbf{m} + [V_1(\mathbf{m}) \ \cdots \ V_p(\mathbf{m})]\theta \tag{4.1}$$

  C'est aussi un modèle linéaire.

Navigation entre modèles. Les modèles fortement paramétrés permettent de décrire précisément des mouvements complexes, mais leur résolution est plus ardue. C'est pour cela que certains auteurs [Kan04, VGBS03] recommandent de **sélectionner le modèle** dynamiquement.

Synthèse. Les deux hypothèses, photométrique et géométrique, permettent d'écrire l'**équation fondamentale du mouvement** dans un problème de suivi, à savoir (cas de l'hypothèse forte) :

$$\forall t \in \mathbb{N}, \forall \mathbf{m} \in \mathcal{R}, I_0(\mathbf{m}) = I_t(T(\theta_t, \mathbf{m})) \tag{4.2}$$

4.1.1.3 La résolution

Cette équation fondamentale est à la base de l'estimation des paramètres du mouvement.

Les erreurs. Plusieurs types d'erreurs peuvent se produire :

- le modèle géométrique est imparfait. Les imperfections peuvent venir du capteur (distorsions) ou des conditions de prise de vues (déplacements imprévus) ;
- l'hypothèse de conservation des niveaux de gris n'est qu'approchée (changements d'éclairage, surfaces non mates, etc.) ;
- l'étape de numérisation est intrinsèquement dégradante (bruit, discrétisation, quantification).

Le critère SSD. La résultante de toutes ces erreurs étant difficile à prédire, on la suppose en général tout simplement gaussienne, additive par rapport aux niveaux de gris (§ 2.4.2).

Dans ces conditions, le critère des moindres carrés (couramment appelé *Sum of Squared Differences*, SSD) est des plus pertinents :

$$e_t(\theta) = \sum_{\mathbf{m}\in\mathcal{R}} \|I_t(T(\theta,\mathbf{m})) - I_0(\mathbf{m})\|^2$$

Par suite, l'estimation de θ_t consiste à trouver :

$$\widehat{\theta_t} = \underset{\theta\in\mathbb{R}^p}{\operatorname{argmin}}\ e_t(\theta) \tag{4.3}$$

Résolution par mise en correspondance. Une voie « mille fois empruntée » par les chercheurs en géométrie multi-vues est la détection dans I_0 (ou I_{t-1}) et I_t d'éléments visuels et leur appariement.

Vigueras *et al.* [VGBS03] se basent sur des points d'intérêt. Ils calculent la pose 3D en les mettant en correspondance et estiment les paramètres par échantillonnage aléatoire. Fait intéressant : une implémentation en temps réel est réaliste sur les ordinateurs actuels.

Mégret et Jolion [MJ02] détectent des « taches » (*blobs*) sur chaque image et les mettent en correspondance en contraignant leurs trajectoires à être régulières (à l'aide d'un arbre d'hypothèses).

Résolution par dérivation de l'équation fondamentale. Cette mise en correspondance n'exploite pas directement l'hypothèse des petits changements. La dérivation de l'équation fondamentale débouche sur une alternative qui pallie à cet inconvénient.

Nous notons :

- $I'(\mathbf{m})$ la dérivée de l'image I au point $\mathbf{m}$, identifiable à un vecteur ligne de $\mathbb{R}^{1\times 2}$. Ce vecteur est la transposée du vecteur gradient de I mesuré en $\mathbf{m}$;
- $T_\theta(\theta,\mathbf{m})$ (resp. $T_m(\theta,\mathbf{m})$) la dérivée partielle de T par rapport à θ (resp. $\mathbf{m}$), mesurée en $(\theta,\mathbf{m})$. Elle est identifiable à la matrice jacobienne de taille $2\times p$ (resp. 2×2).

Connaissant θ_{t-1} (une estimation) et supposant $\Delta = \theta_t - \theta_{t-1}$ petit, le développement de Taylor-Young à l'ordre 1 (pour $\mathbf{m}$ fixé) permet d'écrire, en ne conservant que la partie régulière :

$$I_t(T(\theta_{t-1}+\Delta,\mathbf{m})) \approx I_t(T(\theta_{t-1},\mathbf{m})) + I'_t(T(\theta_{t-1},\mathbf{m}))T_\theta(\theta_{t-1},\mathbf{m})\Delta$$

En utilisant cette approximation, la minimisation SSD (équation (4.3)) se ramène au problème d'optimisation en Δ suivant :

$$\min_{\Delta\in\mathbb{R}^p} \|A_t\Delta - b_t\| \tag{4.4}$$

où, en notant $\mathcal{R} = \{\mathbf{m}_1,...,\mathbf{m}_n\}$:

$$A_t = \left[I'_t(T(\theta_{t-1},\mathbf{m}_i))T_\theta(\theta_{t-1},\mathbf{m}_i)\right]_{i=1..n} \qquad b_t = \left[I_0(\mathbf{m}_i) - I_t(T(\theta_{t-1},\mathbf{m}_i))\right]_{i=1..n}$$

C'est un problème de moindres carrés linéaires ; Δ s'en déduit facilement. Ceci constitue la base de nombreuses techniques de suivi.

Implémentations classiques. Shi et Tomasi [ST94] exploitent (4.4) pour suivre des fenêtres de pixels autour de points d'intérêt, avec soit un modèle translation, soit un modèle affine. Le conditionnement de la matrice A_t du problème aux moindres carrés renseigne sur la pertinence de $\widehat{\Delta}$. Ils l'utilisent comme une mesure de qualité (analogue à celle de Harris [SMB00]) pour sélectionner les points.

Irani *et al.* [IRP94] exploitent (4.4) pour suivre des régions denses avec soit un modèle affine, soit un modèle linéaire en θ et quadratique en $\mathbf{m}$. Une pyramide multirésolution rend l'estimation plus robuste. Ils effectuent simultanément une segmentation de l'image en régions (§ 4.2.1.4).

Hager et Belhumeur ([HB98] § 2.2) évitent le calcul de $I'_t(.)$ faisant intervenir I'_0 et en... voir § 4.1.3.3.

Apprentissage. On peut guider la minimisation par des techniques d'apprentissage. Un algorithme « apprend » préalablement les mouvements qu'il va rencontrer pendant le suivi.

Toujours dans le cadre de l'exploitation de (4.4), Black *et al.* [BYJF97] :

- constituent préalablement un corpus d'apprentissage avec des séquences vidéo ;
- associent un champ de mouvement (calculé par flot optique) $M_i \in \mathbb{R}^{2\times l\times h}$ (les images sont de taille $l \times h$) à chaque paire d'images successives des vidéos du corpus ;
- forment $A = [M_1 \ \cdots \ M_h]$ (h est le nombre total de champs calculés, considérés comme des vecteurs) ;
- calculent la décomposition SVD de A. Les p premières composantes de celle-ci (ACP) sont les champs de mouvement élémentaires $(V_i)_{i=1..p}$

Les champs élémentaires sont combinés linéairement pour obtenir le modèle complet (équation (4.1)).

Ils utilisent cet algorithme de suivi pour :

- des mouvements discontinus. Le corpus contient des images synthétiques représentant deux régions de mouvements distincts ;
- des mouvements du corps humain (jambes, lèvres, ...). Le corpus est alors constitué des mêmes types de vidéos.

Jurie et Dhome [JD02a] étendent la méthode à un modèle homographique (§ 4.1.3.4) pour suivre des motifs plans. Leur algorithme apprend les mouvements possibles sur l'image initiale et en déduit une estimation de A_0^+ qui peut être utilisée pendant tout le suivi.

Traitement de l'incertitude. Choisissant pour modèle de prise de vues une projection orthogonale, Brand et Bhotika [BB01] déterminent la pose (en 3D) d'objets non rigides. La chaîne de calculs qui mène de l'échantillonnage de l'image à l'estimation des paramètres est linéaire, ce qui permet la **propagation des matrices de covariance**. Cette méthode est très efficace pour le suivi de visages, même en présence d'occultations.

4.1.1.4 Ce que nous en retenons

La phase de suivi n'est qu'une des étapes (certes prépondérante en temps de calcul) des applications que nous nous fixons d'exécuter en temps réel (au rythme de la vidéo). En conséquence, une version de l'algorithme de Jurie et Dhome (connu pour sa rapidité), intégrée dans une approche multi-résolution, est une composante essentielle de notre proposition.

4.1.2 Modélisation

4.1.2.1 Le contexte expérimental

Nos expériences sont menées dans le contexte suivant.

Une caméra dont le modèle de prise de vues est une projection centrale filme une cible 3D (**motif plan**, figure 1.3).

L'hypothèse de conservation des niveaux de gris est considérée satisfaite si :

- la surface de la cible 3D est lambertienne (autrement dit, elle diffuse la lumière incidente de manière isotrope) ;
- l'eclairage produit une illumination isotrope.

Le motif plan et la caméra peuvent faire des mouvements quelconques à condition que la cible 3D demeure entièrement visible.

4.1.2.2 La cible 2D

La région cible est un sous-ensemble de points dense, délimité par un quadrilatère $\mathcal{Q}$, de sommets $(\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \mathbf{w}_4)$.

Pendant le suivi, nous utilisons un sous-ensemble épars de $\mathcal{Q}$:

$$\mathcal{R} = \{\mathbf{m}_1, ..., \mathbf{m}_n\} \subset \mathcal{Q}$$

Le modèle de mouvement est supposé valide sur $\mathcal{Q}$ entier, mais en pratique nous utilisons $\mathcal{R}$, pour alléger les calculs.

4.1.2.3 Le modèle photométrique

Nous utilisons l'hypothèse photométrique forte (l'image de référence est I_0). La conservation des niveaux de gris (nous n'utilisons pas les couleurs) est définie par l'équation (4.2).

4.1.2.4 Le modèle géométrique

La cible 3D étant plane, le mouvement de la cible 2D est décrit par une homographie en dimension deux ([HZ01] § 7.1.1 p185). C'est un modèle non linéaire.

Nous utilisons la normalisation $h_9 = 1$ (notations § 3.4.2.1). Le vecteur $\theta_t = [h_1 \cdots h_8]^\top$ contient 8 éléments de la matrice d'homographie :

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{bmatrix}$$

Remarque : Une homographie transforme le quadrilatère $\mathcal{Q}$ en un quadrilatère.

4.1.2.5 Le problème d'estimation de paramètres

À l'image courante I_t de la séquence nous associons un problème d'estimation de paramètres et, conformément au formalisme du chapitre 2 :

- chaque point de référence constitue une « expérience », il y en a n ;
- les paramètres sont les coefficients de l'homographie, à savoir $\beta = \theta_t$ $(p = 8)$;
- les entrées sont les coordonnées des points de référence sur I_0, $x = \mathbf{m}$ et les mesures associées $\widetilde{x_i} = \mathbf{m_i}$;
- le modèle est déterminé par la fonction

$$f : \begin{array}{ccc} \mathbb{R}^2 \times \mathbb{R}^p & \longrightarrow & \mathbb{R} \\ (x, \beta) & \longmapsto & f(x, \beta) = I_t(T(\theta_t, \mathbf{m})) \end{array}$$

- les sorties sont les niveaux de gris sur l'image courante $y = I_t(T(\theta_t, \mathbf{m}))$ et les mesures associées sont

$$\widetilde{y_i} = I_0(T(\theta_0, \mathbf{m}_i)) \qquad i = 1..n$$

Ceci donne la « boîte noire » :

$$\mathbf{m} \in \mathbb{R}^m \rightarrow \boxed{\begin{array}{c} \theta \in \mathbb{R}^p \\ \hline m = 2; p = 8; q = 1 \end{array}} \rightarrow I_t(T(\theta, \mathbf{m})) \in \mathbb{R}^q$$

Le problème au sens des moindres carrés qui résulte de cette formalisation repose sur le critère SSD (§ 4.1.3.2).

4.1.2.6 Relâchement des contraintes

Certaines des hypothèses précédentes sont moins contraignantes que d'autres.

Les distorsions géométriques des objectifs photographiques peuvent être estimées dans une étape d'étalonnage (appendice 7.4) et corrigées pour aboutir à des écarts (par rapport au modèle de la projection centrale) inférieurs au pixel (en évitant les objectifs grand-angle et/ou bas de gamme).

Les variations globales de luminosité sur la surface peuvent être compensées si l'éclairage au lieu d'être isotrope n'est qu'unidirectionnel (rayons parallèles).

Par contre, la planéité de la cible 3D doit être respectée « strictement » (en relativisant par rapport à la distance scène $\leftrightarrow$ caméra).

4.1.2.7 Notations complémentaires

Nous notons

$$F_t(\theta) = \left[I_t(T(\theta, \mathbf{m}_i))\right]_{i=1..n} \tag{4.5}$$

L'équation fondamentale (4.2) et le critère SSD peuvent se réécrire :

$$F_t(\theta_t) = F_0(\theta_0) \tag{4.6}$$

$$e_t(\theta) = \|F_t(\theta) - F_0(\theta_0)\|^2 \tag{4.7}$$

4.1.3 Résolution

Comme θ_0 est connu et θ_t peu différent de θ_{t-1}, il est commode de faire un suivi par degrés : calculer $\widehat{\theta_t}$ revient à estimer $\Delta = \theta_t - \theta_{t-1}$.

Nous supposons que nous pouvons faire une **phase d'apprentissage**, arbitrairement longue, qui n'utilise que la première image. C'est à ce moment que les points de $\mathcal{Q}$ retenus dans $\mathcal{R}$ sont soigneusement choisis.

En phase d'exploitation, pour chaque nouvelle image et au rythme de la vidéo, nous exécutons les opérations propres au suivi. Ces dernières, qui permettent de calculer Δ, font l'objet de cette section.

La plupart des techniques de résolution exposées précédemment sont valables quel que soit le modèle de mouvement. Nous le précisons quand nous utilisons des propriétés spécifiques aux homographies.

4.1.3.1 Choix des points de référence

La tentation est grande de choisir les points d'intérêt habituellement utilisés dans la mise en correspondance d'images, parce qu'ils sont représentatifs de l'image. Cependant, comme ceux-ci s'accumulent autour des discontinuités de I_0, ils sont une source d'instabilité numérique.

Nous utilisons un mélange de p_h points d'intérêt (de Harris, [SMB00]) et de points (dits ♠ **points réguliers**) répartis arbitrairement sur $\mathcal{Q}$. Ici, nous les choisissons sur les nœuds d'une grille de taille $p_\mathrm{r} \times p_\mathrm{r}$ contenue dans $\mathcal{Q}$ (figure 4.1).

Théoriquement, pour estimer les 8 paramètres de l'homographie, 4 points de référence suffisent (4 points → 8 coordonnées → 8 équations). Cependant, pour assurer la robustesse du suivi, nous en choisissons beaucoup plus, de l'ordre de quelques centaines.

4.1.3.2 Optimisation non linéaire (NL)

Le problème de minimisation de e_t (équation (4.7)) est un problème d'estimation non linéaire. Pour le résoudre, nous utilisons l'algorithme de Levenberg-Marquardt (§ 2.6.3.1). La dérivée de F_t en θ, identifiée à une matrice jacobienne, est :

$$F'_t(\theta) = \left[\tfrac{\partial I_t \circ T}{\partial \theta}(\theta, \mathbf{m}_i)\right]_{i=1..n} = \left[I'_t(T(\theta, \mathbf{m}_i))T_\theta(\theta, \mathbf{m}_i)\right]_{i=1..n} \tag{4.8}$$

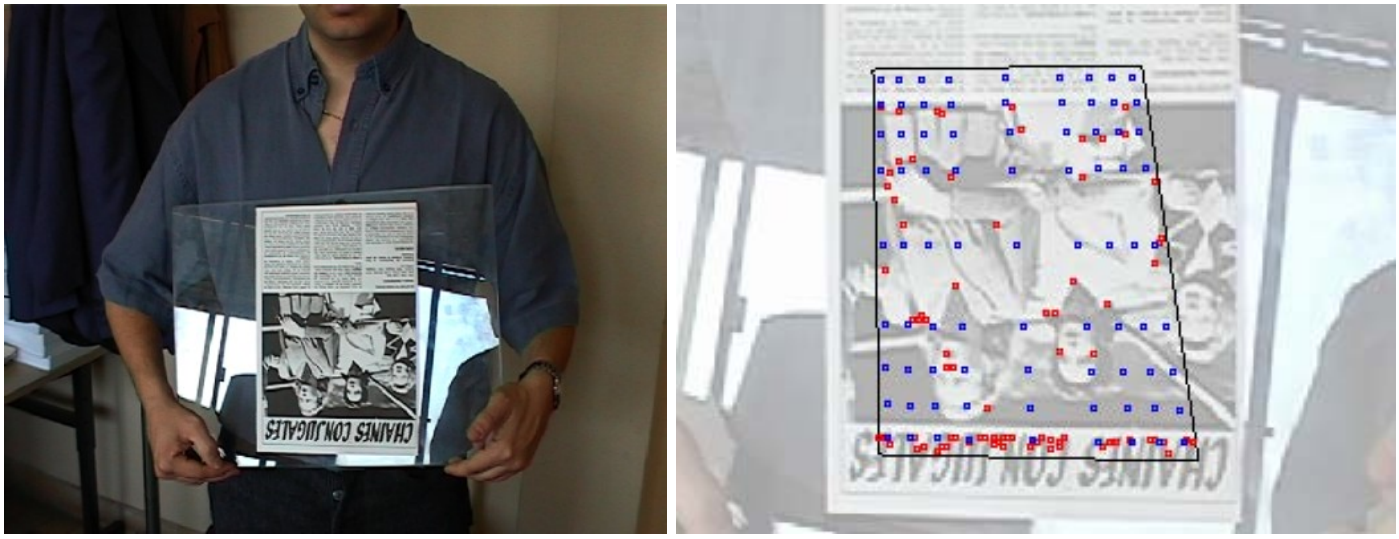


FIG. 4.1: Une image d'une séquence vidéo et ses points de référence. Le cadre représente $\mathcal{Q}$, les points d'intérêt sont en rouge, les points réguliers en bleu.

Le calcul de T_θ ne pose pas de problème. Par contre, à cause de la nature discrète des images, il est nécessaire de travailler sur des valeurs interpolées de I'_t (§ 7.2.2).

La solution $\widehat{\theta_t}$ est retournée par le processus d'estimation dont nous limitons les itérations à n_{it} (de l'ordre de 5).

4.1.3.3 Méthode de Hager et Belhumeur (HB)

Cette méthode repose sur une itération de la méthode de Gauss-Newton (§ 2.6.3.1) :

$$\widehat{\Delta_{\text{GN}}} = F'_t(\widehat{\theta_{t-1}})^+(F_0(\theta_0) - F_t(\widehat{\theta_{t-1}}))$$

Hager et Belhumeur ([HB98], eq13) préfèrent remplacer la dérivée de F_t par celle à l'image précédente, supposée peu différente :

$$\widehat{\Delta_{\text{HB}}} = F'_{t-1}(\widehat{\theta_{t-1}})^+(F_0(\theta_0) - F_t(\widehat{\theta_{t-1}})) \tag{4.9}$$

L'équation (4.2) de conservation des niveaux de gris, écrite pour l'image I_{t-1} et un point $\mathbf{m} \in \mathcal{R}$, donne :

$$I_{t-1}(T(\theta_{t-1}, \mathbf{m})) = I_0(T(\theta_0, \mathbf{m}))$$

en dérivant les deux membres de cette égalité par rapport à $\mathbf{m}$, on obtient :

$$I'_{t-1}(T(\theta_{t-1}, \mathbf{m}))T_m(\theta_{t-1}, \mathbf{m}) = I'_0(T(\theta_0, \mathbf{m}))T_m(\theta_0, \mathbf{m})$$

d'où :

$$I'_{t-1}(T(\theta_{t-1}, \mathbf{m})) = I'_0(T(\theta_0, \mathbf{m}))T_m(\theta_0, \mathbf{m})T_m(\theta_{t-1}, \mathbf{m})^{-1}$$

En combinant ceci avec l'équation (4.8), on obtient l'expression suivante de $F'_{t-1}(\theta_{t-1})$ (c'est l'équation 19 de [HB98]) :

$$F'_{t-1}(\theta_{t-1}) = [\underbrace{I'_0(T(\theta_0, \mathbf{m}_i))T_m(\theta_0, \mathbf{m}_i)}_{A} \ \underbrace{T_m(\theta_{t-1}, \mathbf{m}_i)^{-1}T_\theta(\theta_{t-1}, \mathbf{m}_i)}_{B}]_{i=1..n}$$

Le terme A étant constant et ne dépendant que de l'image de référence I_0, il peut être précalculé. Quant au terme B, Hager et Belhumeur proposent un raccourci de calcul pour le cas où il peut se mettre sous la forme $\Gamma(\mathbf{m})\Sigma(\theta_{t-1})$. Cette factorisation est possible dans leur modèle affine, mais pas dans le modèle homographique. Nous ne pouvons donc pas utiliser efficacement ce calcul.

4.1.3.4 La méthode de Jurie et Dhome (JD)

La méthode de Jurie et Dhome [JD02a] commence par une coûteuse phase d'apprentissage reposant uniquement sur I_0. Elle permet de mettre en relation les petites variations de θ_0 avec celles des niveau de gris qu'elles induisent sur les points de référence. Grâce à l'équation fondamentale (4.2), cette relation peut être utilisée pendant la phase de suivi pour retrouver une petite **perturbation** Δ du vecteur de paramètres à partir de la mesure de $F_0(\theta_0 + \Delta)$

Phase d'apprentissage.

L'approximation linéaire. Les méthodes de Gauss-Newton améliorées (§ 2.6.3.1) reposent sur le fait que, pour une matrice A bien choisie,

$$\Delta \approx A(F_0(\theta_0 + \Delta) - F_0(\theta_0)) \tag{4.10}$$

Dans la méthode de Gauss-Newton, $A = F'_0(\theta_0)^+$. Plutôt que de calculer A de cette manière, Jurie et Dhome [LJDC04] se sont aperçus qu'il était plus robuste et flexible de l'estimer à partir de N « expériences » ($N \gg n$). On est en droit d'espérer :

- plus de robustesse car la partie régulière du développement de Taylor-Young ne donne qu'une approximation valide très localement (il y a le reste !) ;
- plus de flexibilité en estimant A sur le (voire les § 4.1.4.5) voisinage(s) qui convien(nen)t le mieux.

L'expérience k consiste à tirer aléatoirement une perturbation Δ_k et à mesurer

$$\widetilde{g_k} = F_0(\theta_0 + \Delta_k) - F_0(\theta_0)$$

Ensuite, il s'agit d'estimer $\widehat{A}$ qui rende « aussi vraie que possible » la relation

$$\Delta_k = \widehat{A}\widetilde{g_k}$$

Voici les ingrédients du problème d'estimation de paramètres :

- les paramètres à estimer sont $\beta = A$ ($p' = pn$) ;

– les entrées sont les perturbations $x = \Delta$ et les « mesures » associées sont : $\widetilde{x_k} = \Delta_k$;
– le modèle exprime le décalage entre les perturbations injectées et les valeurs reconstruites :

$$f(x, \beta) = A(F_0(\theta_0 + \Delta) - F_0(\theta_0)) - \Delta$$

– les sorties sont identiquement nulles : $\widetilde{y_i} = 0_p$.

La « boîte noire » correspondante est :

$$\Delta \in \mathbb{R}^m \rightarrow \boxed{\begin{array}{c} A \in \mathbb{R}^{p \times n} \\ \hline m' = p; n' = N; p' = p \times n; q' = p \end{array}} \rightarrow A(F_0(\theta_0 + \Delta) - F_0(\theta_0)) - \Delta \in \mathbb{R}^p$$

C'est un problème linéaire multidimensionnel (§ 2.6.2).

Résolution OLS. Dans un premier temps, nous résolvons le problème aux moindres carrés ordinaires. Ceci mène à l'estimation suivante pour A :

$$\widehat{A_{\text{OLS}}} = \underset{A \in \mathbb{R}^{p \times n}}{\text{argmin}} \sum_{k=1}^{N} \| A\widetilde{g_k} - \Delta_k \|^2 \tag{4.11}$$

En notant

$$\begin{array}{rcl} M & = & [\Delta_1 \quad \cdots \quad \Delta_N] \quad \in \mathbb{R}^{p \times N} \\ G & = & [\widetilde{g_1} \quad \cdots \quad \widetilde{g_N}] \quad \in \mathbb{R}^{n \times N} \end{array}$$

ceci peut se réécrire :

$$\widehat{A_{\text{OLS}}} = \underset{A \in \mathbb{R}^{p \times n}}{\text{argmin}} \| AG - M \|^2 \tag{4.12}$$

La résolution consiste à se ramener à la disposition classique des moindres carrés par transposition :

$$\widehat{A_{\text{OLS}}} = \left(\underset{A' \in \mathbb{R}^{n \times p}}{\text{argmin}} \| G^\top A' - M^\top \|^2 \right)^\top = \left(\left(G^\top\right)^+ M^\top \right)^\top = M((G^+)^\top)^\top$$

d'où :

$$\widehat{A_{\text{OLS}}} = MG^+$$

Résolution affinée. L'estimation aux moindres carrés ne correspond pas au type d'erreurs rencontrées dans les mesures. En effet, les mesures faites sur le « second membre » M sont sans erreur : ce sont les valeurs injectées. Par contre, les éléments de la matrice G sont des différences de niveaux de gris : ils sont sujets à une imprécision liée à l'interpolation sous-jacente.

La conséquence logique de cette remarque serait d'envisager l'estimation de A comme solution d'un problème DLS multidimensionnel (§ 2.5.2.3).

Dans les faits, les nombreuses expériences menées dans ce sens sur nos données, en collaboration avec Sabine Van Huffel et Yvan Markovsky (de l'université catholique de ♠ Louvain), permettent de conclure que le meilleur compromis, pour tenir compte de la

relative imprécision du modèle mathématique linéaire sous-jacent à l'équation (4.10), est de rechercher $\widehat{A}$ comme solution d'un problème de moindres carrés totaux (TLS) pondérés dans le cas multidimensionnel (§ 2.6.1.2).

L'estimation TLS prend la forme :

$$\begin{cases} \min \parallel \gamma\Delta_{\mathrm{M}} \parallel^2 + \parallel (1-\gamma)\Delta_{\mathrm{G}} \parallel^2 \\ A \in \mathbb{R}^{p\times n}, \Delta_{\mathrm{M}} \in \mathbb{R}^{p\times N}, \Delta_{\mathrm{G}} \in \mathbb{R}^{n\times N} \\ A(G+\Delta_{\mathrm{G}}) = M + \Delta_{\mathrm{M}} \end{cases}$$

La pondération γ permet de naviguer entre OLS ($\gamma = 0$) et DLS ($\gamma = 1$). En pratique, nous utilisons le coefficient α qui est lié à γ par

$$\gamma = \frac{e^{\alpha}}{e^{\alpha} + e^{-\alpha}}$$

Empiriquement, $\alpha \in [3, 8]$ donne de bons résultats.

Le calcul pratique de $\widehat{A_{\mathrm{OLS}}}$ et $\widehat{A_{\mathrm{TLS}}}$ est décrit au § 2.6.2.

Phase de suivi.

Notations. Appelons T_t la transformation ponctuelle

$$T_t : \begin{array}{ccc} \mathbb{R}^2 & \longrightarrow & \mathbb{R}^2 \\ \mathbf{m} & \longmapsto & T(\theta_t, \mathbf{m}) = \mathbf{m}^{(t)} \end{array}$$

Quel est le problème ? Connaissant :
– les transformations $T_0, ..., T_{t-1}$ (approximativement),
– les images I_0 et I_t,
il s'agit de calculer T_t (nous devrions dire « estimer »).

♠

Une réponse possible

Hypothèse d'application. Si deux points $\mathbf{m}^{(t-1)}$ et $\mathbf{m}^{(t)}$ sont voisins sur I_t, alors leurs antécédents par T_t, $\mathbf{m}$ et $\mathbf{m}'$, sont suffisamment voisins sur I_0 pour que le modèle linéaire appris (équation 4.10) puisse s'appliquer pour déterminer la transformation T' qui les met en bijection (naturellement, T' est calculée à partir des n points de $\mathcal{R}$).

À propos de T'. La définition de la composée de deux applications et la figure 4.2 montrent que :

$$T' = T_t^{-1} \circ T_{t-1} \qquad \text{d'où} \qquad T'_t = T_{t-1} \circ T'^{-1} \tag{4.13}$$

Naturellement nous exploitons les deux propriétés des homographies
– la composée de deux homographies est une homographie,

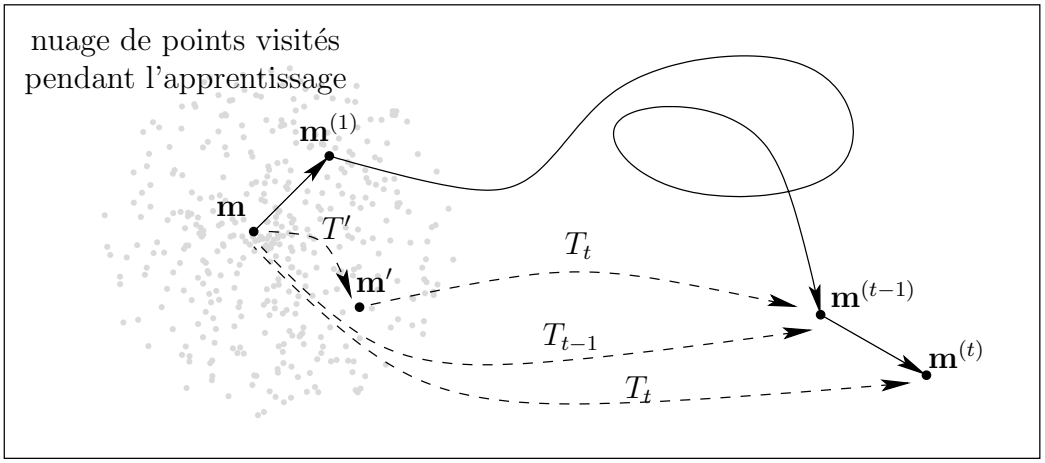


FIG. 4.2: Trajectoire d'un point $\mathbf{m}$ de la région suivie dans les images au cours de la séquence.

– une homographie est bijective et son inverse est une homographie,

pour affirmer que T' est une homographie, dont nous appelons θ' le vecteur de paramètres. Nous notons :

$$\forall i = 1..n, \mathbf{m}_i^{(t)} = T(\theta_t, \mathbf{m}_i) \text{ et } \mathbf{m}'_i = T'(\mathbf{m}_i)$$

Comment calculer T' ? Nous calculons θ' en évaluant l'équation (4.13) avec les données disponibles par exploitation du principe de conservation de niveaux de gris (équation (4.2)) qui permet d'écrire :

$$\forall i = 1..n, I_0(\mathbf{m}'_i) = I_t(\mathbf{m}_i^{(t-1)})$$

et en remplaçant $\mathbf{m}_i$ et $\mathbf{m}_i^{(t-1)}$ par leurs valeurs dans (4.5) :

$$F_0(\theta') = \left[I_0(T'(\mathbf{m}_i))\right]_{i=1..n} = \left[I_t(T_{t-1}(\mathbf{m}_i))\right]_{i=1..n} = F_t(\theta_{t-1})$$

D'où l'expression :

$$\theta' \approx \theta_0 + \widehat{A}(F_0(\theta') - F_0(\theta_0))$$

Et pour finir : T_t. En raisonnant sur les vecteurs de paramètres associés aux homographies et en notant :

– $\theta_1 \bullet \theta_2$ le vecteur de paramètres associé à $T_1 \circ T_2$;

– $\text{inv}(\theta)$ le vecteur de paramètres associé à T^{-1}

(4.13) se traduit par :

$$\widehat{\theta_t} = \widehat{\theta_{t-1}} \bullet \text{inv}(\theta_0 + \widehat{A}(F_t(\theta_{t-1}) - F_0(\theta_0)))$$

ce qui est l'équation (8) de Jurie et Dhome [JD02a].

Remarque : en pratique, l'hypothèse d'application ne peut être vérifiée qu'*a posteriori* en confrontant $\theta - \theta_0$ aux valeurs utilisées pendant la phase d'apprentissage ayant permis l'estimation de $\widehat{A}$.

4.1.4 Implémentation

Nous comparons les trois méthodes (NL, HB et JD) afin de les combiner. Nous les évaluons en termes de **robustesse** et de **précision**.

L'implémentation des algorithmes nécessite d'accéder aux images discrètes, par opposition aux images théoriques manipulées jusqu'à présent. Les techniques permettant de le faire sont décrites dans l'appendice 7.2.2.

4.1.4.1 La méthode NL

Notre implémentation de la méthode NL repose sur celle de l'algorithme de Levenberg-Marquardt dans MINPACK [MGH80]. Nous calculons la fonction et sa matrice jacobienne à l'aide d'un échantillonnage bilinéaire. Nous limitons le nombre d'itérations à 5, au-delà les pas deviennent trop petits (largement en-dessous d'un pixel) pour être significatifs.

Cette implémentation de la méthode NL est très précise, mais elle nécessite une bonne initialisation pour converger (elle n'est pas robuste).

4.1.4.2 La méthode HB

Nous avons implémenté la méthode HB sans l'« astuce » qui permet de précalculer les dérivées sur I_0. Elle consiste simplement en un pas d'optimisation de Gauss-Newton, qui requiert de résoudre un système linéaire surdéterminé aux sens des moindres carrés. Nous utilisons la méthode brute pour échantillonner les images et calculer les dérivées.

Indépendamment de l'implémentation, la méthode HB montre ses limites tant en termes de robustesse que de précision.

4.1.4.3 La méthode JD

Dans cette méthode, les paramètres susceptibles d'ajustement sont :
- le nombre d'expériences N faites pendant la phase d'apprentissage ;
- le type de perturbations appliquées pendant les expériences ;
- la méthode utilisée pour estimer la matrice A ;
- le type d'échantillonnage utilisé pour mesurer les niveaux de gris.

Les perturbations. Pour chaque expérience de la phase d'apprentissage, nous calculons un vecteur $\theta'_0 = \theta_0 + \Delta$ en appliquant une perturbation aléatoire aux sommets $\{\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \mathbf{w}_4\}$ du quadrilatère suivi.

Nous favorisons les translations dans le corpus d'apprentissage pour « coller » à une réalité observée dans nos expériences, à savoir la prépondérance d'une composante de translation dans le mouvement. Concrètement nos perturbations intègrent deux composantes : ♠

1. une translation commune $t \in \mathbb{R}^2$ aux quatre sommets. La variable aléatoire t est centrée et isotrope au sens où :

$$E(t) = 0_2 \qquad \text{et} \qquad \arg(t) \sim \mathcal{U}([-\pi, \pi[)$$

La norme du vecteur est proportionnelle à un facteur d'échelle σ_A. Elle peut :
– être la réalisation d'une variable aléatoire gaussienne

$$\|t\| \sim \mathcal{N}(0, \sigma_\text{A})$$

– atteindre son maximum ailleurs qu'en $(0, 0)$. La « couronne uniforme », par exemple, correspond à

$$\|t\| \sim \mathcal{U}((1-\alpha)\sigma_\text{A}, (1+\alpha)\sigma_\text{A})$$

où $\alpha = 0.25$ donne de bons résultats ;

2. un « déplacement » individualisé pour chaque sommet. Il suit une loi gaussienne centrée de matrice de variance-covariance $\sigma_\text{B}\text{Id}_2$.

L'algorithme 4.1 décrit notre implémentation de la génération d'une perturbation « artificielle » de la géométrie de la cible 2D et du calcul de l'homographie qui en résulte.

$t \leftarrow$ `translation_aléatoire`(σ_A) -- translation commune
Pour i **de** 1 **à** 4 **faire**
  $\mathbf{u}_i \leftarrow T(\theta_0, \mathbf{w}_i) + t +$ `vecteur_aléatoire` (σ_B) -- perturbations individuelles
finpour
$\theta'_0 \leftarrow$ `homographie`$((\mathbf{w}_i)_{i=1..4}, (\mathbf{u}_i)_{i=1..4})$

Fonctions utilisées :
– `translation_aléatoire`(σ) renvoie un vecteur aléatoire bidimensionnel, d'argument indépendant de sa norme. La distribution de la norme peut être ajustée proportionnellement à σ ;
– `vecteur_aléatoire`(σ) « tire » indépendamment les coordonnées d'un vecteur 2D selon une loi gaussienne centrée, d'écart-type σ ;
– `homographie`$((a_i)_{i=1..4}, (b_i)_{i=1..4})$ calcule les coefficients de l'homographie unique qui transforme a_i en b_i pour $i \in [\![1, 4]\!]$.

Algorithme 4.1: « Tirage » aléatoire d'une homographie θ'_0.

Remarque. Nous utilisons souvent un seul coefficient σ, avec $(\sigma_\text{A}, \sigma_\text{B}) = (0.7\sigma, 0.3\sigma)$: plus σ est grand, plus le suivi est robuste mais moins précis et vice-versa.

Astuces numériques.

Conditionnement. Avant d'estimer la matrice A par une méthode OLS, TLS ou DLS, nous nous assurons que la matrice G n'est pas singulière. Pour cela, nous supprimons les lignes de G contenant une proportion de valeurs nulles supérieure à 80 %. Ces lignes correspondent à des points de référence dont le voisinage a un niveau de gris uniforme.

En pratique, c'est suffisant pour que G soit bien conditionnée : son conditionnement ([Cia85] p29) est de l'ordre de 1500 (11 positions binaires sur 53 en double précision).

Représentation de l'homographie. L'expression de Δ comme une différence d'éléments de matrice d'homographie fait que ses composantes sont d'ordres de grandeur très disparates, ce qui est nuisible à la stabilité de l'estimation.

Pour contourner cet inconvénient, lors du calcul de $\widehat{A}$ et son utilisation, nous exprimons l'homographie à l'aide d'un ensemble de paramètres équivalent $\mathbf{d} = (\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4)$, où $\mathbf{d}_i \in \mathbb{R}^2$. Pour une homographie donnée, la relation entre θ et $\mathbf{d}$ est :

$$T(\theta, \mathbf{w}_i) = \mathbf{w}_i + \mathbf{d}_i \qquad \forall i = 1..4$$

c'est-à-dire que $\mathbf{d}_i$ est le déplacement que l'homographie applique au sommet $\mathbf{w}_i$ de la cible 2D. Tous les éléments de ce déplacement sont en pixels et du même ordre de grandeur.

La représentation $\mathbf{d}$ est donc plus équilibrée que la θ mais l'opération de composition est plus difficile à exprimer. Le passage d'une représentation à l'autre est relativement facile.

Échantillonnage. L'algorithme est basé sur de l'échantillonnage brut car :
- la qualité de la méthode JD qui nous intéresse est la robustesse ;
- un échantillon est alors un octet (et non un flottant) qu'il est facile de stocker et de manipuler avec des instructions vectorielles (appendice 7.3).

4.1.4.4 Diagnostic

Il est intéressant d'avoir une évaluation de la qualité de l'estimation fournie par les méthodes de suivi. Pour toutes les méthodes, nous utilisons à cet effet le critère d'optimisation.

De plus, dans le cas JD, nous utilisons un critère de cohérence avec les expériences menées pendant la phase d'apprentissage. Pour cela, nous calculons le déplacement que ♠
l'homographie θ' calculée applique aux sommets $(\mathbf{w}_i)_{i=1..4}$:

$$\mathbf{d}_i = T(\theta', \mathbf{w}_i) - \mathbf{w}_i \qquad \forall i = 1..4$$

En fait, les $\mathbf{d}_i$ sont directement disponibles grâce à la représentation ci-dessus de l'homographie.

L'intensité moyenne de ces déplacements est :

$$d = \frac{1}{4}(\|\mathbf{d}_1\| + \|\mathbf{d}_2\| + \|\mathbf{d}_3\| + \|\mathbf{d}_4\|)$$

En pratique, nous avons déterminé que si $d > 3.5\sigma$ (σ est l'amplitude de la perturbation, voir § 4.1.4.3), alors le résultat en sortie de JD est plus mauvais que l'estimation en entrée.

4.1.4.5 Combinaison

Le résultat $\widehat{\theta}$ d'une méthode peut être utilisé comme valeur initiale à l'entrée d'une autre, en remplaçant $\widehat{\theta_{t-1}}$. En effet, nous utilisons $\widehat{\theta_{t-1}}$ uniquement comme approximation de θ_t sans supposer qu'il concerne l'image précédente.

En conséquence, les méthodes peuvent être appliquées séquentiellement en combinaison avec une approche hiérarchique. Nous appliquons les méthodes les plus robustes en premier, puis les plus précises. Ceci mène à la chaîne de traitements suivante : ♠

1. quelques itérations de JD, avec des σ décroissants ;
2. une estimation de HB ;
3. quelques itérations de la méthode NL.

Après chaque étape d'estimation, nous contrôlons la décroissance du critère e_t. Si le test est satisfaisant, la nouvelle estimation devient l'hypothèse courante, sinon nous gardons l'ancienne estimation. La figure 4.3 résume cette chaîne de traitements.

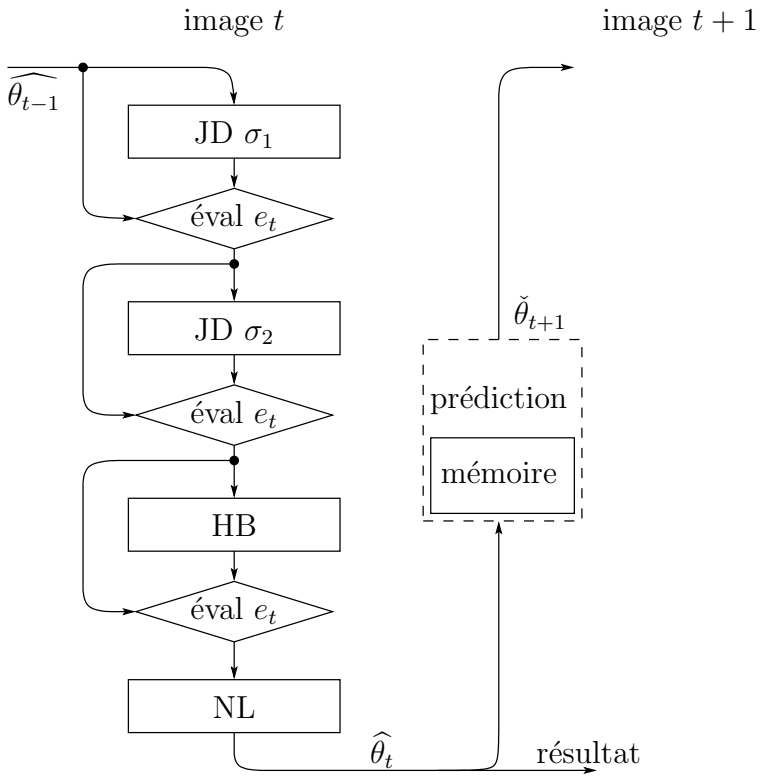


FIG. 4.3: Traitement de l'image I_t. Les données d'entrée arrivent par la gauche (l'image précédente) et sont affinées dans une chaîne de traitements. Elle se compose d'unités de suivi (JD, HB et NL) et d'opérateurs annexes. Les paramètres des homographies empruntent les chemins fléchés. L'entrée d'une unité de suivi est une estimation initiale de θ_t, sa sortie est une nouvelle estimation de θ_t. L'opérateur « éval e_t » sélectionne le meilleur (au sens du critère) θ parmi ses entrées. Un éventuel opérateur de prédiction estime θ_{t+1} d'après l'historique.

4.1.4.6 Prédiction

Si le mouvement de la région cible est suffisamment régulier, il est intéressant de prédire sa position en entrée de la chaîne de traitements de l'image suivante.

Le schéma de prédiction le plus simple consiste à supposer que l'homographie entre I_{t+1} et I_t est la même qu'entre I_t et I_{t-1}. Ceci fournit la prédiction suivante pour θ_{t+1} : ♠

$$\check{\theta}_{t+1} = \theta_t \bullet \mathrm{inv}(\theta_{t-1}) \bullet \theta_t$$

Si les estimations de θ_{t-1} et θ_t sont exactes, ce schéma prédit correctement :
- les translations à vitesse constante ;
- les rotations de centre fixe, à vitesse constante ;
- homothéties de centre fixe et dont le facteur croît géométriquement ;
- les similitudes qui combinent les rotations et homothéties ci-dessus.

La prédiction requiert de maintenir une mémoire des transformations antérieures.

4.1.5 Expérimentation

Nous avons fait des expériences d'abord sur des images de synthèse pour optimiser les paramètres des algorithmes, puis sur des images réelles.

4.1.5.1 Images de synthèse

Nous avons choisi une image *a priori* idéale pour faire le suivi (figure 4.4). Nous avons ensuite perturbé l'image de trois manières différentes :
- rotation : de 1° à 20° entre deux images successives ;
- zoom : nous alternons entre agrandissement et rapetissement de l'image, jusqu'à un facteur 2.5 et par pas variant de 1% à 20%.
- déformation : le motif est déformé comme si son antécédent 3D (plan) avait subi une rotation de plus en plus rapide autour d'un axe coplanaire d'angle variant entre -40° et +40° ;

Les perturbations sont de plus en plus intenses au fur et à mesure du suivi. Par exemple, pour la séquence rotation, l'angle de rotation α entre deux images est (en degrés) : $\alpha = \frac{t}{20} + 1$. En conséquence inéluctable, le suivi échoue toujours au bout d'un certain nombre d'images.

Critère de validation. Comme nous disposons ici de la « vérité terrain, » nous pouvons mesurer directement l'erreur de suivi. Si θ_t est la transformation réelle, $\widehat{\theta_t}$ la version estimée et $(\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \mathbf{w}_4)$ les sommets du quadrilatère suivi, nous définissons l'erreur (critère géométrique) :

$$\varepsilon_t = \sum_{i=1}^{4} \parallel \mathbf{w}_i - T(\mathrm{inv}(\theta_t), T(\widehat{\theta_t}, \mathbf{w}_i)) \parallel^2$$

La région suivie est considérée « perdue » quand ε_t dépasse un seuil s. La robustesse du suivi est caractérisée par la valeur t^* minimale satisfaisant à $\varepsilon_{t^*} > s$. Nous choisissons

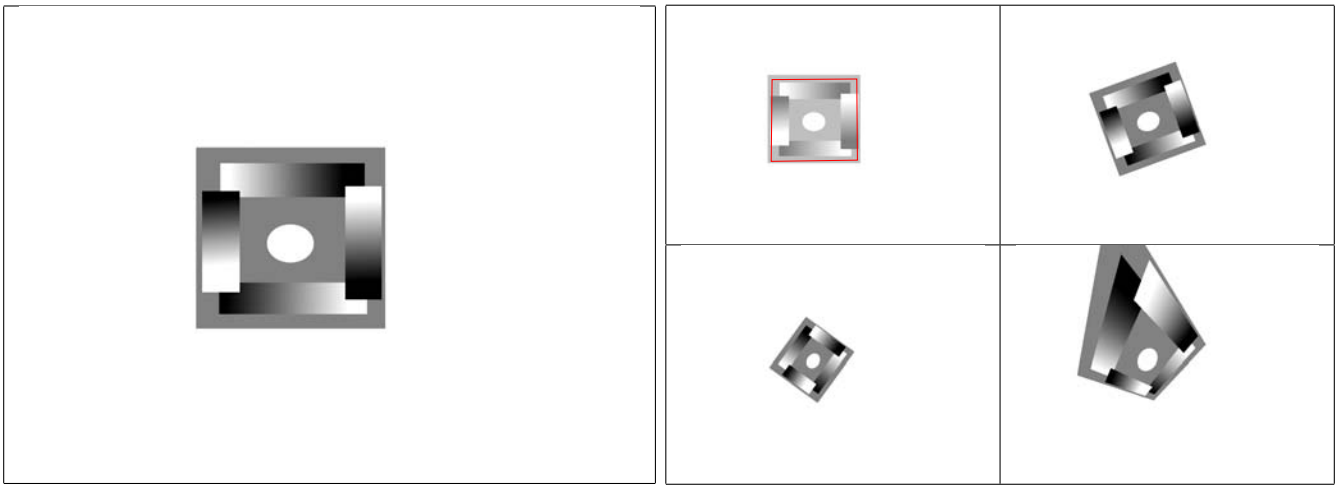


FIG. 4.4: À gauche : l'image synthétique (1024×768 pixels) utilisée pour les tests. À droite, dans l'ordre de lecture : la région suivie $\mathcal{Q}$ (encadrée, elle fait environ 250×250 pixels), l'image après rotation, zoom et déformation.

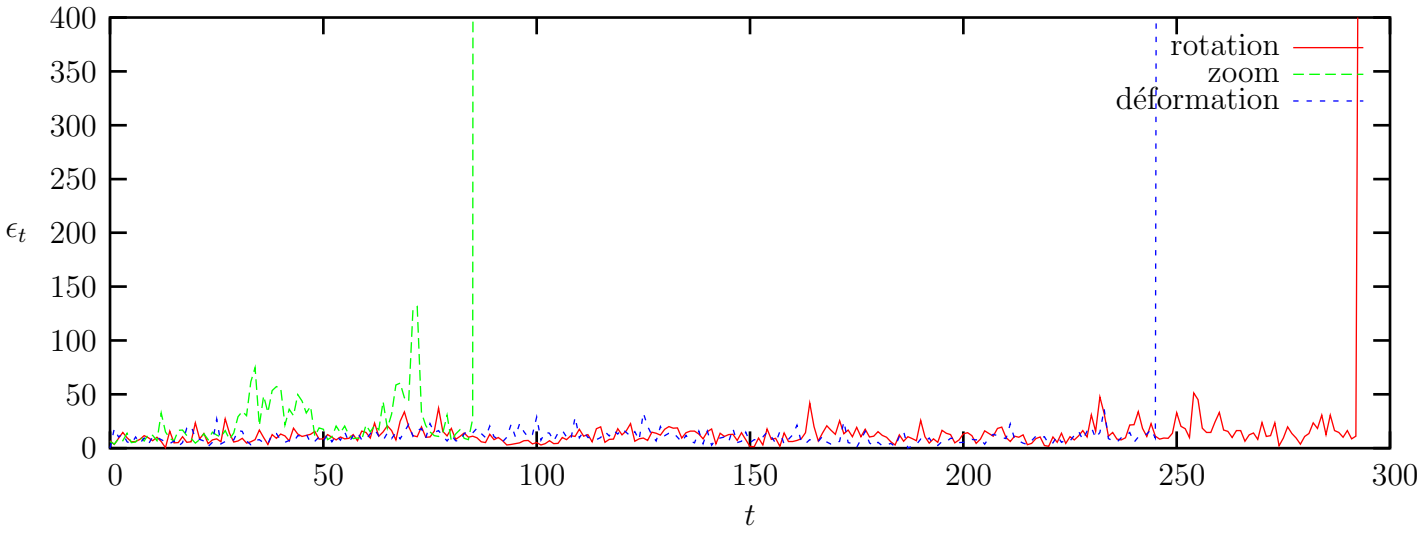


FIG. 4.5: Évolution de l'erreur ε_t en fonction du temps pour les trois séquences, avec notre algorithme.

$s = 400 = 4 \times 10^2$, ce qui représente une erreur quadratique moyenne de 10 pixels par sommet. La précision du suivi est quantifiée par la valeur moyenne $\bar{\varepsilon}$ de $(\varepsilon_t)_{t=1..t^*}$.

4.1.5.2 Évolution du critère

Pour notre algorithme, les résultats sont présentés à la figure 4.5. On voit que le suivi est toujours très précis jusqu'au moment où il échoue.

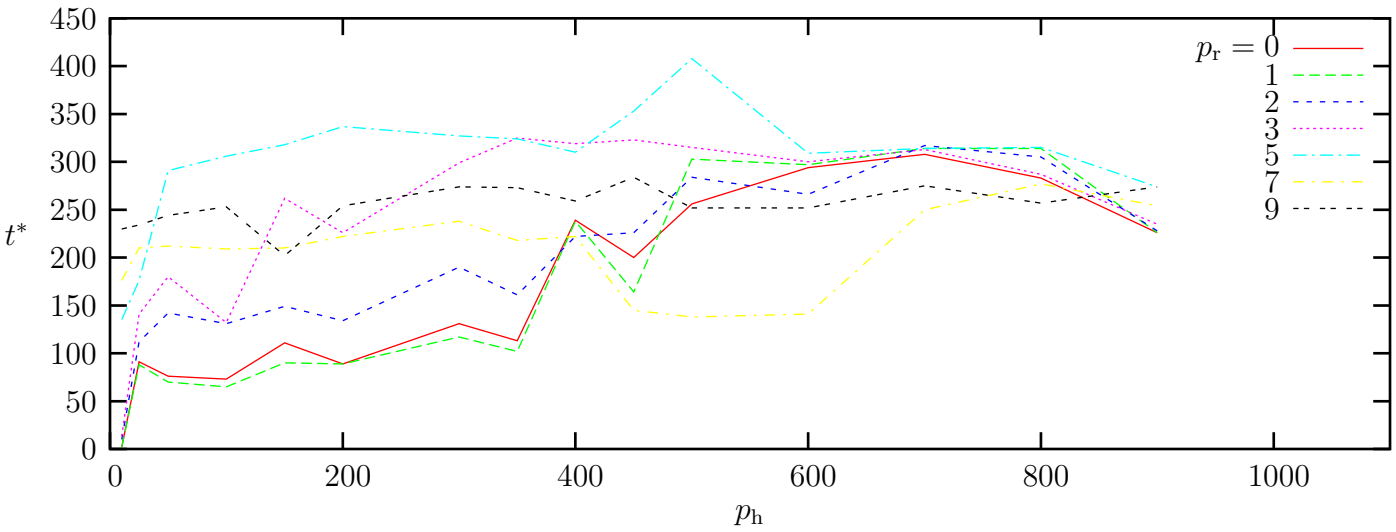


FIG. 4.6: Durée t^* du suivi réussi en fonction du nombre de points d'intérêt p_h et de points réguliers p_r^2.

4.1.5.3 Influence du nombre de points de référence

Nous avons cherché à quantifier le nombre de points de référence nécessaires pour le suivi le plus robuste. Pour cela, nous utilisons notre algorithme sur la séquence « rotation », en faisant varier le nombre p_r^2 de points réguliers et le nombre p_h de points d'intérêt. Le résultat est décrit dans la figure 4.6.

Les résultats sont assez instables. Cependant, on peut voir que :

- pour un nombre de points réguliers donné, il y a une valeur optimale du nombre de points d'intérêt ;
- la valeur optimale de p_r est 5.

Pour cette séquence, les valeurs optimales sont $(p_h, p_r^2) = (500, 25)$. Les résultats sont semblables pour les deux autres séquences.

4.1.5.4 Séquence d'algorithmes

Nous avons enchaîné les algorithmes de diverses manières, pour observer l'effet sur t^* et $\bar{\varepsilon}$. Les résultats sont résumés dans le tableau 4.1.

L'enchaînement qui offre le meilleur compromis robustesse/précision correspond à la chaîne d'estimations intégrant quatre étapes JD associées à des phases d'apprentissage d'amplitude de perturbation :

$$(\sigma_1, \sigma_2, \sigma_3, \sigma_4) = (15, 10, 5, 2.5)$$

séquence	t^*	$\bar{\varepsilon}$
NL	11	40.66
HB	11	70.46
JD 5	85	98.41
HB, NL	19	49.17
JD 5, NL	111	106.84
JD 5, HB	111	30.82
JD 5, HB, NL	121	27.04
JD 5, JD 2.5, HB, NL	134	14.34
JD 10, JD 5, JD 2.5, HB, NL	149	14.98
JD 12.5, JD 10, JD 5, JD 2.5, HB, NL	238	28.10
JD 15, JD 10, JD 5, JD 2.5, HB, NL	278	31.67
JD 20, JD 10, JD 5, JD 2.5, HB, NL	112	10.09

TAB. 4.1: Durée t^* (en images) du suivi réussi et erreur moyenne (en pixels) $\bar{\varepsilon}$ pour différentes combinaisons de méthodes (en utilisant la notation compacte de la figure 4.3).

4.1.5.5 Apport de TLS

Nous avons comparé les performances de notre algorithme en utilisant la méthode OLS ou TLS pendant la phase d'apprentissage des étapes de Jurie et Dhome. Ces performances dépendent fortement du nombre d'expériences (figure 4.7).

Il n'y a pas d'avantage clair à utiliser TLS pour l'estimation de la phase d'apprentissage.

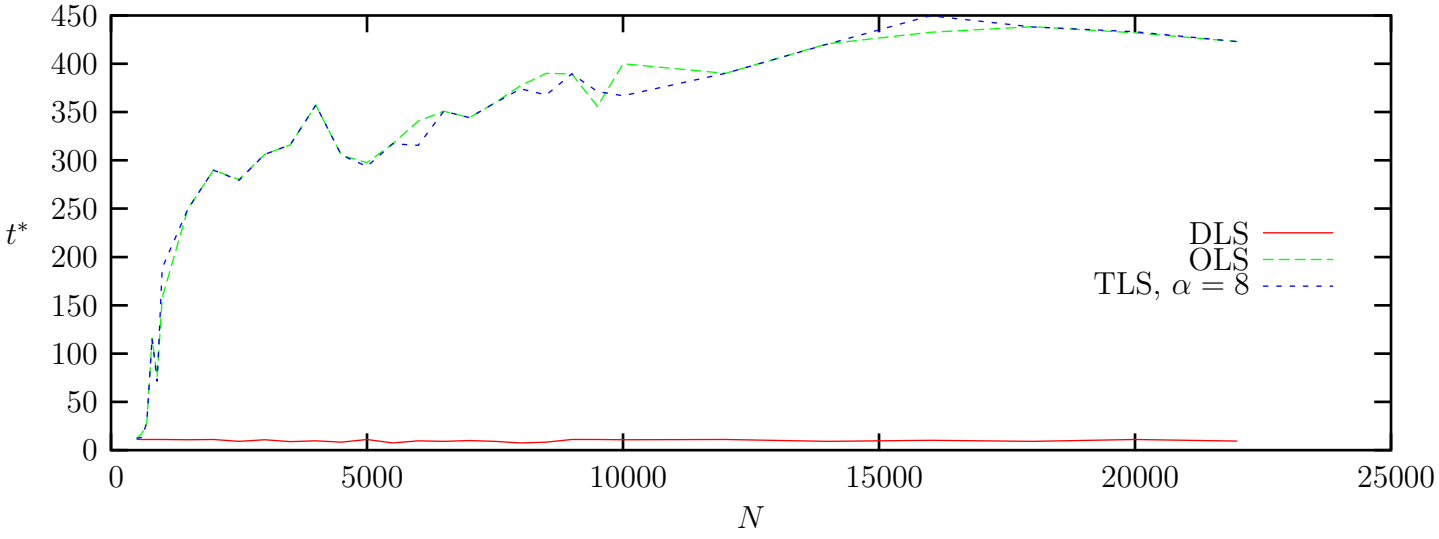


FIG. 4.7: Performances du suivi en fonction de la méthode d'estimation et du nombre N d'expériences.

4.1.5.6 La prédiction

Nous avons comparé le suivi avec et sans prédiction sur la séquence « déformation ». L'évolution du critère réel est montrée sur la figure 4.8.

Le suivi réussit beaucoup plus longtemps avec l'opérateur de prédiction. Pour $t > 300$, les images successives sont tellement dissemblables qu'un observateur expert est lui-même incapable de réaliser le suivi. La prédiction donne des résultats encore meilleurs avec la séquence « rotation ». Une conclusion trop hâtive concernant le bien-fondé de notre méthode de prédiction doit être tempérée par la régularité des mouvements artificiels.

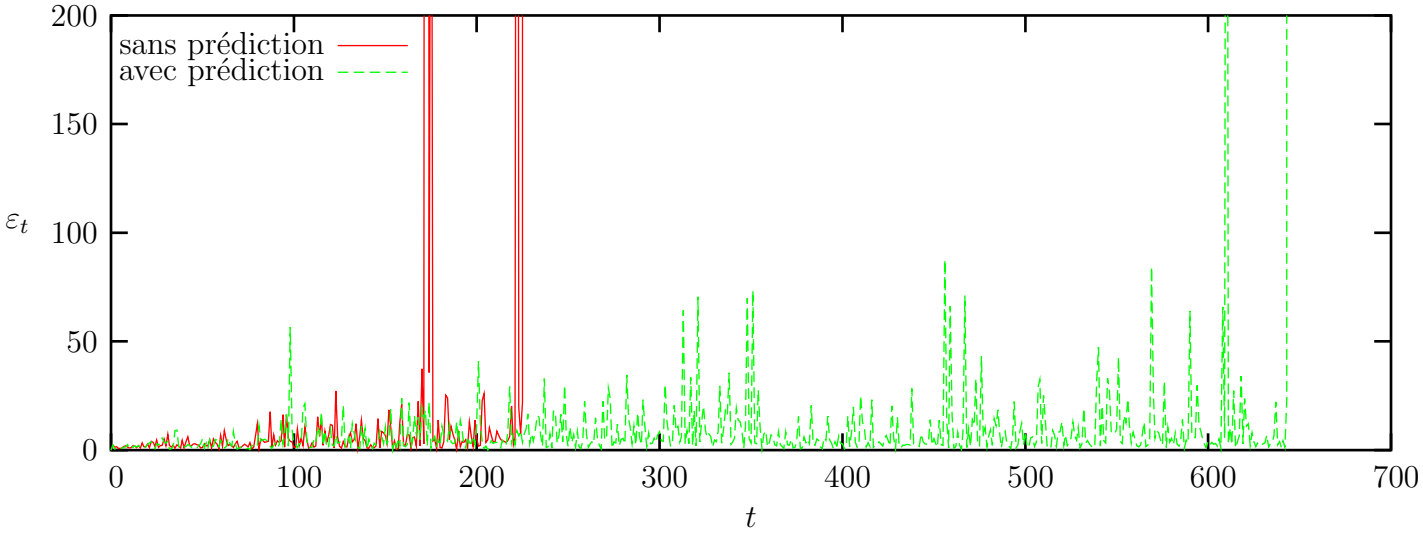


FIG. 4.8: Évolution du critère au cours du suivi avec et sans prédiction.

4.1.5.7 Types de motifs

Nous avons cherché à cerner le « profil » du motif facile à suivre. Pour cela, nous avons généré plusieurs motifs synthétiques périodiques (figure 4.9, première ligne). La fréquence f est exprimée en fonction de la taille de la cible carrée : à $f = 1$, une seule période est visible dans la cible. Le motif est utilisé dans la séquence « rotation » où le centre de rotation coïncide avec celui de la cible.

La grande régularité de certains motifs provoque des instabilités numériques, ce qui nous a amené à ajouter un bruit gaussien centré d'écart-type 0.5 (sur 256 niveaux de gris) aux mesures. Cette procédure améliore la stabilité de l'estimation de la matrice A.

Les résultats sont présentés dans la deuxième ligne de la figure 4.9.

Le motif SIN. C'est un produit tensoriel de deux sinusoïdes discrétisées. Le suivi fonctionne pour $f \in [1/5, 5]$. La valeur maximale $t^* = 1100$ est obtenue pour $f = 0.8$, et correspond à une rotation de 80° entre deux images successives !

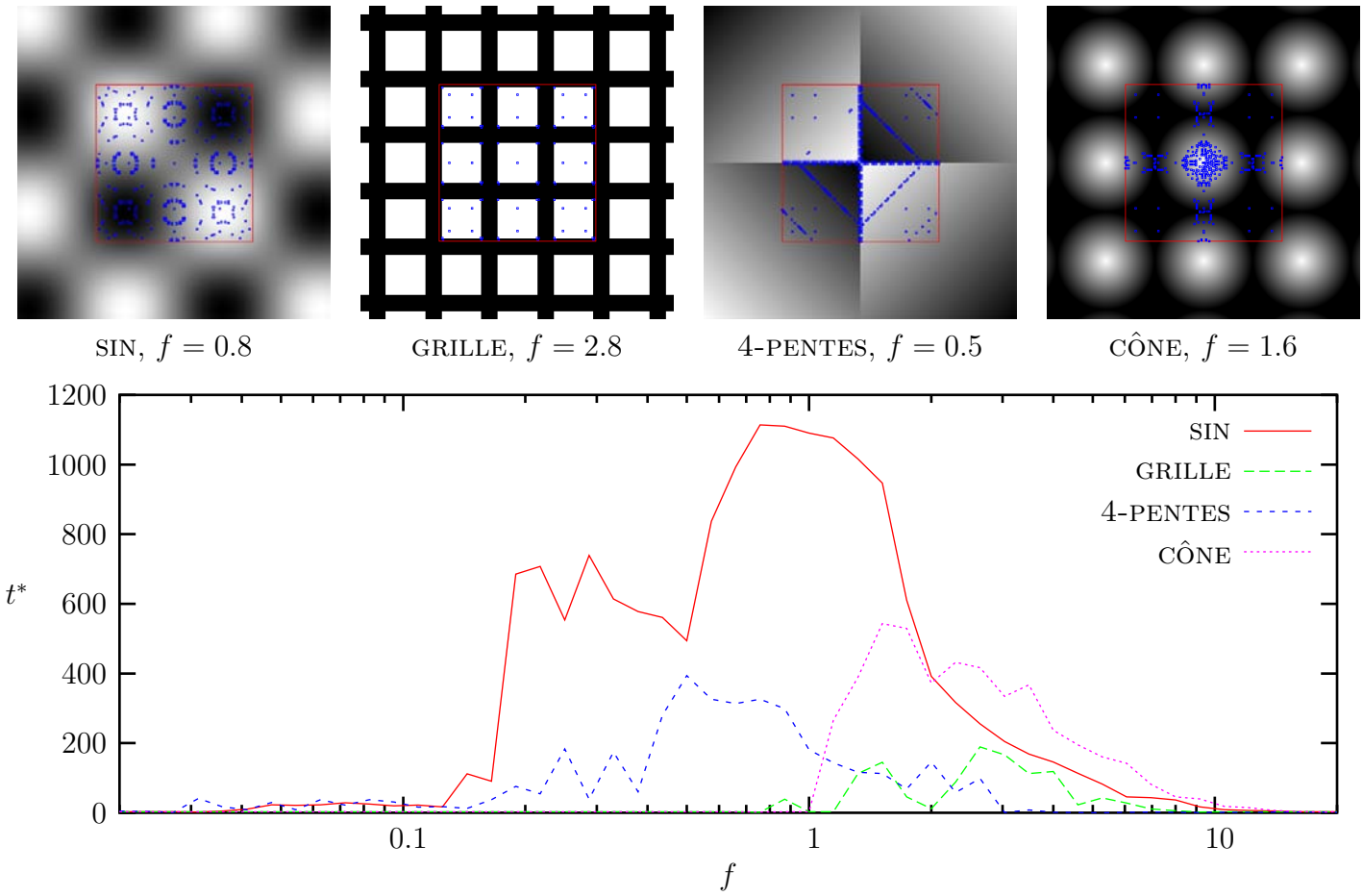


FIG. 4.9: Première ligne : exemples de motifs. La cible est encadrée en rouge et les points de référence sont en bleu. Deuxième ligne : nombre d'images suivies en fonction de la fréquence spatiale du motif, pour les exemples présentés. Les motifs sont représentés à la fréquence où le suivi est le plus efficace.

Le motif GRILLE. Le suivi est relativement inefficace. Il ne devient possible qu'à partir du moment où la cible englobe plusieurs périodes ($f > 1$). L'inefficacité du suivi est aussi due au faible nombre de points trouvés par le détecteur de points d'intérêt. Les résultats sont analogues pour un motif en damier.

Le motif 4-PENTES. Il est constitué de 4 dégradés (le niveau de gris est une fonction affine des coordonnées). Pour $f < 1$, il y a un compromis :
- si $f \approx 1$, il y a une forte dynamique de niveaux de gris mais les bords du motif sont de niveaux de gris uniformes ;
- si $f \approx 0$, on se rapproche d'un motif en damier : il y a un fort contraste entre les régions, mais peu de variations lentes de luminosité.

La position moyenne ($f = 1/2$) est optimale.

Le motif CÔNE. C'est un cône où le niveau de gris d'un point représente la cote.

Bilan. En général, le suivi est impossible si la cible est telle que

$$\exists \theta_{\text{id}} \in \mathbb{R}^p \text{ « petit » }, \forall \mathbf{m} \in \mathcal{Q}, I_0(T(\theta, \mathbf{m})) = I_0(\mathbf{m})$$

car l'algorithme peut trouver $\theta_{\text{id}} \bullet \theta_t$ au lieu de θ_t (« petit » signifie proche de l'identité). Ceci explique que :
- le suivi est difficile si la fréquence est trop élevée ($f > 5$). θ_{id} est alors une translation d'une période du motif, suffisamment petite pour que l'algorithme s'y trompe ;
- le suivi est impossible si les niveaux de gris du motif sont définis par une seule fonction affine. θ_{id} est alors une translation dans une direction orthogonale au vecteur gradient ;
- le suivi est impossible pour le motif SIN quand la vitesse de rotation dépasse 90° entre deux images : θ_{id} est la rotation d'angle 180° et $\theta_{\text{id}} \bullet \theta_t$ est plus proche de θ_{t-1} que θ_t ;
- le suivi est impossible sur le motif CÔNE quand une seule période est visible ($f < 1$). θ_{id} est une rotation quelconque autour du centre du motif.

Sous réserve des remarques ci-dessus, les motifs les plus faciles à suivre sont ceux dont les niveaux de gris varient de manière continue car :
- l'approximation linéaire de $F'(\theta)$, sur laquelle sont basées les méthodes de suivi est définie et valide sur un voisinage plus important ;
- le détecteur de Harris est totalement inopérant dans les régions uniformes.

4.1.5.8 Images réelles

Implémentation. Nous traitons en temps réel un flux vidéo FireWire (provenant d'une caméra DV) sur un PowerPC G4 866, en visualisant les résultats à l'aide d'une interface. Nous avons accéléré la phase d'apprentissage en tirant parti des instructions vectorielles du processeur (appendice 7.3). Elle prend de l'ordre d'une seconde selon le nombre de points de référence.

Les images arrivent à une fréquence de 25 Hz, mais leur traitement doit durer moins de 20 ms parce que la décompression du flux vidéo et l'affichage prennent les 20 ms restantes. Nous ne pouvons pas sauter d'image, parce que des mouvements brutaux de la scène pourraient nous échapper.

Comme on le voit dans le tableau 4.2, les opérations sont assez rapides pour faire plusieurs étapes d'estimation pour chaque image. La méthode de Jurie et Dhome est la plus rapide. La méthode de Hager et Belhumeur est plus lente, parce qu'elle nécessite la décomposition QR d'une matrice $n \times (p+1)$. La méthode non linéaire, bien que plus lente, demeure dans les délais.

nombre de points de référence	125	175	225	325	425	525
échantillonnage	0.074	0.113	0.127	0.181	0.238	0.289
calcul de e_t	0.012	0.015	0.018	0.027	0.031	0.038
calcul de JD	0.209	0.237	0.264	0.351	0.415	0.490
calcul de HB	1.269	1.741	2.272	3.269	4.317	5.328
calcul de NL	1.527	2.024	2.640	3.781	5.095	6.020

TAB. 4.2: Temps d'exécution pour les opérations effectuées au cours du suivi (figure 4.3), en millisecondes. Nous les avons mesurés sur notre combinaison d'algorithmes, en fonction du nombre de points de référence.

Exemples. En tournant les actes de RFIA 2002 (figure 4.10, première image) en tous sens, nous nous sommes rendus compte de l'importance de la planéité de la surface. Nous avons donc collé une feuille texturée et mate sur un carreau de verre. Le suivi devient alors très fiable.

Nous avons fait les expériences en déplaçant le motif plan, plus ou moins vigoureusement, tout en effectuant des zooms (figure 4.10, trois dernières images). Les différents types de mouvement sont représentés (en 2D, ils se traduisent par des translations, des rotations, des zooms, et des déformations dues à la projection).

4.2 Suivi avec occultations

Des éléments peuvent apparaître au cours de la séquence et occulter une partie de la cible 2D. Ces éléments sont quelconques : ils peuvent être des objets, des personnages, mouvants ou non. Ces occultations violent les contraintes énoncées dans le § 4.1.2.1 : un traitement particulier des occultations au cours du suivi s'impose.

Appelant τ_s le rapport entre la surface occultée et la surface de la région cible, nous distinguons trois situations, en fonction de l'ordre de grandeur de τ_s :

1. $\tau_s \leq 5$ %. Nous considérons que le suivi n'est pas perturbé ;
2. 5 % $< \tau_s \leq 50$ %. Le suivi est possible moyennant un traitement approprié. Il est de plus en plus hasardeux à mesure que τ_s augmente ;

FIG. 4.10: Images extraites de quatre séquences vidéo acquises à l'aide d'une caméra DV. Notre algorithme réussit sans difficulté sur ces quatre vidéos.

3. $\tau_s > 50$ %. Le suivi est considéré impossible. Toutefois, sous des hypothèses de continuité du mouvement, il est envisageable de retrouver la région après une occultation temporaire.

Nous traitons le second cas en ayant pour objectif une méthode qui s'intègre « harmonieusement » dans le cadre du suivi sans occultation.

4.2.1 État de l'art

Plusieurs méthodes de suivi robustes aux occultations existent, nous les examinons sous deux angles : la modélisation et la résolution du modèle.

4.2.1.1 Suivi par mise en correspondance

Les paramètres sont estimés à partir de paires de (fenêtres autour de) points d'intérêt homologues.

Les objets occultants apparaissent comme des mesures aberrantes sur certaines paires. Ces aberrances, minoritaires, peuvent être détectées et supprimées grâce à des critères et des méthodes de résolution robustes (§ 2.5.3 et 2.5.4). Rien ne distingue les points issus d'appariements erronés des occultés ([VGBS03] § 1.2).

Cette technique est souvent utilisée pour produire une image panoramique ([BDH03] § 4.2, [SK04] § 3) à partir d'une séquence vidéo.

4.2.1.2 Suivi par « segmentation de mouvement »

Le mouvement global est estimé à partir d'un **champ dense de mouvements locaux**, typiquement des translations obtenues par flot optique.

On considère que les images sont subdivisées en régions subissant des mouvements indépendants. L'une d'elles est la cible, les autres correspondent aux objets occultants. Les régions sont détectées par une segmentation dont le critère d'homogénéité se réfère au mouvement.

La segmentation peut emprunter deux directions :

– la fusion ([DM96] § 2.1) ;

– la division : Galpin *et al.* ([GOM01] § 2.2) en déduisent un maillage adaptatif .

Dans les deux cas apparaît le **problème d'ouverture** : le mouvement est estimé sur une fenêtre qui peut chevaucher plusieurs régions. Pour remédier à ceci, Black et al. ([BYJF97] § 3.1) proposent de détecter spécifiquement les discontinuités de mouvement.

Masson *et al.* [MDJ04] suivent un polyèdre de géométrie connue en calculant un champ de mouvements à partir de nombreuses petites régions (*patchs*) texturées qui recouvrent ses faces. Ils combinent les mouvements élémentaires (similitudes) des patchs pour en déduire la pose du polyèdre. Cette opération de combinaison repose sur un M-estimateur, ce qui permet de rejeter les patchs mal localisés ou occultés.

4.2.1.3 Plusieurs hypothèses

L'algorithme CONDENSATION [IB96] est utilisé pour suivre une cible dans un environnement encombré. Il ne manipule pas les valeurs des paramètres, mais une **distribution de probabilité** de ces paramètres. En cas d'incertitude sur la position de la région, les différentes hypothèses sont représentées jusqu'à ce que l'ambiguïté soit levée. Cette technique permet en particulier de résister aux occultations passagères.

4.2.1.4 Suivi à l'aide d'un masque d'occultation

La cible 2D est ici un ensemble de points (dense ou non). Quand un objet occultant passe devant la cible 3D, la mesure de luminance ou de couleur devient inutilisable en certains points. La technique du **masque d'occultation** ($w \in \{0,1\}^n$ voire $w \in [0,1]^n$) permet de pallier cet inconvénient : $w_i = 0$ signifie que le point $\mathbf{m}_i$ est invisible, $w_i = 1$ qu'il est visible.

Cas où le masque est connu. L'estimation du mouvement peut se faire à partir de l'équation (4.4) où on **supprime la contribution des points occultés** :

$$\begin{cases} \min \|W(A_t\Delta - b_t)\| \\ \Delta \in \mathbb{R}^p \\ W = \mathrm{diag}(w_1, ..., w_n) \end{cases}$$

Dans le cadre d'une approche probabiliste, le masque influence la **variance** (qui code l'incertitude) des mesures utilisées dans l'estimation des paramètres. Si le masque est proche de 1, la variance est faible, s'il tend vers 0, la variance est élevée ([BB01] § 3.0.13).

Cas où le mouvement est connu. On peut utiliser l'hypothèse photométrique pour détecter les points occultés. Le masque w est alors déterminé par comparaison des niveaux de gris mesurés avec ceux de l'image de référence. La détermination la plus simple s'écrit :

$$w_i = s(|I_t(\theta_t, \mathbf{m}_i) - I_0(\mathbf{m}_i)|)$$

où $s : \mathbb{R} \longrightarrow [0,1]$ est une fonction décroissante. Des déterminations plus élaborées sont possibles :

- le masque w_i peut être calculé sur un voisinage en favorisant les points où le gradient est fort, car leur texture est plus reconnaissable ([IRP94] eq (5)) ;
- si les images utilisées sont en couleurs, on peut s'appuyer sur une distance de Mahalanobis ([BDH03] § 6.1) entre vecteurs de composantes primaires. La matrice de variance-covariance est estimée de manière appropriée, pixel par pixel.

Dans le cas d'une région dense, beaucoup d'auteurs appliquent des opérateurs morphologiques pour lisser le masque d'occultation, et en conséquence le rendre localement plus cohérent.

Cas mixte. Ainsi, il existe des techniques pour estimer les paramètres connaissant le masque et réciproquement. La méthode la plus simple pour faire le suivi dans ces conditions consiste pour l'image courante I_t à :

1. estimer les paramètres en utilisant le masque de l'image I_{t-1} ;
2. estimer le masque utile à l'image I_{t+1} à partir des paramètres courants.

Le masque a toujours une image de retard ! Il peut cependant être utile, parce que l'hypothèse des petits changements garantit un mouvement d'amplitude faible de la cible 2D.

Une résolution plus satisfaisante de ce problème de « poule et d'œuf », est d'**itérer les deux estimations** sur chaque image. Ceci peut se faire simultanément avec un autre processus itératif, comme une pyramide multirésolution ([IRP94] § 2.4).

Une autre piste est de poser le problème (4.4) avec un critère de M-estimateur (§ 2.5.3). Le masque apparaît alors explicitement lors de la résolution par moindres carrés repondérés (§ 2.6.3.3). Cette technique a été utilisée :

- pour trouver les coefficients d'une analyse en composantes principales de l'image d'une canette de soda ([BJ98] § 4). La fonction de modération ρ utilisée est celle de l'équation (2.6) où σ décroît au cours des itérations ;
- pour estimer un mouvement affine. Hager et Belhumeur ([HB98] eq.(49)) choisissent une fonction ρ convexe qui assure la convergence globale.

4.2.1.5 Ce que nous en retenons

Nous voulons que le mécanisme de traitement des occultations s'intègre à notre système de suivi, et qu'il soit rapide : nous calculons un masque d'occultation.

Nous adaptons la technique de moindres carrés repondérés de Hager et Belhumeur. En notant i l'indice de l'itération, leur proposition est de transformer l'équation (4.9) en :

$$\widehat{\Delta_i} = F'_{t-1}(\widehat{\theta_{t-1}})^+(W_i(F_0(\theta_0) - F_t(\widehat{\theta_{t-1}})))$$

où $W_i \in \mathbb{R}^{n\times n}$ est mise à jour à chaque itération en utilisant l'approximation linéaire de F_t autour de $\widehat{\theta_{t-1}}$.

Cette forme mène à une optimisation intéressante du traitement dans le cas où F_t peut être décomposée (§ 4.1.3.3), mais ce n'est pas un M-estimateur au sens propre (§ 2.5.3). En effet, un point occulté est traité numériquement comme un point dénué de variation de niveau de gris. La relation liant Δ_i au vecteur de variation de niveaux de gris étant linéaire, une sous-estimation de l'amplitude des mouvements en résulte inévitablement.

4.2.2 Modélisation

Les modèles photométrique et géométriques sont les mêmes qu'au § 4.1.2, mais ils sont restreints à un sous-ensemble de points de référence.

4.2.2.1 Le masque d'occultation

Nous choisissons $w_i \in \{0, 1\}$ car :
- la signification d'un masque continu est peu claire : est-ce une probabilité d'occultation ? un objet occultant translucide ? une fraction de pixel occultée ?
- l'estimation est sous la contrainte du temps réel. Nous calculons un masque plus élaboré pour la détection des « intrus » (§ 5.2) ;
- un masque à valeurs dans $[0, 1]$ ne pourrait être traité dans le cas JD.

Nous utilisons le masque d'occultation de l'image précédente comme estimation initiale pour l'image courante, ce qui est raisonnable lorsque le mouvement de l'objet occultant est « proche » de celui de la cible.

4.2.2.2 Notations

Dans ce qui suit, nous notons :
- n' le nombre de points visibles ;
- $\Omega \in \{0, 1\}^{n' \times n}$ la matrice obtenue en supprimant les lignes de Id_n correspondant aux points occultés ;
- p_i l'indice du $i^{\text{ième}}$ point occulté.

Remarque : Les notations W, $(w_i)_{i=1..n}$ et Ω sont redondantes, en effet :

$$W = \mathrm{diag}(w_1, ..., w_n) \qquad \Omega^\top \Omega = W \qquad \forall i \in [\![1, n']\!], i = \sum_{j=1}^{p_i} w_j \qquad n' = \sum_{i=1}^{n} w_i$$

4.2.3 Résolution

Dans la phase de suivi de la méthode JD, le critère optimisé n'est pas explicite. Nous ne pouvons donc pas y greffer un M-estimateur .

Nous itérons les opérations d'estimation des paramètres et de calcul du masque.

4.2.3.1 Détection des occultations

Tout point de référence $\mathbf{m}$ est considéré occulté si son niveau de gris est en contradiction avec une analyse statistique grossière de son voisinage faite sur I_0 simultanément à la phase d'apprentissage de JD.

L'analyse statistique consiste à calculer l'écart-type σ de

$$\{I_0(T(\theta_0 + \Delta_k, \mathbf{m})) - I_0(\mathbf{m}), k = 1..N\}$$

À l'image courante du suivi, avec les paramètres θ, il y a « contradiction » ($w_i = 0$) si $|I_t(T(\theta, \mathbf{m})) - I_0(\mathbf{m})| > 2.5\sigma$.

Si la distribution des niveaux de gris dans le voisinage de $\mathbf{m}$ est gaussienne de moyenne $I_0(\mathbf{m})$ alors seul 1 % de points dépasse ce seuil.

Cette technique est simple et rapide à l'exécution. Elle a deux inconvénients qui la rendent peu fiable :

– elle ne prend pas en compte le voisinage spatial (les points voisins sont-ils occultés ?) ou temporel (le point était-il occulté à l'image précédente ?) de $\mathbf{m}$;
– elle ne peut pas détecter un objet de niveau de gris proche de ceux au voisinage du point, en particulier si ce voisinage est « trop » texturé.

4.2.3.2 Les cas NL et HB

Dans le cas des méthodes NL et HB, il suffit de retirer les points occultés du corpus des points de référence pour faire l'estimation du mouvement. Comme nous ne pouvons pas utiliser (homographie oblige) la factorisation de Hager et Belhumeur, cette restriction des points de référence ne dégrade pas les performances de l'estimation (au contraire puisque le nombre de points à prendre en compte est inférieur).

4.2.3.3 Le problème dans le cas JD (PJD)

Le traitement des occultations est plus problématique avec la méthode JD, car la matrice $\widehat{A}$ est de taille fixe. Enlever les colonnes correspondant aux points occultés revient à annuler les variations de niveaux de gris sur ces points : on retombe sur le problème de l'approche de Hager et Belhumeur.

♠ Il est nécessaire de prendre en compte les occultations pour remplacer la matrice d'apprentissage $\widehat{A}$ par $\widehat{A}' \in \mathbb{R}^{p\times n'}$ pour obtenir :

$$\widehat{\Delta'} = \widehat{A}'\Omega\widetilde{g}$$

Dans le cas où A est estimée avec OLS (nous ne traitons pas TLS et DLS), nous définissons $\widehat{A}'$ par une modification de l'équation (4.11) :

$$\widehat{A}' = \underset{A\in\mathbb{R}^{p\times n'}}{\operatorname{argmin}} \sum_{k=1}^{N} \parallel A\Omega\widetilde{g}_k - \Delta_k \parallel^2 = M(\Omega G)^+$$

Malheureusement, nous ne pouvons pas calculer $\widehat{A}'$:
– à partir de $\widehat{A}$ et Ω, parce que $\widehat{A}$ ne contient pas assez d'information ;
– pendant la phase d'apprentissage, parce que Ω n'est pas encore connue ;
– pendant la phase de suivi, à cause de la contrainte du temps réel.

4.2.3.4 Résolution de PJD : transformation du problème

Nous revenons à l'expression de $\widehat{A}'$. Puisque G est de rang plein, ΩG l'est aussi, d'où ([LH74] eq(7.24) p39) :

$$\widehat{A}' = MG^\top\Omega^\top(\Omega G G^\top\Omega^\top)^{-1}$$

Les matrices $M' = MG^\top$ et $G' = GG^\top$ peuvent être précalculées pendant la phase d'apprentissage. Ceci nous libère des calculs de complexité en $\mathcal{O}(N)$ pendant la phase de suivi. L'estimation de $\widehat{\Delta'}$ devient :

$$\widehat{\Delta'} = M'\Omega^\top(\Omega G'\Omega^\top)^{-1}\Omega\widetilde{g}$$

ce qui mène à l'algorithme[2] :

$y \leftarrow \Omega\widetilde{g}$ -- 0 flops (changement de forme)
résoudre $\Omega G'\Omega^\top x = y$ en x -- $\mathcal{O}(n'^3)$ flops
$\widehat{\Delta'} \leftarrow M'\Omega^\top x$ -- $2n'p$ flops

La seconde opération est la plus coûteuse de l'algorithme. Nous l'optimisons pour l'exécuter en temps réel, en exploitant les particularités de notre cas de figure. Nous nous fixons les ordres de grandeur suivants : $(n, n', p) = (400, 300, 8)$ et nous budgétisons 6 Mflops (10 ms sur un G4 866 MHz) pour faire ce travail.

4.2.3.5 Résolution de PJD : approche naïve

L'algorithme de référence, sans optimisation, s'écrit :

$H \leftarrow \Omega G'\Omega^\top$ -- 0 flops
$R \leftarrow$ décomposition de Cholesky de H ($H = R^\top R$) -- $n'^3/3$ flops
résoudre $R^\top x' = y$ en x' -- n'^2 flops
résoudre $Rx = x'$ en x -- n'^2 flops

La décomposition de Cholesky est le goulot d'étranglement : $n'^3/3 = 9$ Mflops. Comment l'optimiser ?

4.2.3.6 Résolution de PJD : actualisation négative (*downdating*)

La décomposition de Cholesky $G' = R'^\top R'$ peut être précalculée pendant la phase d'apprentissage. La matrice H s'obtient en supprimant des lignes et des colonnes à G'.

Principe. Les opérations d'actualisation servent à mettre à jour la décomposition d'une matrice quand des lignes ou des colonnes lui sont ajoutées (**actualisation positive**, *updating*) ou retirées (**actualisation négative**, *downdating*).

Approches classiques. Åke Björck ([Bjö96] § 3.2) étudie l'actualisation des décompositions QR et de Cholesky, qui sont liées : si $A = QR$ alors $A^\top A = R^\top R$. Il distingue les cas faciles (ajout d'une ligne et suppression d'une colonne de A) des cas difficiles (ajout d'une colonne et suppression d'une ligne).

Dans les problèmes de moindres carrés linéaires, le retrait d'une expérience nécessite de supprimer une ligne. Les résolutions OLS et TLS sont traitées séparément :

– cas OLS : il s'agit alors du downdating d'une décomposition QR. La routine `dchdd` [Sau72] de LINPACK concrétise des études vieilles de plus de 30 ans, et les recherches récentes portent sur les approches par blocs [EP94a] et leur stabilité [EP94b] ;

[2] Nous comptons une opération flottante (flop) par addition/soustraction et une par multiplication. Les divisions et les extractions de racines carrées, quoique plus coûteuses, sont en nombre négligeable par rapport aux opérations simples. Nous indiquons seulement le terme dominant du nombre d'opérations flottantes. Nous faisons abstraction des accès mémoire.

– cas TLS : l'actualisation concerne une décomposition SVD. Celle-ci étant coûteuse [PH95], l'intérêt s'est reporté sur les décompositions URV et ULV « révélatrices de rang ». Jesse Barlow traite leur actualisation dans [BYZ96] et [BY97]. Selon lui, l'actualisation par blocs dans ce cas TLS est un sujet de recherche vierge. La bibliothèque MATLAB UTV TOOLS, consacrée à ces factorisations, propose des routines de *up(down)dating* ([FHH90] § 4).

Tous les algorithmes d'actualisation ont au mieux une complexité moyenne en $\mathcal{O}(n^2)$ pour la suppression d'**une seule** ligne/colonne.

Dans notre cas. Selon la classification de Björck, notre cas est favorable : il consiste à actualiser une décomposition de Cholesky en supprimant plusieurs colonnes de $A = G^\top$. En général ces colonnes ne sont pas consécutives. En conséquence, le traitement d'un bloc est exceptionnel : nous ne présentons ici que le cas du retrait d'une colonne.

Si $T \in \mathbb{R}^{n\times n}$ est une matrice orthogonale, on peut écrire :

$$\begin{aligned} H &= \Omega G' \Omega^\top \\ R^\top R &= \Omega R'^\top R' \Omega^\top \\ &= \Omega R'^\top T^\top T R' \Omega^\top \\ &= (T R' \Omega^\top)^\top T R' \Omega^\top \end{aligned}$$

Nous choisissons T de manière à rendre $S = TR'\Omega^\top$ triangulaire supérieure, ce qui peut se faire par une séquence de rotations de Givens. Par unicité de la décomposition de Cholesky, R est alors le triangle supérieur de S.

Implémentation. Nous ne manipulons pas T explicitement, nous appliquons seulement les rotations de Givens à S ([GL91] eq 5.1.8 et 5.1.9). Pour des raisons d'efficacité, pendant l'opération, nous représentons la matrice à traiter sous la forme d'un produit $D^{-1/2}S$, où $D = \mathrm{diag}(d) \in \mathbb{R}^{n\times n}$ est modifiée en même temps que S (algorithme 4.2).

Complexité. La complexité de l'algorithme d'actualisation dépend de la répartition des points occultés :

$$C = 2n'^2 + 4\sum_{i=1}^{n'}(p_i - i)(n' - i)$$

Voici quelques cas de figure de répartitions :

– cas meilleur : tous les points occultés sont à la fin de la liste ($p_i = i$) :

$$C = 2n'^2$$

– cas pire : tous les points occultés sont au début de la liste ($p_i = i + n - n'$) :

$$C = 2(n - n')n'(n' - 1) + 2n'^2$$

$S \leftarrow R'\Omega^\top$ -- S est R' privé des colonnes occultées
$d \leftarrow [1 \ \cdots \ 1]^\top \in \mathbb{R}^n$ -- $D = \mathrm{Id}_n$
Pour i **de** 1 **à** n' **faire**
  Pour k **de** $i+1$ **à** p_i **faire**
    `fast_givens`$(d_i, S_{i,i:n'}, d_k, S_{k,i:n'})$ -- coût : $4(n'-i)$ flops
  finpour -- $4(p_i - i)(n'-i)$ flops
finpour -- le triangle supérieur de $D^{-1/2}S = R$
$S' \leftarrow S_{1:n',1:n'}$
$D' \leftarrow \mathrm{diag}(d_{1:n'})$ -- ainsi, $H = R^\top R = S'^\top D'^{-1} S'$
résoudre $S'^\top x' = y$ en x' -- n'^2 flops
résoudre $S'x = Dx'$ en x -- $n'^2 + n'$ flops
La décomposition résultante est $\mathrm{diag}(d)^{-1/2}S$.

Fonction utilisée : la fonction `fast_givens`(d_1, l_1, d_2, l_2) effectue une rotation (ou une réflexion) entre les lignes $d_1^{-1/2}l_1$ et $d_2^{-1/2}l_2$, de manière à annuler l_{21}. En convenant que toutes les variables sont en entrée et en sortie, ceci s'écrit :
$(x_1, x_2) \leftarrow (l_{11}, l_{21})$ -- les éléments qui déterminent des facteurs de rotation
Si $x_2 \neq 0$ **alors** -- sinon, rien à faire
  Si $x_1^2 d_2 \leq x_2^2 d_1$ **alors** -- la deuxième ligne est « dominante »
    $\begin{bmatrix} l_1 \\ l_2 \end{bmatrix} \leftarrow \begin{bmatrix} \frac{d_2 x_1}{d_1 x_2} & 1 \\ 1 & -\frac{x_1}{x_2} \end{bmatrix} \begin{bmatrix} l_1 \\ l_2 \end{bmatrix}$ -- réflexion
    $(d_1, d_2) \leftarrow (1 + \frac{d_2 x_1^2}{d_1 x_2^2})(d_2, d_1)$
  sinon -- la première ligne est « dominante »
    $\begin{bmatrix} l_1 \\ l_2 \end{bmatrix} \leftarrow \begin{bmatrix} 1 & \frac{d_1 x_2}{d_2 x_1} \\ -\frac{x_2}{x_1} & 1 \end{bmatrix} \begin{bmatrix} l_1 \\ l_2 \end{bmatrix}$ -- rotation
    $(d_1, d_2) \leftarrow (1 + \frac{d_1 x_2^2}{d_2 x_1^2})(d_1, d_2)$
  finsi
finsi

Algorithme 4.2: Résolution de $\Omega G' \Omega^\top x = y$ en x par actualisation négative (*downdating*) de la décomposition de Cholesky R' de G'.

– cas moyen : les points occultés sont répartis régulièrement ($p_i = \lfloor in/n' \rfloor$) :

$$\begin{aligned} C &= 4\sum_{i=1}^{n'} \left(\left\lfloor \frac{in}{n'} \right\rfloor - i \right) (n' - i) + 2n'^2 \approx 4\sum_{i=1}^{n'} \left(\frac{in}{n'} - i \right) (n' - i) \\ &\approx \frac{2}{3}(n - n')(n' + 1)(n' + 2) + 2n'^2 \end{aligned}$$

Si le rapport $\alpha = n'/n$ est constant, l'algorithme a une complexité en $\mathcal{O}(n^3)$. Cependant, le coefficient du n^3 est faible : $\frac{2}{3}(1 - \alpha)\alpha^2$, ce qui suggère que, pour les valeurs de n considérées, il peut être intéressant. Si nous considérons le cas moyen avec les chiffres mentionnés, $C \approx 6$ Mflop.

4.2.3.7 Résolution de PJD : approche itérative

Le modèle à l'origine de l'équation $\Omega G' \Omega^\top x = y$ est une approximation (celle de la phase d'apprentissage de JD), ce qui implique que nous pouvons nous contenter d'une approximation de la solution en x (une erreur relative de $\pm 0.1\%$ sur les éléments du vecteur est acceptable). Ceci suggère d'utiliser une résolution itérative.

Méthodes itératives classiques. Nous avons appliqué les méthodes itératives de Jacobi/Gauss-Seidel et du gradient conjugué ([GL91] eq10.1.3 et alg10.2.1). Comme la matrice $H = \Omega G' \Omega^\top$ du système est pleine, le coût de ces algorithmes est déterminé par celui d'un produit matrice-vecteur avec H (il y a un produit par itération qui coûte $2n'^2$ flops).

Dans notre cas, ces méthodes nécessitent de l'ordre de 60 itérations pour converger vers des valeurs utilisables, ce qui correspond à 10.8 Mflops. Ce n'est pas compétitif ; nous devons exploiter les particularités du problème.

Amélioration. Typiquement, on résout un problème linéaire de matrice A à l'aide d'une méthode itérative dans les cas suivants :

– la matrice A est creuse et sa décomposition la rendrait pleine. Comme une solution itérative repose sur des produits avec A, on peut exploiter le caractère creux ;
– on dispose d'une bonne approximation de la solution et quelques itérations permettront de converger ;
– on dispose d'une bonne approximation de A^{-1}. C'est le cas le plus favorable.

Nous faisons « comme si » nous étions dans ce troisième cas car :

– nous pouvons calculer G'^{-1} ;
– nous postulons que $M^{-1} = \Omega G'^{-1} \Omega^\top$ est une approximation utilisable de $(\Omega G' \Omega^\top)^{-1}$.

♠ Nous l'utilisons dans un « **préconditionneur** » pour un algorithme de gradient conjugué ([GL91] algo10.3.1). L'algorithme 4.3 découle de toutes ces considérations.

L'algorithme converge à la précision machine en 15 itérations environ (une itération coûte $4n'^2$ flops). Nous disposons d'un résultat suffisamment précis en 5 itérations, c'est-à-dire en 1.8 Mflop, ce qui est un coût acceptable.

$k \leftarrow 0$; $x \leftarrow 0_{n'}$; $r \leftarrow y$
Tant que $r \neq 0_{n'}$ et $k < k_{\max}$ **faire**
  $z \leftarrow \Omega G'^{-1} \Omega^\top$ -- application du préconditionneur ($2n'$ flops)
  Si $k = 0$ **alors** -- calcul de la direction du prochain pas
    $u \leftarrow \|z\|^2$
    $p \leftarrow z$
  sinon
    $u \leftarrow r^\top z$
    $p \leftarrow z + \frac{u}{u_{\text{prec}}} p$
  finsi
  $a \leftarrow \Omega G' \Omega^\top p$ -- $2n'$ flops
  $\alpha \leftarrow \frac{u}{p^\top a}$
  $x \leftarrow x + \alpha p$ -- mise-à-jour de l'estimation
  $r \leftarrow r - \alpha a$ -- mise-à-jour des résidus
  $k \leftarrow k + 1$; $u_{\text{prec}} \leftarrow u$
fintantque

Algorithme 4.3: Résolution de $\Omega G' \Omega^\top x = y$ en x par une méthode itérative ($k_{\max}$ itérations) de gradient conjugué préconditionné. L'inverse de G' est précalculée.

4.2.3.8 Enchaînement de traitements

Nous pouvons détecter des occultations à deux moments : à la fin de la chaîne de traitements pour l'image courante (figure 4.11, chaîne 1) ou après chaque étape d'estimation (chaîne 2).

A priori, nous ne savons pas quelle chaîne donne les meilleurs résultats. La chaîne 2 tient compte de l'estimation la plus à jour de θ_t, mais si l'estimation est trop imprécise, beaucoup de points pertinents sont considérés occultés.

4.2.4 Expérimentation

4.2.4.1 Le protocole expérimental

L'évaluation d'une approche de traitement des occultations se fait selon le protocole suivant :

- nous employons la séquence « déformation » (§ 4.1.5.1) ;
- nous ajoutons sur chaque image un carré noir. Le carré, de taille $d \times d$, est centré sur un point fixe de l'image (figure 4.12).
- nous effectuons le suivi et mesurons t^*.

4.2.4.2 Les trois traitements

D'abord nous fournissons la « vérité terrain » pour le masque. Nous comparons trois approches des occultations (figure 4.13) :

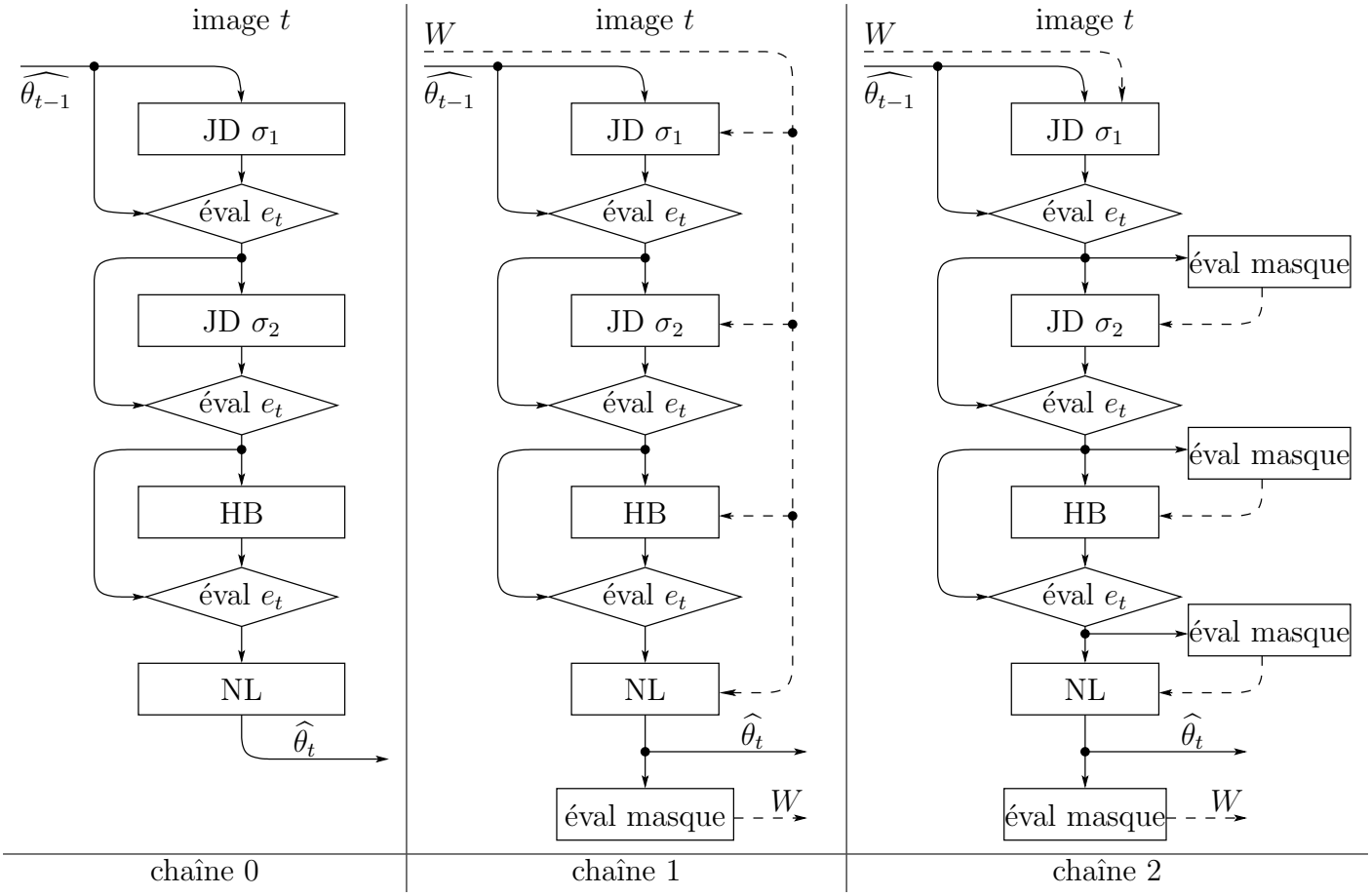


FIG. 4.11: Propositions de chaînes de traitements (avec les notations de la figure 4.3). Les unités de suivi peuvent prendre en compte un masque d'occultation (§ 4.2.3.2 et 4.2.3.3). Les masques d'occultation empruntent les chemins fléchés en pointillés. L'opérateur « éval masque » calcule un masque (§ 4.2.3.1) à partir d'un vecteur de paramètres θ. La chaîne 0 est celle de la figure 4.3. Les chaînes 1 et 2 traitent les occultations.

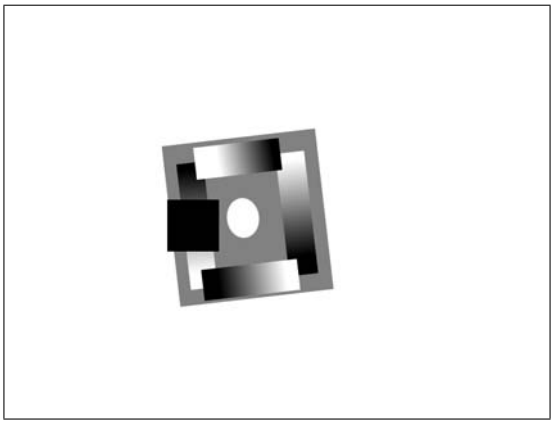


FIG. 4.12: Image de synthèse avec occultation ($d = 100$)

– aucun traitement (c'est la référence). Elle correspond à la chaîne 0 de la figure 4.11 ;
– le traitement de Hager et Belhumeur (forcer à 0 la différence de niveaux de gris sous le masque d'occutation § 4.2.1.5) intégré dans la chaîne 2 ;
– notre approche (supprimer les points occultés dans l'estimation § 4.2.3.3) intégrée à la chaîne 2.

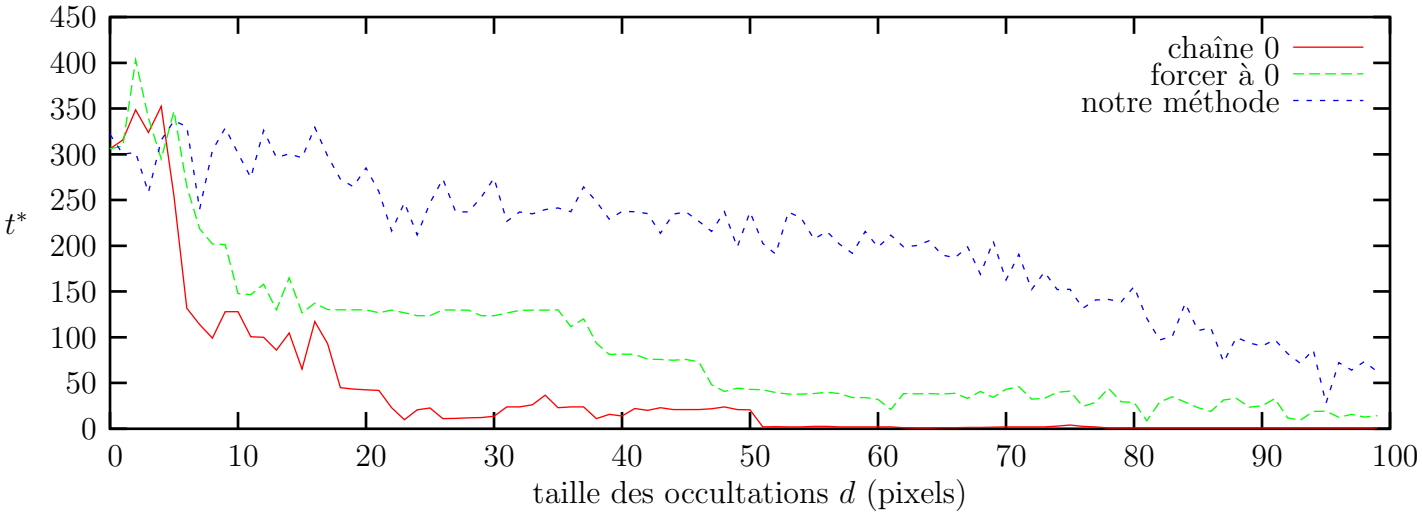


FIG. 4.13: Traitement des occultations en supposant connu le masque. La taille de la cible 2D est environ 250×250 pixels.

Les trois méthodes sont équivalentes pour des petites occultations ($d \leq 5$ pixels).

La chaîne 0 devient rapidement inefficace quand elles deviennent significatives ($d > 20$ pixels).

Le principe de traitement de Hager et Belhumeur montre ses limites comparé à notre solution.

4.2.4.3 Quand évaluer les occultations ?

Nous comparons les chaînes 1 et 2 (figure 4.14), sans utiliser de vérité terrain dans la détection du masque. La chaîne 0 sert de référence.

Le graphe montre que pour une occultation de petite taille, il est préférable de ne pas faire de traitement, probablement parce que des points mal localisés sont classés cachés. Quand les occultations deviennent assez grandes, c'est la chaîne 1 qui « s'en sort le moins mal ».

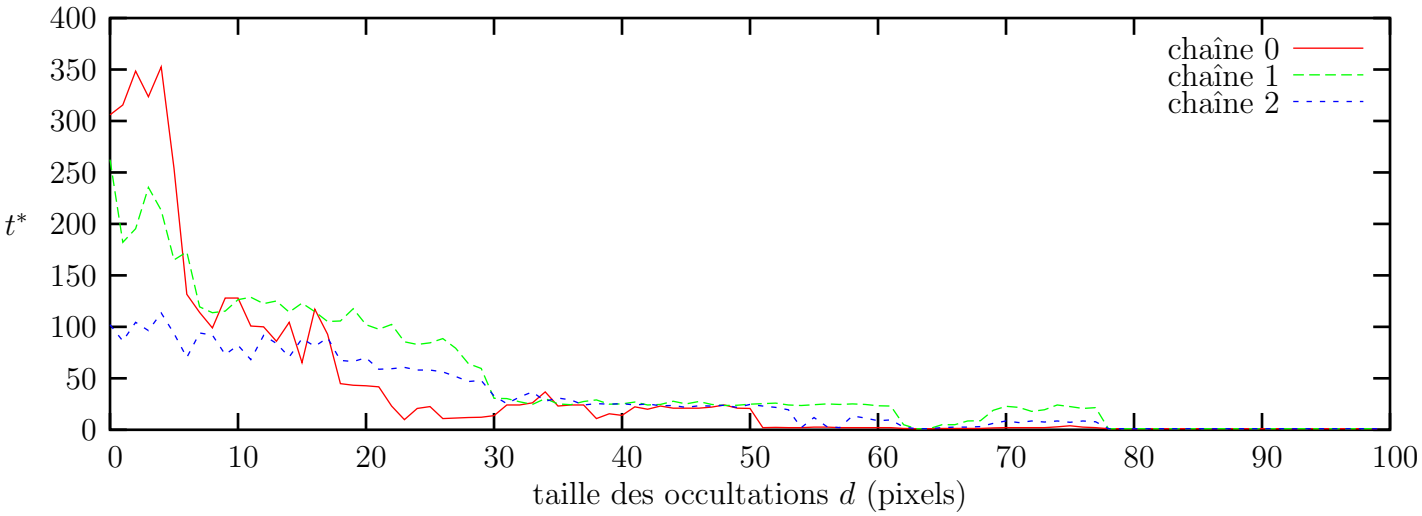


FIG. 4.14: Comparaison des chaînes de traitements.

4.2.4.4 Vitesse du traitement dans le cas JD

Pour choisir la méthode de résolution du système $\Omega G' \Omega^\top x = y$ dans le cas JD (§ 4.2.3.3), nous avons comparé les vitesses d'exécution pour les différentes méthodes en faisant varier le nombre de points visibles (figure 4.15).

On voit que la solution naïve (décomposition) est nettement plus lente. La méthode itérative (avec 5 itérations) est la plus rapide, sauf quand les points occultés sont peu nombreux (moins de 80) où l'actualisation négative devient intéressante. L'actualisation profite d'un schéma d'accès à la mémoire plus efficace que la méthode itérative.

4.2.5 Conclusion

Dans l'ensemble, le traitement des occultations par cette méthode est tout de même très coûteux en temps de calcul ; nous perdons un des avantages de la méthode JD : son

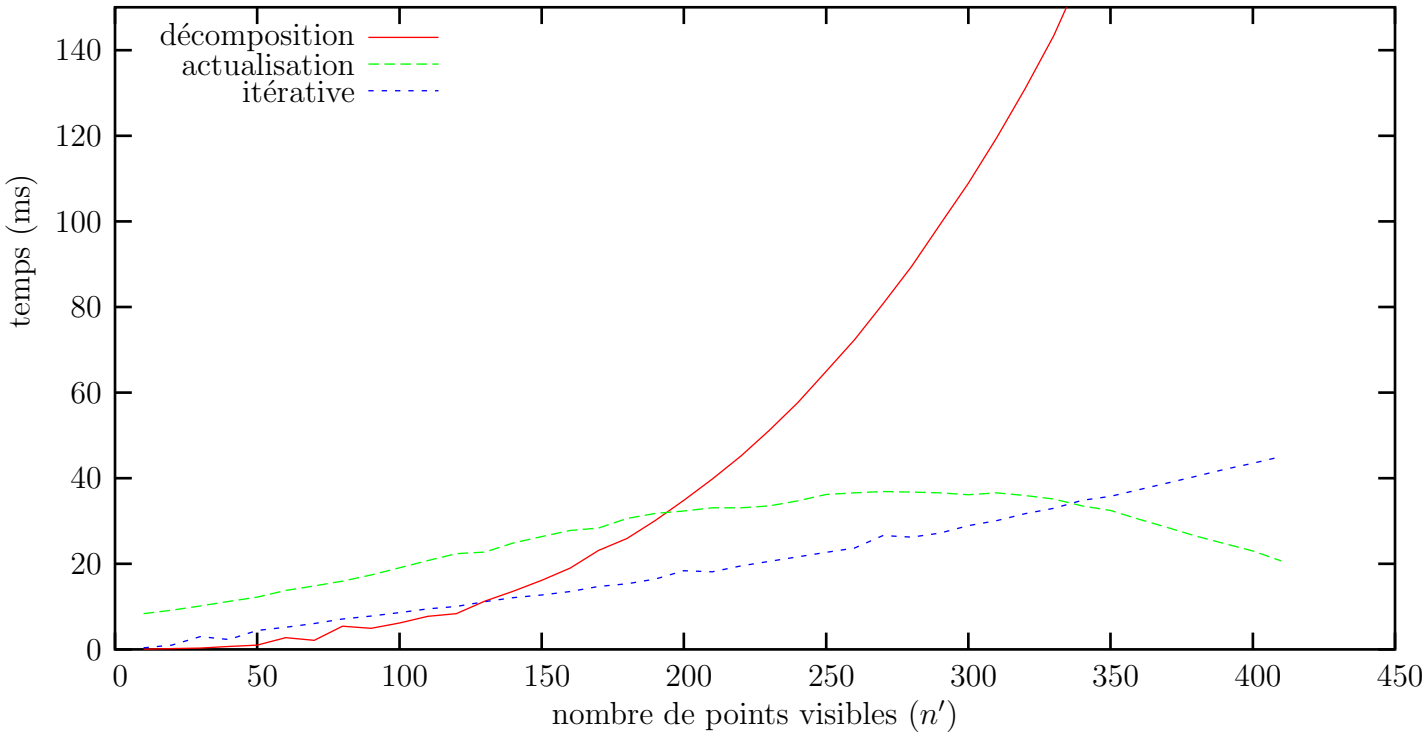


FIG. 4.15: Coût d'une étape de suivi JD en fonction du nombre de points visibles (n' sur $n = 425$). Les trois méthodes comparées sont la décomposition complète (§ 4.2.3.5), l'actualisation négative (§ 4.2.3.6) et la méthode itérative basée sur le gradient conjugué préconditionné (§ 4.2.3.7).

efficacité. Une exécution en temps réel (moins de 20 ms pour toute la chaîne de traitements) nécessite de faire des compromis : utiliser une machine plus rapide et/ou moins de points de référence.

Chapitre 5

Applications

Dans ce chapitre, nous présentons deux techniques supplémentaires qui participent à la fois de la création d'images panoramiques et du suivi robuste : le **suivi dans une image panoramique** et la **segmentation par « soustraction pixel à pixel »**. Ce type de segmentations donne lieu à deux applications :

- le codage vidéo ;
- la réalité augmentée.

5.1 Suivi dans une image panoramique

Nous nous plaçons dans le contexte évoqué dans l'introduction (§ 1.2.2 et figure 1.1).

Le protocole.

- Nous disposons d'une ou plusieurs vues panoramiques d'une scène (figure 5.1), ramenées à des projections centrales planes de centres optiques communs (§ 3.5.3) ;
- la scène et l'éclairage sont figés, mais des objets et/ou des personnages s'y ajoutent ;
- une caméra filme cette scène. Les mouvements admissibles du capteur sont ceux décrits au § 3.4.2.1 : elle peut changer de direction de visée et de distance focale. Si la caméra est montée sur un pied, le nombre de degrés de liberté peut être limité à 3 (deux angles de rotation –inclinaison et azimut– et un facteur de zoom).

Le but. Il s'agit de mettre en correspondance les images de la séquence avec l'image panoramique de référence. Cette tâche est un problème de suivi (deux images successives sont « proches ») à base d'images géométriquement liées par des homographies 2D. ♠

5.1.1 Modélisation

5.1.1.1 Relations avec le suivi d'un motif plan

En considérant que **l'image panoramique en entier comme la cible 2D**, nous retrouvons le contexte du suivi d'un motif plan sans restriction aucune. Nous traitons le

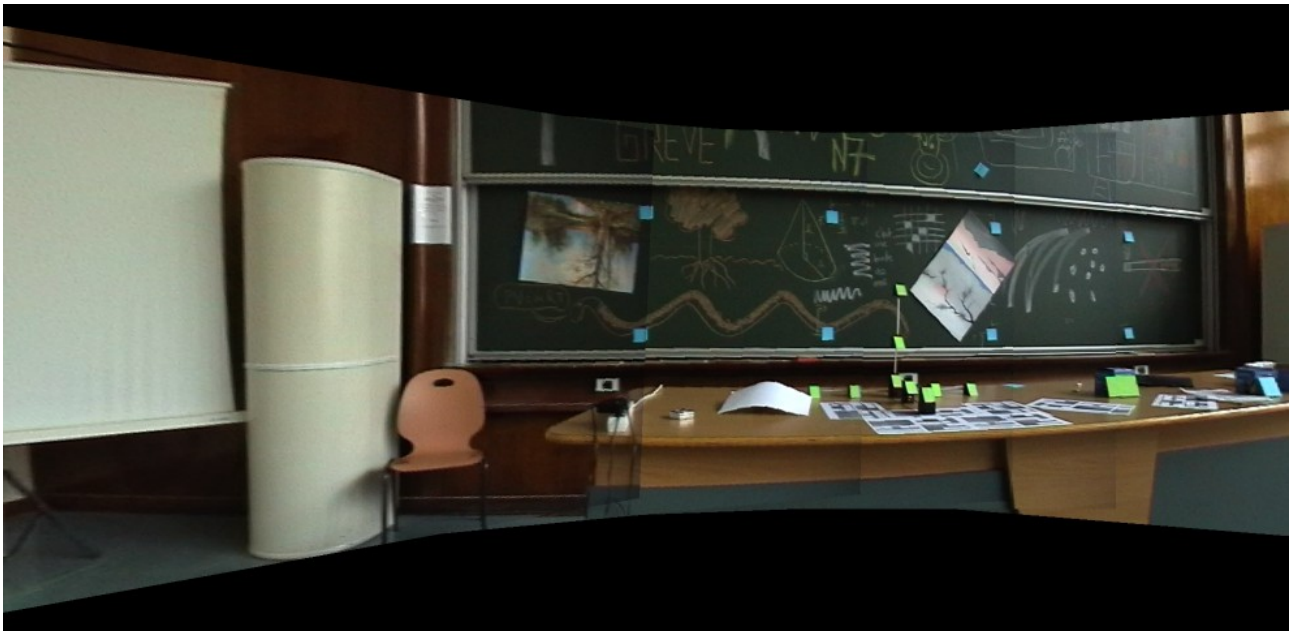


FIG. 5.1: Une image panoramique de référence.

cas d'une image panoramique unique I_{p}.

La transformation initiale θ_0 entre I_{p} et l'image I_0 est supposée connue. Ce n'est pas l'identité. Elle peut être obtenue :

- manuellement. Avant de faire bouger la caméra, l'utilisateur situe son image par rapport à la vue de référence ;
- automatiquement. Le contexte est parallèle à celui du calcul de mosaïques et plus généralement de l'indexation d'images [MS01].

À la différence du suivi d'un motif plan, l'image panoramique n'est jamais entièrement dans le champ visuel de la caméra : seule une fraction des points de référence de la cible 2D se retrouve sur les images de la séquence. Cette fraction est trop faible pour y appliquer les traitements développés pour les occultations (§ 4.2).

5.1.1.2 Les tuiles

Nous appuyons le suivi sur **plusieurs** cibles 2D (des **tuiles**) qui recouvrent la vue de référence. Cette approche par tuiles indépendantes est une solution classique à la contrainte de fixité des points de référence dans la méthode JD ([NLJD02], [MDJ04]).

Nous utilisons des tuiles (figure 5.2) :

- rectangulaires, disposées en quinconce ;
- assez petites pour que plusieurs d'entre elles soient visibles sur chaque image de la séquence ;
- assez grandes pour préserver la pertinence de l'estimation.

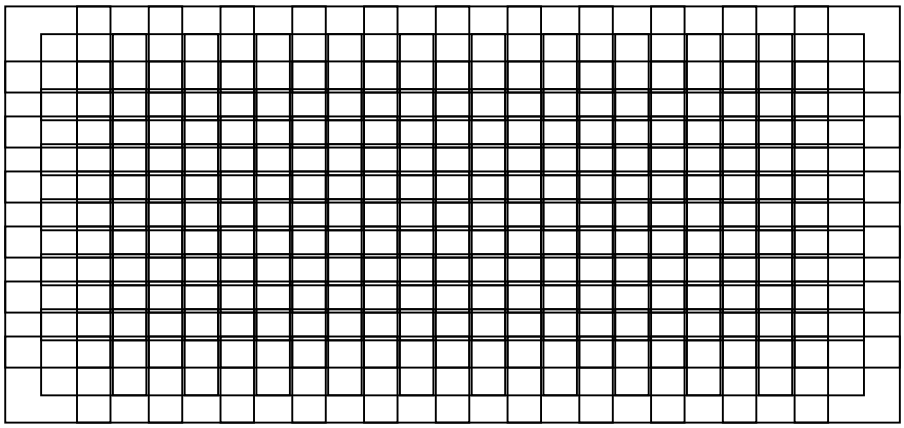


FIG. 5.2: Disposition des tuiles couvrant l'image panoramique (figure 5.1).

5.1.2 Résolution

Pour chaque tuile, nous choisissons un ensemble de points de référence et nous exécutons une phase d'apprentissage.

5.1.2.1 Combinaison des tuiles

Pendant la phase de suivi nous traitons l'image I_t. Nous utilisons $\widehat{\theta_{t-1}}$ pour détecter les tuiles entièrement visibles sur I_{t-1}. Nous estimons θ_t pour chacune de ces tuiles. Ensuite, nous sélectionnons (figure 5.3) les r tuiles qui :

- sont visibles. Seule une marge autour de la tuile peut sortir du champ ;
- ont peu de points occultés ;
- donnent des résultats de suivi raisonnables. Nous utilisons pour cela le critère d'estimation. Dans le cas JD, nous vérifions en plus la compatibilité de la perturbation avec celle appliquée lors de la phase d'apprentissage (§ 4.1.4.4).

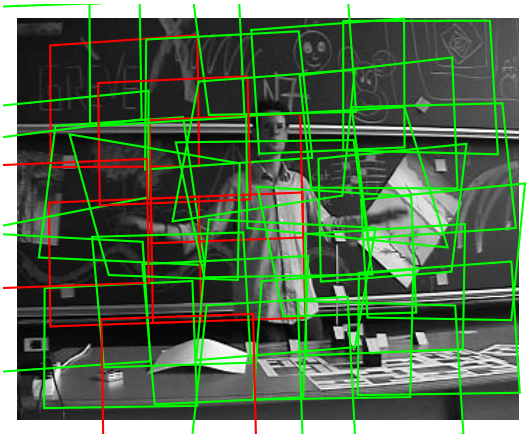


FIG. 5.3: Tuiles utilisées pendant le suivi sur une image. En vert, celles utilisées pendant la combinaison des paramètres θ. En rouge, les rejetées).

Pour combiner les résultats du suivi sur chaque tuile, nous utilisons ses 4 sommets. Avec r tuiles, ceci donne $4r$ correspondances de points et $8r$ équations en θ_t. Nous résolvons ces équations au sens des moindres carrés : les aberrations sont supposées rejetées par l'étape de sélection des tuiles.

5.1.2.2 Chaînes d'estimation

La combinaison des tuiles peut être faite à plusieurs moments :
- à la fin des étapes. Le suivi est fait indépendamment sur chaque tuile et les résultats sont combinés à la fin des traitements (figure 5.4, chaîne 1) ;
- après chaque étape. Les résultats du suivi pour chaque tuile sont combinés dès qu'une nouvelle estimation est disponible (figure 5.4, chaîne 2).

De la même manière que dans le traitement des occultations (§ 4.2.3.8), la meilleure chaîne est difficile à déterminer. La chaîne 2 permet de rattraper des erreurs d'estimation dans une étape en cas de tuile occultée. Cependant, si une tuile est occultée, la divergence peut n'être évidente qu'après plusieurs étapes. Or, la chaîne 2 prend en compte cette tuile qui contamine alors les autres résultats.

5.1.3 Expérimentation

5.1.3.1 Données de synthèse.

Nous avons fait des expériences de suivi sur des images de synthèse. Elles représentent des boules avec une texture en dégradé sur fond noir (figure 5.5). Les orientations successives de la caméra décrivent un mouvement en spirale, de plus en plus rapide.

Évaluation du critère d'optimisation. Nous avons vérifié l'adéquation du critère d'optimisation (disponible pendant l'exécution de l'algorithme) avec le critère basé sur la vérité terrain (§ 4.1.5.1).

La figure 5.6 montre que ces deux critère sont cohérents entre eux. Le critère d'optimisation est donc pertinent pour évaluer les occultations.

Taille des tuiles. La taille optimale pour les tuiles pour cette séquence est 130×120 pixels. Le vecteur séparant deux tuiles a pour coordonnées $(75, 70)$. Les tuiles se recouvrent donc fortement.

5.1.3.2 Séquence réelle

Les expériences sur des images réelles portent sur des scènes d'intérieur (l'environnement est bien contrôlé). La correspondance initiale θ_0 entre I_0 et l'image panoramique I_p est fixée à la main.

La figure 5.7 présente une séquence suivie à l'aide de notre algorithme.

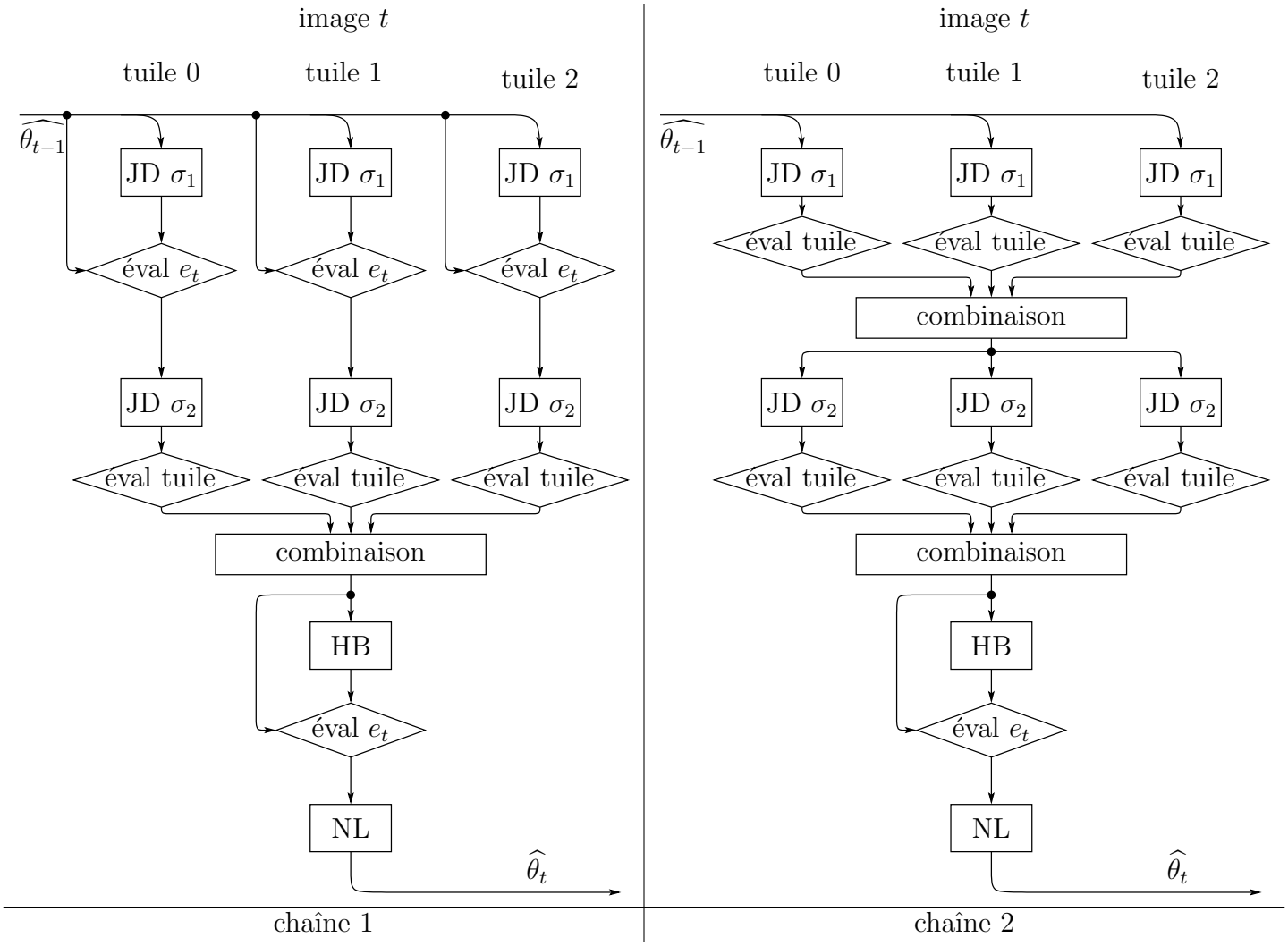


FIG. 5.4: Chaînes d'estimation des paramètres θ pour trois tuiles, avec les notations de la figure 4.11. L'opérateur « éval tuile » calcule le critère et teste les conditions de sélection de la tuile (§ 5.1.2.1). En cas de sélection, la tuile est utilisée par l'opération « combinaison » qui synthétise les paramètres de l'homographie. Sinon, la tuile n'est pas utilisée.

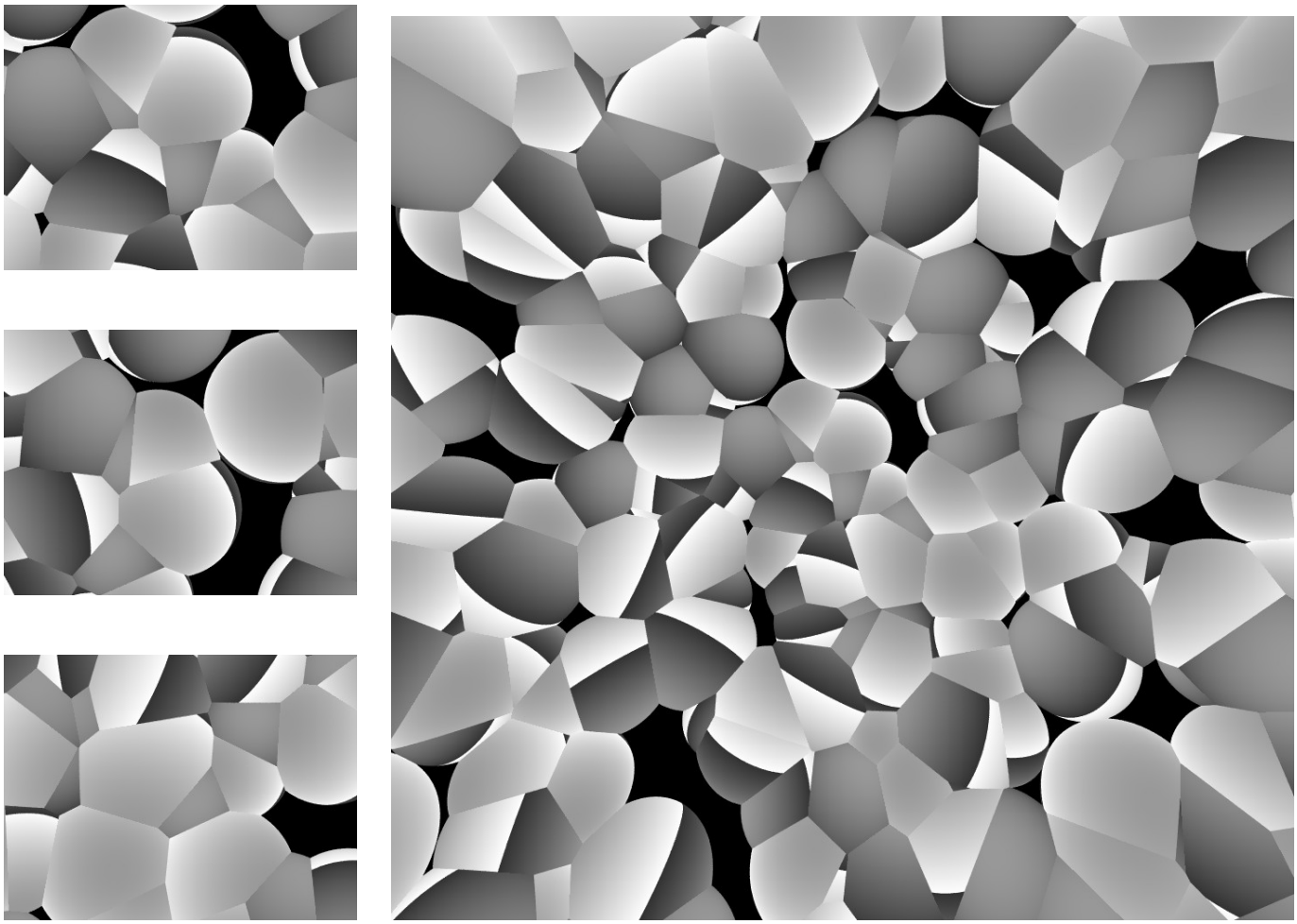


FIG. 5.5: Images de synthèse utilisées pour le suivi dans des images panoramiques : les images 0, 50 et 100 de la séquence (de taille 400×300 pixels et d'angle de vue horizontal 30°) et l'image panoramique (taille 1024×1024, angle de vue horizontal 90°).

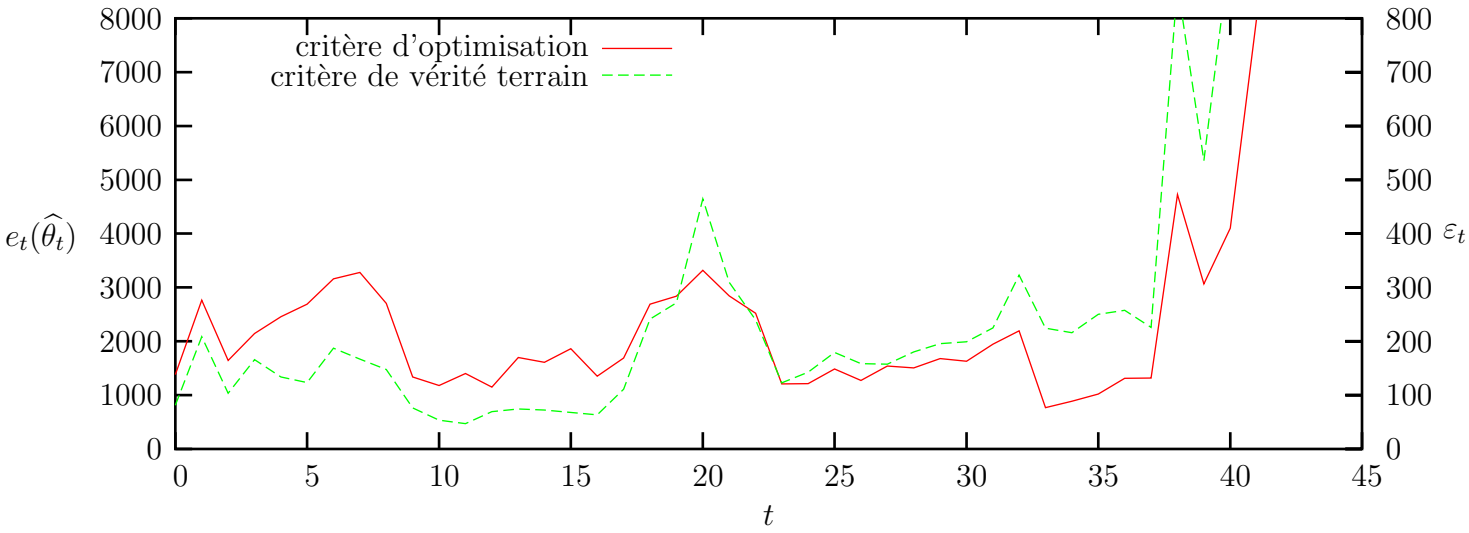


FIG. 5.6: Évolution du critère d'optimisation $e_t(\widehat{\theta}_t)$ et de la vérité terrain ε_t au cours d'une séquence vidéo synthétique.

FIG. 5.7: Image panoramique de référence et images d'une séquence vidéo suivie avec notre méthode.

5.2 Détection d'« intrus »

La segmentation a pour but d'extraire les zones occultantes de l'image I_t en cours de traitement. Elle fournit un masque dense superposable à I_t, c'est-à-dire une image binaire qui est à 1 pour les parties non occultées et à 0 pour les autres, censées représenter des objets ou des personnages intrus.

Dans cette partie, l'image panoramique de référence I_p et l'image I_t sont en couleurs ; ce sont des fonctions

$$I_\mathrm{p}, I_t : \mathbb{R}^2 \to \mathbb{R}^3$$

D'après l'équation fondamentale du mouvement (4.2), pour tout point $\mathbf{m}$ non occulté sur l'image courante,

$$I_\mathrm{p}(T(\mathrm{inv}(\theta_t), \mathbf{m})) = I_t(\mathbf{m}) \tag{5.1}$$

La détection des points occultés exploite cette équation en opérant une « soustraction » entre les membres de l'égalité (de manière analogue au § 4.2.3.1).

Cette « soustraction » ne peut avoir lieu qu'après un alignement géométrique (ou compensation de mouvement) et photométrique (ou chromatique).

5.2.1 Alignement géométrique

Pour obtenir un masque dense superposable à I_t, nous re-projetons I_p sur I_t de manière à amener les points homologues en coïncidence.

L'image panoramique re-projetée sur l'image courante satisfait à :

$$\forall \mathbf{m}, \qquad I_{\mathrm{p,g},t}(\mathbf{m}) = I_\mathrm{p}(T(\mathrm{inv}(\theta_t), \mathbf{m})) \tag{5.2}$$

En pratique, l'algorithme de re-projection des images discrètes est indiqué à l'appendice § 7.2.3.

5.2.2 Alignement photométrique

L'équation (5.1) n'est pas vérifiée exactement si les capteurs utilisés pour l'image panoramique et la vidéo ont des caractéristiques photométriques différentes.

L'alignement photométrique consiste à modifier les couleurs des points de I_t pour les faire coïncider avec celles des points homologues de I_p.

5.2.2.1 Modèle d'alignement

Nous appliquons une transformation affine aux trois couleurs primaires de I_t pour obtenir $I_{t,\mathbf{p}}$:

$$\forall \mathbf{m}, \qquad I_{t,\mathrm{p}}(\mathbf{m}) = A_\mathrm{p} I_t(\mathbf{m}) + b_\mathrm{p}$$

Les paramètres de l'alignement sont $A_\mathrm{p} \in \mathbb{R}^{3\times 3}$ et $b_\mathrm{p} \in \mathbb{R}^3$.

Ce modèle affine est relativement grossier. Les modèles plus fins sont basés sur une « correction gamma » ([Nye92] 7.2.1).

5.2.2.2 Étalonnage photométrique

Nous estimons les paramètres $A_\mathrm{p} \in \mathbb{R}^{3\times3}$ et $b_\mathrm{p} \in \mathbb{R}^3$ lors d'un étalonnage photométrique. ♠
Il est réalisé sur les images $I_{\mathrm{p,g,0}}$ et I_0, supposée sans occultation.

Le problème d'estimation. Conformement au cadre du § 2.2 :
- chaque point $(\mathbf{m}_i)_{i=1..N}$ de l'image I_0 représente une « expérience » ;
- les entrées sont les couleurs sur I_0 : $x = I_0(\mathbf{m})$;
- les sorties sont les couleurs sur $I_{\mathrm{p,g,0}}$: $y = I_{\mathrm{p,g,0}}(\mathbf{m})$;
- les paramètres du modèle sont $A_\mathrm{p} \in \mathbb{R}^{3\times3}$ et $b_\mathrm{p} \in \mathbb{R}^3$ ($p = 12$).

Ce qui mène à la « boîte noire » suivante :

$$x \in \mathbb{R}^3 \rightarrow \boxed{\begin{array}{c} A_\mathrm{p} \in \mathbb{R}^{3\times3}, b_\mathrm{p} \in \mathbb{R}^3 \\ \hline m = 3; n = N; p = 12; q = 3 \end{array}} \rightarrow y = A_\mathrm{p}x + b_\mathrm{p} \in \mathbb{R}^3$$

Les erreurs. Les erreurs dont il faut tenir compte sont :
- la saturation des couleurs, qui se produit quand la caméra est pointée en direction de la lumière. Le point concerné apparaît blanc, sans nuance. Nous traitons ce type d'erreur en supprimant les expériences dont la mesure en entrée ou en sortie correspond à la couleur blanche ;
- les mesures aberrantes sur les pixels mal superposés. Le risque est fort dans les régions de gradient élevé, où une petite erreur de mise en correspondance (qui peut être inférieure au pixel) introduit une forte déviation des couleurs ;
- le bruit photométrique sur les mesures de couleur. Nous le modélisons comme un bruit gaussien sur les entrées et les sorties.

Pour traiter ce dernier type d'erreurs, nous nous plaçons dans le cadre ODR :

$$P_\mathrm{p} \left\{ \begin{array}{l} \min \sum_{i=1}^{N} \|\delta_i\|^2 + \|\varepsilon_i\|^2 \\ A_\mathrm{p} \in \mathbb{R}^{3\times3}, b_\mathrm{p} \in \mathbb{R}^3, \forall i = 1..N, \delta_i \in \mathbb{R}^3, \varepsilon_i \in \mathbb{R}^3 \\ A_\mathrm{p}(\widetilde{x}_i - \delta_i) = \widetilde{y}_i - \varepsilon_i \end{array} \right.$$

Résolution. En utilisant les valeurs exactes correspondant aux mesures, la contrainte d'égalité du problème (P_a) peut se mettre sous la forme :

$$\underbrace{\begin{bmatrix} A_\mathrm{p} & b_\mathrm{p} \end{bmatrix}}_{X^\top} \underbrace{\begin{bmatrix} x_1 & \cdots & x_N \\ 1 & \cdots & 1 \end{bmatrix}}_{A^\top} = \underbrace{\begin{bmatrix} y_1 & \cdots & y_N \end{bmatrix}}_{B^\top}$$

qui nous ramène à l'équation classique « $AX = B$ ». Les erreurs sont sur la matrice B en entier et sur les trois premières colonnes de A. C'est donc un problème TLS structuré.

C'est un cas favorable car sa structure révèle un problème LS/TLS mixte pour lequel existe une solution analytique ([SVH02] 3.6.3 p92).

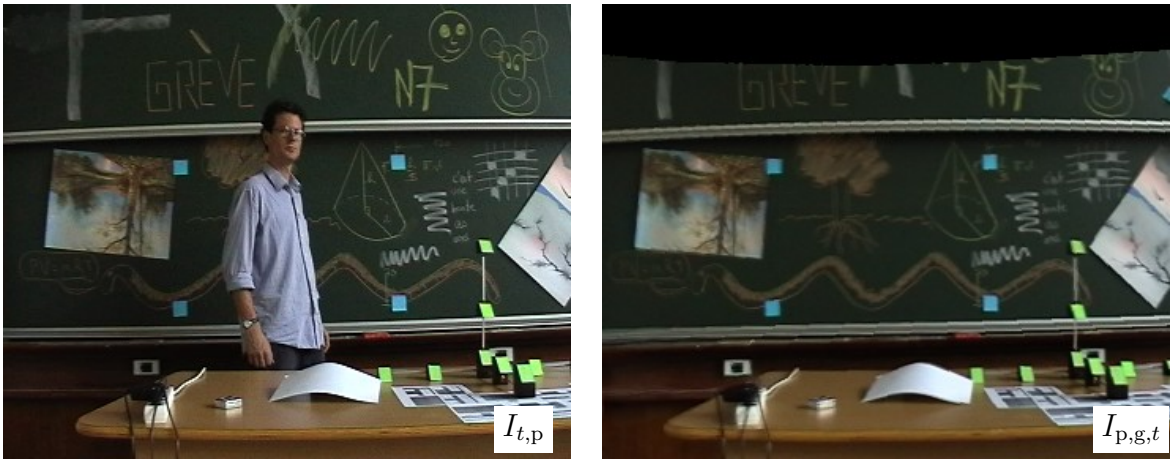


FIG. 5.8: Image de la séquence vidéo après alignement photométrique et image panoramique de référence (figure 5.1) après alignement géométrique.

5.2.3 « Soustraction » de l'arrière-plan

En l'absence d'occultation, l'équation (5.1) se traduit par $I_{t,\mathrm{p}} = I_{\mathrm{p,g},t}$.

Dans le cas contraire (figure 5.8), la différence $I_{t,\mathrm{p}} - I_{\mathrm{p,g},t}$ révèle les objets occultants : ils sont aux points où elle est non négligeable.

En pratique (algorithme 5.1) :

- nous supposons les zones occultantes « contrastées » par rapport aux occultées ;
- nous lissons $I_{t,\mathrm{p}} = I_{\mathrm{p,g},t}$ à l'aide d'un opérateur d'ouverture morphologique pour résister aux imprécisions photométrique (échantillonnage discret) et géométrique (suivi).

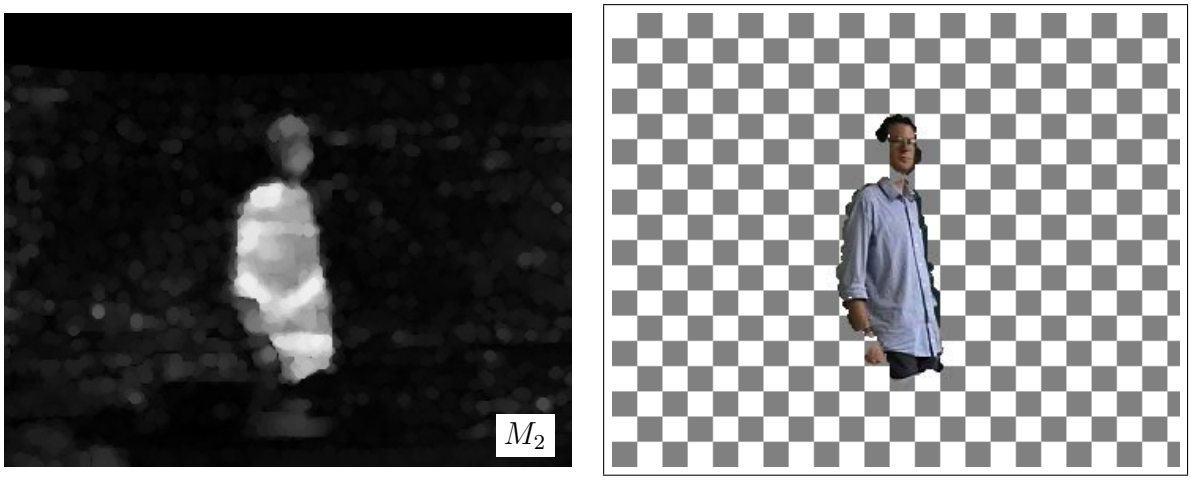


FIG. 5.9: Masque d'occultation avant application du seuil (algorithme 5.1) et premier plan extrait (les parties transparentes sont représentées en échiquier gris et blanc).

La figure 5.9 montre un résultat. Le masque est correct sauf sur la tête, qui paraît détachée du corps. Ceci est dû au fait que la couleur de $I_{\mathrm{p,g},t}$ (la craie brune) est très proche de celle de la peau sur $I_{t,\mathrm{p}}$.

Pour $\mathbf{m} \in [\![0, l-1]\!] \times [\![0, h-1]\!]$ **faire**
  $M_1(\mathbf{m}) \leftarrow \texttt{max3}(I_{\mathrm{p,g},t}(\mathbf{m}) - I_{t,\mathrm{p}}(\mathbf{m}))$ -- soustraction d'arrière-plan
finpour
$M_2 \leftarrow \texttt{dilatation}(\texttt{érosion}(M_1))$ -- ouverture morphologique
Pour $\mathbf{m} \in [\![0, l-1]\!] \times [\![0, h-1]\!]$ **faire**
  $M_3(\mathbf{m}) \leftarrow \texttt{seuillage}(M_2(\mathbf{m}), s)$
finpour
-- En sortie, M_3 vaut 1 aux points occultés, 0 sinon

Fonctions utilisées :
- `max3`(v) renvoie $\max\{|v_1|, |v_2|, |v_3|\}$;
- les opérateurs morphologiques `dilatation` et `érosion` utilisent un disque comme élément structurant (appendice 7.5.2.4) ;
- `seuillage`(v, s) transforme le niveau de gris $v \in \mathbb{R}$ en masque :

$$\texttt{seuillage}(v, s) = \begin{cases} 1 & \text{si } v > s \\ 0 & \text{sinon} \end{cases}$$

Algorithme 5.1: Algorithme de segmentation par « soustraction » entre l'image panoramique de référence $I_{\mathrm{p,g},t}$ et l'image courante $I_{t,\mathrm{p}}$. Le seuil s est à ajuster manuellement.

Pour rebondir sur la contrainte du temps réel qui est la caractéristique majeure de notre contribution, nous affirmons que l'implémentation (appendice 7.5) de l'algorithme 5.1 « sature » les ordinateurs personnels actuels. Au delà, la « goutte d'eau ferait déborder le vase ».

5.3 Codage vidéo

Nous utilisons notre méthode pour envoyer par un réseau une séquence vidéo au fur et à mesure de son acquisition.

5.3.1 État de l'art (très rapide)

5.3.1.1 Codage fond-forme

Beaucoup de recherches actuelles traitent de la compression vidéo **par objets** : différentes régions de la scène sont codées par des méthodes dédiées, choisies pour leur efficacité sur un type d'images donné.

Dans ce contexte, Irani *et al.* ([IAB+96] §2.2) ont proposé de coder séparément le premier plan et l'arrière-plan de séquences vidéo sportives. L'arrière-plan est composé d'une

vue panoramique. Des techniques de suivi et de segmentation diverses ont été proposées pour constituer un tel encodeur ([IAB+96] §2.2, [DM96], [BDH03]).

5.3.1.2 MPEG-4

Les normes de codage vidéo actuelles (les formats WINDOWS MEDIA, QUICKTIME, REALMEDIA, dans une certaine mesure FLASH, et surtout MPEG-4) ne se limitent pas à ranger une piste vidéo et audio dans un fichier. Un document est composés d'un nombre arbitraire de **médias** (texte, image, séquence vidéo, séquence audio, etc.) organisés dans une **présentation multimédia**.

La norme de codage MPEG-4, propose, en plus des possibilités des formats vidéo traditionnels :

- de nouveaux médias, comprenant des objets 3D, des images panoramiques, des sources audio localisées, des images vectorielles, etc. Ces médias sont des « objets » : ils peuvent être ajoutés, supprimés, manipulés indépendamment ;
- un langage de description de scène, BIFS (*BInary Format for Scenes* [PE02] § 4). Il permet de spécifier comment les médias sont composés entre eux en espace (où afficher ?) et en temps (quand jouer ?). Le langage décrit ussi toutes les interactions avec l'utilisateur (lecture, stop, clic sur un lien, etc.) ;
- un moyen de publier ces données sur le réseau. Elles (médias, BIFS) sont transférées sous forme de **flux** (*stream* en anglais). Chaque donnée composant une présentation est décomposée en paquets. Les paquets des différents flux sont envoyés dans un ordre choisi par un multiplexeur pour que :
  - la présentation puisse être jouée progressivement (*streaming* en anglais). Par exemple le début d'une vidéo peut être affiché avant que toute la séquence soit reçue, comme sur une vraie télévision,
  - les erreurs de transmission par le réseau affectent le moins possible la qualité de la présentation.

Ces ajouts s'accompagnent de nombreuses améliorations des normes de codage pour les séquences vidéo (format H264 et format dédié aux visages), les images (JPEG2000) et les séquences audio (format dédié à la parole).

Des lecteurs (*players*) MPEG-4 plus ou moins complets (`http://www.envivio.com`) sont disponibles pour le grand public.

La norme MPEG-4 spécifie le fonctionnement du décodage pour chaque média mais pas celui de l'encodage. Ceci laisse un grande latitude du côté de l'encodage, ce qui explique les nombreuses recherches portant sur les techniques d'encodage pour les différents médias.

5.3.1.3 Ce que nous en retenons

Dans le cadre de MPEG-4, la composition visuelle nous intéresse particulièrement. Deux médias visuels peuvent être superposés lors de la visualisation de la présentation. À un instant donné, chacun des médias, en partie transparent, est dessiné dans un plan

objet vidéo (**VOP** pour *Video Object Plane* en anglais) séparé. Les deux VOP sont ensuite superposés pour former l'image finale.

Nous pouvons transmettre l'arrière-plan (l'image panoramique) et le premier plan (les objets occultants) dans des flux séparés. Le flux vidéo de premier plan peut être produit et affiché en temps réel ([PE02], p356).

5.3.2 Notre technique de codage

La figure 5.10 détaille notre architecture d'encodage (côté serveur) et de décodage (côté client). Les flux de données contenant les différents médias sont multiplexés, envoyés sur le réseau, et reconstitués du côté client par le « dé-multiplexeur ». Les composantes maison ont pour nom « génération de la vue panoramique », « suivi » et « segmentation ».

5.3.2.1 Traitement de l'image panoramique

L'image panoramique est transformée en flux par un codeur progressif exploitant la multirésolution : à partir d'un premier jet de données, le décodeur peut reconstituer l'image en résolution réduite. La suite du flux (4) ajoute progressivement des détails à l'image. De cette manière, l'affichage de l'arrière-plan peut commencer avant la fin de la transmission des données.

Le suivi fournit, pour chaque image I_t de la séquence les paramètres θ_t de l'homographie associée. Ils sont envoyés dans un flux (0). Du côté client (boîte « re-projection »), ils constituent les paramètres d'une homographie qui, appliquée à la vue panoramique, la ramène dans le repère de l'image de la séquence (équation 5.2). Ceci constitue le VOP/0, celui de l'arrière-plan.

5.3.2.2 Le premier plan

La segmentation fournit une séquence d'images du premier plan, avec les masques d'occultation correspondants. La technique retenue dans MPEG-4 repose sur une partition de l'image en macroblocs (MB).

Codage. Pour les images-clés, l'image et le masque sont codés séparément (flux (1) et (2)) :

- le masque est considéré comme une image à 1 bit par pixel, codée sans perte. Un MB est soit entièrement noir, soit entièrement blanc, soit partagé, auquel cas un codeur arithmétique s'en charge ([PE02] § 8.3.1) ;
- seuls les MB non occultés de l'image (la texture) sont codés, à l'aide d'une transformée en cosinus discrète (DCT). S'ils sont partiellement occultés, seuls les pixels visibles sont codés, avec une DCT unidimensionnelle ([PE02] § 8.3.3).

Le masque et la texture sont tous les deux interpolés en temps, entre les images-clés. Pour cela, un flux (3) fournit un vecteur de translation pour chaque MB.

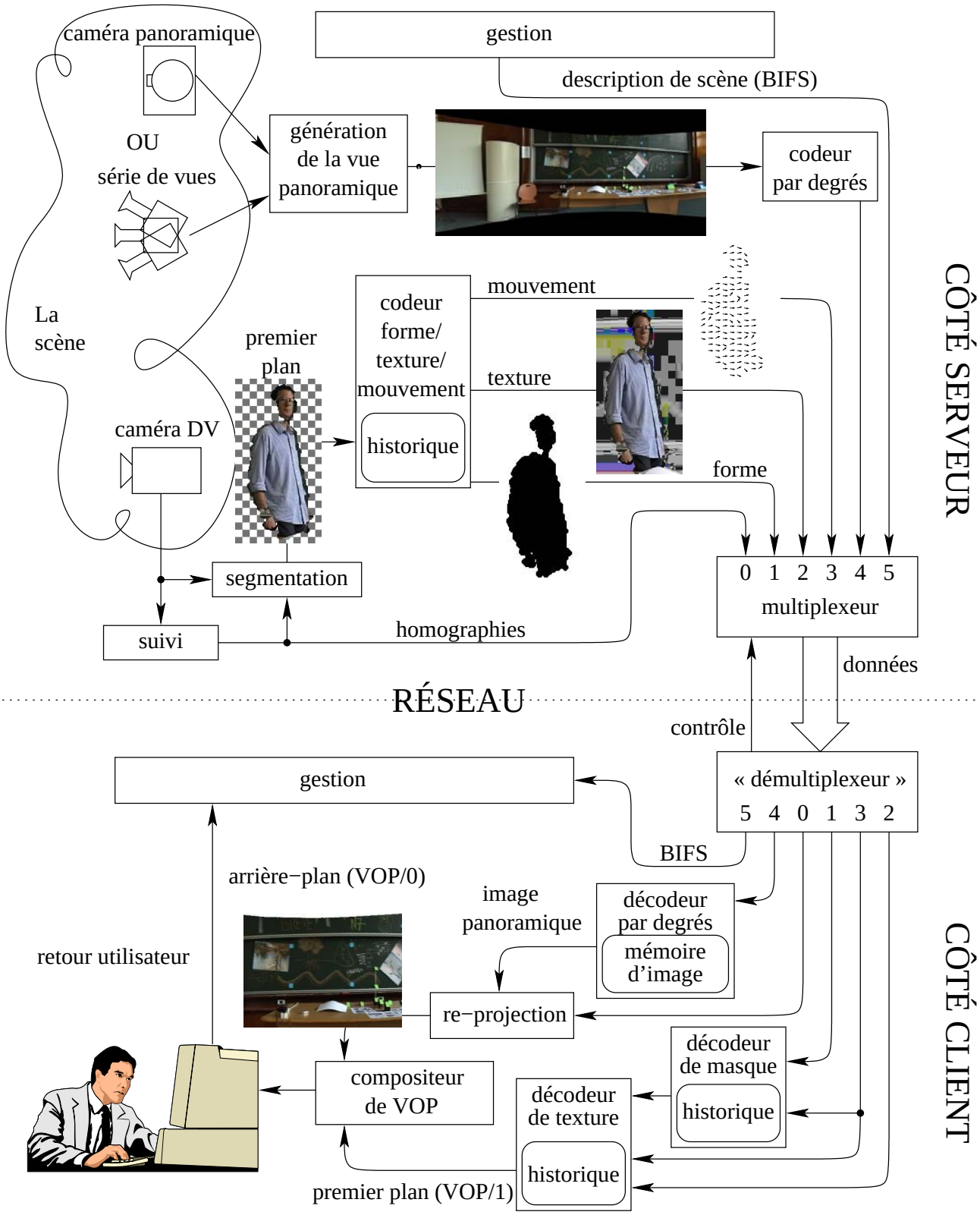


FIG. 5.10: Architecture de codage et de décodage d'une séquence vidéo prise dans nos conditions expérimentales.

Décodage. Du côté client, le masque est décodé en premier. Lui succède (car il en dépend) le décodage de la texture. En sortie, la zone texturée non masquée est transparente et constitue le VOP/1.

5.3.2.3 Superposition

Les deux VOP sont superposés par un compositeur de VOP, et affichés sur l'écran de l'utilisateur. La figure 5.11 présente un exemple d'image obtenue lors de cette combinaison. La segmentation de l'image est celle de la figure 5.9. Les erreurs de segmentation se traduisent par des parties du VOP/1 qui semblent disparaître.

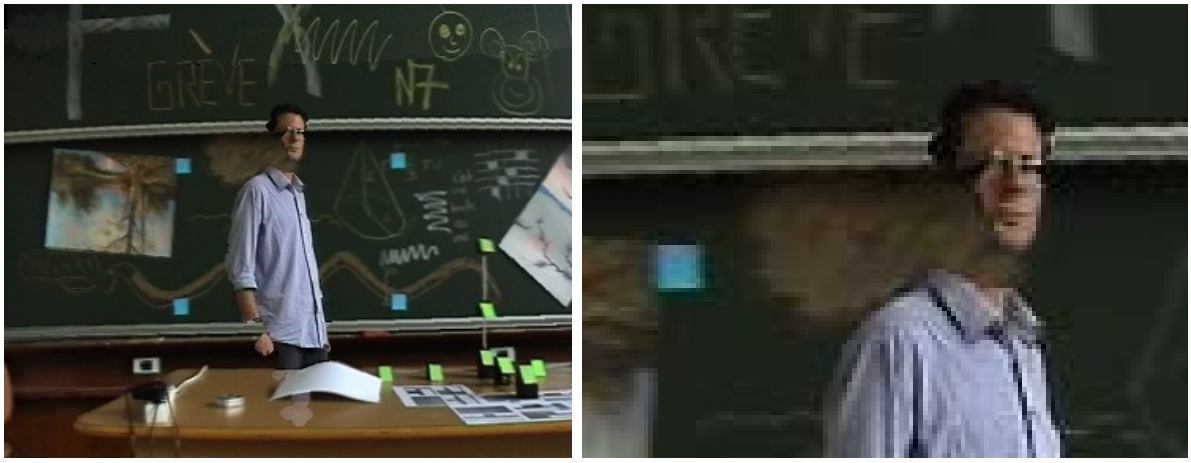


FIG. 5.11: Image d'une séquence vidéo reconstruite en combinant la référence panoramique re-projetée et un objet au premier plan, et vue détaillée qui met en évidence une erreur de segmentation : une partie de la tête n'est pas transmise.

5.3.2.4 Contrôle du lecteur MPEG-4

Coté serveur, un composant de gestion envoie un flux de description de scène au format BIFS (5). Le composant de gestion du client utilise ces informations pour activer la chaîne de décodeurs qui traitent les autres flux de la présentation. Les données BIFS indiquent aussi au composant comment réagir aux actions de l'utilisateur.

5.3.3 Conclusion

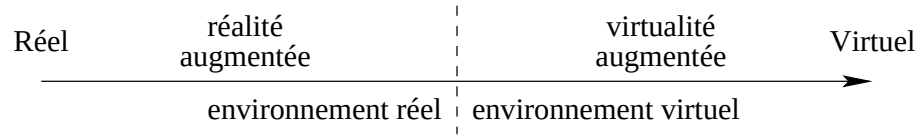
Tous ces composants peuvent fonctionner en temps réel, puisque les flux sont envoyés et affichés simultanément : c'est du *streaming* (accès aux données multimédia en flux continu [Gri03] § 2). Le programme d'affichage, côté client, est un lecteur MPEG-4 générique ; aucun décodeur ou programme spécifiques ne sont nécessaires.

Avec un codeur générique, toutes les images doivent être entièrement encodées. Grâce à notre approche, une fois l'arrière-plan transmis, seul le premier plan nécessite d'être envoyé pour chaque image. La bande passante utilisée pour transmettre les flux s'en trouve réduite *de facto*.

5.4 Réalité augmentée

5.4.1 État de l'art (en toute modestie)

Beaucoup de recherches et de logiciels visent à mélanger des éléments graphiques réels et virtuels dans une scène. Très approximativement, ces combinaisons peuvent être placées sur une échelle (le « continuum réel-virtuel » de Milgram, [Mas04] p3) :



Dans cette échelle :
- le **réel** correspond à des images produites par une caméra ;
- le **virtuel** comprend uniquement des images de synthèse ;
- la **réalité augmentée** consiste à ajouter dans des images réelles des éléments de synthèse (les **augmentations**). Par exemple, de nombreux effets spéciaux de films visent à rajouter des objets 3D dans une scène filmée ;
- la **virtualité augmentée** consiste à ajouter dans un environnement synthétique un objet réel. Dans cette catégorie se trouvent les logiciels de communication où les participants sont représentés par des figurines (des **avatars**) dans une scène virtuelle. La tête de ces avatars est souvent une photo du visage du participant qu'ils représentent : c'est l'élément réel.

Ces deux derniers cas sont regroupés sous le terme de « réalité mixte », ils nécessitent à la fois des techniques d'analyse et de synthèse d'images. Nous nous intéressons particulièrement à la réalité augmentée, parce que c'est la scène réelle qui détermine la position et l'orientation de la caméra, la partie de synthèse devant s'y adapter.

Pour produire les images de synthèse, on utilise une scène virtuelle comprenant les augmentations, une caméra virtuelle et des sources de lumières. L'image produite par la caméra virtuelle est combinée avec l'image réelle. Dans le cas le plus simple, la combinaison revient simplement à superposer l'image virtuelle, partiellement transparente, à l'image réelle.

Azuma *et al.* ([ABB+01] p35) classifient les logiciels de réalité augmentée selon :
- le type d'application. Le logiciel doit-il fonctionner en temps réel (cas des visites virtuelles) ou hors-ligne (cas des effets spéciaux) ? Selon eux, seules les applications interactives relèvent de la réalité augmentée ;

- le type de données en entrée. L'incontournable caméra est-elle étalonnée ? y-a-t-il des capteurs dédiés de position et d'orientation ? des marqueurs ? des données *a priori* sur la scène ?
- le support d'affichage. Est-ce un écran d'ordinateur ou des lunettes semi-transparentes sur lesquelles peuvent être affichées des augmentations ?

Dans notre cas, l'application est en temps réel, la seule entrée est le flux vidéo et l'affichage se fait sur un écran.

5.4.1.1 La visualisation en réalité augmentée

La production d'images en réalité augmentée nécessite de mette en cohérence l'image de la scène réelle et les augmentations. Ce problème a deux aspects : géométrique et photométrique ([Gib04] p3).

La cohérence photométrique. Elle concerne la visualisation des couleurs :

- l'environnement lumineux de la scène virtuelle doit être le même que celui de la scène réelle. Ainsi, les objets ont une luminosité et une ombre propre compatibles avec leur environnement ;
- les interactions lumineuses entre objets doivent être prises en compte. Les ombres portées des objets virtuels sur les objets réels (et vice-versa) sont synthétisées.

La visualisation hors-ligne réaliste des augmentations est possible depuis le début des années '90 en utilisant le lancer de rayons. De nos jours, elle peut se faire en temps réel ([GCHH03] § 2) grâce à :

- une utilisation astucieuse des cartes vidéo, dont la vitesse croît « exponentiellement » et qui peuvent faire des opérations de plus en plus complexes ;
- des hypothèses simplificatrices tant sur le modèle d'éclairage que sur les objets dans les zones d'ombre.

Quelle que soit la technique appliquée, un modèle 3D approximatif de la scène réelle (objets et lumières) est nécessaire.

La cohérence géométrique. Elle est soumise à deux contraintes :

- les paramètres intrinsèques des caméras réelle et virtuelle doivent être identiques, ainsi que leur pose dans leurs scènes respectives (la position est définie à un facteur d'échelle près). De cette manière, les objets synthétiques immobiles dans la scène virtuelle paraissent immobiles dans la vue augmentée, quel que soit le mouvement de la caméra ;
- les interactions entre objets réels et virtuels doivent être réalistes. Ceci concerne tout particulièrement les occultations et les collisions.

Nous nous intéressons à cet aspect géométrique.

5.4.1.2 Les contraintes géométriques

Pour réaliser l'augmentation des séquences vidéo, le modèle géométrique de la caméra virtuelle peut être déduit d'un suivi visuel de points de référence immobiles dans la scène. Cette approche qui nécessite de connaître entièrement la trajectoire 2D (dans la vidéo) des points de référence, permet d'estimer simultanément les coordonnées 3D de ces derniers et la pose de la caméra pour chaque image. Elle ne peut donc pas fonctionner en temps réel.

Faugeras ([Fau98] p23) augmente des vidéos avec des objets synthétiques qui paraissent immobiles par rapport à un objet réel de référence. Les occultations de l'objet de synthèse par des objets réels ne sont pas gérées, mais grâce à un modèle 3D simplifié de l'objet de référence, une ombre portée peut être introduite.

Le problème des occultations d'objets virtuels par des objets réels est relativement facile à traiter quand un modèle 3D complet de la scène est disponible.

Lepetit et Berger [LB00] utilisent l'information minimale pour traiter les occultations, à savoir le résultat d'un détourage des objets réels occultants sur quelques « images-clés ». Ces régions sont interpolées pour les images intermédiaires. Lors de la visualisation, elle sont affichées au premier plan, masquant les augmentations. Comme la technique nécessite une action de l'utilisateur sur les « images-clés » de la vidéo augmentée, elle ne peut pas fonctionner en temps réel.

5.4.1.3 Le temps réel

La réalité augmentée en temps réel doit satisfaire à deux conditions :

la causalité. Le traitement de l'image courante ne dépend pas des images suivantes de la vidéo ;

la rapidité. Les images sont traitées au rythme de la vidéo (en moins de 40 ms).

La réalité augmenté en temps réel est possible en utilisant des informations *a priori* sur la scène, par exemple :

- les paramètres intrinsèques de la caméra et quatre points sur un plan texturé. Sur cette base, Simon et Berger ([SB02] § 3) ajoutent un objet solidaire de ce plan. Le suivi peut être rendu plus robuste en étendant le traitement à plusieurs plans du même objet rigide ;
- la définition d'un rectangle dans la scène. Il sert de mire d'étalonnage (paramètres intrinsèques) de la caméra ([SB02] § 4). En y rajoutant la dimension 3D d'un objet (par exemple la longueur d'un des côtés du rectangle), le facteur d'échelle peut être estimé ([PrM04] § 3). Les objets virtuels sont alors définissables dans un espace métrique ;
- le modèle 3D d'un objet de référence. Lepetit *et al.* ([LVTF03] § 2) procèdent à une phase d'apprentissage s'appuyant sur un modèle 3D de l'objet et quelques images de référence (*keyframes*) de celui-ci, pour détecter des points caractéristiques de sa texture. Lors de l'expérience de réalité augmentée, pendant l'initialisation et quand le suivi échoue, la phase d'apprentissage leur permet de localiser la cible sans utiliser

l'hypothèse de petits changements. Ils présentent des exemples d'applications où l'objet de référence est un objet manufacturé, un bâtiment ou un visage.

Ces informations sont d'autant plus faciles à obtenir que l'environnement est contrôlé (typiquement, un contexte urbain).

5.4.1.4 Ce que nous en retenons

Dans le contexte de la réalité augmentée, nous nous attachons à satisfaire les contraintes géométriques. Nous pouvons faire ceci en temps réel grâce à notre cadre expérimental très simplifié : le centre optique de la caméra est immobile.

D'un point de vue géométrique, ceci permet de :

- remplacer les paramètre (intrinsèques et extrinsèques) de la caméra par ceux d'une homographie ;
- réduire l'information à connaître sur l'image réelle et les augmentations à deux dimensions.

D'un point de vue photométrique, l'absence d'information 3D limite les possibilités d'adaptation des lumières à une correction de la balance des blancs.

5.4.2 Réalité augmentée en 2D

Notre système de réalité augmentée repose sur une **superposition** d'images partiellement transparentes : les **plans**.

5.4.2.1 Les plans

À chaque plan sont associés une image et un masque d'occultation, ou « canal alpha ». Mathématiquement, à l'image I en couleurs nous adjoignons un canal alpha J

$$J : \mathbb{R}^2 \longrightarrow [0, 1]$$

Si $J(\mathbf{m})$ vaut 1, le plan est opaque en ce point, et sa couleur est $I(\mathbf{m})$. S'il vaut 0, le plan est transparent. Les valeurs intermédiaires indiquent une transparence partielle.

Les plans que nous manipulons sont (voir aussi la figure 1.1) : ♠

- l'image panoramique de référence alignée géométriquement $I_{\mathrm{p,g},t}$, qui est entièrement opaque ;
- l'image au premier plan extraite $I_{t,\mathrm{p}}$, extraite en calculant le masque d'occultation à l'aide de l'algorithme 5.1 ;
- l'image de synthèse d'une augmentation qui est opaque seulement aux points où l'objet est visible ;
- une image panoramique modifiée, entièrement opaque.

5.4.2.2 La superposition

L'image S, superposition des plans $I_1, ..., I_n$ de canaux alpha $J_1, ..., J_n$ est définie par le schéma itératif :

$$\begin{cases} S_0 & \text{est } \textit{ad hoc} \text{ (c'est par exemple un motif d'échiquier gris et blanc)} \\ S_i : & \mathbf{m} \rightarrow S_i(\mathbf{m}) = (1 - J_i(\mathbf{m}))S_{i-1}(\mathbf{m}) + J_i(\mathbf{m})I_i(\mathbf{m}) \end{cases} \tag{5.3}$$

La superposition est $S = S_n$.

Les occultations sont induites par :
- l'ordre de superposition des plans ;
- les parties transparentes de ceux-ci.

Ces plans transparents sont analogues aux empilements de feuilles de cellulose utilisés traditionnellement dans la création de dessins animés. Cependant, à l'opposé de cette technique, les transformations applicables aux images ne se limitent pas à des translations mais à des homographies.

5.4.3 Expérimentation

Nous avons écrit deux scénarios de réalité augmentée.

5.4.3.1 Insertion d'un objet de synthèse

Il s'agit d'ajouter un objet de synthèse dans la séquence vidéo pour aboutir à une intégration en harmonie (géométrique) avec le « décor » existant. Cet objet est en premier plan et ne subit aucune occultation en provenance des composantes réelles.

Pour cela, il suffit de superposer les plans suivants :

1. l'image courante de la séquence vidéo $I_{t,\mathrm{p}}$, sans masque d'occultation. La segmentation n'est pas utilisée ;
2. un plan contenant l'image artificielle d'une augmentation. Ce peut être :
   - un objet 3D virtuel dont l'image est calculée en temps réel ;
   - une image panoramique précalculée porteuse de l'augmentation.

La figure 5.12 illustre ce scénario. Le mouvement du bureau virtuel est cohérent avec le reste de la scène : il reste fixe par rapport au tableau.

5.4.3.2 Extraction d'intrus

Il s'agit d'extraire de la scène les objets/personnages occultants et les représenter au premier plan d'un décor virtuel.

Pour cela, il suffit de superposer les plans suivants :

1. une image panoramique modifiée. Un dessinateur peut la préparer à partir de photographies ou d'éléments synthétiques ;
2. l'objet occultant extrait de la séquence vidéo.

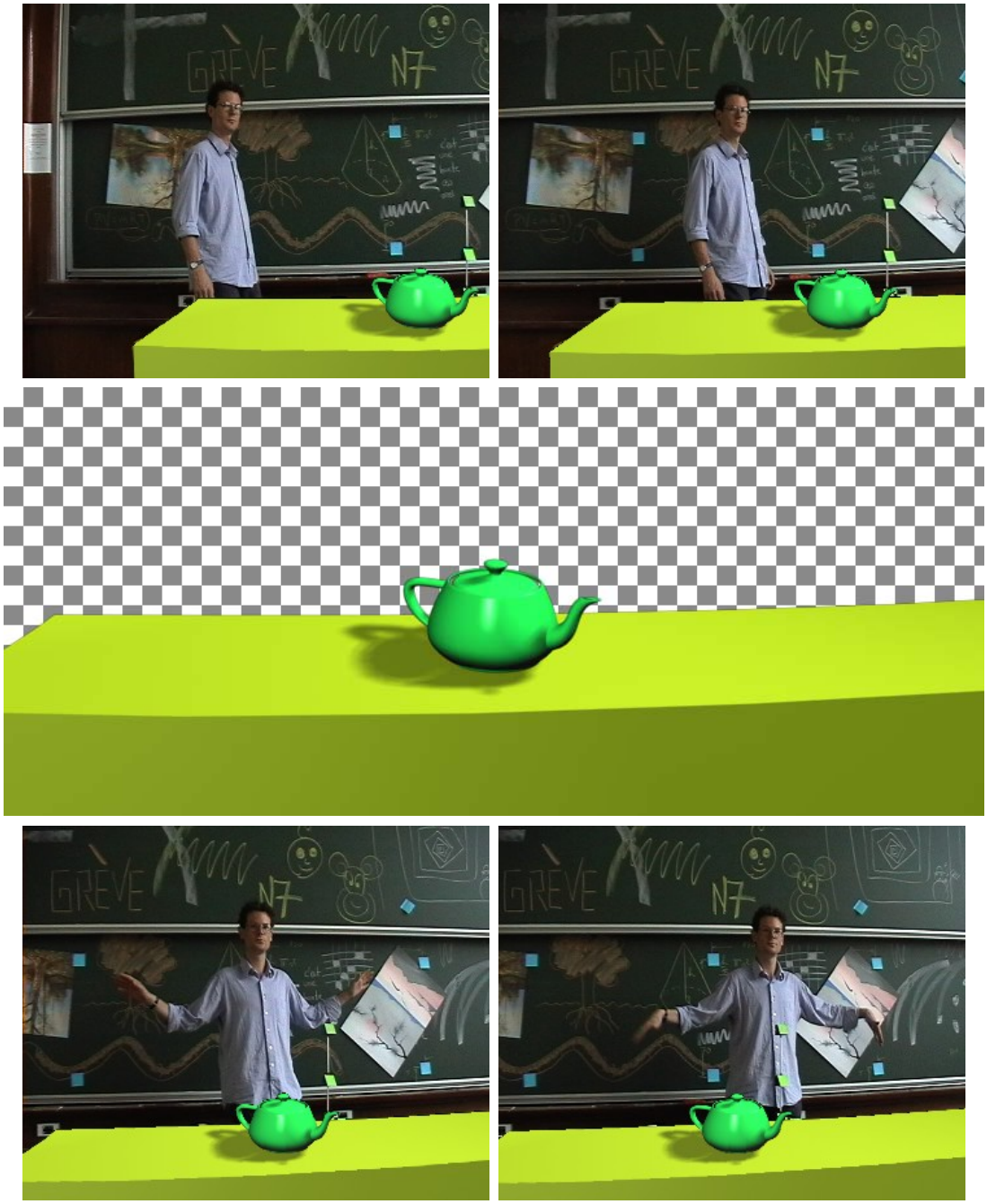


FIG. 5.12: Réalité augmentée avec insertion d'un objet de synthèse : l'objet de synthèse (au milieu) et quatre images de la vidéo augmentée.

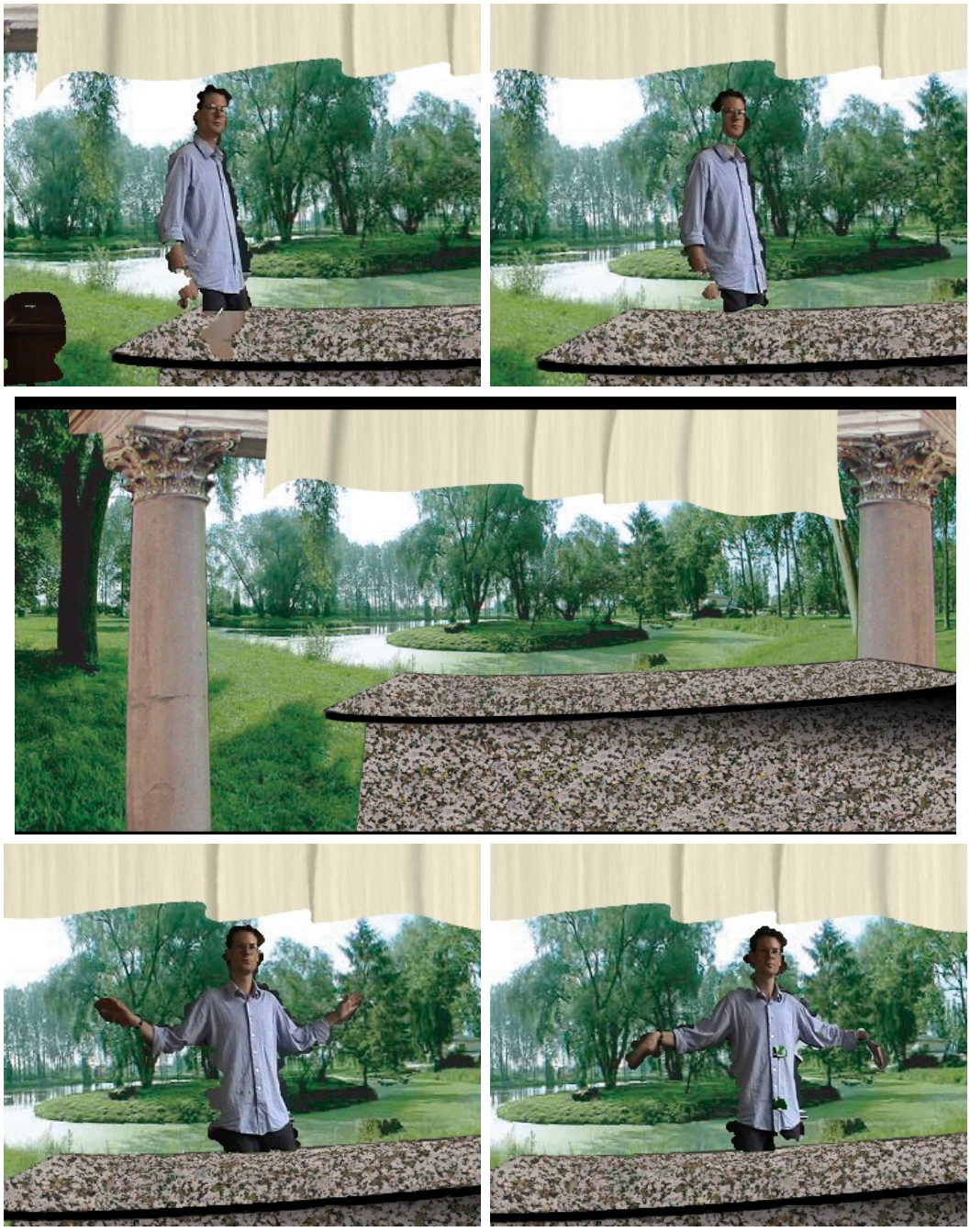


FIG. 5.13: Réalité augmentée avec extraction de l'objet occultant : image panoramique de synthèse (au milieu) et images de la séquence où l'arrière-plan est remplacé par celle-ci. La séquence vidéo complète et les quatre images retenues sont celles de la figure 5.12.

Ce scénario correspond au cas présenté dans l'introduction (§ 1.1).

La figure 5.13 illustre le scénario. L'image panoramique modifiée est un mélange artistique d'éléments synthétiques et de photographies. Ce « faux fond » a le même mouvement apparent que la scène réelle. Le déplacement du personnage ne permet pas de déceler le trucage !

Chapitre 6

Conclusion

Grâce à notre « suivi robuste, en temps réel, de séquences vidéo » au sein d'une image panoramique de référence nous avons pu développer des applications qui visent à « composer des objets visuels ».

6.1 La composition d'objets visuels

Les deux contextes. Nous traitons la composition dans deux contextes généraux :

- le codage vidéo par objets. L'obstacle est alors la « détection » d'un objet (ie. d'une région associée, dans un sens à définir) et son suivi spatio-temporel ;
- la réalité augmentée. Le nœud du problème est alors de caler (et de maintenir ce calage !) les paramètres extrinsèques (voire intrinsèques) du capteur de synthèse sur ceux du capteur réel.

Les simplifications. Notre approche opérant dans des conditions spécifiques et simplificatrices (caméra à centre optique fixe, § 5.1) nous a permis d'unifier ces deux « interprétations » de la composition d'objets visuels :

- dans un sens (codage, § 5.3), le calcul des « homographies inter-images » (c'est le titre de la thèse) permet le suivi de « l'objet fond » ;
- dans l'autre sens (réalité augmentée, § 5.4), les homographies calculées rassemblent l'information utile sur les paramètres de la caméra.

Les autres objets. Ils ne suivent pas le modèle de mouvement, mais nous les manipulons afin d'aboutir à nos applications de « composition d'objets visuels ». Le traitement (§ 5.2) doit porter, explicitement ou implicitement, sur leurs paramètres de forme, de texture, de position dans la scène et de mouvement.

6.2 Comment avons-nous « attaqué » le problème ?

Nous avons adopté systématiquement la même démarche dans l'approche de ces différents points noirs.

L'idée force est d'abord de ne pas traiter ces problèmes complexes dans toute leur généralité : par exemple, 30 ans de travaux imparfaits sur des méthodes générales de segmentation d'images nous incitent à un peu d'humilité concernant notre tâche d'identification des objets qui suivent le modèle.

De même, alors que le modèle de projection plane 3D $\rightarrow$ 2D englobe tous les paramètres intrinsèques et extrinsèques de la caméra, nous le ramenons à une homographie 2D dans le cas où :

- la cible d'intérêt de l'espace 3D est plane (chapitre 4) ;
- il est nécessaire et suffisant de comparer deux images provenant d'une caméra dont le centre optique est fixe (chapitre 5).

L'estimation de paramètres. À l'instar de nombreux auteurs de travaux actuels au sein de la communauté de vision par calculateur, nous avons privilégié une approche où l'estimation de paramètres nous permet :

- de réunir adroitement un large ensemble d'informations ou de **données de référence**, à savoir les observations, les mesures, les expériences faites hors ligne, les images de référence ou d'apprentissage ;
- de choisir un ou plusieurs modèles mathématiques paramétrés pouvant expliquer (sur le plan géométrique, photométrique, temporel, etc.) la réunion non hasardeuse de ces données de références ;
- d'identifier mathématiquement et noter en noir sur blanc l'ensemble des **approximations sous-jacentes** aux divers modèles et, dans un même élan, l'ensemble des **erreurs numériques** susceptibles de contaminer ces informations ;
- de représenter de manière compacte et **robuste** ces données par un simple vecteur de paramètres ;
- de valider autant que faire se peut la démarche par l'expérience. Elle s'applique d'abord sur des données parfaitement contrôlées (de synthèse), afin de « fortifier le modèle nouveau-né », puis sur les « vrais » problèmes rencontrés sur des images de scènes réelles.

Le temps réel. Cette démarche systématique étant par ailleurs placée dans un cadre fortement contraint par le temps, nous avons décliné ces principes de plusieurs façons pour que nos résultats numériques soient délivrés « juste à temps ».

Ainsi, lorsque l'acquisition d'informations de référence est coûteuse, nous avons privilégié une approche d'apprentissage « hors ligne » des modèles. Lorsque les étapes numériques de résolution sont lourdes, un **effort d'implémentation** a dû être fait.

6.3 Contributions

Nos contributions sont nombreuses, à défaut d'être fondamentales. Relevons :
- une technique de calcul de la SVD partielle pour les matrices contenant des niveaux de gris (§ 2.6.2.2 et 7.3) ;
- une solution du problème DLS multidimensionnel et la technique de calcul associée (§ 2.6.2.3) ;
- l'estimateur LTTS, combinaison de LTS et TLS (§ 2.6.4.1) ;
- une approche hiérarchique par régions (§ 3.4.4) et une estimation non linéaire itérée de l'homographie (§ 3.4.5.4) pour la génération de mosaïques ;
- des améliorations de la méthode de suivi de Jurie et Dhome. Parmi elles :
  - l'estimation de la matrice d'apprentissage par TLS (§ 4.1.3.4),
  - une étude du type d'expériences à appliquer pendant l'apprentissage (§ 4.1.4.3),
  - un diagnostic sur le résultat du suivi (§ 4.1.4.4),
  - un agencement hiérarchique de plusieurs méthodes (§ 4.1.4.5),
  - un schéma de prédiction simple (§ 4.1.4.6) ;
- une formulation et sa résolution du problème de la suppression de points de référence dans la méthode de Jurie et Dhome (§ 4.2.3.3). Pour cela, nous avons développé deux méthodes d'actualisation négative (*downdating*) multiple d'une décomposition de Cholesky : une analytique (§ 4.2.3.6) et une itérative (§ 4.2.3.7) ;
- une expérimentation reposant sur la fabrication à la maison d'une image de référence panoramique de la scène (§ 5.1) ;
- une technique d'étalonnage photométrique avec un modèle affine (§ 5.2.2.2) ;
- une formulation de la réalité augmentée 2D en termes de plans transparents (§ 5.4.2) ;
- une technique de calcul sur processeur graphique de la dilatation et de l'érosion morphologiques dans les cas séparable et non-séparable (§ 7.5.2.4).

6.4 Perspectives

« Les perspectives n'engagent que ceux qui y croient. » Ainsi (à peu près) sonne l'adage désabusé d'un homme politique célèbre. Nous nous permettrons néanmoins d'ouvrir quelques pistes dans la continuité de notre travail.

Apprentissage. Comment mieux choisir nos données de référence ? (ensemble d'apprentissage par exemple). Pour le moment, nous n'avons que des résultats empiriques sur la manière dont doit être mené l'apprentissage.

Modèle de mouvement. Comment enrichir le modèle de mouvement ? Il y a plusieurs pistes, consistant à introduire plus de connaissance sur les objets suivis :
- modèles basés sur des maillages (plus compacts qu'un champ dense de mouvements) ;
- modèles déformables (de gestes, articulaires) adaptés au suivi d'un personnage ;
- modèles de textures (tissus par exemple).

Stabilisation du suivi. Comment ajuster le compromis entre un suivi « tremblant » et réactif (sans lissage) et un suivi stabilisé mais qui peut avoir quelques images de retard (avec lissage) ? Notre méthode pourrait être régularisé en donnant plus de place à la prédiction.

Suivi basé sur une image panoramique brute. Dans la version actuelle, l'image de référence est obtenue par re-projection d'une ou plusieurs images brutes. Elle est échantillonnée une deuxième fois pendant le suivi. Pour éviter ce double échantillonnage, l'algorithme doit pouvoir traiter directement l'image brute.

Approches des occultations. Pour traiter les occultations, vaut-il mieux chercher à supprimer les points occultés ou rassembler ces points en petites régions et détecter les régions occultées ? Nous n'avons exploré que la première approche.

Implémentation. Nos techniques de suivi et de visualisation ne sont pour certaines qu'à l'état de prototypes. Il faudrait les intégrer dans un « vrai » logiciel.

6.5 Articles

Le travail de cette thèse a été émaillé de publications présentées dans des conférences. Certaines décrivent l'état courant du travail. D'autres sont des collaborations, sur des thèmes voisins, qui ont inspiré des extensions à la thèse. Nous les mentionnons par thèmes :

- la générations de mosaïques : [DCT01], [DCT02]. La section 3.4 de la thèse est une version complétée de ces articles ;
- le suivi d'un motif plan : [DCT03], [DCT04]. La section 4.1 est une version étendue et complétée de ces articles ;
- les vidéos structurées (ou hypervidéos) : [GCD01b], [GCD01a], [GCD02]. Ce thème a aiguillé notre travail sur le codage en direction de MPEG-4 (partie 5.3.1.2) ;
- l'accélération vectorielle : [Dou03]. L'appendice 7.3 est une version condensée de cet article ;
- la caméra panoramique tournante : [DPCC03]. L'article décrit cette caméra (§ 3.3.2) et expose les principes du codage ;
- les applications : [DC03], [DDCM04]. Les articles concernent le codage et la réalité augmentée, décrits en détail dans les sections 5.3 et 5.4.

Chapitre 7

Appendices

Ce chapitre rassemble un ensemble disparate de thèmes à dominante technique.

7.1 Approcher une rotation par une translation

L'algorithme de génération de mosaïques approche l'homographie qui lie deux images par une translation (§ 3.4.4). L'effet d'une rotation 3D de la caméra sur son image en 2D peut être approché par une translation, surtout si l'angle de rotation est faible.

Nous montrons ceci pour une rotation autour de l'axe vertical.

Géométrie. Nous considérons deux images d'une scène immobile prises par la caméra qui tourne d'un angle α autour de l'axe vertical. Nous utilisons les repères suivants (figure 7.1) :

- en 3D : Z est l'axe optique de la caméra, X et Y correspondent aux directions horizontale et verticale de l'image. L'axe de rotation est Y ;
- en 2D : les coordonnées sont mesurées sur le plan image orthogonal à l'axe Z à une distance 1 du centre optique de la caméra.

Soit un point M dans la scène. Ses coordonnées 3D sont (X, Y, Z) et celles de sa projection sont (x, y). Après rotation, M se transforme en M', dont les coordonnées 3D et 2D sont respectivement (X', Y', Z') et (x', y'). Les équations de projection et de rotation sont :

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix} \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} X'/Z' \\ Y'/Z' \end{bmatrix}$$

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \times \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

En combinant les deux, il vient :

$$\left\{ \begin{array}{lll} x' & = \frac{X\cos\alpha + Z\sin\alpha}{-X\sin\alpha + Z\cos\alpha} & = \frac{x\cos\alpha + \sin\alpha}{-x\sin\alpha + \cos\alpha} \\ y' & = \frac{Y}{-X\sin\alpha + Z\cos\alpha} & = \frac{y}{-x\sin\alpha + \cos\alpha} \end{array} \right.$$

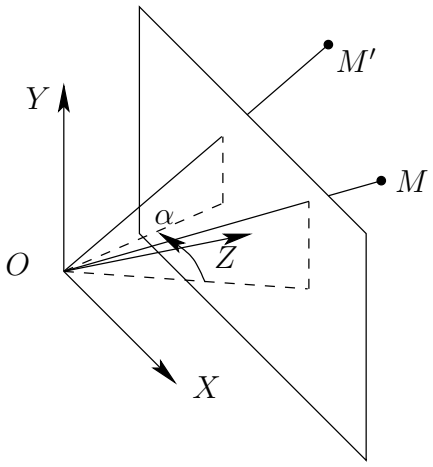


FIG. 7.1: Rotation d'un point autour de l'axe Y.

Le développement limité d'ordre 1 en α de l'expression donne :

$$\begin{cases} x' \approx x & +\alpha & +x^2\alpha \\ y' \approx y & & +xy\alpha \end{cases} \tag{7.1}$$

Analyse. Si le point principal est au centre de l'image, les coordonnées x et y varient dans des intervalles centrés en 0. Nous choisissons donc la translation subie par $(x, y) = (0, 0)$, ce qui donne le vecteur translation approximatif $(x_t, y_t) = (\alpha, 0)$. Cette approximation est valide si :

- l'angle α est petit (pour justifier le développement limité) ;
- les coefficients x^2 et xy sont petits (à cause de l'équation (7.1)). Pour un angle de vue de 60° et un rapport 4/3 entre la hauteur et la largeur de l'image, les coordonnées vérifient :

$$-1/2 < x < 1/2 \qquad \text{et} \qquad -3/8 < y < 3/8$$

Ceci implique qu'il vaut mieux éviter les indices visuels aux bords de l'image.

Ces résultats peuvent être généralisés à toute combinaison de rotations autour des axes X et Y.

Dans le cas de zooms et de rotations autour de l'axe Z, il n'existe pas de translation approchée meilleure que l'identité. Ils ne nous intéressent pas en premier lieu parce qu'ils ne permettent pas de « découvrir » une scène (comme le fait un mouvement de caméra panoramique).

Cette approximation est très souvent utilisée dans les dessins animés, où des mouvements de translation simulent des balayages panoramiques.

7.2 Échantillonnage et re-projection d'images discrètes

7.2.1 Représentation des images

Représentation théorique. Dans tout le texte, nous représentons l'image I en couleurs sous la forme d'une fonction :

$$I : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$$

Pour le point $\mathbf{m} \in \mathbb{R}^2$, le vecteur $v = I(\mathbf{m})$ représente les composantes de couleurs rouge (v_1), verte (v_2) et bleue (v_3) du point.

De la même manière, une image en niveaux de gris I prend la forme :

$$I : \mathbb{R}^2 \longrightarrow \mathbb{R}$$

et $I(\mathbf{m})$ est la luminosité du point $\mathbf{m}$.

Dans la suite, nous ne parlons que d'images en niveaux de gris, l'extension à la couleur se fait simplement en dupliquant les opérations sur les trois couleurs primaires.

Sur calculateur. En pratique, sur calculateur, les images sont représentées par un tableau de taille $l \times h$ contenant des entiers ou des flottants. Nous le notons sous forme de fonction (plutôt que de matrice) :

$$\widetilde{I} : [\![0, l-1]\!] \times [\![0, h-1]\!] \longrightarrow S$$

où S est l'ensemble des valeurs représentables par le type de données. Le plus souvent, le tableau est constitué de valeurs sur un 1 octet ($S = [\![0, 255]\!]$) ou de flottants ($S = \mathbb{R}$).

Dans la suite, nous considérons que $S = \mathbb{R}$, la conversion en octets peut se faire par arrondi et écrêtement.

Comme $\widetilde{I}$ est un tableau, dans les algorithmes nous nous permettons d'écrire l'affectation

$$\widetilde{I}(\mathbf{m}) \leftarrow v \in \mathbb{R}$$

7.2.2 Échantillonnage et dérivation

L'image mesurée $\widetilde{I}$ est une version dégradée de l'image théorique I. La relation entre I et $\widetilde{I}$ dépend de diverses propriétés du capteur :

- la surface d'intégration d'un pixel du CCD ;
- le temps d'intégration ;
- le bruit du capteur.
- la fonction qui transfert l'énergie lumineuse en signal électrique ;
- l'échantillonnage de ce signal.

Malgré ces éléments perturbateurs, nous cherchons à reconstruire une valeur approchée $\widehat{I}$ de I à partir de $\widetilde{I}$. Nous cherchons aussi une valeur approchée $\widehat{I}'$ de la dérivée I'.

Soient $(x, y) \in [0, l[\times[0, h[$ et $(u, v) = (\lfloor x \rfloor, \lfloor y \rfloor)$; nous calculons $\widehat{I}(x, y)$ et $\widehat{I}'(x, y)$ de plusieurs façons. Pour les points extérieurs au pavé $[0, l[\times[0, h[$, un traitement dépendant de l'application doit être utilisé.

7.2.2.1 Échantillonnage brut

C'est la technique la plus simple et la plus rapide :

$$\widehat{I}(x,y) = \widetilde{I}(u,v)$$

Cette expression n'est pas dérivable, nous calculons une « dérivée » par différences finies. Pour les différences finies centrées, il vient :

$$\widehat{I'}(x,y) = \left[\frac{\widetilde{I}(u+d,v) - \widetilde{I}(u-d,v)}{2d} \quad \frac{\widetilde{I}(u,v+d) - \widetilde{I}(u,v-d)}{2d} \right]$$

où $d \in \mathbb{N}$ est un facteur de distance sur laquelle on mesure la dérivée (en général, $d = 1$).

7.2.2.2 Interpolation bilinéaire

Dans cette technique, l'image est supposée coïncider avec la version discrète sur les points entiers (qui vérifient $(x,y) = (u,v)$). Pour les autres points, il faut utiliser une interpolation :

$$\widehat{I}(x,y) = \begin{array}{l} (1+u-x)(1+v-y)\widetilde{I}(u,v) + \\ (x-u)(1+v-y)\widetilde{I}(u+1,v) + \\ (1+u-x)(y-v)\widetilde{I}(u,v+1) + \\ (x-u)(y-v)\widetilde{I}(u+1,v+1) \end{array}$$

La dérivée $\widehat{I'}$ est définie par dérivation directe de $\widehat{I}$.

7.2.2.3 Par convolution

Si l'image est de résolution suffisante mais bruitée, elle peut être échantillonnée par convolution avec un noyau gaussien ([Mar82] p54) :

$$\widehat{I}(x,y) = (\widetilde{I} \otimes \Gamma)(u,v)$$

où le noyau Γ, séparable, est défini par :

$$\Gamma_{ij} = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{i^2+j^2}{2\sigma^2} \right) \qquad i,j \in [\![-k,k]\!]$$

L'écart-type σ est à définir selon l'intensité du bruit et l'étendue de la convolution ; k est à choisir en fonction de σ.

Cette expression n'est pas continue. Sa dérivée approchée $\widehat{I'}$ est donc calculé par différence finies (ce qui peut se faire par convolution avec le masque dérivé par différences finies).

7.2.3 Re-projection

Un problème apparenté à l'échantillonnage est le calcul pratique de l'image J définie par déformation de l'image I :

$$J(\mathbf{m}) = I(f(\mathbf{m})) \qquad \text{où } f : \mathbb{R}^2 \longrightarrow \mathbb{R}^2 \tag{7.2}$$

L'image discrète $\widetilde{J}$ est de taille $l_{\mathrm{J}} \times h_{\mathrm{J}}$. Selon le choix de la méthode d'échantillonnage et du traitement des points extérieurs, le résultat $\widehat{J}$ peut varier (algorithme 7.1).

Pour $\mathbf{m} \in [\![0, l_{\mathrm{J}} - 1]\!] \times [\![0, h_{\mathrm{J}} - 1]\!]$ **faire**
  $\mathbf{m}' \leftarrow f(\mathbf{m})$
  Si $\mathbf{m}' \notin [0, l[\times[0, h[$ **alors**
    $\widehat{J}(\mathbf{m}) \leftarrow v$ -- v est une valeur arbitraire signifiant « inconnu »
  sinon
    $\widehat{J}(\mathbf{m}) \leftarrow \widehat{I}(\mathbf{m}')$ -- en utilisant une des méthodes d'échantillonnages du § 7.2.2
  finsi
finpour

Algorithme 7.1: Calcul approché de la transformation de l'image I par la fonction f (équation (7.2)).

7.3 Multiplication matricielle rapide

Pour la résolution de problèmes de moindres carrés multidimensionnels (§ 2.6.2), nous avons besoin d'une technique de calcul rapide du produit

$$A^\top A$$

où la matrice $A \in \mathbb{R}^{m \times n}$ est allongée : $m \gg n$.

Dans le cas de la phase d'apprentissage de JD (§ 4.1.3.4), les éléments de A sont des différences de niveaux de gris codés sur un octet.

7.3.1 Codage de la matrice

Les différences de niveaux de gris sont des entiers de $[\![-255, 255]\!]$. Les éléments de A peuvent donc être codés sur des entiers signés de 2 octets.

Dans notre cas, nous simplifions le calcul de $A^\top A$ en observant que ces différences sont de la forme

$$A = G - ug^\top$$

où

- la matrice $G \in [\![0, 255]\!]^{m \times n}$ contient des niveaux de gris,
- le vecteur $g \in [\![0, 255]\!]^n$ contient aussi des niveaux de gris ($g = F_0(\mu_0)$) ;

– le vecteur $u \in \mathbb{R}^m$ ne contient que des 1.

En conséquence :

$$A^\top A = G^\top G + gx^\top + x^\top g$$

où $x = \frac{m}{2}g - G^\top u$. Le vecteur $G^\top u$ contient la somme des éléments de G, colonne par colonne. Le calcul coûteux devient celui de $G' = G^\top G$. La matrice G contient des niveaux de gris, elle peut être stockée sur des entiers de 1 octet.

7.3.2 Calcul par blocs

Comme G' est symétrique, nous ne calculons que son triangle supérieur. La matrice G est stockée par colonnes (à la FORTRAN), donc les éléments de G' sont le résultat du produit scalaire de deux vecteurs d'entiers. Ceci donne l'algorithme de base 7.2.

```
Pour i de 1 à n faire
    a ← G_{1:m,i}                          -- manipulation de pointeur
    Pour j de 1 à n faire
        b ← G_{1:m,j}
        s ← 0                              -- accumulateur entier non signé de 4 octets
        Pour k de 1 à m faire
            s ← s + a_k × b_k
        finpour
        G'_{ij} ← s
    finpour
finpour
```

Algorithme 7.2: Algorithme basique de calcul de $G' = G^\top G$, où G est une matrice d'entiers non signés de 1 octet, disposés par colonnes.

Cet algorithme naïf peut être amélioré, comme le montre la figure 7.2. Les trois algorithmes effectuent le même nombre de calculs et d'accès mémoire, mais dans des ordres différents. Nous pouvons choisir celui qui fait les accès les plus locaux pour exploiter le cache de données [DAB00]. Par exemple, si nous avons un cache (niveau 2) de 4 ko, alors :

– l'algorithme naïf accède une grande partie de la mémoire séquentiellement plusieurs fois parce que pour une colonne i, toutes les colonnes $j \geq i$ sont parcourues,

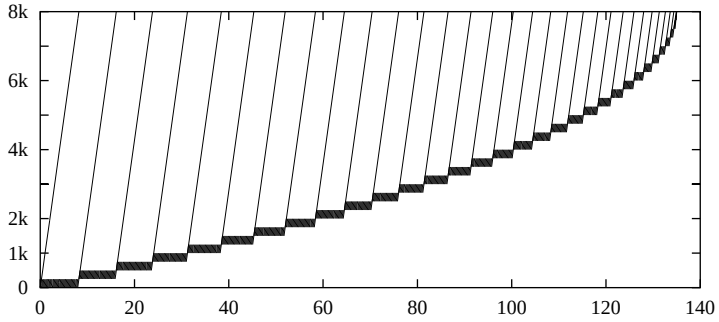
– l'algorithme bloc 1 fait que deux ensembles de b_s colonnes sont stockés dans le cache la plupart du temps. Ainsi, les trois boucles internes ont toutes les données nécessaires dans le cache.

Si nous avons en plus un cache de niveau 1 (plus rapide) de 1 ko, l'accès est encore plus local : $2 \times b_\mathrm{s}$ tronçons de colonnes de b_t éléments peuvent être stockés dedans : la boucle la plus interne peut se faire à partir de ce cache.

Nous avons implémenté les algorithmes par blocs dans le cas général (n non multiple de b_s, boucles indépendantes pour les cas aux bords, etc.). Nous avons trouvé le couple $(b_\mathrm{s}, b_\mathrm{t})$ le plus rapide empiriquement.

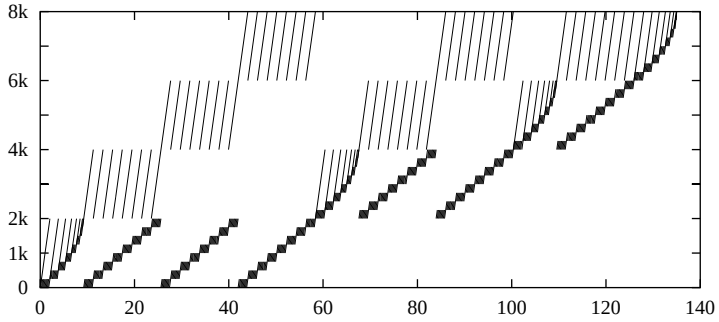
Algorithme naïf

Pour i **de** 1 **à** n
  Pour j **de** i **à** n
    Pour k **de** 1 **à** n
      accès à $im + k$ et $jm + k$



Algorithme bloc 1

Pour i_b **de** 1 **à** n **par** b_s
  Pour j_b **de** i_b **à** n **par** b_s
    Pour i **de** i_b **à** $i_\mathrm{b} + b_\mathrm{s} - 1$
      Pour j **de** $\max(j_\mathrm{b}, i)$ **à** $j_\mathrm{b} + b_\mathrm{s} - 1$
        Pour k **de** 1 **à** n **faire**
          accès à $im + k$ et $jm + k$



Algorithme bloc 2

Pour i_b **de** 1 **à** n **par** b_s
  Pour j_b **de** i_b **à** n **par** b_s
    Pour k_b **de** 1 **à** m **par** b_t
      Pour i **de** i_b **à** $i_\mathrm{b} + b_\mathrm{s} - 1$
        Pour j **de** $\max(j_\mathrm{b}, i)$ **à** $j_\mathrm{b} + b_\mathrm{s} - 1$
          Pour k **de** k_b **à** $k_\mathrm{b} + b_\mathrm{t} - 1$
            accès à $im + k$ et $jm + k$

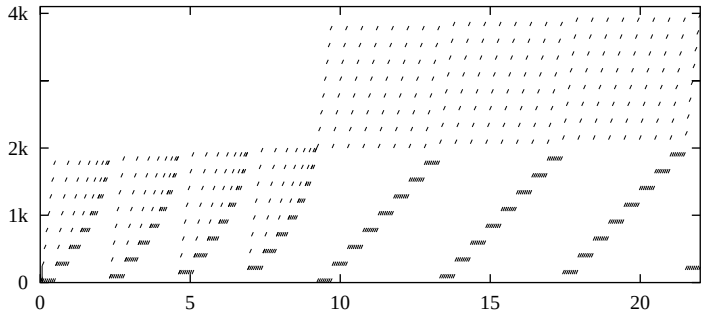


FIG. 7.2: Algorithmes naïf et par blocs et diagrammes d'accès à la mémoire associés. Les algorithmes sont simplifiés pour montrer seulement les opérations de lecture. Pour améliorer la lisibilité, la taille de la matrice est $m \times n = 256 \times 32$, $(b_\mathrm{s}, b_\mathrm{t}) = (8, 64)$. Les diagrammes montrent à quelle adresse (ordonnée, en kilo-octets) les composantes des vecteurs sont lues pour chaque multiplication (abscisse, en milliers).

7.3.3 Instructions vectorielles

Jusqu'ici, nous manipulons des entiers de 8 bits individuellement sur des machines dont les registres font de 32 à 64 bits : quel gâchis !

Nous avons implémenté les produits scalaires des boucles internes sur des unités vectorielles (PowerPC/Altivec, PC/SSE2, UltraSparc/VIS). Ces unités ont des registres de 8 ou 16 octets dont les contenus peuvent être manipulés comme des vecteurs d'entiers ou de ♠

flottants [Dou03].

Les unités vectorielles peuvent être utilisées de quatre manières :

- laisser le compilateur engendrer du code vectoriel. Seul le compilateur de Intel (ICC) sur PC en est capable, sous certaines conditions ;
- exploiter des bibliothèques numériques ou de traitement de signal qui exploitent ces instructions. La fonction proposée applicable à notre cas est le produit scalaire de vecteurs d'entiers (le produit matriciel n'existe qu'en version flottante). Malheureusement, le coût de l'appel d'une fonction à faire pour toutes les paires de lignes est lourd, surtout dans le cas bloc 2 ;
- utiliser des macrocommandes (sans appel de fonction) pour accéder aux instructions vectorielles. C'est possible sur les trois plateformes ;
- écrire en langage d'assemblage (indispensable avec GCC sur PC).

Les données manipulées par les unités vectorielles doivent être alignées naturellement (leur adresse est multiple de 8 ou 16 octets), cette contrainte s'applique aux colonnes des matrices.

7.3.4 Résultats

La table 7.1 présente les résultats pour les trois plateformes. Elles sont comparables parce qu'elles ont été achetées à peu près au même moment (sauf la Sun, qui est nettement plus récente) et dans les mêmes classes de prix. La version bloc 2 déroulé correspond à l'algorithme bloc 2 où les trois boucles internes sont déroulées explicitement (un programme génère le code C déroulé).

Les résultats montrent que le PC est beaucoup plus rapide que le Mac et le Sun. L'unité vectorielle du Sun n'a pas évolué depuis 1995, elle est limitée à 64 bits et partage ses ressources avec l'unité flottante.

L'interface la plus commode pour les instructions vectorielles est sur le Mac : c'est une extension des types de données C. Les données vectorielles peuvent être manipulées (presque) aussi simplement que des données scalaires. L'ensemble des instructions Altivec est aussi le plus riche des trois.

Les calculs sur les très grandes matrices peuvent être améliorés par des approches « diviser pour régner » (en particulier le produit matriciel de Strassen [Raz02, CK00]). Elles permettent de faire moins de multiplications-accumulations quand les matrices sont grandes.

7.4 Étalonnage d'une caméra à focale variable

Nous utilisons une caméra à projection centrale plane pendant la construction de la mosaïque d'images. Les caméras réelles souffrent de distorsions qui ne sont pas prises en compte par ce modèle de projection. Dans la suite, nous corrigeons les distorsions indépendantes de la profondeur qui peuvent s'exprimer comme une déformation de l'image en 2D. Ces distorsions dépendent de la distance focale (facteur de zoom) de la caméra.

plate-forme	Mac	PC		Sun
CPU	G4 866 MHz	Pentium 4, 1.8 GHz		USparc 3 Cu, 900 MHz
Caches (L1/L2/L3)	32+32 k/256 k/2 M	48+8 k/256 k		32+64 k/8 M
registres vectoriels	32 × 128 bits	8 × 128 bits		32 × 64 bits
OS	Mac OS X 10.2.4	Linux 2.4.18		Solaris 9.0
compilateurs	GCC 3.1	GCC 3.2	Intel CC 7.0	Sun Workshop Pro 6.2
scalaire entier	3838	1647	1360	4082
scalaire flottant	7280	1449	1459	2491
bibliothèque	NA	518	518	1401
auto-vectorisé	NA	NA	445	NA
vectoriel	2078	471	445	2783
bloc 2	1666	1443	1083	3349
(b_s, b_t)	(16,256)	(2,32)	(4,32)	(16,256)
bloc 2 vectoriel	542	385	172	1550
(b_s, b_t)	(16,64)	(8,32)	(4,128)	(8,512)
idem, déroulé	365	170	NI	NI
(b_s, b_t)	(4,16)	(4,8)		

TAB. 7.1: Comparaison par algorithme et par plate-forme des temps de calcul de $G^\top G$ (en ms). G est de taille 5000×400 (NA=non applicable, NI=non implémenté).

7.4.1 Le modèle de distorsions

Le modèle le plus simple est donné par les distorsions primaires de Seidel ([Thi94] II.B.3.d). Ce sont des déformations radiales, dues au fait que le système optique ne peut se ramener qu'approximativement à une lentille mince. Pour un point image dont les coordonnées théoriques sont (x_t, y_t), la version déformée est :

$$\begin{cases} x = x_t + dr^2(x_t - x_0) \\ y = y_t + dr^2(y_t - y_0) \end{cases} \quad \text{où } r^2 = (x_t - x_0)^2 + (y_t - y_0)^2,$$

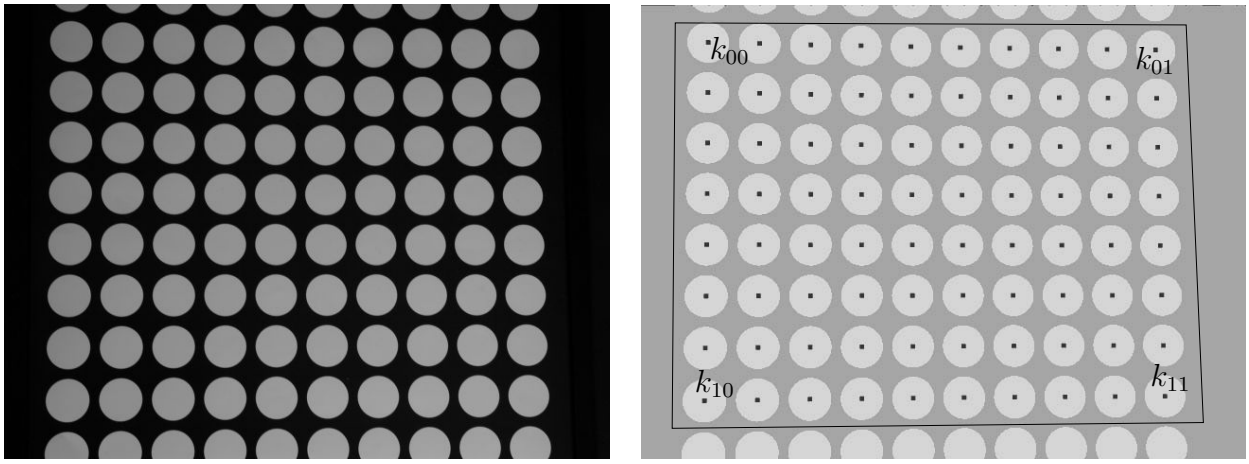
le point $(x_0, y_0) \in \mathbb{R}^2$ est l'origine des distorsions (si le système optique a une symétrie radiale parfaite, l'origine coïncide avec le point principal), et $d \in \mathbb{R}$ est le coefficient de distorsion. Quand $d > 0$ on parle de distorsions en barillet, sinon elles sont « en tonneau ». Ces trois paramètres doivent être estimés.

7.4.2 Contexte expérimental

Les expériences permettant d'estimer les distorsions sont basées sur une mire d'étalonnage plane ([Thi94], II.E.2.a). C'est une image de 100 disques blancs sur un fond noir, organisés en pavage carré (figure 7.3(a)). La mire est imprimée avec une grande précision. Pendant la prise de vues, nous nous assurons que :

– les directions du pavage sont approximativement alignées avec les directions horizontale et verticale de l'image ;

– la caméra voit complètement un sous-ensemble rectangulaire de disques (le **rectangle de référence**). Tous les autres disques sont soit invisibles soit visibles partiellement.



(a) (b)

FIG. 7.3: (a) : image de la mire d'étalonnage utilisée pour estimer les distorsions. (b) : les centres des disques et le rectangle de référence.

7.4.3 Identification des disques

L'estimation des paramètres repose sur la mise en correspondance des coordonnées 3D des centres des disques (dans un repère quelconque) avec leurs coordonnées image. Nous traitons séparément les canaux rouge, vert et bleu de l'image. Les traitements sont donc exposés pour une image en niveaux de gris.

7.4.3.1 Coordonnées sur l'image

Nous extrayons l'histogramme de l'image en calculant un seuil entre le mode blanc (correspondant aux disques) et le mode noir (le fond). Nous utilisons pour cela la méthode d'Otsu, implémentée d'après la fonction `graythresh` de Matlab. Des classes de pixels connexes (au sens du voisinage 4) dont le niveau de gris dépasse le seuil sont construites. Elles sont ensuite sélectionnées si :

– elles sont assez grandes (typiquement plus de 1000 pixels) pour ne pas être du bruit ;
– elles ne touchent pas le bord (sinon, la position du centre serait incertaine).

Chaque groupe correspond à un disque : c'est une ellipse discrétisée. Chaque ellipse fournit un « centre », qui est le barycentre des pixels du groupe pondérés par leurs niveaux de gris.

La sortie de cette étape est un ensemble de coordonnées en pixels (les « centres ») : $K = \{(x_1, y_1), ..., (x_n, y_n)\}$.

7.4.3.2 Coordonnées 3D

Les disques les plus faciles à identifier sont les sommets du rectangle de référence. Nous trouvons leurs numéros ainsi : ♠

$$k_{00} = \underset{k\in[\![1,n]\!]}{\operatorname{argmin}}\ x_k + y_k \qquad k_{01} = \underset{k\in[\![1,n]\!]}{\operatorname{argmin}}\ x_k - y_k$$

$$k_{10} = \underset{k\in[\![1,n]\!]}{\operatorname{argmax}}\ x_k - y_k \qquad k_{11} = \underset{k\in[\![1,n]\!]}{\operatorname{argmax}}\ x_k + y_k$$

En 3D, les centres des disques sont les sommets d'un rectangle, donc ils forment un repère où k_{ij} $(i, j \in \{0, 1\})$ a pour coordonnées $(i, j, 0)$. Nous calculons l'homographie T_0 qui transforme (i, j) en $(x_{k_{ij}}, y_{k_{ij}})$. Cette homographie est une approximation de la fonction de projection de la caméra sans distorsion. Elle permet d'identifier les coordonnées des centres des autres disques.

Le rectangle de référence compte $(n_\mathrm{u}+1)\times(n_\mathrm{v}+1)$ disques. Nous passons à un repère 3D où le disque au sommet k_{ij} a pour coordonnées $(n_\mathrm{u}i, n_\mathrm{v}j, 0)$. Dans ce repère, les centres de tous les disques ont (théoriquement) des coordonnées entières. Si le modèle homographique était parfait, les coordonnées (u, v) du point numéro k vérifieraient :

$$(x_k, y_k) = T_0(u/n_\mathrm{u}, v/n_\mathrm{v}) = T_1(u, v)$$

C'est pour cela que nous calculons les coordonnées 3D idéales (u_k, v_k) par la formule :

$$(u_k, v_k) = E(T_1^{-1}(x_k, y_k))$$

où $E : \mathbb{R}^2 \to \mathbb{Z}^2$ arrondit les éléments d'un vecteur à l'entier le plus proche.

En réalité n_u et n_v sont inconnus. Nous les cherchons en minimisant la distance entre les centres des disques mesurés et leurs coordonnées théoriques (la minimisation peut se faire séparément sur n_u et n_v) :

$$\min_{(n_\mathrm{u},n_\mathrm{v})\in\mathbb{N}^2} \sum_{k=1}^{n} \|E(T_1^{-1}(x_k, y_k)) - T_1^{-1}(x_k, y_k)\|_1$$

La sortie de cette étape est un ensemble de coordonnées correspondantes :

$$L = \{(x_k, y_k, u_k, v_k) \in \mathbb{R}^2 \times \mathbb{Z}^2, k \in [\![1, n]\!]\}$$

7.4.4 Estimation des distorsions

Le modèle de projection à partir du système de coordonnées de la mire que nous utilisons est défini par :

$$f : \begin{array}{ccccc} \mathbb{Z}^2 & \to & \mathbb{R}^2 & \to & \mathbb{R}^2 \\ (u, v) & \mapsto & (x_\mathrm{t}, y_\mathrm{t}) & \mapsto & (x, y) \end{array}$$

La première étape est une homographie issue du modèle de caméra à projection centrale plane, la seconde ajoute les distorsions. Il y a 11 paramètres influant sur ce modèle, 8

pour l'homographie (combinaison de paramètres intrinsèques et extrinsèques) et 3 pour les distorsions (paramètres intrinsèques). Nous estimons ces paramètres à partir des n disques, considérés comme des expériences, ce qui donne la « boîte noire » :

$$(u,v) \in \mathbb{Z}^m \rightarrow \boxed{\begin{array}{c}(\theta, x_0, y_0, d) \in \mathbb{R}^{8+3} \\ \hline m = 2; p = 11; q = 2\end{array}} \rightarrow f(u,v) = (x,y) \in \mathbb{R}^q$$

Les erreurs intervenant dans cette estimation sont :

- les imperfections de la mire, imparfaitement plane ou mal imprimée ;
- les erreurs dues à la discrétisation de l'image ;
- le centre d'un disques ne se projette pas exactement sur le centre de l'ellipse image[1] ;
- d'autres distorsions interviennent.

En revanche, les conditions expérimentales très contrôlées font que le risque de rencontrer des aberrances dues à des appariements erronés est négligeable. Comme les erreurs sont équitablement réparties entre les entrées et les sorties du modèle, nous utilisons une régression en distance orthogonale (§ 2.5.2.2) pour estimer les paramètres.

La résolution du problème est itérative. Nous initialisons les paramètres de la manière suivante :

- les 8 paramètres d'homographie sont initialisés avec h_1^{-1},
- les 3 paramètres de distorsion sont initialisés avec (x_0, y_0) au centre de l'image et $d = 0$.

Les données en entrée et en sortie sont fournies par L et nous utilisons ODRPACK [BBRS92] pour estimer les paramètres. Comme les valeurs initiales sont proches de la solution et le système est surdéterminé, le processus converge vite.

7.4.5 Résultats

Nous avons testé notre procédure sur deux appareils photo digitaux (un Canon Powershot A60 et un Nikon Coolpix 4500). Nous avons fait des prises de vue à différents facteurs de zoom. L'information de distance focale est stockée dans les métadonnées EXIF des fichiers JPEG engendrés par les appareils photo (Cf. `http://www.exif.org`).

La prise en compte des distorsions a permis de réduire l'erreur de positionnement maximale sur les centres des disques de 10 pixels (resp. 7 pixels) à 0.2 pixels sur les images du Nikon (resp. Canon), pour des images de 2272×1704 pixels (resp. 1600×1200).

En guise d'exemple, nous avons affiché le paramètre de distorsion d sur la figure 7.4. Nous avons estimé l'équation d'une hyperbole qui passe par ces points (en enlevant un point aberrant). Le choix de cette conique est empirique.

L'origine (x_0, y_0) des distorsions est à peu près constante sur les trois couleurs primaires.

[1] Jusqu'ici et dans la suite, nous avons supposé que le centre des disques se projette sur le centre de l'ellipse image du disque. Ce n'est le cas, en toute rigueur, que pour les transformations affines (c'est-à-dire quand le plan image de la caméra est parallèle à la mire). Sinon, ce n'est qu'une approximation mais elle est suffisamment bonne pour que nous l'utilisions.

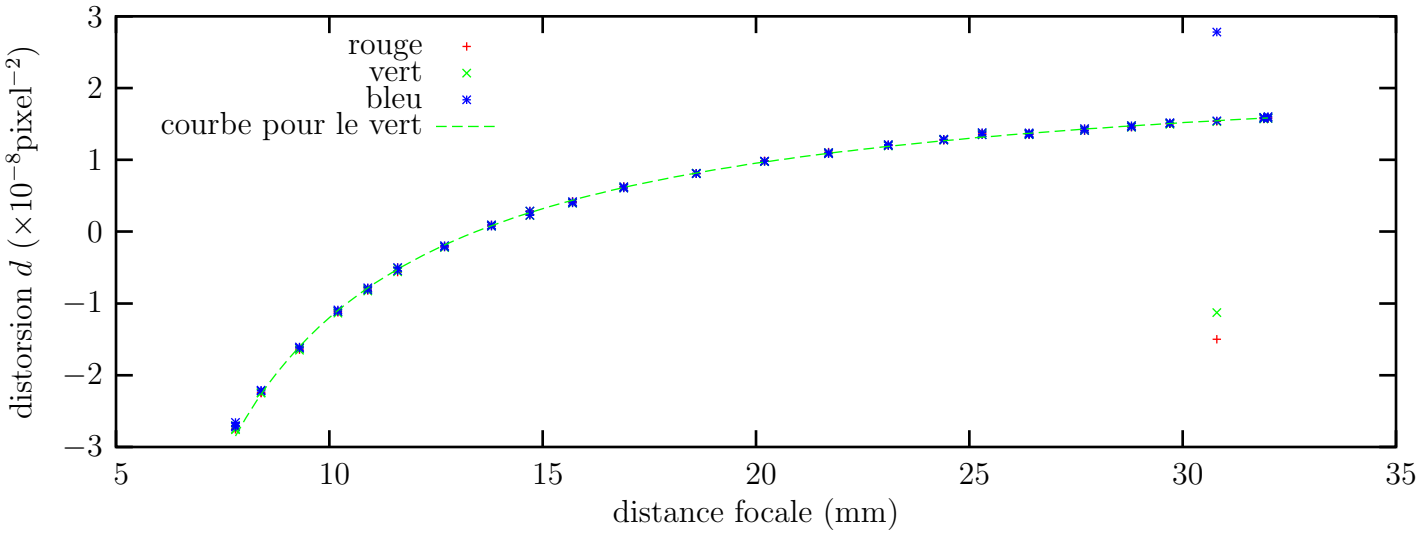


FIG. 7.4: Intensité des distorsions radiales en fonction de la distance focale sur un appareil photo Nikon. La courbe qui passe par les points est une hyperbole.

7.5 Implémentation dédiée de traitements d'images

Les opérations de re-projection et de segmentation peuvent être prises en charge par les cartes d'affichage actuelles, qui ont les qualités suivantes :

- elles sont très rapides, parce que leur processeur a plusieurs unités d'exécution. Elles exploitent ainsi le parallélisme naturel des opérations sur les images ;
- elles peuvent fonctionner en parallèle avec le processeur central.

Nous exploitons ces possibilités en utilisant la bibliothèque OPENGL et son extension IMAGING [SA01].

7.5.1 Description de la bibliothèque

7.5.1.1 Les données

OPENGL peut gère deux types d'images stockées dans la mémoire de la carte vidéo :

les tampons sont les images sur lesquelles on dessine et qui peuvent être affichées (en toute rigueur, ce sont des *color buffers*) ;

les textures sont des images qu'on peut dessiner dans un tampon (en toute rigueur, ce sont des *2D textures*). Lors de cette opération, elles sont dupliquées par pavage rectangulaire et délimitées par un polygone quelconque.

Les images peuvent être transférées de la mémoire centrale vers les deux types d'image (le chemin en sens inverse est moins efficace pour des raisons matérielles).

Les textures et les tampons peuvent être classés selon l'information associée à chaque pixel :

- luminance : à chaque pixel correspond une seule luminance ;
- RGB : trois couleurs primaires (rouge, vert et bleu) ;
- RGBA : les trois couleurs plus une valeur de **alpha**, qui sert généralement pour indiquer une transparence.

Dans notre cas, chaque canal est codé sur 1 octet et les valeurs de pixels considérées sont dans $[0, 1]$. Un octet à 0 (resp. 255) correspond à une valeur sur l'image de 0.0 (resp. 1.0). Si un calcul appliqué sur une image renvoie une valeur $x \notin [0, 1]$, elle est « élaguée », c'est-à-dire remplacée par clamp(x), où la fonction clamp est définie par :

$$\begin{array}{rccl} & \mathbb{R} & \longrightarrow & [0,1] \\ \text{clamp} : & x & \longmapsto & \frac{|x|-|x-1|-1}{2} = \left\{ \begin{array}{ll} 0 & \text{si } x < 0 \\ x & \text{si } 0 \leq x \leq 1 \\ 1 & \text{si } x > 1 \end{array} \right. \end{array}$$

Le fonctionnement de OpenGL est défini par une machine à états. Son état courant comprend en particulier le tampon sélectionné pour l'écriture (souvent celui qui va être affiché) et la texture sélectionnée pour la lecture. Beaucoup d'opérations mentionnées par la suite sont des changements d'état de la machine.

7.5.1.2 Fonctionnalités

Au cours du dessin d'une texture dans un tampon, l'image peut subir les adaptations suivantes :

- elle peut être déformée par une homographie à l'aide d'un échantillonnage brut ou bilinéaire ([SA01] 2.10.4). L'état de la machine comporte une matrice $M_{\text{tex}} \in \mathbb{R}^{4\times4}$ qui rassemble les coefficients de l'homographie (sa taille 4×4 lui permet de gérer aussi les textures en 3D) ;
- elle peut être combinée (opération *blend*) avec le contenu courant du tampon par une somme, une somme pondérée, un min/max, ... ([SA01] 4.1.7). La combinaison est définie par un **mode de combinaison** qui fait partie de l'état de la machine. Nous notons pour un pixel :
  - C la couleur du tampon,
  - S et A la couleur et la valeur alpha de la texture à dessiner.

  Nous indiquons le mode de combinaison par une équation du type :

  $$C \leftarrow (1 - A)C + AS$$

  qui signifie : en chaque pixel, la valeur de C est remplacée par la valeur de cette expression.

Quand une partie d'un tampon est transférée vers une texture 2D ([SA01] 3.8.2), elle peut être modifiée de différentes manières :

- par une convolution, optimisée dans le cas séparable ([SA01] p106) ;
- par une transformation affine appliquée sur ses couleurs ([SA01] p111) ;

7.5.2 Utilisation de la bibliothèque

Nous indiquons ici comment la bibliothèque peut être utilisée pour réaliser les opérations décrites dans les sections 5.2 et 5.4.

7.5.2.1 Superposition de plans

L'opération de superposition de l'équation (5.3) est simple à réaliser si les plans $(I_i, J_i)_{i=1..n}$ sont représentés par des textures RGBA. Le tampon joue le rôle d'accumulateur sur lequel s'empilent les images combinées par une somme pondérée (algorithme 7.3).

sélectionner et effacer le tampon de destination
sélectionner le mode de combinaison $C \leftarrow (1-A)C + AS$
Pour i **de** 1 **à** n **faire**
  Si le plan i doit être re-projeté selon θ **alors**
    $M_{\text{tex}} \leftarrow$`hom44`$(\theta)$
  sinon
    $M_{\text{tex}} \leftarrow \text{Id}_4$
  finsi
  dessiner la texture contenant (I_i, J_i)
finpour

Fonction utilisée : $M =$`hom44`(θ) renvoie une matrice 4×4 qui applique l'homographie de paramètres θ aux textures dessinées :

$$M = \begin{bmatrix} \theta_1 & \theta_2 & 0 & \theta_3 \\ \theta_4 & \theta_5 & 0 & \theta_6 \\ 0 & 0 & 1 & 0 \\ \theta_7 & \theta_8 & 0 & 1 \end{bmatrix}$$

Algorithme 7.3: Superposition de plans $(I_i, J_i)_{i=1..n}$.

7.5.2.2 La différence

Le calcul de la différence, canal par canal, entre deux images I et I' intervient dans l'algorithme 5.1 (fonction `max3`). Il s'agit, pour les vecteurs de couleurs v et $v' \in \mathbb{R}^3$ mesurés en un pixel des deux images, de calculer :

$$w = \begin{bmatrix} |v_1 - v'_1| \\ |v_2 - v'_2| \\ |v_3 - v'_3| \end{bmatrix} \tag{7.3}$$

La bibliothèque n'offre pas l'opération « valeur absolue ». Elle peut seulement calculer $\text{clamp}(v_i - v'_i)$. C'est pour cela que nous ramenons l'équation (7.3) à l'expression :

$$w_i = \text{clamp}(v_i - v'_i) + \text{clamp}(v'_i - v_i) \qquad i = 1..3$$

L'opération d'addition pourrait être remplacée par un max.

Cette expression est calculée par l'algorithme 7.4.

`combiner`$(C \leftarrow S, I)$
`combiner`$(C \leftarrow C - S, I')$
copier le contenu du tampon dans une texture I''
`combiner`$(C \leftarrow S, I')$
`combiner`$(C \leftarrow C - S, I)$
`combiner`$(C \leftarrow C + S, I'')$
le résultat est dans le tampon

Fonction utilisée : la fonction `combiner`(op, I) :

1. sélectionne le mode de combinaison précisé par op ;
2. dessine la texture I

Algorithme 7.4: Calcul de la différence en valeur absolue entre les images I et I'.

7.5.2.3 Le seuillage

Le seuillage intervient dans la fonction `seuillage` de l'algorithme 5.1. Nous proposons une version qui travaille sur une image en couleurs.

La copie d'une image à partir d'un tampon est accompagnée par une transformation affine des couleurs. Pour un pixel dont le triplet de couleurs est v et la valeur alpha α, elle s'écrit :

$$w = \mathrm{clamp}\left(M_\mathrm{c}[v_1 \quad v_2 \quad v_3 \quad \alpha]^\top\right)$$

où :

- la matrice $M_\mathrm{c} \in \mathbb{R}^{3\times 4}$ est définie dans l'état de la machine (par défaut, ce sont les trois premières lignes de l'identité) ;
- la fonction clamp applique l'élagage à toutes les composantes du vecteur.

À cause de leur représentation en mémoire, les valeurs v_i et du seuil s sont telles que $255v_i, 255s \in \mathbb{Z}$. Nous avons donc deux chaînes d'équivalences :

$v_i > s$	$v_i \leq s$
$255(v_i - s) > 0$	$255(v_i - s) \leq 0$
$255(v_i - s) \geq 1$	
$\mathrm{clamp}(255(v_i - s)) = 1$	$\mathrm{clamp}(255(v_i - s)) = 0$

Le calcul de la fonction `seuillage` sur les trois composantes séparément revient à positionner

$$M_\mathrm{c} = 255 \begin{bmatrix} 1 & & & -s \\ & 1 & & -s \\ & & 1 & -s \end{bmatrix}$$

La valeur de α peut être définie comme une constante au lieu d'être lue dans le tampon : $\alpha = 1$

Cependant, il n'est pas réaliste d'avoir un masque différent sur les trois couleurs primaires. Pour calculer un masque commun, avec un seuil $0 \leq s \leq 3$ sur la somme des trois composantes, il suffit d'utiliser la matrice

$$M_{\mathrm{c}} = 255 \begin{bmatrix} 1 & 1 & 1 & -s \\ 1 & 1 & 1 & -s \\ 1 & 1 & 1 & -s \end{bmatrix}$$

7.5.2.4 La dilatation et l'érosion

Définitions. L'**addition de Minkowski** ([SM94] eq(II.6)) de deux ensembles $B \subset \mathbb{Z}^2$ et $B' \subset \mathbb{Z}^2$ est :

$$B \oplus B' = \{\mathbf{m} + \mathbf{m}', (\mathbf{m}, \mathbf{m}') \in B \times B'\}$$

c'est un opérateur commutatif et associatif. L'élément neutre est $\{(0,0)\}$.

Nous utilisons une représentation de l'image sous forme de tableau (§ 7.2.1).

La dilatation d'une image en niveaux de gris I par un ensemble $B \subset \mathbb{Z}^2$ (l'**élément structurant**) est l'image[2] :

$$I \oplus B(\mathbf{m}) = \max\{I(\mathbf{m} + \mathbf{m}'), \mathbf{m}' \in B\}$$

Le niveau de gris des points extérieurs à l'image est supposé égal à $-\infty$, qui est l'élément neutre du max.

L'érosion est définie de la même manière :

$$I \ominus B(\mathbf{m}) = \min\{I(\mathbf{m} + \mathbf{m}'), \mathbf{m}' \in B\}$$

Ici, les points extérieurs sont à $+\infty$.

Nous utilisons la propriété d'« associativité » suivante ([SM94] eq(II.15) et (II.25)) :

$$\begin{aligned} I \oplus (B \oplus B') &= (I \oplus B) \oplus B' \\ I \ominus (B \oplus B') &= (I \ominus B) \ominus B' \end{aligned} \tag{7.4}$$

La « distributivité » s'écrit ([SM94] eq(II.16) et (II.26)) :

$$\begin{aligned} I \oplus (B \cup B') &= \max(I \oplus B, I \oplus B') \\ I \ominus (B \cup B') &= \min(I \ominus B, I \ominus B') \end{aligned} \tag{7.5}$$

où les opérateurs min et max s'appliquent pixel à pixel.

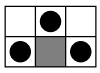
Dans la suite, nous traiterons uniquement l'érosion ; l'extension à la dilatation est triviale. Nous exprimons le coût des algorithmes en fonction de :

- C_{des} : le coût du dessin d'une texture ;
- C_{cp} : le coût de la copie d'une image dans le tampon vers une texture.

[2] Dans [SM94] (eq(II.11)), la dilatation et l'érosion par un élément structurant plan se notent respectivement $I \oplus \check{g}_B$ et $I \ominus \check{g}_B$.

Ces deux opérations prennent, en pratique, sensiblement le même temps.

Nous représentons quelquefois les éléments structurants graphiquement. Par exemple (l'origine est indiquée par une case grise) :

$$M = \text{  } = \{(-1,0),(0,1),(1,0)\}$$

Érosion simple. Un moyen simple permet de calculer l'érosion d'une image I par un élément structurant B : il suffit de dessiner I avec des translations correspondant aux points de B (algorithme 7.5).

L'algorithme coûte $\mathrm{Card}(B)C_{\mathrm{des}}$. Il fonctionne bien si l'ensemble B contient peu d'éléments. C'est le cas pour le voisinage 4, où

$$B = \{(0,0),(-1,0),(1,0),(0,-1),(0,1)\}$$

choisir $\mathbf{m}_0$ dans B
$M_{\mathrm{tex}} \leftarrow$`translation`$(\mathbf{m}_0)$
dessiner I -- inutile si $\mathbf{m}_0 = (0,0)$ et l'image est déjà dans le tampon.
sélectionner le mode de combinaison $C \leftarrow \min(C,S)$
Pour $\mathbf{m} \in B - \{\mathbf{m}_0\}$ **faire**
    $M_{\mathrm{tex}} \leftarrow$`translation`$(\mathbf{m})$
    dessiner I
finpour
l'image $I \ominus B$ est dans le tampon.

Fonction utilisée : `translation`(x,y) calcule une matrice d'homographie qui applique une translation à une image :

$$M = \begin{bmatrix} 1 & & & x \\ & 1 & & y \\ & & 1 & 0 \\ & & & 1 \end{bmatrix}$$

Algorithme 7.5: Calcul de l'érosion d'une image I par un ensemble B.

Érosion séparable. Nous qualifions B de **séparable pour la dilatation** s'il peut être décomposé (de manière non triviale) sous la forme :

$$B = B_1 \oplus \cdots \oplus B_n$$

Dans ce cas, d'après l'équation (7.4), l'érosion par B est **séparable** ; elle vaut :

$$I \ominus B = (\cdots (I \ominus B_1) \ominus \cdots) \ominus B_n$$

L'algorithme 7.6 exploite cette expression pour calculer l'érosion. Il coûte

$$\sum_{i=1}^{n} \mathrm{Card}(B_i) C_{\mathrm{des}} + (n-1) C_{\mathrm{cp}}$$

Cette quantité est en général moins grande que $\mathrm{Card}(B)$, donc l'algorithme est plus rapide que l'érosion simple.

calculer l'érosion de I par B_1 (algorithme 7.5)
Pour i **de** 2 **à** n **faire**
    copier l'image du tampon dans une texture I'
    calculer l'érosion de I' par B_i
finpour
l'image $I \ominus B$ est dans le tampon.

Algorithme 7.6: Calcul de l'érosion d'une image I par un ensemble B séparable.

Applications. L'érosion par un voisinage 8 est séparable :

$$B = [\![-1,1]\!] \times [\![-1,1]\!] = \{(0,-1),(0,0),(0,1)\} \oplus \{(-1,0),(0,0),(1,0)\}$$

Ce cas peut être optimisé : lors de la deuxième érosion, en choisissant $\mathbf{m}_0 = (0,0)$, il n'est pas nécessaire de dessiner l'image. Le coût de l'érosion pour le voisinage 8 est alors $5C_{\mathrm{des}} + C_{\mathrm{cp}}$, ce qui est presque aussi rapide que les $5C_{\mathrm{des}}$ du voisinage 4.

La famille d'éléments structurants $(C_n)_{n\in\mathbb{N}}$ carrés définie par :

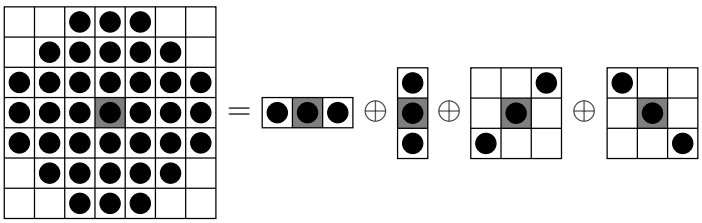
$$C_n = [\![0, 2^n - 1]\!] \times [\![0, 2^n - 1]\!]$$

peut être obtenue récursivement :

$$C_0 = \{(0,0)\} \qquad \text{et} \qquad C_{n+1} = C_n \oplus \{(0,0),(2^n,0),(0,2^n),(2^n,2^n)\}$$

Le coût d'une érosion par C_n, en utilisant la même optimisation que pour le voisinage 8, est $n(C_{\mathrm{cp}} + 3C_{\mathrm{des}})$. C'est une manière peu coûteuse de faire une érosion sur un grand voisinage.

Beaucoup d'approximations de disques simples sont séparables, par exemple :



Érosion composée. Quand l'élément structurant est compliqué, il peut être exprimé par une combinaison d'érosions séparables et d'unions. Cette combinaison peut être représentée par un **graphe de dilatation**. C'est un graphe orienté sans cycle. Les flèches sont étiquetées avec des valeurs de $\mathbb{Z}^2$. Il peut être vu de deux manières, selon qu'il manipule des éléments structurants ou des images. ♠

Vue « élément structurant ». À chaque nœud est associé un élément structurant (sa valeur), qu'il transmet par ses flèches de sortie. Les nœuds sont de deux types :

nœud initial I : il est unique et n'a pas d'entrée. L'élément structurant associé est $\{(0,0)\}$;

nœud de calcul U : si les flèches en entrée convoient les éléments structurants $B_1, ..., B_n$ et portent les étiquettes $\mathbf{m}_1, ..., \mathbf{m}_n$, alors l'élément structurant associé est :

$$B = (B_1 \oplus \{\mathbf{m}_1\}) \cup \cdots \cup (B_n \oplus \{\mathbf{m}_n\})$$

Le résultat du graphe est la valeur du nœud final (il est entouré deux fois).

La figure 7.5 présente un exemple de graphe d'élément structurant calculé par cette méthode.

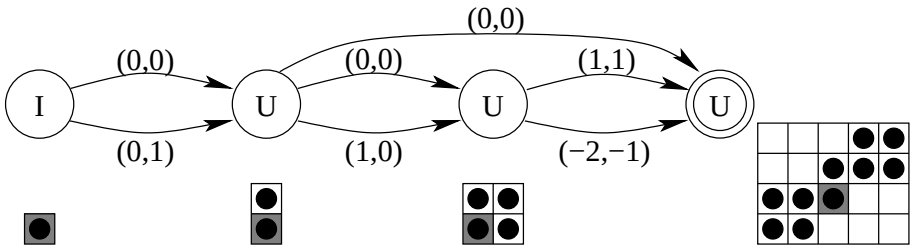


FIG. 7.5: Exemple de graphe de dilatation calculant un élément structurant complexe. Les valeurs des nœuds sont indiquées en-dessous.

Vue « image ». Un graphe de dilatation peut aussi être lu comme traitant des images. La valeur de chaque nœud est une image. Les types de nœuds sont alors :

nœud I : l'image I en entrée.

nœud U : si les flèches en entrée convoient les images $I_1, ..., I_n$ et portent les étiquettes $\mathbf{m}_1, ..., \mathbf{m}_n$, alors l'image associée au nœud est :

$$I_{\text{nd}} = \min\{I_1 \oplus \{\mathbf{m}_1\}, ..., I_n \oplus \{\mathbf{m}_n\}\} \tag{7.6}$$

où l'opérateur min s'applique pixel à pixel.

Intérêt pour le calcul. Les deux sémantiques du graphe de dilatation sont liées : pour chaque nœud, la valeur B_{nd} au sens des éléments structurants et la valeur I_{nd} au sens des images vérifient :

$$I_{\text{nd}} = I \ominus B_{\text{nd}}$$

D'un point de vue informatique, le graphe décrit un programme qui calcule l'érosion de l'image I par l'élément structurant du nœud final. La compilation de ce programme consiste à :

1. simplifier le graphe : supprimer les branches mortes, fusionner les sous-expressions communes, simplifier les expressions, etc. ;
2. déterminer l'ordre des instructions ;
3. allouer des registres aux valeurs. Ici, les registres sont des textures, l'accumulateur est le tampon ;
4. traduire les opérations en langage machine. L'opération U consiste à dessiner les images en entrée dans le tampon (algorithme 7.7).

$M_{\text{tex}} \leftarrow$`translation`$(\mathbf{m}_1)$
dessiner I_1 `--` optimisation : inutile si $\mathbf{m}_1 = (0,0)$ et I_1 est déjà dans le tampon.
sélectionner le mode de combinaison $C \leftarrow \min(C, S)$
Pour i **de** 2 **à** n **faire**
    $M_{\text{tex}} \leftarrow$`translation`$(\mathbf{m}_i)$
    dessiner I_i
finpour
le résultat est dans le tampon

Fonction utilisée : `translation` est définie dans l'algorithme 7.5

Algorithme 7.7: Calcul de l'image I_{nd} correspondant à un nœud U à partir des images en entrée $I_1, ..., I_n$ (représentées par des textures) et les étiquettes associées $\mathbf{m}_1, ..., \mathbf{m}_n$ (équation (7.6)).

Le graphe de dilatation est une généralisation des deux cas précédents : l'érosion simple s'exprime comme un nœud U et l'érosion séparable comme une chaîne de nœuds U.

Exemple. Cet algorithme est utile pour former des les éléments structurants complexes, notamment si ils contiennent des trous. La figure 7.6 en est un exemple.

Trouver le graphe de dilatation qui mène à une forme donnée avec le moins possible d'opérations est un problème d'exploration d'arbre. Sa résolution en un temps raisonnable requiert une heuristique d'exploration et des règles pour éliminer le plus vite possible les branches sans intérêt.

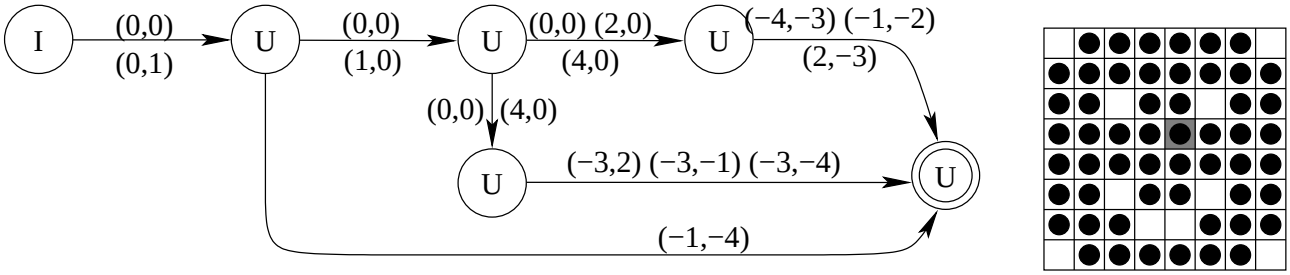


FIG. 7.6: Élément structurant complexe nécessitant un graphe de dilatation pour sa réalisation. Si plusieurs flèches relient le même nœud, elles sont fusionnées en une.

7.5.3 Conclusion

Les possibilités de l'extension IMAGING sont ici exploitées au mieux. Cependant, ces dernières années, les cartes graphiques suivent une évolution qui rend cette approche obsolète : leurs processeurs deviennent de plus en plus « généralistes ». Aucune bibliothèque ne peut plus donner, en quelques fonctions habilement paramétrées, accès à toutes leurs possibilités.

Les nouvelles interfaces de cartes vidéo proposent des langages de programmation pour décrire les opérations à appliquer pour calculer la couleur d'un pixel dans le tampon. Ces langages ressemblaient à de l'assembleur (extension *fragment program*) mais deviennent de plus en plus haut niveau (tels le langage Cg de NVIDIA [nVi04] et le *GL shading language* [KBR04]). L'algorithme 7.4 pourrait particulièrement en bénéficier : il requiert 6 parcours des images, alors que l'opération à réaliser est une simple opération pixel à pixel.

7.6 Notations

Nous précisons ici les notations les plus inhabituelles utilisées dans le document. Les exposants sont toujours utilisés pour les puissances (et non pour des indices). Le séparateur décimal est le point. Les termes en anglais (ou d'autres langues étrangères) sont en *slanted*. Les renvois à d'autres parties de ce document sont toujours précédés du § (le signe paragraphe), même s'il s'agit de sections, de sous-sections, etc.

Nous notons :

$\mathbb{R}, \mathbb{Z}, \mathbb{N}$	respectivement l'ensemble des réels, des entiers et des entiers positifs
$C^{m\times n}$	l'ensemble des matrices de m lignes et n colonnes sur le corps C. Nous notons les matrices entre crochets : $\begin{bmatrix} 1 & \sqrt{2} & 1 \\ & & 2/5 \end{bmatrix} \in \mathbb{R}^{2\times 3}$ Les cases vides contiennent 0.
$\|.\|$	la norme 2 pour les vecteurs et la norme de Frobénius pour les matrices.

A^+	la pseudo-inverse de la matrice A.
$0_{m,n}$	la matrice nulle de m lignes et n colonnes.
Id_n	la matrice identité de taille $n \times n$, et l'application linéaire associée.
$\mathrm{diag}(a_1, ..., a_n)$	la matrice carrée diagonale d'éléments $a_1, ..., a_n$.
$A^\top$	la transposée de la matrice A.
A_{ij}	l'élément à la ligne i et la colonne j de la matrice A.
$A_{i_1:i_2,j_1:j_2}$	le sous-bloc de la matrice défini par les coordonnées $(i,j) \in [\![i_1, i_2]\!] \times [\![j_1, j_2]\!]$.
$\lfloor a \rfloor$ et $\lceil a \rceil$	les arrondis du réel a par défaut et par excès, respectivement
$\mathrm{Ker}(A)$ et $\mathrm{Im}(A)$	le noyau et l'image de l'application définie par la matrice A
$[A \quad B]$	la concaténation des matrices A et B qui ont le même nombre de lignes
$X \sim \mathcal{L}$	pour « la variable aléatoire X suit la distribution $\mathcal{L}$ »
$x \sim \mathcal{L}$	pour « la valeur x est tirée aléatoirement suivant la distribution $\mathcal{L}$ »
$\mathcal{U}(a, b)$	la distribution uniforme entre a et b
$\mathcal{N}(\mu, \sigma^2)$	la distribution normale de moyenne μ et de variance σ^2
$\mathcal{N}(\mu, \Sigma^2)$	la distribution vectorielle normale de moyenne μ et de matrice de variance-covariance Σ^2
$\widetilde{x}$ et $\widehat{x}$	si on veut insister sur le fait que la valeur x est mesurée (tilde) ou estimée (chapeau)
$E[X]$	l'espérance de la variable aléatoire X
$\arg(x, y)$	l'angle dans $[-\pi, \pi[$ entre l'axe horizontal et la droite passant par l'origine et (x, y)
$\mathrm{Card}(A)$	le nombre d'éléments de l'ensemble A.
f'	est la dérivée de la fonction f. Si $f : \mathbb{R}^n \to \mathbb{R}^m$ alors $f'(x)$ est identifiable à sa matrice jacobienne, d'où $$f' : \mathbb{R}^n \to \mathbb{R}^{m \times n}$$ On note quelquefois $$f'(x_0) = \frac{\partial f(x)}{\partial x}(x_0)$$
$[\![n, m]\!]$	l'ensemble des entiers de n à m inclus : $[\![n, m]\!] = [n, m] \cap \mathbb{Z}$. Nous écrivons quelquefois $i = n..m$ pour $i \in [\![n, m]\!]$.
$A - B$	la différence ensembliste : $A - B = \{a \in A / a \notin B\}$
$\begin{cases} \max f(x) \\ x \in A \\ g(x) \in B \end{cases}$	le problème qui consiste à optimiser l'expression dans la première ligne sur les variables mentionnées à la seconde, soumises aux contraintes de la troisième.

$\left[E_i\right]_{i=1..n}$ la matrice dont les lignes sont définies par les vecteurs lignes E_i :

$$\left[E_i\right]_{i=1..n} = \begin{bmatrix} E_1 \\ \vdots \\ E_n \end{bmatrix}$$

Bibliographie

[AA89] Arslane Abi-Ayad. *Calibrage statique et dynamique de caméras. Application à la manipulation d'objets polyédriques par un robot sous le contrôle d'une tête de vision stéréoscopique*. PhD thesis, INP, Toulouse, 1989.

[ABB+01] Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *Computer Graphics and Applications*, 21(6) :34–47, 2001.

[ABC92] Anestis Antoniadis, Jacques Berruyer, and René Carmona. *Régression non linéaire et applications*. Economica, Paris, 1992.

[BA96] Michael J. Black and P. Anandan. The robust estimation of multiple motions : parametric and piecewise-smooth flow fields. *CVIU*, 63(1) :75–104, 1996.

[BAHH92] James R. Bergen, P. Anandan, Keith J. Hanna, and Rajesh Hingorani. Hierarchical model-based motion estimation. In *Proceedings of ECCV*, pages 237–252. Springer-Verlag, may 1992.

[BB01] Matthew Brand and Rahul Bhotika. Flexible flow for 3d nonrigid tracking and shape recovery. In *IEEE Computer Society Conference on Computer vision and Pattern Recognition (CVPR)*, volume 1, pages 315–322, december 2001.

[BBRS92] Paul T. Boggs, Richard H. Byrd, Janet E. Rogers, and Robert B. Schnabel. User's reference guide for odrpack version 2.01 (software for weighted orthogonal distance regression. Technical Report NISTIR 92-4834, US Department of Commerce, Gaithersburg, MD, june 1992.

[BBS87] Paul T. Boggs, Richard H. Byrd, and Robert B. Schnabel. A stable and efficient algorithm for nonlinear orthogonal distance regression. *SIAM Journal on Scientific and statistical computing*, 8(6) :1052–1078, 1987.

[BDH03] Adrien Bartoli, Naveet Dalal, and Radu Horaud. Motion panoramas. Technical Report RR–4771, INRIA, March 2003.

[Bjö96] Åke Björck. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia (PA), 1996.

[BJ98] Michael J. Black and Allan D. Jepson. Eigentracking : robust matching and tracking of articulated objects using a view-based representation. *IJCV*, 26(1) :63–84, 1998.

[Bjö02] Åke Björck. *Recent Advances in Total Least Squares Techniques and Errors-in-Variables Modeling : Analysis, Algorithms and Applications (Sabine van Huffel and P. Lemmerling eds.)*, chapter QR factorization of the Jacobian in some structured nonlinear least squares problems, pages 225–234. Kluwer Academic Publishers, 2002.

[BK01] Ryad Benosman and Sing Bing Kang, editors. *Panoramic Vision, Sensors, Theory and Applications*. Springer, New York, 2001.

[BL02] Matthew Brown and David G. Lowe. Invariant features from interest point groups. In *proceedings of the British Machine Vision Conference*, 2002.

[BL03] Matthew Brown and David G. Lowe. Recognising panoramas. In *proceedings of ICCV*, 2003.

[Bla98] Jérôme Blanc. *Synthèse de nouvelles vues d'une scène 3D à partir d'images existantes*. PhD thesis, INP, Grenoble, janvier 1998.

[BN01] S. Baker and S. K. Nayar. *Panoramic Vision, Sensors, Theory and Applications (Ryad Benosman and Sing Bing Kang editors)*, chapter Single Viewpoint Catadioptric Cameras, pages 39–71. Springer, New York, 2001.

[BW88] D.M. Bates and D.G. Watts. *Nonlinear Regression Analysis and Its Applications*. John Wiley & Sons, 1988.

[BY97] Jesse L. Barlow and Peter A. Yoon. *Recent Advances in Total Least Squares Techniques and Errors-in-Variables Modeling (Sabine van Huffel ed.)*, chapter Solving Recursive TLS problems using the rank-revealing ULV decomposition, pages 117–126. SIAM, Philadelphia, PA, 1997.

[BYJF97] Michael J. Black, Y. Yacoob, Allan D. Jepson, and D. J. Fleet. Learning parameterized models of image motion. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 561–567, june 1997.

[BYZ96] Jesse L. Barlow, Peter A. Yoon, and Hongyuan Zha. An algorithm and a stability theory for downdating the ULV decomposition. *BIT Numerical Mathematics*, 39 :15–40, 1996.

[CC03] Ariel Choukroun and Vincent Charvillat. Bucketing techniques in robust regression for computer vision. In *Scandinavian Conference on Image Analysis*, pages 609–616, 2003.

[Cha97] Vincent Charvillat. *Stéréovision incrémentale appliquée à la modélisation tridimensionnelle d'objets (de l'intérêt de méthodes de régression robuste dédiées)*. PhD thesis, INP, Toulouse, 1997.

[Cia85] P. G. Ciarlet. *Introduction à l'analyse numérique matricielle et à l'optimisation*. Masson, 1985.

[CK00] Richard Crandall and Jason Klivington. Fast matrix algebra on apple g4. Technical report, Apple, Feb 2000.

[CL03] Robert Collins and Yanxi Liu. On-line selection of discriminative tracking features. Technical Report CMU-RI-TR-03-12, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, April 2003.

[CR88] R.J. Carrol and D. Ruppert. *Transformation and Weighting in Regression*. CRC Press, 1988.

[CRM00] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *proceedings of CVPR*, volume 2, pages 142–149, Hilton Head Island, South Carolina, 2000.

[CW82] R. Dennis Cook and Sanford Weisberg. *Residuals and Influence in Regression*. Chapman & Hall,, 1982.

[DAB00] Michel Daydé, Partick Amestoy, and Philippe Berger. Conception des logiciels numériques pour calculateurs haute-performance. Technical report, ENSEEIHT, Toulouse, 2000.

[DC03] Matthijs Douze and Vincent Charvillat. Real-time tracking of video sequences in a panoramic view for object-based video coding. In *Scandinavian Conference on Image Analysis (SCIA 2001)*, Göteborg, Sweden, july 2003.

[DCT01] Matthijs Douze, Vincent Charvillat, and Bernard Thiesse. Mosaïques d'images par approximations successives. In *Journées Francophones des Jeunes Chercheurs en Vision par Ordinateur - ORASIS'2001, Cahors*, 2001.

[DCT02] Matthijs Douze, Vincent Charvillat, and Bernard Thiesse. Des mosaïques plus universelles, plus précises. In *Reconnaissances des Formes et Intelligence Artificelle (RFIA'02)*, volume 2, pages 597–606, january 2002.

[DCT03] Matthijs Douze, Vincent Charvillat, and Bernard Thiesse. Comparaison et intégration de trois algorithmes de suivi de motifs plans. In *Journées Francophones des Jeunes Chercheurs en Vision par Ordinateur - ORASIS'2003, Gérardmer, France*. LORIA, INRIA-Lorraine, May 2003.

[DCT04] Matthijs Douze, Vincent Charvillat, and Bernard Thiesse. Plus de robustesse et de précision dans le suivi temps réel de motifs plans. In *Reconaissances des Formes et Intelligence Artificelle (RFIA'04)*, Toulouse, january 2004.

[DDCM04] Christophe Dehais, Matthijs Douze, Vincent Charvillat, and Géraldine Morin. Augmented reality through real-time tracking of video sequences using a panoramic view. In *Proceedings of ICPR*, Cambridge, UK, august 2004.

[DHS01] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley-Interscience, New York, second edition, 2001.

[DM96] Frédéric Dufaux and Fabrice Moscheni. Background mosaicking for low-bitrate video coding. In *proceedings of ICIP*, volume 1, pages 673–676, Lausanne, Switzerland, september 1996.

[Dou03] Matthijs Douze. Accélération simd d'un calcul matriciel : comparaison de trois plateformes. In *Colloque des doctorants toulousains*, 2003.

[DPCC03] Matthijs Douze, Philippe Puech, Vincent Charvillat, and Jean Conter. Suivi temps-réel de séquences vidéo dans un panoramique pour le codage par objets. In *Compression et représentation des signaux audiovisuels (Coresa 2003)*, january 2003.

[DS83] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice Hall, Englewood Cliffs (NJ), 1983.

[ECT98] G. J. Edwards, T. F. Cootes, and C. J. Taylor. Face recognition using active appearance models. In *European Conference on Computer Vision*, pages 581–695, 1998.

[EP94a] Lars Eldén and Haesun Park. Block downdating of least squares solutions. *SIAM Journal on Matrix Analysis and Applications*, 15(3) :1018–1034, 1994.

[EP94b] Lars Eldén and Haesun Park. Perturbation analysis for block downdating of a Cholesky decomposition. *Numerische Mathematik*, 68 :457–467, 1994.

[Far01] G. Farnebäck. Very high accuracy velocity estimation using orientation tensors, parametric motion, and simultaneous segmentation of the motion field. In *Proceedings of the Eighth IEEE International Conference on Computer Vision*, volume 1, pages 171–177, Vancouver, Canada, July 2001.

[Fau98] Olivier Faugeras. De la géométrie au calcul variationnel : théorie et application de la vision tridimensionnelle. In *Conférence invitée de RFIA*, pages 15–34, 1998.

[FBJ98] David J. Fleet, Michael J. Black, and Allan D. Jepson. Motion feature detection using steerable flow fields. In *IEEE Computer Society Conference on Computer vision and Pattern Recognition (CVPR)*, pages 274–281, june 1998.

[FHH90] Ricardo D. Fierro, Per Christian Hansen, and Peter Søren Kirk Hansen. UTV Tools : Matlab templates for rank-revealing UTV decompositions. *Numerical Algorithms*, 20 :165–194, March 1990.

[Fle00] Roger Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 2nd edition, 2000.

[FLP01] O. Faugeras, Q.T. Luong, and T. Papadopoulo. *The geometry of multiple images*. MIT Press, 2001.

[Ful87] W.A. Fuller. *Measurement Error Models*. John Wiley & Sons, 1987.

[GCD01a] Romulus Grigoraş, Vincent Charvillat, and Matthijs Douze. Contenus multimedia à mémoire. In *Compression et représentation des signaux audiovisuels (Coresa 2001)*, Dijon, novembre 2001.

[GCD01b] Romulus Grigoraş, Vincent Charvillat, and Matthijs Douze. Self-adaptive multimedia content. In *EURASIP Conference on Multimedia Communications and Services (EURASIP '01)*, Budapest, september 2001.

[GCD02] Romulus Grigoraş, Vincent Charvillat, and Matthijs Douze. Optimizing hypervideo navigation using a markov decision process approach. In *ACM Multimedia 2002*, Juan-les-pins, december 2002.

[GCHH03] Simon Gibson, Jon Cook, Toby Howard, and Roger Hubbold. Rapid shadow generation in real-world lighting environments. In *Proceedings of the 14th Eurographics workshop on Rendering*, pages 219–229. Eurographics Association, 2003.

[Gib04] Simon Gibson. Illumination capture and rendering for augmented reality. In *Workshop de RFIA*, 2004.

[GL80] Gene H. Golub and Charles F. Van Loan. An analysis of the total least squares problem. *SIAM Journal of Numerical Analysis*, 17 :883–893, 1980.

[GL87] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, 1987.

[GL91] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. John Hopkins University Press, Baltimore and London, second edition, 1991.

[GMW81] P.E. Gill, W. Murray, and M.H. Wright. *Practical Optimization*. Academic Press, 1981.

[Gol89] David Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[GOM01] Frank Galpin, Lionel Oisel, and Luce Morin. Génération d'un modèle vrml texturé à partir de séquences non calibrées. In *Journées Francophones des Jeunes Chercheurs en Vision par Ordinateur - ORASIS'2001, Cahors*, 2001.

[GR70] Gene H. Golub and C. Reinsch. Singular value decomposition and least squares solutions. *Numerical Mathematics*, 14 :403–420, 1970.

[Gri03] Romulus Grigoraş. *Supervision de flux pour les contenus hypermédia : optimisation de politiques de préchargement et ordonnancement causal*. PhD thesis, INP, Toulouse, 2003.

[HB98] Gregroy D. Hager and Peter N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE pami*, 20(10) :1025–1039, 1998.

[HJ85] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[Hor86] P. Horn. *Robot Vision*. MIT Press, Cambridge, Massachusetts, 1986.

[Hou74] A. S. Householder. *The Theory of Matrices in Numerical Analysis*. Dover Publications, New York, 1974.

[HT51] H.W.Kuhn and A.W. Tucker. Non linear programming. In *Proceedings of the second Berkeley symposium on mathematical statistics and probability (J. Neyman ed.)*. University of California Press, 1951.

[Hub81] Peter J. Huber. *Robust Statistics*. Wiley Series in Probability and Mathematical Statistics, 1981.

[Huf97] Sabine Van Huffel, editor. *Recent Advances in Total Least Squares Techniques and Errors-in-Variables Modeling : Analysis, Algorithms and Applications*. SIAM, 1997.

[HV87] Sabine Van Huffel and Joos Vanderwalle. An efficient and reliable algorithm for computing the singular subspace of a matrix, associated with it smallest singular values. *Journal of Computational and Applied Mathematics*, 19 :313–330, 1987.

[HV91] Sabin Van Huffel and Joos Vanderwalle. The total least squares problem : Computational aspects and analysis. *SIAM*, 1991.

[HZ93] Sabine Van Huffel and Hongyuan Zha. An efficient total least squares algorithm based on a rank-revealing two-sided orthogonal decomposition. *Numerical Algorithms*, 4 :101–133, 1993.

[HZ01] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, London, first edition, 2001.

[HZ03] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, London, second edition, 2003.

[IAB+96] Michal Irani, P. Anandan, J. R. Bergen, Rakesh Kumar, and Steve Hsu. Efficient representation of video sequences and their applications. *Signal Processing : Image Communication*, 8 :327–351, may 1996.

[IB96] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *proceedings of ECCV*, pages 343–356, Cambridge, UK, 1996.

[IRP94] Michal Irani, Benny Rousso, and Shmuel Peleg. Computing occluding and transparent motions. *IJCV*, 12(1) :5–16, 1994.

[JD02a] Frédéric Jurie and Michel Dhome. Hyperplane approximation for template matching. *IEEE PAMI*, 24(7) :996–1000, 2002.

[JD02b] Frédéric Jurie and Michel Dhome. Un algorithme de "template matching" simple et efficace. In *Actes de RFIA*, pages 59–65, 2002.

[JLL94] G. Giraudon J.-L. Lotti. Adaptative window algorithm for aerial image stereo. In *Proceedings of the 12th conference on pattern recognition*, 1994.

[Kan95] Kenichi Kanatani. *Geometric Computation for Machine Vision*. Oxford University Press, Oxford, 1995.

[Kan96] Kenichi Kanatani. *Statistical Optimization for Geometric Computation : Theory and Practice*. Elsevier Science, Amsterdam, 1996.

[Kan04] Kenichi Kanatani. Uncertainty modeling and model selection for geometric inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26 :1307–1319, October 2004.

[Kar39] W. Karush. Minima of functions of several variables with inequalities as side conditions. Master's thesis, University of Chicago, Chicago, 1939.

[KBR04] John Kessenich, Dave Baldwin, and Randi Rost. *The OpenGL Shading Language*. 3Dlabs, April 2004.

[KK04] Yasushi Kanazawa and Kenichi Kanatani. Image mosaicing by stratified matching. *Image and Vision Computing*, 22 :93–103, 2004.

[KWT88] M. Kass, A. Witkin, and D. Terzopoulos. Snakes : Active contour models. *IJCV*, 1 :321–332, 1988.

[Lan97] Z.-D. Lan. *Méthodes robustes en vision : applications aux appariements visuels*. PhD thesis, INP, Grenoble, 1997.

[Lar02] Larousse, editor. *Le Petit Larousse*. Bordas, Paris, 2002.

[LB00] Vincent Lepetit and Marie-Odile Berger. A semi automatic method for resolving occlusions in augmented reality. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, june 2000.

[Lev44] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quaterly Applied Mathematics*, 2 :164–168, july 1944.

[LH74] C.L. Lawson and R.J. Hanson. *Solving Least Squares Problems*. Prentice Hall, 1974.

[Lhu00] Maxime Lhuillier. *Modélisation pour la sythèse d'images à partir d'images*. PhD thesis, INP, Grenoble, 2000.

[LJDC04] J. T. Lapresté, F. Jurie, M. Dhome, and F. Chaumette. Nouvelle approche pour le calcul du jacobien inverse en asservissement visuel 2d. In *RFIA*, pages xx–yy, january 2004.

[LKF+02] Qiong Liu, Don Kimber, Jonathan Foote, Lynn Wilcox, and John Boreczky. Flyspec : A multi-user video camera system with hybrid human and automatic control. In *proceedings of ACM Multimedia*, Juan-les-Pins, december 2002.

[LR03] C.M. Laguna and R.Marti. *Scatter Search : Methodology and Implementations*. Kluwer Academic Publishers, 2003.

[Lue69] David G. Luenberger. *Optimization by Vector Space Methods*. John Wiley & Sons, 1969.

[LVTF03] Vincent Lepetit, Luca Vacchetti, Daniel Thalmann, and Pascal Fua. Fully automated and stable registration for augmented reality applications. In *Proceedings of ISMAR*, Tokyo, Japan, september 2003.

[Mar63] Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *J. Soc.Indust. Appl. Math.*, 11(2) :431–441, june 1963.

[Mar82] David Marr. *Vision*. Freeman & Co., New York, 1982.

[Mar93] Fernando Martinez. *MMT, un système de stéréovision anthropomorphe pour le calcul de cartes denses de profondeur*. PhD thesis, INP, Toulouse, 1993.

[Mas04] Guillaume Masserey. Virtualité augmentée : un forme de système mixte – caractérisation et mise en œuvre dans les arts plastiques. Master's thesis, Université Paul Sabatier, Toulouse, 2004.

[MDJ04] Lucie Masson, Michel Dhome, and Frédéric Jurie. Robust real time tracking of 3d objects. In *Proceedings of ICPR*, Cambridge, UK, august 2004.

[Mee01] Peter Meer. Robust techniques for computer vision (a tutorial). Technical report, Rutgers Univesity, New Jersey, 2001.

[Mee04] Peter Meer. *Emerging Topics in Computer Vision (Gerald Medioni and Sing Bing Kang eds.)*, chapter Robust Techniques for Computer Vision. Prentice Hall, 2004.

[MGH80] Jorge J. Moré, Burton S. Garbow, and Kenneth E. Hillstrom. User Guide for MINPACK-1. Technical Report ANL–80–74, Argonne National Laboratory, 97000 south cass avenue, Argonne, Illinois 60439, August 1980.

[MJ02] Rémi Mégret and Jean-Michel Jolion. Tracking scale-space blobs for video description. *IEEE Multimedia*, 9(2) :34–43, 2002.

[MJD03] Lucie Masson, Frédéric Jurie, and Michel Dhome. Suivi de motifs texturés : approche hybride texture/contours. In *Actes de ORASIS*, pages 231–238, 2003.

[MK01] C. Matsunaga and K. Kanatani. Calibration of a moving camera using a planar pattern : Optimal computation, reliability evaluation and stabilization by the geometric aic. *Electronics and Communications in Japan, Part 3 : Fundamental Electronic Science*, 84(7) :12–21, 2001.

[Mor78] Jorge Moré. The levenberg-marquardt algorithm : implementation and theory. In *Numerical Analysis. Proceedings Biennal Conference, Dundee*, volume 630, pages 105–116. Springer Verlag, 1978.

[MS01] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *International Conference on Computer Vision*, pages 525–531, july 2001.

[NLJD02] Éric Noirfalise, Jean-Thierry Lapresté, Frédéric Jurie, and Michel Dhome. Real-time registration for image mosaicing. In *Proceedings of the British Machine Vision Conference 2002, BMVC 2002, Cardiff, UK, 2-5 September 2002*. British Machine Vision Association, 2002.

[nVi04] nVidia, editor. *Cg toolkit user's manual. A Developer's Guide to programmable Grpahics*. nVidia, January 2004.

[Nye92] Adrian Nye. *Xlib Programming Manual volume one*. O'Reilly & Associates, Inc., 1992.

[PE02] Fernando Pereira and Touradj Ebrahimi, editors. *The MPEG-4 Book*. Pearson Education, Upper Saddle River, NJ, 2002.

[PH95] Haesun Park and Sabine Van Huffel. Two-way bidiagonalization scheme for downdating the singular value decomposition. *Linear Algebra and Its Applications*, 222 :23–39, 1995.

[PrM04] Muriel Pressigout and Éric Marchand. Model-free augmented reality by virtual visual servoing. In *Proceedings of ICPR*, Cambridge, UK, august 2004.

[PSc01] T. Pajdla, T. Svoboda, and V. Hlaváč. *Panoramic Vision, Sensors, Theory and Applications (Ryad Benosman and Sing Bing Kang editors)*, chapter Epipolar Geometry of Central Panoramic Catadioptric Cameras, pages 73–102. Springer, New York, 2001.

[Rat83] D.A. Ratkovsky. *Nonlinear regression modeling*. Marcel Dekker, 1983.

[Raz02] Ran Raz. On the complexity of matrix product. In *Proceedings of the thiryfourth annual ACM symposium on Theory of computing*, pages 144–151. ACM Press, 2002.

[RD99] P. J. Rousseeuw and K. Van Driessen. Computing lts for large data sets. Technical report, University of Antwerp, 1999.

[RDO01] Edmond Ramis, Claude Deschamps, and Jacques Odoux. *Cours de mathématiques*, volume 2. Dunod, 2001.

[RL87] P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley & sons, 1987.

[Rob03] Paul Robert, editor. *Petit Robert*. Dictionnaires le Robert, Paris, 2003.

[SA96] H. S. Sawhney and S. Ayer. Compact representation of videos through dominant and multiple motion estimation. *IEEE pami*, 18(8) :814–830, 1996.

[SA01] Mark Segal and Kurt Akeley. *The OpenGL Graphics System : A Specification*. Silicon Graphics, 1.3 edition, 2001.

[Sau72] Michael A. Saunders. Large scale linear programming using the Cholesky factorization. Technical Report CS 252, Stanford University Computer Science Department, 1972.

[SB02] Gilles Simon and Marie-Odile Berger. Reconstructing while registering : a novel approach for markerless augmented reality. In *International Symposium on Mixed and Augmented Reality - ISMAR'02, Darmstadt, Germany*, Sep 2002.

[SHK98] H. S. Shawhney, S. Hsu, and R. Kumar. Robust video mosaicing through topology inference and local to global alignment. In *proceedings of ECCV*, 1998.

[SK04] Yasuyuki Sugaya and Kenichi Kanatani. Extracting moving objects from a moving camera video sequence. to be presented in the Symposium on Sensing via Image Information, june 2004.

[SM94] Michel Schmitt and Juliette Mattioli. *Morphologie mathématique*. Masson, 1994.

[SMB00] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors. *IJCV*, 37(2) :151–172, 2000.

[SS97] H.-Y. Shum and Richard Szeliski. Panoramic image mosaicing. Technical Report MSR-TR-97-23, Microsoft Research, September 1997.

[SS01] H.-Y. Shum and Richard Szeliski. *Panoramic Vision, Sensors, Theory and Applications (Ryad Benosman and Sing Bing Kang editors)*, chapter Construction

of panoramic image mosaics with global and local alignment, pages 227–268. Springer, New York, 2001.

[ST85] Hubert Schwetlick and Volker Tiller. Numerical methods for estimating parameters in nonlinear models with errors. *Technometrics*, 27(1) :17–24, february 1985.

[ST94] Jiambo Shi and Carlo Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593–600, 1994.

[SVH02] Philippe Lemmerling Sabine Van Huffel. *Total Least Squares and Errors-in-Variables Modeling : Analysis, Algorithms and Applications*. Kluwer Academic Publishers, 2002.

[SW89] G.A.F. Seber and J. Wild. *Nonlinear Regression*. John Wiley & Sons, 1989.

[SZ02] F. Schaffalitzky and Andrew Zisserman. Multi-view matching for unordered image sets or "how do I organize my holiday snaps ?". In *proceedings of ECCV*, 2002.

[TD03] Philip. H. S. Torr and Colin Davidson. Impsac : A synthesis of importance sampling and random sample consensus. *PAMI*, 25(3) :354–365, 2003.

[Thi94] Bernard Thiesse. Calibrage géométrique d'un système –caméra+numériseur– d'acquisition d'images numériques. Technical report, ENSEEIHT, Toulouse, 1994.

[UES01] Matthew Uyttendaele, Ashley Eden, and Richard Szeliski. Eliminating ghosting and exposure artifacts in image mosaics. In *proceedings of ICPR*, 2001.

[Var62] R. S. Varga. *Matrix Iterative Analysis*. Prentice Hall, Englewood Cliffs (NJ), 1962.

[VGBS03] Javier-Flavio Vigueras-Gomez, Marie-Odile Berger, and Gilles Simon. Calibration multiplanaire d'une caméra : augmenter la stabilité en utilisant la sélection de modèles. In *Journées Francophones des Jeunes Chercheurs en Vision par Ordinateur - ORASIS'2003, Gérardmer, France*, pages 147–156. LORIA, INRIA-Lorraine, May 2003.

[Wat82] G. A. Watson. Numerical methods for linear orthogonal L_p approximation. *IMA Journal of Numerical Analysis*, 2 :275–287, 1982.

[WB01] Greg Welch and Gary Bishop. Course 8 : An introduction to the kalman filter. In *Tutorial at ACM SIGGRAPH*, 2001.

[WBC03] Oliver Williams, Andrew Blake, and Roberto Cipolla. A sparse probabilistic learning algorithm for real-time tracking. In *proceedings of ICCV*, Nice, France, october 2003.

[Yvi96] Yann Yvinec. *Appariement dense de stéréogrammes issus d'un capteur binoculaire non calibré*. PhD thesis, INP, Toulouse, 1996.

[Zha97] Z. Zhang. Parameter estimation techniques :a tutorial with applications to conic fitting. *Image and Vision Computing*, 15(1), 1997.