Université de Montréal

Reconstruction active et passive en vision par ordinateur

par

Jean-Philippe Tardif

Département d'informatique et de recherche opérationnelle

Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures

en vue de l'obtention du grade de

Philosophiae Doctor (Ph.D.)

en informatique

Août, 2007

Université de Montréal

Faculté des études supérieures

Cette thèse intitulée :

Reconstruction active et passive en vision par ordinateur

présentée par

Jean-Philippe Tardif

a été évaluée par un jury composé des personnes suivantes:

Neil Stewart
(président-rapporteur)

Sébastien Roy
(directeur de recherche)

Jean Meunier
(membre du jury)

Marc Pollefeys
(examinateur externe)

Neil Stewart
(représentant du doyen de la FES)

Thèse acceptée le 30 août 2007

RÉSUMÉ

Dans cette thèse, nous proposons des solutions à plusieurs problèmes rencontrés en reconstruction 3D active et passive en vision par ordinateur. Nos contributions portent sur trois grands thèmes soient l'auto-calibrage, la reconstruction et la factorisation. Le premier thème est le calibrage et l'auto-calibrage des caméras sténopé, grand angle et *fisheye* ainsi que d'une grande partie des caméras catadioptriques possiblement d'angle de vue plus grand que 180° et à points de vue multiples. Tout d'abord, nous proposons le modèle de caméra *radial symétrique général* à projection centrale ou non-centrale. Celui-ci permet une représentation unifiée des fonctions de projection de toutes ces caméras. Une caméra est modélisée par un ensemble de caméras sténopé virtuelles ayant le même axe optique, chacune d'elles associées à un cercle de l'image centré au centre de distorsion. Le modèle permet de représenter toutes sortes de distorsions radiales symétriques en permettant aux caméras virtuelles de posséder des longueurs focales différentes. Cette variation de longueur focale peut être représentée par une fonction discrète ou paramétrique. Nous proposons un total de trois algorithmes de calibrage planaire du modèle central et un pour le modèle non-central. De plus, nous proposons deux algorithmes d'auto-calibrage du modèle central. Le premier utilise des images de lignes alors que le deuxième utilise des correspondances entre deux images ou plus d'une scène planaire. Le deuxième thème est la mise en correspondance active robuste entre un projecteur multimédia et une caméra, à l'aide de motifs de lumière structurée. Notre contribution est une approche probabiliste de décodages des motifs. Celle-ci corrige automatiquement les erreurs commises à l'étape de binarisation des images. Elle permet aussi un lissage des codes tout en préservant les discontinuités. Le troisième thème est le problème non-linéaire

de factorisation de matrice avec données manquantes. Un algorithme permettant d'effectuer cette tâche est un outil très utile puisque plusieurs problèmes de vision par ordinateur peuvent être formulés de cette manière. On compte, par exemple, les problèmes de reconstruction 3D pour les modèles affine et perspectif, ainsi que la reconstruction sous illumination variable et inconnue. La factorisation est généralement résolue en lançant plusieurs minimisation itératives initialisées à des point de départ choisis aléatoirement. Il s'agit d'un processus lent ou chaque minimisation converge la plupart du temps vers un minimum local. Nous proposons une nouvelle méthode *batch*, qui procède en accumulant des contraintes sur le premier des facteurs à partir de sous-blocs complets de la matrice de mesure. Celles-ci permettent de formuler l'estimation de ce facteur sous forme d'un problème de minimisation convexe. Nous proposons une généralisation des contraintes de fermeture, de nouvelles contraintes duales appelées *contraintes de base* ainsi qu'un algorithme pour trouver des sous-blocs complets. La méthode est très rapide et donne des résultats presque optimaux. Par ailleurs, nos expériences démontrent une amélioration significative du taux de convergence des méthodes itératives lorsqu'initialisées à l'aide de notre méthode. De plus, nous proposons une variante de nos contraintes spécifiquement formulée pour le problème de reconstruction 3D passive. Une attention particulière est donnée au modèle affine, pour lequel nous obtenons de très bons résultats même sans raffinement itératif de la reconstruction.

Mots clés : calibrage, auto-calibrage, vision omnidirectionnelle, distorsion radiale, mise en correspondance, lumière structurée, factorisation, reconstruction 3D, vision par ordinateur.

ABSTRACT

In this thesis, we propose solutions for several problems encountered in the field of active and passive 3D reconstruction in computer vision. Our contributions are related to three topics: self-calibration, reconstruction and factorization.

The first topic relates to the problem of calibration and self-calibration of pin-hole, wide-angle and fisheye cameras, as well as many catadioptric cameras possibly with view angle larger than 180° and also with a non-single viewpoint. We propose the *general radially symmetric* camera model with central or non-central projection. It provides a unified representation of the projection function of all the above cameras. The camera is modeled using a set of virtual pinhole cameras with identical optical axis, each of them associated with a circle of the image centered in the distortion center. Many kinds of distortion can be represented by allowing the virtual cameras to have different focal length. This variation can be represented either by a discrete or a parametric function. We propose three plane-based calibration algorithms for the central-projection model and another one for the non-central case. Furthermore, we propose two self-calibration algorithms for the case of central projection. The first one is a plumbline method and the second one uses feature correspondences in at least two views of a planar scene.

The second topic concerns the active matching between a multimedia projector and a camera using structured light patterns. Our contribution is a probabilistic approach to decode the patterns that can correct errors induced by the binarization of the images. Furthermore, it provides smoothing of the codes while preserving discontinuities.

The third topic is the non-linear problem of matrix factorization with missing

data. An algorithm performing this task is very useful tool because many vision problem can be formulated in this way. Among them, there are the problems of affine and perspective 3D reconstruction as well as 3D reconstruction under varying and unknown illumination. In general, the factorization problem is solved by launching many iterative minimization initialized using random solutions. The process is slow and, most of the time, the minimization get stuck into a local minimum. We propose a novel batch algorithm that proceeds by computing constraints on the first factor using complete sub-blocks of the measurement matrix. These are combined together to allow the estimation of the first factor by solving a convex minimization problem. We propose generalized closure constraints, new dual constraints called *basis constraints* and an algorithm for finding complete sub-blocks. The algorithm is very fast and provides close to optimal solutions. Furthermore, our method can provide a starting point for iterative methods and significantly improve their convergence rate. Also, we provide alternative closure and basis constraints specifically designed for the problem of passive 3D reconstruction. We perform a careful analysis of the affine model, for which we obtain very good results even without refining the reconstruction by using an iterative algorithm.

Keywords: calibration, self-calibration, omnidirectional vision, radial distortion, matching, structured light, factorization, 3D reconstruction, computer vision.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLES

REMERCIEMENTS

Je veux tout d'abord remercier le Professeur Sébastien Roy pour m'avoir convaincu d'entreprendre ce défi et qui m'a offert l'opportunité de travailler dans son laboratoire de recherche. J'apprécie particulièrement ses encouragements, son support et sa grande ouverture en me permettant de travailler sur un large éventail de sujets, de collaborer avec d'autres chercheurs et de visiter d'autres centres de recherche.

Je remercie aussi chaudement Peter Sturm pour m'avoir accueilli à deux reprises dans le groupe de recherche Perception de l'INRIA Rhône-Alpes et pour m'avoir encouragé tout le long de ce doctorat. Merci aussi à Adrien Bartoli pour son invitation à venir travailler avec lui à deux occasions au CNRS de Clermont-Ferrand. Ces quatre voyages ont été marquants pour la réalisation de mon doctorat. Évidemment, je ne peux oublier Martin Trudeau pour son aide, sa patience et ses encouragements durant ces quatre années.

Je voudrais remercier le Professeur Marc Pollefeys pour avoir accepté avec enthousiasme d'être l'examinateur externe de cette thèse, de même que le Professeurs Neil Stewart et Jean Meunier pour avoir accepté de former le reste du jury. Merci aux FQRNT et surtout au CRSNG pour leur support financier au cours de ces années d'études.

Finalement, je voudrais exprimer ma plus grande gratitude à ma famille et et à mes amis pour leurs encouragements, en particulier ceux de mes parents, Robert et Ginette, et de ma soeur Lyne, qui ont toujours été là pour moi. Je dois aussi beaucoup à Mé-Linh pour sa présence, sa patience, son humour et son amour, et qui reste ma plus profonde source d'inspiration.

Chapitre 1

INTRODUCTION

La *vision 3D par ordinateur* est un domaine de recherche relativement jeune qui a pour objectif l'inférence d'informations 3D à partir d'images. Celui-ci fait partie de la *vision par ordinateur* dont l'objectif beaucoup plus large est d'inférer tout type d'information à partir d'images. Généralement, ces images sont acquises par des appareils de capture numérique contrôlés par un utilisateur ou un système robotisé, puis, elles sont transférées à un ordinateur. Ce dernier effectue une variété d'opérations sur les images pour en extraire l'information de façon automatique. Les applications de la vision par ordinateur touchent à la fois à la reconnaissance et la détection d'objets, à la restauration d'images dégradées et à la construction de modèles 3D, pour n'en nommer que quelques-uns. En vision 3D par ordinateur, l'information est associée à la profondeur des objets dans une image ou à la construction de modèles 3D. De plus, elle concerne aussi l'appareil de capture d'images : sa position, son orientation et les caractéristiques du système d'acquisition, comme par exemple, la valeur du zoom.

Dans cette thèse, nous nous intéressons à ce problème de reconstruction 3D dans plusieurs contextes. En général, on peut classer les méthodes de reconstruction 3D en deux grandes catégories : *active* et *passive*. À l'intérieur de ces classes, les méthodes sont généralement départagées par le type de connaissance connue *a priori* au sujet de la scène ou de l'appareil d'acquisition. Le type d'information varie largement : il peut s'agir d'un objet de la scène de dimension connue, de contraintes sur le type de mouvement que l'appareil d'acquisition peut effectuer, ou, d'information sur la

position relative entre deux appareils. En mode actif, le système d'acquisition interagit avec la scène au moyen de lasers ou de projecteurs multimédia utilisés, par exemple, pour projeter des motifs de lumière sur les objets. D'ailleurs, les méthodes actives sont suffisamment perfectionnées pour être utilisées dans l'industrie. Cependant, leur applicabilité est assez limitée. Entre autres, elles sont difficilement utilisables pour reconstruire des objets de très grandes tailles ou très éloignés.

Les méthodes de reconstruction 3D passives, quant à elles, minimisent le plus possible l'interaction de l'appareil d'acquisition avec la scène. Le système d'acquisition ne modifie ni l'aspect ni la position des objets, qui sont généralement de dimension inconnue. Toutefois, il est commun d'ajouter à la scène un objet de dimension ou de position connue. Cet ajout a pour objectif de faciliter l'inférence de l'information 3D. Un exemple typique est l'utilisation d'un objet de taille connue permettant d'estimer un modèle de caméra décrivant les caractéristiques de l'appareil d'acquisition. Cette approche se nomme *calibrage* de caméra. Bien que les méthodes passives soient plus flexibles que les méthodes actives, elles sont néanmoins moins précises. Aujourd'hui, il est toutefois possible d'estimer les paramètres d'un modèle de caméra par *auto-calibrage*, *i.e.* sans même utiliser d'objets connus. Ce type d'innovation permet d'envisager des applications comme la navigation automatique de véhicules ainsi que la reconstruction automatique d'objets aussi larges que des bâtiments, ou même des quartiers tout entier.

Que ce soit pour une méthode active ou passive, le système d'acquisition d'images est choisi en fonction du contexte de la reconstruction. Certaines méthodes actives utilisent des lasers, alors que d'autres, des projecteurs multimédia. Les systèmes d'acquisition d'images sont encore plus variés : caméras conventionnelles, appareils médicaux de toutes sortes comme les machines à rayon X, images provenant de satellite et captées à différentes longueurs d'ondes. Dans cette thèse, nous utilisons des appareils photo et vidéo numérique conventionnels ou omnidirectionnels. Ces derniers se caractérisent par un angle de vue beaucoup plus large que les premiers. Pour ce

Titre	Conf./Journal	Présentation	%
Calibration of Cameras with Radially Symmetric Distortion	OMNIVIS2005/PAMI	Oral/Soumis	–
Self-calibration of a General Radially Symmetric Distortion Model	ECCV 2006	Poster	21.4%
Plane Based Self-calibration of Radial Distortion	ICCV 2007	Oral	3.9%
MRF Formulation for Coded Structured Light	3DIM 2005	Oral	31%
Batch Algorithms for Matrix Factorization with Missing Data	NIPS 2007	Soumis	–
Algorithms for Batch Matrix Factorization with Application to Structure-from-Motion	CVPR 2007	Poster	28.8%

TAB. **1.1.** Liste des publications reproduites dans cette thèse.

% :	Taux d'accepation
OMNIVIS :	Workshop on omnidirectional vision, camera networks, and non-classical cameras
PAMI :	Pattern Analysis and Machine Intelligence
ECCV :	European Conference on Computer Vision
ICCV :	IEEE International Conference on Computer Vision
3DIM :	International Conference on 3D Digital Imaging and Modeling
NIPS :	Conference on Neural Information Processing Systems
CVPR :	IEEE Conference on Computer Vision and Pattern Recognition

qui est des méthodes actives, nous faisons appel soit à un projecteur multimédia conventionnel ou à un écran d'ordinateur.

Cette thèse par articles propose des contributions originales aux problèmes de reconstruction 3D active et passive. Les articles reproduits dans cette thèse sont donnés à la table 1.1. Les recherches ont touché à trois sujets reliés à ces thématiques : calibrage et auto-calibrage de caméras omnidirectionnelles, méthode de reconstruction robuste par lumière structurée et algorithmes de factorisation de matrice creuse.

1.1 Organisation

La thèse est divisée en quatre grandes parties. La première consiste en une introduction des concepts fondamentaux nécessaires à la lecture la thèse. Chacune des parties suivantes est constituée d'une revue de la littérature suivie d'un ou plusieurs articles présentant nos contributions sur les trois sujets mentionnés plus haut. Voici un aperçu des quatre parties de la thèse :

(a) (b) (c)

FIG. 1.1. **Trois types de senseur auxquels nous nous intéressons : (a) grand angle, (b)** *fisheye*, **(c) catadioptrique.**

(a) (b) (c)

FIG. 1.2. **Trois images de caméra de type (a) grand angle, (b)** *fisheye*, **(c) catadioptrique.**

1.1.1 Concepts de base et notions d'optimisation continue

Au chapitre 2, nous présentons notre notation et donnons un résumé des concepts fondamentaux de vision 3D par ordinateur nécessaires à la lecture de la thèse. Au chapitre 3, nous décrivons plusieurs problèmes d'optimisation auxquels nous avons fait face, ainsi que plusieurs des méthodes de minimisation utilisées pour les résoudre.

1.1.2 Calibrage et auto-calibrage de caméra omnidirectionnelle

Dans cette partie, nous nous intéressons aux problèmes de calibrage et d'auto-calibrage de caméras grand angle et omnidirectionnelle. Plus précisément, nous proposons un modèle de caméra et des algorithmes permettant de retrouver les paramètres de caméras *fisheye* et des caméras catadioptriques les plus répandues. Nous nous intéressons surtout à l'estimation des *paramètres internes*, ceux qui décrivent la formation de l'image. La figure 1.1 illustre les trois modèles d'objectifs de caméra, grand angle, *fisheye* et catadioptrique, auxquels nous nous intéressons. Des exemples de photos prisent à l'aide de ces objectifs sont montrés à la figure 1.2. Contrairement aux photos traditionnelles, on remarque que la projection de droites (par exemples provenant des bâtiments) résultent en des courbes dans les photos. Au chapitre 4, nous présentons une sélection de plusieurs modèles de distorsion d'image. Par la suite, nous présentons quelques algorithmes, chacun d'eux permettant de calibrer un de ces modèles en utilisant une grille de points connus. Au chapitre 5, nous présentons notre nouveau modèle de caméra permettant de représenter les caméras qu'elles soient traditionnelle, grand angle, *fisheye* ou catadioptrique. Il permet aussi d'unifier plusieurs modèles de caméra catadioptrique à points de vue multiples. Nous proposons trois nouveaux algorithmes pour en estimer les paramètres. Au chapitre 6, nous résumons les méthodes de calibrage et auto-calibrage de plusieurs modèles de caméra à l'aide d'images de lignes. Puis, au chapitre 7, nous présentons une méthode de calibrage et d'auto-calibrage de notre modèle. Cette méthode se démarque par sa grande simplicité et par son efficacité. Le chapitre 8 résume plusieurs méthodes d'auto-calibrage à partir de points de correspondances entre images. Nous proposons une méthode dans cette catégorie au chapitre 9.

(a) (b)

FIG. **1.3. Projection immersive obtenues à l'aide d'un système de projecteur multiples sur des surfaces cylindriques ou sphériques. a) Vue complète de l'écran de l'intérieur à l'aide d'une caméra catadioptrique. b) Vue de l'extérieur de l'écran contenant une projection immersive.**

1.1.3 Mise en correspondance active

Lorsque la position de la caméra est connue, il est possible, en principe, d'effectuer une reconstruction dense des objets. Ceci consiste à obtenir des modèles 3D complets de ces objets, ou bien, d'estimer la profondeur des objets à chaque pixel des images. Dans cette partie, nous nous intéressons à ce type de reconstruction et proposons une nouvelle méthode active. Celle-ci utilise la lumière structurée pour obtenir des correspondances denses entre l'image de la caméra et celle d'un projecteur multimédia. Ces correspondances peuvent, entre autres, être utilisées pour effectuer une reconstruction dense. Elles sont aussi utiles pour certaines méthodes de calibrage de systèmes multi-projecteurs sans reconstruction explicite de l'écran, tel que discuté §10.3. Un exemple de résultat produit par un tel système est montré à la figure 1.3. Au chapitre 10, nous présentons quelques méthodes actives de mise en correspondances denses. Notre contribution, à ce titre, est présentée au chapitre 11. Il s'agit d'une méthode probabiliste d'estimation des correspondances. Celle-ci est utile en reconstruction 3D, mais, elle est particulièrement bien adaptée aux systèmes multi-projecteurs.

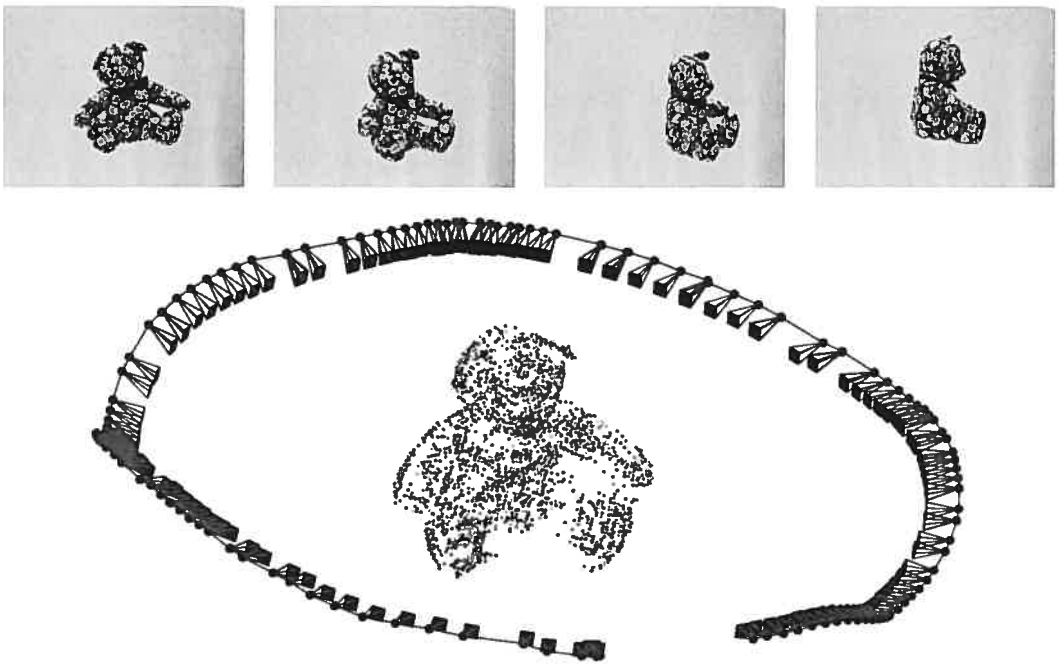


FIG. **1.4. Reconstruction 3D d'un ourson en peluche. (Haut) Les images 1, 20,40 et 60 de la séquence. (Bas) Reconstruction sous forme de nuage de points et trajectoire de la caméra au cours de la séquence d'images.**

1.1.4 Reconstruction 3D non-calibrée et factorisation de matrice avec données manquantes

Plusieurs des problèmes rencontrés en vision par ordinateur peuvent être formulés sous forme d'un problème de factorisation d'une matrice, plus précisément, d'approximation d'une matrice par une seconde matrice d'un rang donné. Le problème de reconstruction 3D d'une caméra non-calibrée et en mouvement[1] en est un des meilleurs exemples. L'algorithme prend en entrée un ensemble de trajectoires de points saillants obtenues dans une séquence d'images. Ces trajectoires sont combinées pour former une matrice de mesure. La factorisation permet de trouver deux facteurs correspondant à la trajectoire de la caméra et à un nuage de points 3D appartenant à la scène

[1] Structure-from-Motion

reconstruite, tel qu'illustré à la figure 1.4. Dans plusieurs problèmes, la matrice qui doit être factorisée est incomplète (jusqu'à plus de 95% de données manquantes dans certains cas), ce qui rend la factorisation beaucoup plus difficile (en ce sens qu'il n'y pas de solution analytique). Au chapitre 12, nous présentons plusieurs méthodes de factorisation de matrice avec une attention particulière à celles qui sont spécifiques au problème de reconstruction 3D. Suivent deux contributions aux chapitres 13 et 14. La première est une méthode générale permettant de factoriser toute matrice avec donnéee manquantes, alors que la deuxième propose une solution spécifique au problème de la reconstruction 3D.

1.2 Contributions des coauteurs

Calibration of Cameras with Radially Symmetric Distortion (chap. 5)

Peter Sturm a proposé une partie de la théorie dans les sections 2 et 3 et a participé à la rédaction. Martin Trudeau a aidé à formuler et à rédiger la preuve de l'annexe 1. Finalement, Sébastien Roy a participé à la relecture de la version courte de l'article [151].

Self-calibration of a general radially symmetric distortion model (chap. 7)

La théorie a été développée avec l'aide de Peter Sturm et il a proposé des corrections à l'article. Sébastien Roy a participé à la relecture de l'article.

Plane based self-calibration of radial distortion (chap. 9)

Peter Sturm a offert ces commentaires sur la théorie et a aidé à la rédaction de l'article. Sébastien Roy a participé à la relecture de l'article.

MRF formulation for coded strctured light (chap. 11)

Sébastien Roy a proposé l'idée originale et a participé à la rédaction de l'article. Martin Trudeau a aussi participé à la rédaction de l'article.

Batch Algorithms for Matrix Factorization with Missing Data (chap. 13)

Adrien Bartoli a aidé à élaborer la théorie de la section 13.4 et a participé à la rédaction. Sébastien Roy a participé à la relecture de l'article et proposé des modifications à l'algorithme 1.

Algorithms for Batch Matrix Factorization with Application to Structure-from-Motion (chap. 14)

Adrien Bartoli a proposé l'idée originale, aidé à élaborer la théorie et participé à la relecture de l'article. Martin Trudeau a aidé à la rédaction, Nicolas Guilbert a aidé à l'élaboration de l'idée originale avec Adrien Bartoli, et Sébastien Roy a participé à la relecture de l'article.

Chapitre 2

CONCEPTS FONDAMENTAUX ET OUTILS MATHÉMATIQUES

Cette section présente les outils mathématiques et les concepts géométriques de base de vision par ordinateur qui sont essentiels à la lecture de cette thèse. Les conventions ainsi que la notation employées sont tout d'abord présentées et suivies d'une brève introduction à la géométrie projective. Nous y décrivons les éléments de base et leurs transformations. Nous présentons aussi les deux modèles de caméra les plus utilisés, soit le modèle perspectif et le modèle affine. Plus de détails sur la géométrie pour la vision par ordinateur sont présentés aux références [42, 60, 85].

2.1 Conventions et notation

Tout au long du texte, nous référons aux équations simplement par leur numéro placé entre parenthèse, *e.g.* (2.1). Le caractères '§' peut se lire 'section', et nous utilisons des crochets pour les références. Nous utilisons des caractères standards en minuscules pour représenter les scalaires, *e.g.* s. Les matrices sont en caractères *sans-serif* et en majuscules, *e.g.* M. Pour représenter les éléments (scalaires) d'une matrice, nous utilisons généralement la même lettre, et des crochets pour former la matrice, comme dans l'exemple :

$$\mathsf{M}_{(m\times n)} = \begin{bmatrix} M_{11} & \dots & M_{1n} \\ \vdots & \ddots & \vdots \\ M_{m1} & \dots & M_{mn} \end{bmatrix} = \begin{bmatrix} \mathbf{M}^1 & \dots & \mathbf{M}^n \end{bmatrix} = \begin{bmatrix} \mathbf{M}_1^\mathsf{T} \\ \vdots \\ \mathbf{M}_m^\mathsf{T} \end{bmatrix} \tag{2.1}$$

c'est-à-dire que $\mathbf{M}^i$ est la $i^{\text{ième}}$ colonne de M et $\mathbf{M}_j$ est la $j^{\text{ième}}$ rangé. Pour distinguer les vecteurs des matrices, nous les mettons en gras, *e.g.* $\mathbf{v}$, et nous utilisons des parenthèses :

$$\mathbf{v}_m = \begin{pmatrix} v_1 \\ \vdots \\ v_m \end{pmatrix} = \begin{pmatrix} v_1 & \dots & v_m \end{pmatrix}^\mathsf{T}$$

où, $\mathbf{v}^\mathsf{T}$ est la transposée de $\mathbf{v}$. Cette même règle s'applique aussi aux nombres de telle sorte qu'un vecteur et une matrice de zéros s'écrivent respectivement $\mathbf{0}$ et 0. La taille des matrices ou vecteurs est donnée en indice entre parenthèse comme en (2.1). L'espace projectif de dimension n est référé par $\mathbb{P}^n$, alors que l'espace euclidien de même dimension est $\mathbb{R}^n$. Dans $\mathbb{P}^n$, les éléments sont des $(n+1)$-vecteurs définis à un facteur d'échelle (non-nul) près. Par exemple, les vecteurs $\mathbf{u}, \mathbf{v} \in \mathbb{P}^n$ représentent le même point s'il existe un facteur d'échelle α non-nul tel que $\mathbf{u} = \alpha\mathbf{v}$. Dans ce cas, nous écrivons $\mathbf{u} \propto \mathbf{v}$. Nous utilisons $\mathbb{P}^2$ pour représenter les éléments sur le plan image ou sur des plans en 3D et $\mathbb{P}^3$ pour des points en 3D. Pour distinguer clairement les points en coordonnés euclidiennes des points en coordonnées projectives, nous leur ajoutons un "tilde", *e.g.* $\tilde{\mathbf{v}}$. Un vecteur $\mathbf{v} \succ 0$ ou $\mathbf{v} \succeq 0$ signifie que chaque élément de $\mathbf{v}$ est plus grand ou plus grand ou égal à 0. Une matrice $\mathsf{M} \succ 0$ signifie qu'elle est définie positive, *i.e.* $\forall \mathbf{x} \neq 0 : \mathbf{x}^\mathsf{T}\mathsf{M}\mathbf{x} > 0$ et $\mathsf{M} \succeq 0$ signifie qu'elle est semi-définie positive, *i.e.* $\forall \mathbf{x} : \mathbf{x}^\mathsf{T}\mathsf{M}\mathbf{x} \geq 0$. L'ensemble des matrices symétriques de dimension $(k \times k)$ est noté $\mathbb{S}^k$, alors que $\mathbb{S}^k_+$ représente les matrices symétriques semi-définies positives ($\succeq$) et $\mathbb{S}^k_{++}$ celles qui sont positives définies ($\succ$). La matrice (3×3) $[\mathbf{v}]_\times$ est défini pour tous vecteurs à trois coordonnée :

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix},$$

et $[\mathbf{v}]_\times \mathbf{u} = \mathbf{v} \times \mathbf{u}$, où $\times$ est le produit vectoriel.

Normes

Nous faisons usage de deux normes : la norme 2 pour les vecteurs :

$$||\mathbf{v}||_2 = \sqrt{\sum_i v_i^2}$$

et la norme de Frobenius pour les matrices :

$$||\mathsf{M}||_F = \sqrt{\sum_{ij} M_{ij}^2}$$

et il arrive souvent que les indices 2 et F ne soient pas employés.

2.2 Géométrie projective

Nous manipulerons des éléments (point, ligne, etc) dans plusieurs espaces : euclidien, métrique, affine et projectif. L'espace euclidien correspond au monde qui nous entoure. N'importe quelle mesure (distance, angle) dans cet espace correspond à la réalité. Dans l'espace métrique, la notion de distance n'est définie qu'à un facteur d'échelle inconnu. Lorsqu'on fait une reconstruction 3D d'un objet, la reconstruction métrique implique qu'on ne connaît pas sa taille. La passage à une reconstruction euclidienne ne peut se faire qu'en connaissant la taille réelle d'une partie de la scène 3D reconstruite, comme par exemple, la distance entre deux points de l'objet ou bien la taille du capteur CCD de la caméra. Dans l'espace affine, le parallélisme est conservé, alors que les angles ne correspondent pas nécessairement à la réalité. Finalement, dans l'espace projectif, tous les repères auxquels nous sommes habitués disparaissent.

Nous ferons un usage intensif de notions de géométrie projective algébrique. Cette théorie nous permet de représenter les éléments de base et leur manipulation (trans-

formation, projection) sous la forme vectorielle et matricielle. Les éléments qui nous intéressent sont : le point, la ligne, le plan, la conique et la quadrique. Pour simplifier les explications, nous ne ferons aucune distinction entre les éléments eux-mêmes et leur représentation vectorielle ou matricielle. Par exemple, $\mathbf{p}$ est à la fois un point image dans $\mathbb{P}^2$ et un vecteur à trois coordonnées. L'utilisation de l'espace projectif plutôt que l'espace euclidien offre deux avantages importants. Le premier avantage est la représentation transparente des points à l'infini. En fait, l'espace projectif $\mathbb{P}^n$ permet de représenter l'espace $\mathbb{R}^n$ augmenté des points à l'infini, ceux-ci représentés par un seul élément dual : la ligne à l'infini $\mathbf{l}_\infty \propto (0,0,1)^\mathsf{T}$ dans $\mathbb{P}^2$ et le plan infini $\boldsymbol{\pi}_\infty \propto (0,0,0,1)^\mathsf{T}$ dans $\mathbb{P}^3$. Ainsi, l'ensemble des points à l'infini dans $\mathbb{P}^2$ et $\mathbb{P}^3$ sont donnés respectivement par $\mathbf{p}^\mathsf{T}\mathbf{l}_\infty = 0$ et $\mathbf{P}^\mathsf{T}\boldsymbol{\pi}_\infty = 0$. Nous reviendrons plus loin sur le concept de dualité. Le deuxième avantage est que toutes les transformations affines peuvent être représentées par une matrice. Par exemple, la transformation affine (rotation + translation) de $\tilde{\mathbf{P}}$ vers $\tilde{\mathbf{P}}'$ est $\tilde{\mathbf{P}}' = \mathsf{R}_{(3\times3)}\tilde{\mathbf{P}} + \tilde{\mathbf{T}}$, où, $\tilde{\mathbf{P}}, \tilde{\mathbf{P}}', \tilde{\mathbf{T}} \in \mathbb{R}^3$, et $\tilde{\mathbf{T}}$ et R sont des transformations de rotation et translation. En projectif, on peut écrire :

$$(\tilde{\mathbf{P}}', 1)^\mathsf{T} \propto \begin{bmatrix} \mathsf{R} & \tilde{\mathbf{T}} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} (\tilde{\mathbf{P}}, 1)^\mathsf{T}.$$

L'ensemble des points $\mathbf{p} \in \mathbb{R}^n$ peut être représenté dans $\mathbb{P}^n$ en définissant $\mathbf{p}' = \alpha(\mathbf{p}^\mathsf{T}, 1)^\mathsf{T}$ $\alpha \neq 0$, c'est-à-dire par l'ajout d'une $(n+1)^{\text{ième}}$ coordonnée de valeur 1 et en multipliant par un scalaire non-nul fini α. Un point $\mathbf{p}' = (x_1, ..., x_{n+1})^\mathsf{T} \in \mathbb{P}^n$ peut être converti en élément de $\mathbb{R}^n$ tel que $\mathbf{p} = (\frac{x_1}{x_{n+1}}, ..., \frac{x_n}{x_{n+1}})$ seulement s'il n'est pas à l'infini, *i.e.* $x_{n+1} \neq 0$. Finalement, le point correspondant au $(n+1)$-vecteur $(0, ..., 0)^\mathsf{T}$ ne fait pas partie de $\mathbb{P}^n$.

Nous décrivons dans les section suivantes les entités de bases qui sont utilisées dans ce document ainsi que leurs transformations.

2.2.1 Transformations et projection

L'utilisation de matrices permet d'effectuer des transformations, des projections et même, dans certains cas, des déprojections (*cf.*[60, 137] pour plus d'explications). Une matrice $(n \times n)$ effectue une transformation d'un espace $\mathbb{P}^{n-1}$, alors qu'une matrice $(n \times m)$ permet la projection de l'espace $\mathbb{P}^{m-1}$ dans $\mathbb{P}^{n-1}$. Les matrices utilisées pour des transformations image sont donc des (3×3), la projection du 3D vers une image sont des (3×4), et les transformations du monde 3D sont des (4×4). Dans ce qui suit, nous utilisons $\mathsf{M}_{(3\times4)}$ et $\mathsf{H}_{(4\times4)}$ ou $\mathsf{H}_{(3\times3)}$, selon le contexte.

Les deux projections typiques en vision par ordinateur sont : perspective et affine. Nous verrons plus loin que celles-ci permettent de modéliser plusieurs types de caméras. La projection perspective effectue le passage $\mathbb{P}^3 \rightarrow \mathbb{P}^2$ à l'aide d'une matrice (3×4) qui est définie à un facteur d'échelle près :

$$\mathsf{M}_{\text{persp}} \propto \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \end{bmatrix}.$$

Quant à elle, la projection affine effectue la projection $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ à l'aide d'une matrice (2×4) de huit degrées de liberté :

$$\mathsf{M}_{\text{aff}} = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \end{bmatrix}.$$

En effet, pour que cette projection soit valide, on impose que la dernière coordonnée des points 3D projectifs soit égale à 1. Autrement dit, ce sont des points euclidiens. La matrice de projection affine est en fait un matrice de projection perspective de la

forme :

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Une explication rigoureuse est donnée dans [60]. L'important est de savoir qu'une projection affine est effectuée à partir d'un centre situé à l'infini. En effet, la coordonnée projective de ce centre $\mathbf{C}$ est donnée par le noyau droit de la matrice de projection. Ainsi, dans le cas d'une matrice affine, $\mathbf{C}$ doit vérifier $(0,0,0,1)\mathbf{C} = 0$, d'où, $C_4 = 0$. Une autre manière d'exprimer la même chose est à partir d'une projection de $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ à l'aide d'une matrice (2×3), suivie d'une translation :

$$\tilde{\mathbf{p}} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \end{bmatrix} \tilde{\mathbf{P}} + \begin{pmatrix} M_{14} \\ M_{24} \end{pmatrix}.$$

2.2.2 Point et ligne dans $\mathbb{P}^2$ et concept de dualité

On peut interpréter tout élément de $\mathbb{P}^2$ à la fois comme un point ou une ligne. Cette double interprétation fait appel à un principe très important retrouvé en géométrie projective appelé *dualité*. Celui-ci peut s'énoncer de la manière suivante : à chaque théorème de géométrie projective correspond un théorème dual, construit à partir de l'original, en inter-changeant le rôle des points et des lignes. Par exemple, un point $\mathbf{p}$ est situé sur une ligne $\mathbf{l}$ si $\mathbf{l}^\mathsf{T}\mathbf{p} = 0$. Par le principe de dualité, l'ensemble des lignes $\mathbf{l}$ qui s'intersectent en $\mathbf{p}$ sont données par $\mathbf{p}^\mathsf{T}\mathbf{l} = 0$. Aussi, la droite $\mathbf{l}$ passant par deux points est donnée par $\mathbf{p}_1 \times \mathbf{p}_2$, alors que l'intersection de deux lignes donne un point $\mathbf{p} \propto \mathbf{l}_1 \times \mathbf{l}_2$ (possiblement à l'infini).

2.2.3 Point et plan dans $\mathbb{P}^3$

Dans $\mathbb{P}^3$, les éléments représentent soit un point, soit un plan. Un point $\mathbf{P}$ est sur un plan $\boldsymbol{\pi}$ si et seulement si $\mathbf{P}^\mathsf{T}\boldsymbol{\pi} = 0$. Le plan formé par trois points $\mathbf{P}_i$, $i =$

$1, ..., 3$ non-colinéaires est donné par le noyau droit de $[\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3]^\top$. Une équation similaire permet de trouver le point d'intersection de trois plans s'il existe un tel point. Lorsqu'un élément de $\mathbb{P}^3$ est interprété comme un point, on peut y appliquer une transformation M selon $\mathbf{P} \leftarrow \mathsf{M}\mathbf{P}$. Lorsqu'interprété comme un plan, on peut le transformer selon $\boldsymbol{\pi} \leftarrow \boldsymbol{\pi}\mathsf{M}^{-1}$.

2.2.4 Matrice et vecteur de Plücker

Dans $\mathbb{P}^3$, la représentation d'une ligne n'est pas aussi naturelle qu'en $\mathbb{P}^2$. La façon la plus simple est d'utiliser la forme paramétrique, c'est-à-dire qu'une ligne est construite à partir de deux points 3D distincts : $\mathbf{l}(\lambda) = \lambda\tilde{\mathbf{P}}_1 + (1-\lambda)\tilde{\mathbf{P}}_2$. Cependant, cette représentation comporte l'inconvénient que la ligne n'est pas décrite par un seul vecteur ou matrice comme pour les formes implicites discutées jusqu'à maintenant. Une telle représentation existe toutefois ; il s'agit des *matrices* ou des *vecteurs de Plücker.*

Il est possible de construire une matrice de Plücker décrivant une ligne joignant des points 3D $\mathbf{P}_1, \mathbf{P}_2 \in \mathbb{P}^3$ à partir de la formule : $\mathsf{L}_{(4\times4)} = \mathbf{P}_1\mathbf{P}_2^\top - \mathbf{P}_2\mathbf{P}_1^\top$. La projection d'une telle ligne dans l'espace $\mathbb{P}^2$ par une matrice de projection M est effectué selon

$$[\mathbf{v}]_\times = \mathsf{M}\mathsf{L}\mathsf{M}^\top, \mathbf{v} \in \mathbb{P}^2.$$

Ceci a l'avantage de donner directement l'image de la projection de la ligne. Le vecteur de Plücker défini à partir de deux points 3D $\mathbf{P}$ et $\mathbf{Q}$ ($\mathbf{P} \neq \mathbf{Q}$) est donné par les éléments de L

$$\Big(\underbrace{L_{12}, L_{13}, L_{14}}_{\mathbf{u}^\top}, \underbrace{L_{23}, L_{42}, L_{34}}_{\mathbf{w}^\top}\Big)$$

et vérifie toujours que $\mathbf{u}^\top\mathbf{w} = 0$.

2.3 Conique et quadrique

2.3.1 Conique

Une conique est l'image de l'intersection entre un cône droit et un plan, situé dans le système de coordonnées de ce plan. Selon l'orientation de ce dernier par rapport au cône, différentes formes de courbes sont obtenues. Parmi les coniques non-dégénérées, on distingue la parabole, l'hyperbole, l'ellipse et le cercle. Une ligne, un point et deux lignes sont aussi des coniques dégénérées. L'équation implicite d'une conique est :

$$a\,x^2 + d\,x + b\,x\,y + c\,y^2 + e\,y + f = 0 \tag{2.2}$$

où, (x, y) est un point situé sur la courbe et (a, b, c, d, e, f) constituent les paramètres de la conique. La même conique peut être représentée plus généralement pour des points de $\mathbb{P}^2$ sous forme matricielle, soit

$$\mathsf{C} \propto \begin{bmatrix} a & \frac{b}{2} & \frac{d}{2} \\ \frac{b}{2} & c & \frac{e}{2} \\ \frac{d}{2} & \frac{e}{2} & f \end{bmatrix}$$

et un point $\mathbf{p} \in \mathbb{P}^2$ sur la courbe vérifie $\mathbf{p}^\mathsf{T}\mathsf{C}\mathbf{p} = 0$. Il existe aussi des coniques complexes formées de points complexes, mais elles ne nous sont pas utiles. Une conique C peut-être transformée par une matrice $\mathsf{H}_{(3\times3)}$ selon $\mathsf{C} \leftarrow \mathsf{H}^{-\mathsf{T}}\mathsf{C}\mathsf{H}^{-1}$. Finalement, il existe aussi des coniques duales, notées C^*, définies non pas pour les points, mais pour des lignes, *i.e.* une ligne $\mathsf{l} \in \mathbb{P}^2$ est tangente à la courbe si $\mathsf{l}^\mathsf{T}\mathsf{C}^*\mathsf{l} = 0$. Pour les coniques non-dégénérées, $\mathsf{C}^* = \mathsf{C}^{-1}$. Les coniques seront surtout utiles au chapitre 5.

2.3.2 Décomposition

Une conique non-dégénérée peut être décomposée sous la forme suivante :

$$\mathsf{C} \propto (\mathsf{TR})^{-\top} \begin{bmatrix} \frac{1}{a^2} & 0 & 0 \\ 0 & \kappa\frac{1}{b^2} & 0 \\ 0 & 0 & -1 \end{bmatrix} (\mathsf{TR})^{-1}$$

où, $\mathsf{R}_{(3\times3)}$ est un rotation 2D donnée par le deuxième suivi du premier vecteur singulier droit de la partie (2×2) supérieure gauche de C. La matrice $\mathsf{T}_{(3\times3)}$ est une translation 2D donnée par :

$$(t_x, t_y, 1)^\top \propto \mathsf{C}^{-1}(0,0,1)^\top.$$

En posant $\mathsf{C}' \propto (\mathsf{TR})^\top \mathsf{CTR}$, telle que $C'_{3,3} = -1$, on obtient :

$$\begin{aligned} a &= 1/\sqrt{|C'_{1,1}|} \\ b &= 1/\sqrt{|C'_{2,2}|} \\ \kappa &= \text{sign}(C'_{2,2}) \\ c &= \sqrt{a^2 - \kappa b^2} \end{aligned}$$

où,

$$\text{sign}(x) = \begin{cases} 1 & x \leq 0 \\ -1 & x < 0 \end{cases}$$

$(\pm c, 0)$ sont les coordonnées des points focaux la conique en position canonique.

2.3.3 Dualité ellipse—hyperbole

Il existe une dualité entre ellipses et hyperboles. Pour une conique d'équation

$$\mathsf{E} \propto \text{diag}(\alpha, \beta, -1), \qquad \alpha, \beta \neq 0, \alpha \neq \beta$$

i.e. en position canonique, correspond une seconde conique donnée par

$$\mathsf{H} \propto \mathrm{diag}(\alpha\beta, \beta(\alpha-\beta), \alpha-\beta).$$

Selon la valeur des paramètres α et β, ces coniques sont des hyperboles ou des ellipses. Lorsque E est une ellipse, H est une hyperbole et, à l'inverse, si E est une hyperbole, alors H est une ellipse. Plus intéressant encore, il existe une relation géométrique entre ces deux coniques, illustrée à la figure 2.1. Plaçons les coniques dans des plans perpendiculaires entre eux, de sorte que leur axe majeur respectif forme l'intersection des plans. Plus précisément, E est placée en 3D dans le plan $Z = 0$: de telle sorte que $\mathbf{P} \in \mathbb{P}^3$ appartient à la courbe s'il vérifie que $\mathbf{P}^\mathsf{T}\mathrm{diag}(E_{11}, E_{22}, 0, E_{33})\mathbf{P} = 0, P_3 = 0$; de façon similaire, H est en 3D dans le plan $Y = 0$: $\mathbf{P}^\mathsf{T}\mathrm{diag}(H_{11}, 0, H_{22}, H_{33})\mathbf{P} = 0, P_2 = 0$. Dans cette configuration, on peut interpréter H comme l'ensemble des positions où, ayant placé la pointe d'un cône droit, il existe une orientation du cône permettant d'engendrer E comme étant l'intersection du cône avec le plan $Z = 0$.

2.3.4 Estimation de conique

Le problème d'estimation d'une conique à partir de points n'est pas un problème aussi facile qu'il n'y paraît. La méthode la plus simple consiste à minimiser la somme des moindres carrées de l'erreur algébrique donnée par l'équation implicite (2.2). Il est cependant plus difficile d'effectuer l'estimation selon un critère géométrique tel que la minimisation de la distance des points à la conique. Nous renvoyons un lecteur intéressé aux références [20, 38, 43, 69, 70, 124].

2.3.5 Quadrique

Les surfaces quadriques sont l'extension naturelle des coniques au monde 3D. Elles incluent les paraboloïdes, les hyperboloïdes, les ellipsoïdes et les sphères pour les formes non-dégénérées. Le cône est aussi une quadrique (dégénérée) très utile en

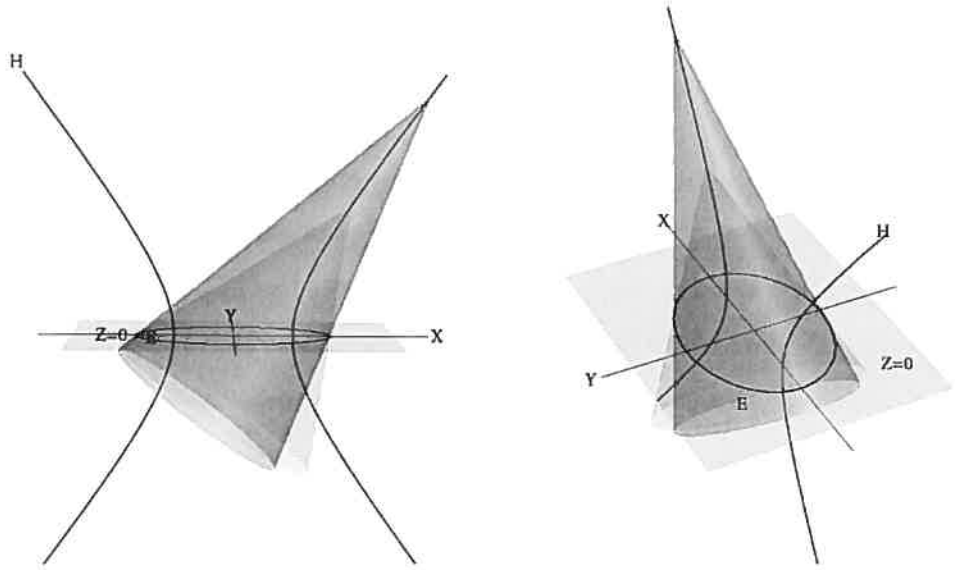


FIG. **2.1. Dualité ellipse—hyperbole : ellipse E, hyperbole H correspondante et deux cônes droits ayant leur pointe sur H dont l'orientation et l'ouverture est choisie pour générer E dans le plan $Z = 0$.**

vision.

Une formulation sous forme matricielle est possible en utilisant des points 3D projectifs. La matrice Q d'une quadrique est symétrique et de taille (4×4) et un point $\mathbf{P} \in \mathbb{P}^3$ de sa surface vérifie l'équation $\mathbf{P}^\mathsf{T}\mathsf{Q}\mathbf{P} = 0$. Une quadrique est transformée par une transformation projection $\mathsf{H}_{(4\times4)}$ selon $\mathsf{Q} \leftarrow \mathsf{H}^{-\mathsf{T}}\mathsf{Q}\mathsf{H}^{-1}$. Comme pour les coniques, une quadrique duale non-dégénérée est donnée par $\mathsf{Q}^* = \mathsf{Q}^{-1}$. La projection de la quadrique passe nécessairement par sa duale $\mathsf{C}^* \propto \mathsf{M}^\mathsf{T}\mathsf{Q}^*\mathsf{M}$, où, C^* est l'image de la quadrique, une conique dans sa forme duale.

2.4 Homographie

Les homographies sont très utilisées pour représenter des transformations projectives entre deux plans images. Considérons deux plans π_1 et π_2 dans l'espace 3D :

pour tous points $\mathbf{p}_2 \in \mathbb{P}^2$ dans le système de coordonnées de $\boldsymbol{\pi}_2$, sa projection sur $\boldsymbol{\pi}_1$ dans une certaine direction est donnée par la relation linéaire projective $\mathbf{p}_1 \propto \mathsf{H}_{21}\mathbf{p}_2$, où, H_{21} est une matrice (3×3) (voir figure 2.2). Sauf pour les cas de projection dégénérée, cette matrice est de rang 3 et donc inversible. Une homographie est définie à un facteur d'échelle près. Par conséquent, elle a huit degrés de liberté. Chaque correspondance entre les deux plans donnant deux contraintes sur l'homographie, il en faut nécessairement quatre pour la définir sans ambiguïté.

Projection d'un plan

Si une matrice de projection $\mathsf{M}_{(3\times4)}$ effectue la projection de l'image d'un plan sur un autre plan, il existe une homographie reliant les systèmes de coordonnées du plan et de l'image. Celle-ci est donnée par :

$$\mathsf{H} = \begin{bmatrix} \mathsf{M}^1 & \mathsf{M}^2 & \mathsf{M}^4 \end{bmatrix}.$$

En effet, un point de ce plan est donné en 3D par $(X, Y, 0, H)^\mathsf{T}$. On vérifie alors que :

$$\mathsf{M}(X, Y, 0, H)^\mathsf{T} = \mathsf{H}(X, Y, H)^\mathsf{T}.$$

2.5 *Formation d'image et modèles*

L'acquisition d'une image 2D d'un monde 3D par une caméra, ou tout autre système d'acquisition, est le fruit d'un processus complexe touchant l'optique et l'électronique. Les modèles de caméra utilisés en vision sont utiles pour décrire les aspects géométriques du *processus de formation d'image*. Il s'agit de modèles géométriques plus ou moins complexes permettant de relier ce qui se trouve dans l'image avec les objets en 3D.

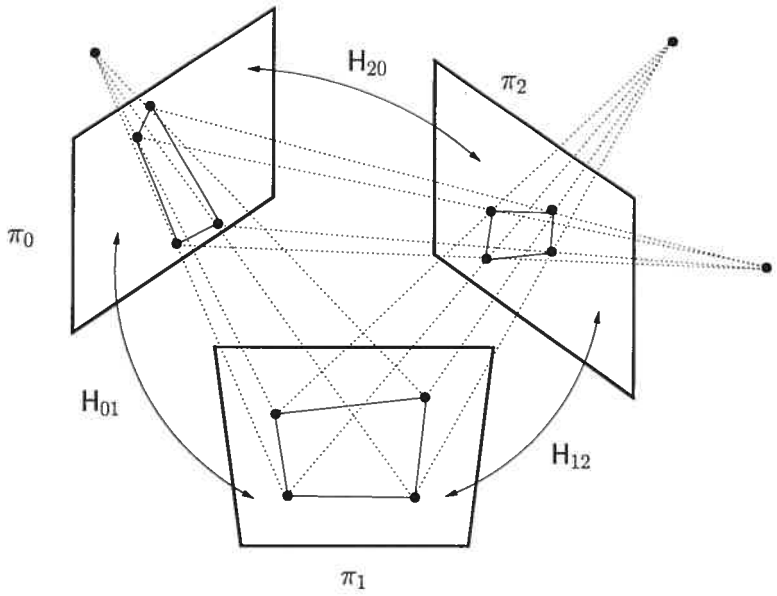


FIG. 2.2. **Relation linéaire projective entre les plans π_0, π_1 et π_2. H_{ij} est la relation entre le plan π_i et π_j. La relation inverse est $H_{ji} = H_{ij}^{-1}$.**

2.5.1 Optique de Gauss

Lorsque la lumière est capturée par l'objectif d'une caméra, elle traverse plusieurs groupes de lentilles. Le modèle à lentille mince reproduit approximativement les phénomènes qui se produisent en supposant que l'objectif n'est composé que d'une seule lentille mince. Celui-ci suppose que la réfraction de la lumière ne se fait qu'en un seul point de la lentille, plutôt qu'à son entrée et à sa sortie. Un modèle à lentille mince peut donc être caractérisé par la position de la lentille et la distance des deux points focaux (frontal et arrière) par rapport à la lentille (figure 2.3). Cette simplification permet deux observations. Tout d'abord, l'ensemble des rayons émanant d'un point 3D et traversant la lentille se recoupent en un même point (figure 2.3a). De plus, pour un ensemble de points situés sur un plan perpendiculaire à l'axe optique, l'intersection de leurs rayons se fait dans le même plan (figure 2.3b).

Ce modèle permet d'introduire le concept de *mise au point*. Lorsque le plan image

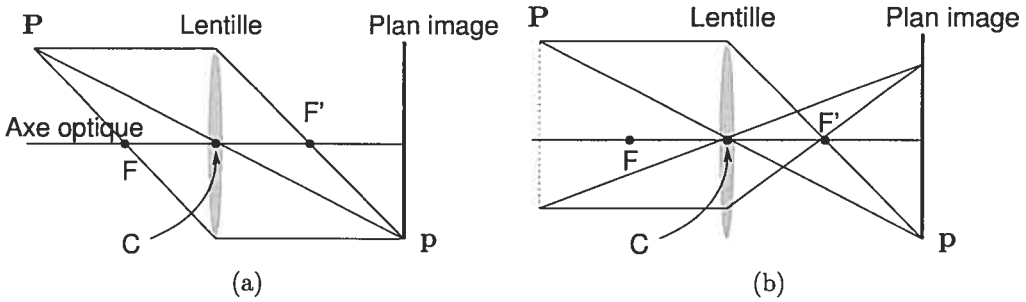


FIG. **2.3. Modèle de caméra à lentille mince. (a) Les trois rayons principaux en partance de P se coupent en p sur le plan image. (b) Pour deux points sur un plan, le premier est au point focal si et seulement si le deuxième l'est aussi.**

passe par l'intersection de rayons provenant d'un point 3D, on dit que ce dernier est au point focal. Finalement, résultat de la deuxième observation, un seul plan 3D perpendiculaire à l'axe optique est réellement au point focal. En photographie (ou en pratique), on parle plutôt de profondeur de champs. Il s'agit d'un intervalle de distances de l'avant à l'arrière du plan au point focal. L'espace défini par ce domaine correspond à l'ensemble des points au "foyer" pour la caméra.

Une autre caractéristique importante au sujet du modèle est que pour un point 3D, il existe plusieurs rayons de projection menant au point image. De plus, il est possible de bloquer une partie des rayons et tout de même d'obtenir une image de ce point 3D. En vision conventionnelle, ce modèle est souvent trop complexe et l'optique de Gauss est très peu utilisée. Cependant, certains travaux démontrent qu'il faut garder en tête ce genre de phénomène lorsque l'ouverture de l'objectif est très grand (*cf.*§4.4) [82].

2.5.2 Modèle sténopé

Le modèle sténopé[1], illustré figure 2.4, est le modèle de caméra le plus utilisé en vision. Il repose sur une simplification du phénomène associé à l'optique des lentilles minces faisant l'hypothèse d'une ouverture infiniment petite à la place de l'objectif

[1] *Pin-hole*

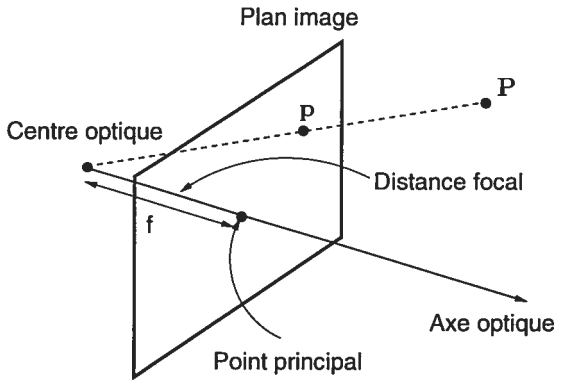


FIG. 2.4. **Modèle de caméra sténopé.**

de la caméra. Chaque point 3D est alors projeté exactement à un seul endroit du plan image. Par conséquent, la notion de mise au point (focus) n'existe plus et, de façon équivalente, la caméra a une profondeur de champs infinie. En pratique, les caméras ayant une optique de qualité sans aucune distorsion d'image en plus d'une ouverture très petite sont bien représentées par le modèle sténopé[2].

La matrice de projection d'une caméra sténopé est de type perspective et possède 11 degrés de liberté (ddl). On décrit une caméra située à la position $\tilde{\mathbf{C}} \in \mathbb{R}^3$ de l'espace par sa matrice de projection $\mathsf{M}_{(3\times4)} = \mathsf{K}[\mathsf{R}|-\mathsf{R}\tilde{\mathbf{C}}]$ où, $\mathsf{K}_{(3\times3)}$ est la matrice des paramètres internes (5 ddl) et les paramètres externes, $\mathsf{R}_{(3\times3)}$ et $\tilde{\mathbf{C}}$, sont respectivement l'orientation (3 ddl chacun) et la position (3 ddl). La matrice $[\mathsf{R}|-\mathsf{R}\tilde{\mathbf{C}}]$ effectue la transformation (rotation et translation) d'un point 3D du monde en un point 3D dans le système de coordonnées caméra. Il s'agit d'un espace où la caméra est située à l'origine, l'axe optique est aligné avec l'axe Z et les axes du plans image sont parallèles aux axes X et Y. La projection d'un point 3D $\mathbf{P} \in \mathbb{P}^3$ en un point image $\mathbf{p} \in \mathbb{P}^2$ se fait selon :

$$\mathbf{p} = h(x, y, 1)^\top \propto \mathsf{M}\mathbf{P} = \mathsf{K}[\mathsf{R}|-\mathsf{R}\tilde{\mathbf{C}}]\mathbf{P}.$$

[2] Une ouverture plus petite ou égale à F11 donne généralement de bons résultats.

Puisque h est la composante homogène de la coordonnée, le point image est (x, y). La déprojection d'un point image donne un rayon 3D défini par l'équation paramétrique :

$$\mathbf{P}(\lambda) = \mathsf{M}^{+} + \lambda\mathbf{C} = \underbrace{\mathsf{M}^{\top}[\mathsf{M}\mathsf{M}^{\top}]^{-1}}_{\mathsf{M}^{\dagger}} + \lambda\mathbf{C},$$

où, λ varie de 0 à l'infini et $\mathsf{M}^{\dagger}$ est la pseudo-inverse de M (*cf.*§3.4.1). La matrice K est une matrice triangulaire supérieure de la forme :

$$\begin{bmatrix} \alpha f & \alpha \cos\theta & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

où, α est le ratio d'aspect, f la focale, θ l'inclinaison des pixels et (c_x, c_y) le point principal, *i.e.* l'intersection de l'axe optique de la caméra avec le plan image.

2.5.3 Modèle de caméra affine

Le modèle de caméra affine inclue les caméras orthographique et perspective faible. Leur matrice de projection est donnée de façon générale par :

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & Z_r \end{bmatrix}$$

ce qui correspond à un projection orthographique des points sur le plan Z_r suivie d'une projection perspective classique. En choisissant $Z_r = 1$, on obtient donc une projection orthographique. En choisissant un Z_r plus grand, on fait l'hypothèse que la profondeur des points 3D est de Z_r. On peut aussi y ajouter à gauche une matrice de paramètres internes traditionnelle.

Le noyau droit de ce genre de matrice de projection nous indique cependant que

le centre de projection est un point à l'infini. Cela peut se comprendre en observant qu'une caméra affine est une caméra perspective de matrice de projection $\mathsf{K}\mathsf{R}[\mathtt{I}|-\tilde{\mathbf{C}}]$, dont le centre de projection est reculé vers l'arrière sur l'axe optique et dont la focale est augmentée graduellement (*cf.*figure 2.5). Ceci est équivalent à modifier la matrice de projection comme suit :

$$\mathsf{M}_t = \mathsf{K}\begin{bmatrix} \mathbf{R}_1^\mathsf{T} & -\mathbf{R}_1^\mathsf{T}(\tilde{\mathbf{C}} - t\,\mathbf{R}_3) \\ \mathbf{R}_2^\mathsf{T} & -\mathbf{R}_2^\mathsf{T}(\tilde{\mathbf{C}} - t\,\mathbf{R}_3) \\ \mathbf{R}_3^\mathsf{T} & -\mathbf{R}_3^\mathsf{T}(\tilde{\mathbf{C}} - t\,\mathbf{R}_3) \end{bmatrix}$$

où, t est le déplacement le long de l'axe optique donné par $\mathbf{R}_3$. On peut alors montrer qu'en poussant ce déplacement à l'infini, on obtient une matrice de projection de la forme :

$$\lim_{t\to\infty} \mathsf{M}_t = \mathsf{K}\begin{bmatrix} \mathbf{R}_1^\mathsf{T} & -\mathbf{R}_1^\mathsf{T}\tilde{\mathbf{C}} \\ \mathbf{R}_2^\mathsf{T} & -\mathbf{R}_2^\mathsf{T}\tilde{\mathbf{C}} \\ \mathbf{0}^\mathsf{T} & d_0 \end{bmatrix} = \underbrace{\mathsf{K}\mathrm{diag}\left(\frac{1}{d_0}, \frac{1}{d_0}, 1\right)}_{\mathsf{K}'} \begin{bmatrix} \mathbf{R}_1^\mathsf{T} & -\mathbf{R}_1^\mathsf{T}\tilde{\mathbf{C}} \\ \mathbf{R}_2^\mathsf{T} & -\mathbf{R}_2^\mathsf{T}\tilde{\mathbf{C}} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix}$$

où d_0 est simplement un facteur d'échelle pour conserver la taille de l'image et K' est une nouvelle matrice de paramètre interne. Finalement, on peut réécrire le tout à l'aide d'une matrice de projection affine (2×4) :

$$\tilde{\mathbf{p}} = \mathsf{K}'\begin{bmatrix} \mathbf{R}_1^\mathsf{T} & -\mathbf{R}_1^\mathsf{T}\tilde{\mathbf{C}} \\ \mathbf{R}_2^\mathsf{T} & -\mathbf{R}_2^\mathsf{T}\tilde{\mathbf{C}} \end{bmatrix} \tilde{\mathbf{P}},$$

comme nous l'avons vu §2.2.1.

2.5.4 Autres modèles

Une description exhaustive de plusieurs modèles de caméra assez couramment utilisés, mais différents des modèles de caméra sténopé et affine, est donnée à la

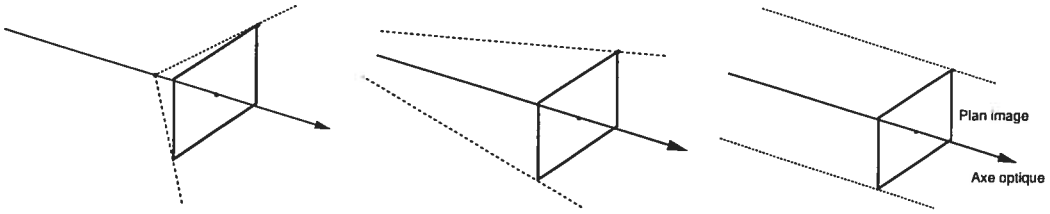


FIG. **2.5. Effet de déplacement du centre optique le long de l'axe optique à une distance infinie du plan image. De gauche à droite : grand angle de vue, petit angle de vue et affine.**

référence [60]. Nous verrons au chapitre 4 qu'il existe aussi une panoplie de modèles de projection beaucoup plus élaborés et mieux adaptés aux caméras à très grand angle de vue. À quelques exceptions près, les modèles ont un point en commun : chaque pixel est associé à un seul rayon 3D d'échantillonage. Cette contrainte est nécessaire pour que la projection d'un point 3D dans une image demeure assez facile à définir. Néanmoins, pour certains modèles complexes, il n'est pas toujours simple d'effectuer cette opération de projection.

2.6 Matrices fondamentale et essentielle

Pour deux images de caméra sténopé d'une même scène prise de points de vues différents, il existe une relation entre les points de correspondances. Une correspondance est la projection d'un même point 3D dans deux images de caméra.

Cette relation, appelée relation épipolaire[3], est décrite par les *matrices fondamentale*[4] F et *essentielle*[5] E. Pour que de telles matrices soient définies, les centres de projection des deux caméras doivent être différents (translation non-nulle), sinon la relation de mise en correspondance est dégénérée et s'explique plutôt avec une transformation linéaire projective (§2.4).

[3] Epipolar relation

[4] Fundamental matrix

[5] Essential matrix

2.6.1 Matrice fondamentale

Prenons deux caméras M et M′, de centres de projection respectifs $\mathbf{C}$ et $\mathbf{C}'$ ($\mathbf{C} \neq \mathbf{C}'$). Un point $\mathbf{p}$ dans la première image est déprojetée en 3D sur la ligne paramétrique $\mathbf{P}(\lambda) = \mathsf{M}^{\dagger}\mathbf{p} + \lambda\mathbf{C}$. À l'aide de cette formule, il est utile de définir deux points situés sur la ligne : $\mathsf{M}^{\dagger}\mathbf{p}$ et $\mathbf{C}$ (pour $\lambda = \infty$). En projetant ces deux points dans la seconde image, puis en les joignant, on obtient la projection de $\mathbf{P}(\lambda)$:

$$\mathbf{l}' = \underbrace{(\mathsf{M}'\mathbf{C})}_{\mathbf{e}'} \times (\mathsf{M}'\mathsf{M}^{\dagger}\mathbf{p}),$$

où, $\mathbf{e}'$ est *l'épipole* dans la deuxième image, défini comme étant la projection du centre optique de la première caméra dans l'image de la deuxième. De façon similaire, on peut définir l'épipole $\mathbf{e}$ dans la première image. L'équation peut alors se réécrire :

$$\mathbf{l}' = \mathbf{e}' \times (\mathsf{M}'\mathsf{M}^{\dagger}\mathbf{p}) = [\mathbf{e}']_{\times}\mathsf{M}'\mathsf{M}^{\dagger}\mathbf{p}.$$

Finalement, comme $\mathbf{p}'$ est quelque part sur $\mathbf{l}'$, nous obtenons que :

$$\mathbf{p}'^{\mathsf{T}}\mathbf{l}' = \mathbf{p}'^{\mathsf{T}} \underbrace{[\mathbf{e}']_{\times}\mathsf{M}'\mathsf{M}^{\dagger}}_{\mathsf{F}_{(3\times3)}}\mathbf{p} = 0$$

où, F est appelée matrice fondamentale entre les deux caméras. En résumé, si $\mathbf{p}$ et $\mathbf{p}'$ sont les images d'un même point 3D observé par deux caméras, ils satisfont la relation $\mathbf{p}'^{\mathsf{T}}\mathsf{F}\mathbf{p} = 0$.

2.6.2 Matrice essentielle

Lorsque les paramètres internes de la caméra sont connus, il existe une relation épipolaire calibrée représentée par la matrice essentielle E. À la différence de la matrice fondamentale, E agit sur les points caméra plutôt que sur les points images. Cette

matrice est donnée par $E_{(3\times3)} = K'^{T}FK$.

Chapitre 3

ÉLÉMENTS D'OPTIMISATION CONTINUE

3.1 Introduction

Ce chapitre présente la plupart des problèmes d'optimisation auxquels nous avons été confrontés au cours de l'élaboration de cette thèse ainsi que des méthodes d'optimisation que nous avons utilisées pour les solutionner. Le terme *optimisation* est utilisé dans plusieurs contextes de la vision par ordinateur. En informatique, on y réfère souvent comme le processus permettant d'améliorer l'efficacité d'un système ou d'un programme. Cette tâche d'optimisation comporte donc des aspects autant algorithmiques, particulièrement reliés la notion de *complexité*, que pratiques, tel l'utilisation la plus efficace du processeur et de la mémoire de l'ordinateur. D'un point de vue mathématique, l'optimisation réfère à la minimisation (ou maximisation) d'une fonction objectif sujette ou non à des contraintes. Bien entendu, nous sommes le plus souvent intéressés par la valeur et la localisation du ou des minima de la fonction. En *optimisation numérique*, les aspects informatiques et mathématiques de l'optimisation sont considérés puisque l'objectif est de calculer les minima d'une fonction le plus efficacement possible. Pour une description plus complète de l'optimisation numérique, consulter [22].

Les problèmes d'optimisation peuvent être classés en deux catégories : convexe ou non-convexe. En général, les problèmes convexes sont beaucoup plus faciles à solutionner que les problèmes non-convexes. De plus, les méthodes de résolution de problèmes non-convexes requièrent le plus souvent un estimé du minimum global pour être fiable. Malheureusement, les problèmes de vision par ordinateur requiert presque toujours l'optimisation de fonctions non-convexes. Une grande part des recherches

consiste à approcher le mieux possible les fonctions non-convexes par des fonctions convexes. Idéalement, la solution correspondant au minimum global de la fonction convexe se trouve assez près de celle correspondant au minimum de la fonction non-convexe.

3.2 Cadre général de l'optimisation

On peut décrire un problème d'optimisation comme la minimisation d'une fonction objectif $f : \mathbb{R}^n \rightarrow \mathbb{R}$ sujette à des contraintes d'inégalité $g_i : \mathbb{R}^n \rightarrow \mathbb{R}$ et des contraintes d'égalité $h_j : \mathbb{R}^n \rightarrow \mathbb{R}$. Dans sa forme standard, un tel problème s'écrit[1] :

$$\begin{aligned} \arg\min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{tel que} \quad & g_i(\mathbf{x}) \leq 0, i = 1, \ldots, m \\ & h_j(\mathbf{x}) = 0, j = 1, \ldots, p. \end{aligned} \tag{3.1}$$

L'ensemble des points qui sont définis pour f et qui satisfont aux contraintes g_i et h_i constitue le domaine de f, noté dom$f$. À noter que les contraintes d'égalité ne sont pas formellement nécessaires puisqu'elles peuvent toujours être remplacées par des inégalités. Selon la forme de f et des g_i et h_j, un tel problème d'optimisation est plus ou moins difficile à résoudre. Dans les cas les plus simples, il existe une solution analytique et dans les plus difficiles, le problème est NP-complet. Par exemple, il existe une solution analytique lorsque f est une fonction quadratique à plusieurs variables et qu'il n'y pas de contrainte. Le problème est toutefois NP-difficile dans presque toutes les situations où les éléments de $\mathbf{x}$ sont contraints à prendre la valeur 0 ou 1. Il s'agit alors d'un problème de programmation binaire où les contraintes h_j prennent la forme $h_j(\mathbf{x}) = x_j^2 - x_j$. Un problème peut aussi être non-borné ou *infaisable* (impossible à solutionner), *i.e.* lorsque dom$f$ est vide.

[1] Cette forme implique qu'il y ait des 0 dans la partie droite des égalités et inégalités.

3.2.1 Minimum global et local, point critique

En général, nous sommes intéressés par le ou les minima globaux d'une fonction. On dit de $\mathbf{x}^*$ qu'il est un *minimum global* si :

$$\forall \mathbf{x} \in \mathrm{dom} f : f(\mathbf{x}^*) \leq f(\mathbf{x}).$$

On dit de $\mathbf{x}^*$ qu'il est un *minimum local* de f si :

$$\exists\ \phi \quad \text{tel que} \quad \forall \mathbf{x} \in \mathrm{dom} f : \|\mathbf{x} - \mathbf{x}^*\| \leq \phi \text{ et que nous avons} f(\mathbf{x}^*) \leq f(\mathbf{x}).$$

Par cette définition, il est possible d'avoir plusieurs minima globaux à la fois. La notion de minimum global n'est pas particulièrement importante dans le contexte générale d'optimisation puisqu'elle ne donne pas de méthode pour vérifier si un point $\mathbf{x}$ est un minimum global ou non (à moins que la fonction soit convexe). À l'opposé, il est facile de vérifier si un point est un minimum local. Ceci fait appel à la notion de *point critique*, *point stationnaire* et *col*[2]. Un point stationnaire est un point pour lequel le gradient de la fonction égale 0. Un point critique est soit un point stationnaire, soit un point où la dérivé de la fonction n'existe pas. Finalement, un col est un point stationnaire qui est aussi un point d'inflexion (*cf*.figure 3.1).

Le théorème de Fermat nous donne des conditions nécessaires (mais pas suffisantes) pour identifier les minima locaux. Le point $\mathbf{x}_0$ est un minimum local si l'une des conditions suivantes est respectée :

- $\mathbf{x}_0$ est un point stationnaire ;
- f n'est pas différentiable en $\mathbf{x}_0$;
- $\mathbf{x}_0$ est sur la frontière de $\mathrm{dom} f$;

Pour identifier les minima locaux d'une fonction, il est possible d'utiliser le test

[2] *Saddle point*

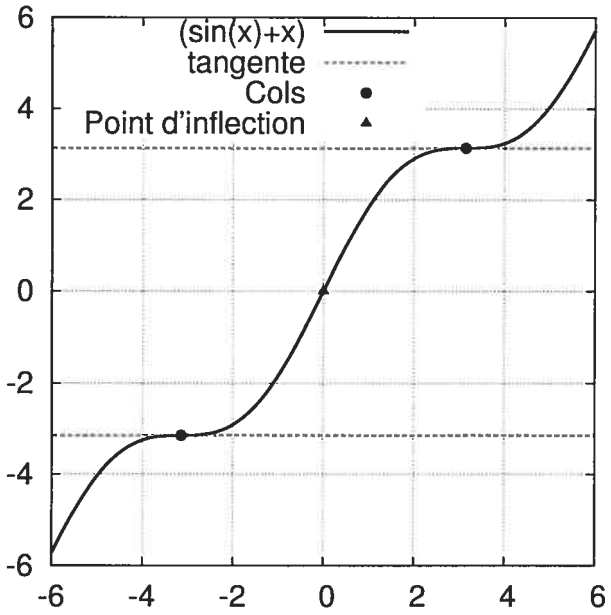


FIG. 3.1. **Exemple de point d'inflexion et de cols.**

de la dérivée seconde. Le test utilise la matrice Hessienne de la fonction f

$$\nabla^2 f(\mathbf{x}) = D\nabla f(\mathbf{x}) = \left(\frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j}\right).$$

Ainsi, un point stationnaire $\mathbf{x}_0$ est :

- un minimum local si les valeurs propres de $\nabla^2 f(\mathbf{x}_0)$ sont toutes positives ;
- un maximum local si les valeurs propres de $\nabla^2 f(\mathbf{x}_0)$ sont toutes négatives ;
- un col si les valeurs propres de $\nabla^2 f(\mathbf{x}_0)$ sont positives et négatives,

et le test échoue lorsque $\nabla^2 f(\mathbf{x}_0)$ est singulière. Pour une fonction f à une variable x et doublement différentiable, une condition suffisante pour qu'un point soit un minimum est donc que

$$f_0'(x) = 0 \text{ et } f_0''(x) > 0.$$

De plus, si x est un minimum local, il est possible qu'il soit aussi un minimum global.

3.2.2 Problèmes convexes et non-convexes

Un problème non-convexe est constitué d'une fonction objectif ou d'un domaine non-convexe. En général, ces problèmes sont difficiles à résoudre. Pour qu'un problème soit convexe, il doit être constitué d'une fonction objectif convexe sur un domaine convexe. En général, les problèmes convexes sont assez faciles à optimiser, du moins dans les cas qui nous intéressent ici. Une fonction convexe f vérifie que :

$$\forall \mathbf{x}, \mathbf{x}' \in \mathrm{dom} f, \theta \in [0,1] : f(\theta \mathbf{x} + (1-\theta)\mathbf{x}') \leq \theta f(\mathbf{x}) + (1-\theta) f(\mathbf{x}')$$

i.e. que l'approximation de Taylor du premier ordre est une sous-estimation globale de la fonction, et que :

$$\forall \mathbf{x} : \nabla^2 f(\mathbf{x}) \succ 0,$$

i.e. que la matrice Hessienne est définie positive. Ces propriétés impliquent que la fonction ne possède qu'un seul minimum (global). Le problème (3.1) est convexe si f, g_i et h_j sont convexes pour tout i, j.

3.3 Problèmes convexes

Nous portons notre attention sur différentes variantes de deux problèmes rencontrés à maintes reprises au cours de cette thèse : *quadratique* et *homogène linéaire*, avec ou sans contrainte et creux[3] ou non.

[3] *Sparse*

3.3.1 Problèmes quadratiques

Un problème quadratique[4] (QP), dans sa forme générale, a la forme suivante :

$$\begin{aligned} \arg\min_{\mathbf{x}} \quad & \frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathsf{Q}\mathbf{x} + \mathbf{q}^{\mathsf{T}}\mathbf{x} + r \\ \text{tel que} \quad & \mathsf{G}_{(m\times n)}\mathbf{x} \preceq \mathbf{h} \\ & \mathsf{A}_{(p\times n)}\mathbf{x} = \mathbf{b}. \end{aligned}$$

À notez que les contraintes sont maintenant linéaire. Lorsque $\mathsf{Q} \in \mathbb{S}^k_+$ (de dimension $(k \times k)$), il est possible que le problème possède plusieurs minima globaux. Lorsque $\mathsf{Q} \in \mathbb{S}^k_{++}$, il n'y a qu'un seul minimum. Ces deux situations ont été rencontrées au chapitre 9. Finalement, si Q possède à la fois des valeurs propres positives et négatives, ce problème devient NP-complet [108].

Programmation linéaire

Les problèmes quadratiques tels que Q égale 0 sont appelés *problèmes linéaires*[5] (LP). Naturellement, une méthode permettant de résoudre un problème de programmation quadratique peut aussi résoudre un problème de programmation linéaire. Il existe toutefois des algorithmes spécifiques aux problèmes LP telle la méthode du simplexe, et, probablement encore mieux, la méthode du point intérieur de Karmarkar [2]. Une méthode s'appliquant aux problèmes QP est présentée §3.4.8.

---

[4] *Quadratic program*

[5] *Linear program*

QP sans contrainte : régression linéaire

Le problème de régression linéaire au sens des moindres carrés est un problème QP sans contrainte. En effet, on cherche $\mathbf{x}$ telle que la fonction suivante est minimisée :

$$\|\mathsf{A}\mathbf{x} - \mathbf{b}\|_2^2 = \mathbf{x}^\mathsf{T} \underbrace{\mathsf{A}^\mathsf{T}\mathsf{A}}_{\mathsf{Q}} \mathbf{x} \underbrace{-2\mathbf{b}^\mathsf{T}\mathsf{A}}_{2\mathbf{q}^\mathsf{T}} \mathbf{x} + \underbrace{\mathbf{b}^\mathsf{T}\mathbf{b}}_{2r}.$$

Notez ici que $\mathsf{A}^\mathsf{T}\mathsf{A} \in \mathbb{S}^k_{++}$ si A est de plein rang, ce qui implique que le minimum global est unique.

Optimisation non-linéaire et non-convexe

Une des méthodes d'optimisation non-linéaire les plus populaires est la méthode de Newton. Nous la résumons §3.4.7. Cette méthode peut être employée pour l'optimisation de fonctions non-linéaires convexes ou non. Pour le moment, l'important est de savoir que cette méthode fonctionne par itération, chacune d'elles étant un problème quadratique avec ou sans contrainte linéaire. Ainsi, la résolution de problèmes quadratiques est essentielle pour la minimisation de fonctions non-linéaires en général.

3.3.2 Systèmes linéaires homogènes

Lorsqu'un problème QP sans contrainte a $\mathbf{q} = \mathbf{0}$ ou, de façon équivalente, une régression linéaire a $\mathbf{b} = \mathbf{0}$, la solution optimale au problème est $\mathbf{x} = \mathbf{0}$. Évidemment, cette solution ne nous intéresse pratiquement jamais. On ajoute alors une contrainte à la norme de $\mathbf{x}$ pour éviter cette solution. Ce problème possède une solution simple basée sur la SVD (voir plus bas). On cherche donc :

$$\arg\min_{\mathbf{x}} \|\mathsf{A}\mathbf{x}\| \quad \text{tel que} \quad \|\mathbf{x}\| = 1.$$

3.4 Méthodes de résolution de problèmes convexes

Dans cette section, nous décrivons différents algorithmes de minimisation de plusieurs types de problèmes quadratiques ou linéaires homogènes.

3.4.1 Décomposition en valeurs singulières (SVD)

La décomposition en valeur singulière[6] (SVD) d'une matrice réelle $\mathsf{A}_{(m\times n)}, m \geq n$ est donnée par :

$$\mathsf{A} = \mathsf{U}_{(m\times n)}\Sigma(V_{(n\times n)})^{\mathsf{T}}$$

où, $\mathsf{U}\mathsf{U}^{\mathsf{T}} = \mathtt{I}$, $\mathsf{V}\mathsf{V}^{\mathsf{T}} = \mathtt{I}$ et $\Sigma = \mathrm{diag}(\sigma_1, ..., \sigma_n)$, σ_i étant les valeurs singulières de A telles que

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0, \quad \sigma_{r+1} = ... = \sigma_n = 0$$

et r est le rang de A. Les colonnes de U sont les vecteurs singuliers gauches et les colonnes de V sont les vecteurs singuliers droits. Ces vecteurs peuvent être interprétées respectivement comme des bases orthonormales de l'espace rangée et de l'espace colonne de A. Cette propriété sera utile aux chapitres 13 et 14.

Pseudo-inverse

La pseudo-inverse $\mathsf{A}^{\dagger}$ d'une matrice A est souvent définie par la formule analytique $\mathsf{A}^{\dagger} = (\mathsf{A}^{\mathsf{T}}\mathsf{A})^{-1}\mathsf{A}^{\mathsf{T}}$. Cette formule est toutefois sujette a des instabilités numériques. De plus, elle n'est pas applicable lorsque $\mathsf{A}^{\mathsf{T}}\mathsf{A}$ est singulière, c'est-à-dire quand A n'est pas de plein rang. Une meilleure définition, utilisant la SVD de $\mathsf{A} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$, est :

$$\mathsf{A}^{\dagger} = \mathsf{V}\Sigma^{\dagger}\mathsf{U}^{\mathsf{T}}.$$

[6] *Singular Value Decomposition*

Puisque Σ est une matrice diagonale, sa pseudo-inverse peut être calculée de façon efficace par la formule :

$$\sigma_{ii}^{\dagger} = \begin{cases} 0 \text{ pour } \sigma_{ii} = 0 \\ {\sigma_{ii}}^{-1} \text{ sinon.} \end{cases}$$

La pseudo-inverse est utile notamment pour la résolution de systèmes d'équations linéaires avec un critère des moindres carrées.

3.4.2 QP sans contrainte : méthode par pseudo-inverse

Nous cherchons $\mathbf{x}$ tel que la fonction $\frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathsf{Q}\mathbf{x} + \mathbf{q}^{\mathsf{T}}\mathbf{x} + r$ est minimisée. Lorsque $\mathsf{Q} \succ 0$, il est possible de la décomposer en $\mathsf{Q} = \mathsf{A}^{\mathsf{T}}\mathsf{A}$ par la décomposition de Cholesky [51]. Ignorant les termes constants et en multipliant par 2, on peut ainsi réécrire le problème sous forme de régression linéaire : $\|\mathsf{A}\mathbf{x} - \mathbf{b}\|_2^2$, où, $\mathbf{b} = \mathsf{A}^{-\mathsf{T}}\mathbf{q}$. Ce problème possède une solution unique donnée par :

$$\mathbf{x}^* = \mathsf{A}^{\dagger}\mathbf{b},$$

car A est de plein rang. Lorsque $\mathsf{Q} \succeq 0$, il y a plusieurs minima globaux. Pour le démontrer, il est plus simple d'utiliser le gradient de la fonction qui doit être égale à $\mathbf{0}$:

$$\mathsf{Q}\mathbf{x} + \mathbf{q} = \mathbf{0}$$

ce qui revient à résoudre : $\mathsf{Q}\mathbf{x} = -\mathbf{q}$. Comme Q n'est pas de plein rang, soit il n'y a pas de solution (la fonction est non-bornée), soit il existe une infinité de solutions données par :

$$\mathbf{x}^*(\mathbf{s}) = -\mathsf{Q}^{\dagger}\mathbf{q} + \mathsf{V}'\mathbf{s}, \quad \mathbf{s} \in \mathbb{R}^{n-r}$$

où, V' est constitué des colonnes de V correspondant aux $(n-r)^{\text{ième}}$ valeurs singulières nulles de la SVD $\mathsf{Q} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$. En d'autres mots, les colonnes de V' forment une base orthonormale du noyau de Q.

3.4.3 QP sans contrainte avec matrice creuse : méthode spécifique

Le choix d'un algorithme de minimisation dépend de la structure et de la taille de A. Il existe plusieurs publications sur le sujet (un bon résumé est fourni dans la documentation de la fonction *mldivide* de MATLAB).

Matrice creuse par blocs

Lorsque la matrice A est de très grande taille et creuse, il est parfois possible de construire des algorithmes de résolution spécifiques aux problèmes qui exploitent la structure creuse de A. Une telle situation a été rencontrée aux chapitres 13 et 14. Par exemple, on cherche :

$$\arg\min_{X,Y} \left\| \underbrace{\begin{pmatrix} A & B^T \\ B & D \end{pmatrix}}_{\mathcal{A}} \begin{pmatrix} X \\ Y \end{pmatrix} - \begin{pmatrix} E \\ F \end{pmatrix} \right\|^2 . \tag{3.2}$$

où, $\mathcal{A}$ est une matrice creuse en forme de flèche, *i.e.* A et B sont denses et D est bloc diagonale. L'objectif est de résoudre (3.2) sans explicitement inverser la matrice $\mathcal{A}$. Nous avons :

$$AX + B^TY = E \tag{3.3}$$

$$BX + DY = F. \tag{3.4}$$

En résolvant Y dans (3.4) et en remplaçant dans (3.3), on obtient après simplification :

$$\left(A - B^TD^{-1}B\right) X = E - B^TD^{-1}F,$$

ce qui permet de calculer X efficacement en prenant soin d'inverser D bloc par bloc. On peut ensuite calculer Y en substituant X dans (3.3).

3.4.4 QP avec contraintes linéaires

Un problème quadratique avec seulement des contraintes linéaires peut aussi se résoudre de façon analytique :

$$\begin{aligned} \arg\min_{\mathbf{x}} \quad & \frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathsf{Q}\mathbf{x} + \mathbf{q}^{\mathsf{T}}\mathbf{x} + r \\ & \mathsf{A}_{(p\times n)}\mathbf{x} = \mathbf{b}. \end{aligned}$$

À l'aide des multiplicateurs de Lagrange, on obtient que la solution doit satisfaire :

$$\mathsf{Q}\mathbf{x} + \mathbf{q} + \mathsf{A}^{\mathsf{T}}\lambda = 0, \quad \mathsf{A}\mathbf{x} = \mathbf{b}, \quad \lambda \in \mathbb{R}^p$$

où, λ est un multiplicateur de Lagrange. Ces équations peuvent être écrites sous forme matricielle :

$$\underbrace{\begin{bmatrix} \mathsf{Q} & \mathsf{A}^{\mathsf{T}} \\ \mathsf{A} & 0 \end{bmatrix}}_{\mathsf{P}} \begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} = \begin{pmatrix} -\mathbf{q} \\ \mathbf{b} \end{pmatrix}.$$

où, P est carré. Par conséquent, on peut solutionner ce problème exactement.

Cette méthode analytique est utilisée entre autres par la méthode de Newton permettant de minimiser des fonctions avec des contraintes linéaires.

3.4.5 Systèmes homogènes : méthode par SVD

La solution d'un système homogène :

$$\arg\min_{\mathbf{x}} \|\mathsf{A}\mathbf{x}\| \quad \text{tel que} \quad \|\mathbf{x}\| = 1 \tag{3.5}$$

est donnée par la dernière colonne de V de la SVD de $\mathsf{A} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$, *i.e.* le vecteur singulier droit correspondant à la plus petite valeur singulière. Nous donnons ici deux justifications pour ce résultat.

Argument par SVD

L'essentiel de l'argument utilise la propriété d'orthonormalité de U et V qui stipule que pour une telle matrice M, $\forall \mathbf{z} : \|\mathsf{M}\mathbf{z}\| = \|\mathbf{z}\|$. Ainsi :

$$\begin{aligned} \|\mathsf{A}\mathbf{x}\| &= \left\|\mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}\mathbf{x}\right\| \\ &= \left\|\Sigma\mathsf{V}^{\mathsf{T}}\mathbf{x}\right\|. \end{aligned}$$

En renommant $\mathbf{y} = \mathsf{V}^{\mathsf{T}}\mathbf{x}$, notre nouveau problème est :

$$\arg\min_{\mathbf{y}} \|\Sigma\mathbf{y}\| \text{ sujet à } \left\|\mathsf{V}^{\mathsf{T}}\mathbf{y}\right\| = \|\mathbf{y}\| = 1$$

et comme Σ est diagonale avec éléments ordonnés en ordre décroissant, la solution minimale est donnée par

$$\mathbf{y} = \begin{pmatrix} 0 & \dots & 0 & 1 \end{pmatrix}^{\mathsf{T}}.$$

Puisque $\mathsf{V}\mathsf{V}^{\mathsf{T}} = \mathtt{I}$, $\mathbf{x}$ est donné par :

$$\mathsf{V}\begin{pmatrix} 0 & \dots & 0 & 1 \end{pmatrix}^{\mathsf{T}}$$

i.e. la dernière colonne de V.

Argument par multiplicateur de Lagrange

Notons que la fonction objectif et la contrainte $\|\mathbf{x}\| = 1$ peuvent être réécrites :

$$\mathbf{x}^{\mathsf{T}}\mathsf{A}^{\mathsf{T}}\mathsf{A}\mathbf{x} \quad \text{tel que} \quad 1 - \mathbf{x}^{\mathsf{T}}\mathbf{x} = 0.$$

Nous pouvons donc formuler le problème comme celui de résoudre exactement :

$$\begin{aligned} \frac{\partial}{\partial \mathbf{x}} \left(\mathbf{x}^T \mathsf{A}^T \mathsf{A} \mathbf{x} + \lambda (1 - \mathbf{x}^T \mathbf{x}) \right) &= \\ 2 \left(\mathsf{A}^T \mathsf{A} \mathbf{x} - \lambda \mathrm{I} \right) \mathbf{x} &= 0. \end{aligned}$$

où, λ est un multiplicateur de Lagrange. Les solutions sont données par les vecteurs propres de $\mathsf{A}^T\mathsf{A}$ telles que λ est leur valeur associée. En multipliant l'équation précédente à gauche par $\mathbf{x}^T$, on obtient que

$$\|\mathsf{A}\mathbf{x}\|^2 = \mathbf{x}^T \mathsf{A}^T \mathsf{A} \mathbf{x} = \mathbf{x}^T \lambda \mathbf{x},$$

ce qui signifie que l'erreur est minimale pour le vecteur propre correspondant à la plus petite valeur propre. L'argument se conclut en notant que les vecteurs singuliers droits de A sont égaux aux vecteurs propres de $\mathsf{A}^T\mathsf{A}$ et que les valeurs singulières sont les valeurs propres au carré.

3.4.6 Système homogène : matrice creuse

Lorsque la matrice A est creuse et de très grande taille, par exemple plus que (2000×2000), une solution au problème (3.5) basée sur une SVD est peu efficace ou carrément impossible avec un espace mémoire raisonnable (quelques centaines de mégaoctets). Par ailleurs, il arrive très souvent que seuls les quelques premiers ou derniers vecteurs singuliers nous intéressent. Dans ce cas, il est préférable d'utiliser une méthode approximative par itérations. Un tel algorithme est disponible dans la librairie ARPACK [79], laquelle est utilisée par la méthode *eigs* de Matlab. La méthode employée se nomme *Implicitly restarted Arnoldi Method* et est basée sur la théorie de Krylov sur la projection des sous-espaces [5, 78].

Une deuxième méthode consiste à transformer le problème homogène en régression sans contrainte. Pour ce faire, un élément de $\mathbf{x}$ est fixé à une valeur non-nulle (1 en

général). Le problème devient :

$$\arg \min_{\mathbf{x}_{\{1\ldots n\}\setminus\{\rho\}}} \left\| \mathsf{A}^{\{1\ldots n\}\setminus\{\rho\}} \mathbf{x}_{\{1\ldots n\}\setminus\rho} + \mathbf{A}^{\{\rho\}} \mathbf{x}_{\{\rho\}} \right\|^2 \tag{3.6}$$

où, A^S est la matrice constituée des colonnes de A d'indices dans S et '\' est la différence d'ensembles. L'estimation pour $\mathbf{x}_{\{1\ldots n\}\setminus\{\rho\}}$ est ensuite combinée à $\mathbf{x}_{\{\rho\}}$ et le vecteur résultant est normalisé. En pratique, il faut tester aléatoirement plusieurs choix pour ρ puisqu'il est possible que $\mathbf{x}_{\{\rho\}}$ soit de valeur 0. Cette approche s'avère particulièrement efficace si la matrice $\mathsf{A}^{\{1\ldots n\}\setminus\{\rho\}}$ est d'une forme particulière pouvant être exploitée, comme expliqué §3.4.3.

3.4.7 Optimisation avec contraintes d'égalité : méthode de Newton

Lorsqu'un problème QP ne possède aucune contrainte, une solution analytique utilisant la pseudo-inverse peut être employée. Avec des contraintes linéaires d'égalité, il existe aussi une solution analytique. Lorsque la fonction objectif f est une fonction convexe pas nécessairement quadratique, il est possible de trouver le minimum global par des méthodes itératives, par exemple, la descente de gradient ou la méthode de Newton. Cette dernière est particulièrement populaire pour sa rapidité de convergence. Lorsqu'aucune contrainte n'est associée à la fonction objectif, cet algorithme est plutôt simple. Il s'agit d'un algorithme itératif où chaque étape revient à résoudre un problème quadratique.

Nous ne décrirons pas en détails la méthode de Newton puisqu'elle est bien connue. Chaque itération de l'algorithme effectue une approximation quadratique locale de la fonction f. Cette approximation est :

$$\hat{f}(\mathbf{x}+\mathbf{v}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^\mathsf{T}\mathbf{v} + \frac{1}{2}\mathbf{v}^\mathsf{T}\nabla^2 f(\mathbf{x})\mathbf{v}$$

pour $\mathbf{x}$ fixe. Trouver λ revient à déterminer les minima locaux de l'approximation

quadratique. Dérivant $\hat{f}$ on obtient la solution analytique :

$$\frac{\partial}{\partial \mathbf{v}} \hat{f}(\mathbf{x}+\mathbf{v}) = \nabla f(\mathbf{x}) + \nabla^2 f(\mathbf{x})\mathbf{v} = 0$$

d'où,

$$\mathbf{v} = -\nabla^2 f(\mathbf{x})^{-1} \nabla f(\mathbf{x}).$$

Cette méthode peut être adaptée pour les cas d'optimisation sous contraintes d'égalité $\mathsf{A}\mathbf{x} = \mathbf{b}$. En supposant que la solution courante respecte les contraintes, l'approximation quadratique doit aussi inclure les contraintes :

$$\mathsf{A}(\mathbf{x}+\mathbf{v}) = \mathbf{b}$$

comme montré §3.4.4.

3.4.8 Optimisation avec contraintes d'inégalité : méthode du point intérieur

Avec des contraintes d'inégalité, les méthodes analytiques font place à des méthodes itératives avec garantie de convergence au minimum global. Dans cette section, nous présentons une *méthode du point intérieur* appelé la *méthode de la barrière* similaire à celle implantée par la librairie SEDUMI [140].

Le problème d'optimiser une fonction objectif convexe $f(\mathbf{x})$ sous contraintes linéaire d'inégalité $g_i(\mathbf{x}) \leq 0$, $g_i : \mathbb{R}^n \to \mathbb{R}$ et d'égalité $\mathsf{A}\mathbf{x} = \mathbf{b}$, est réécrit comme celui de minimiser la fonction non-convexe :

$$\arg\min_{\mathbf{x}} f(\mathbf{x}) + \sum_{i}^{m} I_{-}(g_i(\mathbf{x})), \quad \text{tel que} \quad \mathsf{A}\mathbf{x} = \mathbf{b}$$

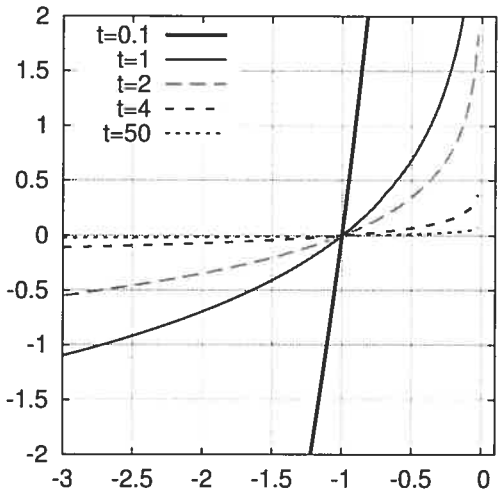


FIG. **3.2. Approximation convexe de la fonction de barrière $\hat{I}_-$ pour différentes valeurs de t.**

où, I_- est appelé la *fonction de barrière* :

$$I_-(a) = \begin{cases} 0 & \text{if } a \leq 0 \\ \infty & \text{sinon.} \end{cases}$$

La méthode de la barrière propose de remplacer I_- par une approximation convexe, doublement différentiable :

$$\hat{I}_-^t(a) = (-1/t)\log(-a), \quad t > 0$$

pour un certain choix de scalaire t. Cette fonction est illustrée à la figure 3.2. On observe que plus t est grand, plus l'approximation est proche de la fonction originale, *i.e.* $\lim_{t\to\infty} \hat{I}_-^t(a) = I_-(a)$. Il s'en suit naturellement un algorithme itératif où t est initialisé à une petite valeur et est augmenté à chaque étape, jusqu'à convergence. Une grande valeur pour t n'est pas choisie dès le départ pour de raisons de stabilité numérique. Cela ferait en sorte que la matrice Hessienne calculée par l'algorithme de

ENTRÉES
- $\mathbf{x}^{(0)} \in \mathrm{dom} f$
- $t^{(0)} > 0$ (approximation de la barrière)
- $\lambda > 1$ (facteur de croissance de t)
- ϵ (petite valeur)
- K (nombre maximal d'itérations)

ALGORITHME

Pour $k = 1$ à K itérations
- trouver $\mathbf{x}^* = \arg\min_{\mathbf{x}} f(\mathbf{x}) + \sum_i^m (-1/t^{(k-1)}) \log(-g_i(\mathbf{x}))$ tel que $\mathsf{A}\mathbf{x} = \mathbf{b}$ par la méthode de Newton avec $\mathbf{x}^{(k-1)}$ comme point de départ
- remplacer $\mathbf{x}^{(k)} = \mathbf{x}^*$
- $t^{(k)} = \lambda t^{(k-1)}$
- critère d'arrêt : si $m/t < \epsilon$

FIG. **3.3. Méthode de la barrière.**

Newton serait mal conditionnée. L'algorithme est présenté à la figure 3.3.

On observe que le critère d'arrêt n'est relié qu'aux contraintes. Ceci est normal puisque les itérations sont faites pour les satisfaire. Le minimum global $\mathbf{x}^*$ de la fonction objectif est donné automatiquement par l'algorithme de Newton.

Initialisation

L'algorithme doit être absolument initialisée avec un point de départ dans le domaine de f. Un tel point est trouvé par des méthodes dites *phase I*. La méthode la plus simple consiste à utiliser la méthode de la barrière pour trouver :

$$\begin{aligned} \{x^*, s^*\} = \arg\min_{\mathbf{x},s} \quad & s \\ \text{tel que} \quad & g_i(\mathbf{x}) \leq s \\ & \mathsf{A}\mathbf{x} = \mathbf{b}. \end{aligned}$$

La méthode d'optimisation peut être initialisé en choisissant $\mathbf{x} = \mathsf{A}^\dagger \mathbf{b}$ et $s = \max_i(g_i(\mathbf{x}))$. La valeur de s^* nous informe de la faisabilité du problème, *i.e.* si $s* \leq 0$, alors $\mathbf{x}^*$ est

dans dom$f$, sinon, le domaine de f est vide. Certaines variations de cet algorithme ont des propriétés utiles. Par exemple, lorsque le domaine de f est vide, certaines d'entre elles fournissent des indices quant aux contraintes qui devraient être relaxées ou enlevées pour que le problème puisse avoir une solution.

3.5 Problèmes non-convexes

3.5.1 Factorisation de matrice pleine

Nous nous intéressons au problème d'approximation d'une matrice M par la plus proche matrice M′ de rang r au sens des moindres carrés. Ceci revient à trouver :

$$\arg\min_{\mathsf{M}'} \left\|\mathsf{M}_{(n\times m)} - \mathsf{M}'\right\|_F^2, \quad \text{tel que} \quad \text{rang}(\mathsf{M}') = r.$$

Comme toute matrice de rang r peut être décomposée eu deux matrices $\mathsf{M}' = \mathsf{A}_{(n\times r)}\mathsf{B}_{(r\times m)}$, nous intéressons plutôt au problème (non-convexe) équivalent de trouver :

$$\arg\min_{\mathsf{A},\mathsf{B}} \left\|\mathsf{M}_{(n\times m)} - \mathsf{A}_{(n\times r)}\mathsf{B}_{(r\times m)}\right\|_F^2 \quad \text{tel que} \quad \text{rang}(\mathsf{A}) = \text{rang}(\mathsf{B}) = r. \tag{3.7}$$

Par ailleurs, nous sommes très souvent intéressés par les facteurs A et B et pas seulement par M′. Malgré le fait que le problème soit non-convexe, il existe une solution efficace basée sur la SVD. Pour mieux comprendre pourquoi une telle solution existe, il faut étudier les points critiques et les minima globaux et locaux de la fonction objectif (3.7) [134]. Tout d'abord, notons qu'il existe une infinité de solutions, résultant de l'ambiguïté de factorisation. À partir d'une solution AB, A′B′ est une solution de même coût s'il existe une matrice R carré $(r \times r)$ de plein rang telle que $\mathsf{A}' = \mathsf{AR}$ et $\mathsf{B}' = \mathsf{R}^{-1}\mathsf{B}$. Par conséquent, il est toujours possible de modifier une factorisation pour que les rangées de B soient orthonormales (*i.e.* $\mathsf{BB}^\mathsf{T} = \mathsf{I}_{(r\times r)}$) et, par conséquent, que $\mathsf{A}^\mathsf{T}\mathsf{A} = \Lambda$ soit une matrice diagonale.

Ceci est assez facile à vérifier. Tout d'abord, on peut toujours trouver une matrice R permettant d'orthonormaliser $B \leftarrow RB$, nous obligeant à transformer $A \leftarrow AR^{-1}$. Il faut donc simplement démontrer que A est orthogonal. Notons la SVD de $AB = U\Sigma V^T$. Comme V est orthonormal, il existe une matrice orthonormale O telle que $V^T = OB$. Ainsi, $A = U\Sigma O$ et donc $A^TA = O^T\Sigma^TU^TU\Sigma O = O^T\Sigma^2O$, qui est orthogonal.

Posons E le résidu de la fonction objectif :

$$E(A, B) = \|M - AB\|^2 = \mathrm{tr}\left(M^TM - 2M^TAB + B^TA^TAB\right)$$

La fonction admet un point critique[7] lorsque les dérivées partielles en A et B s'annulent :

$$\begin{aligned} \frac{\partial E}{\partial A} &= 2ABB^T - 2MB^T = 0 \\ \frac{\partial E}{\partial B} &= 2B^TA^TA - 2M^TA = 0 \end{aligned}$$

À cause de l'ambiguïté de factorisation, nous pouvons porter notre attention au cas où B est orthonormale. Nous avons donc que $\frac{\partial E}{\partial A} = A - MB^T = 0$, d'où $A = MB^T$. En remplaçant A dans $\frac{\partial E}{\partial B}$, nous avons :

$$B^T\Lambda - M^TMB^T = 0.$$

Autrement dit, les rangées de B sont des vecteurs propres de M^TM et Λ une matrice diagonale de valeurs propres de M^TM. De façon plus générale, une solution pour A et B est valide si leurs colonnes sont des combinaisons linéaires des vecteurs propres de MM^T et M^TM, respectivement. À partir de cette constatation, il est possible de voir la solution en terme de la SVD de $M = U\Sigma V^T$. Les matrices A et B sont un point critique s'ils sont des combinaisons linéaires des colonnes de $U\Sigma$ et V, respectivement.

[7] Plus précisément un point stationnaire, puisque la fonction est différentiable partout.

Comme nous sommes intéressés seulement par les points critiques correspondant aux minima globaux de la fonction, il nous faut explorer la valeur de E à chacun des points critiques. Notons l'ensemble des valeurs propres par $\mathbf{q} = \{\sigma_i | i = 1, ..., m\}$. Pour une solution AB, où B est dans l'espace des vecteurs propres correspondant à un choix de r valeurs propres $\mathbf{q}'$ parmi les valeurs propres de $\mathsf{M}^\mathsf{T}\mathsf{M}$, on peut montrer que l'erreur de la solution est donnée par la somme des autres valeurs propres ($\sum_{i \in (\mathbf{q} \setminus \mathbf{q}')} \sigma_i$). Ainsi, les minima globaux sont données en choisissant les vecteurs propres correspondant aux plus grandes valeurs propres.

Il est ensuite possible de démontrer qu'un choix différent parmi les vecteurs propres pour construire une solution donne nécessairement un col et jamais un minimum local. Par conséquent, tous les minima sont globaux, même si la fonction n'est pas convexe.

En résumé, l'ensemble des solutions de (3.7) peut être calculé à l'aide de la SVD de M. Posons $\mathsf{M} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$, la SVD de M. Une solution valide pour A et B est donnée par les r première colonnes de U et les r premières rangées de $\Sigma\mathsf{V}^\mathsf{T}$, respectivement. Le reste des solutions est donné par l'ensemble des matrices AR et $\mathsf{R}^{-1}\mathsf{B}$ tel que $\mathsf{R}_{(r \times r)}$ est inversible.

La formulation d'un problème sous forme de factorisation de matrice est assez couramment utilisée. Des exemples seront donnés aux §§6.2.6, 8.2.6 et d'autres au chapitre 12.

3.5.2 Factorisation de matrice avec données manquantes

Au chapitre 12, nous nous intéresserons au problème de factorisation de matrice avec données manquantes. L'approximation de rang r d'une matrice avec données manquantes M s'écrit formellement :

$$\arg\min_{\mathsf{A},\mathsf{B}} \left\| \mathsf{W}_{(n \times m)} \odot \left(\mathsf{M}_{(n \times m)} - \mathsf{A}_{(n \times r)} \mathsf{B}_{(r \times m)} \right) \right\|, \tag{3.8}$$

où W est une matrice de poids et $\odot$ est la multiplication élément par élément[8] (aussi nommé produit d'Hadamard). Nous sommes surtout intéressés par le cas où W est une matrice binaire où w_{ij} est 0 si m_{ij} est connu et 1 dans le cas contraire. Aucune solution analytique à ce problème n'est connue. Le problème est approché soit par des méthodes numériques itératives ou des méthodes par approximation convexe. Ces méthodes sont le sujet des chapitres 12, 13 et 14.

[8] Element-wise product

Chapitre 4

MODÈLES ET CALIBRAGE DE CAMÉRAS OMNIDIRECTIONNELLES

4.1 Introduction

Le problème de *calibrage* d'une caméra consiste à décrire à l'aide d'un modèle le processus de formation de ses images. De façon générale, un modèle prend la forme d'une fonction de projection ou de déprojection. Une fonction de projection prend en entrée un point 3D et retourne la position de sa projection dans l'image. Une fonction de déprojection, aussi appelé *modèle inverse*, prend en entrée une position dans l'image et retourne un rayon d'échantillonage : un point de départ et une direction. Ce rayon constitue l'ensemble des points 3D dont la projection se retrouve à ce point image. Idéalement, un modèle de caméra peut être utilisé à la fois pour la projection et la déprojection. Cependant, ce n'est pas toujours le cas comme pour les modèles génériques, §4.6.5.

Les paramètres d'un modèle sont divisés en deux groupes : *internes*[1] et *externes*[2]. Ces derniers décrivent la position et l'orientation de la caméra par rapport à un repère du monde. Les paramètres internes, quant à eux, décrivent le processus de formation de l'image. Ils incluent typiquement, la distance focale, le point principal et les paramètres associés à la distorsion (fonction et centre de distorsion). Le modèle de caméra sténopé est le modèle de formation d'image le plus utilisé en vision par ordinateur. Il inclut jusqu'à cinq paramètres internes placés à l'intérieur d'une matrice de transformation image affine K (§2.5.2). Cependant, certaines aberrations de lentille

[1] Intrinsics

[2] Extrinsics

font en sorte que le modèle sténopé décrit mal le processus de formation d'images. Elles ont pour principales causes les imperfections du système optique des caméras. Pour prendre en compte ce genre de défauts, des paramètres de distorsion non-linéaire d'image de type radiale et tangentielle peuvent être combinés au modèle sténopé, *cf*.§§4.3.1 et 4.3.2.

Dans cette thèse, nous nous intéressons aux caméras omnidirectionnelles, *i.e.* ayant un très grand angle de vue, parfois plus grand que 180°. À cause de leur optique, ces caméras sont mal décrites par le modèle sténopé, même avec des paramètres de distorsion d'images. D'ailleurs, lorsque l'angle de vue de la caméra est plus grand que 180°, c'est-à-dire que l'échantillonnage de la caméra couvre plus de la moitié de la surface d'une sphère, il est impossible d'employer les modèles classiques de distorsion d'image.

Dans ce chapitre, nous nous intéressons à différents modèles de paramètres internes et à des algorithmes permettant de les estimer à partir d'images d'objets de dimensions connues. Plusieurs modèles et méthodes de calibrage de caméras ayant des distorsions ont pour objectifs la *rectification* des images. Ce processus consiste à corriger les images obtenues par une caméra, de telle sorte qu'elles pourraient être le fruit d'une caméra sténopé.

Organisation

Nous discutons brièvement comment sont conçus les systèmes de caméras omnidirectionnelles les plus populaires §4.2. Nous présentons ensuite un grand nombres de modèles de caméras ou de distorsion : pour les *fisheye* §4.3, pour les caméras catadioptriques centrales §4.4, les caméras génériques centrales §4.5 et finalement, pour les caméras non-centrales §4.6. Nous présentons ensuite, §4.7, un résumé de quelques méthodes de calibrage de caméra, celles qui sont le plus reliées à nos contributions du chapitre 5, brièvement décrites §4.8.

4.2 Augmenter l'angle de vue des caméras

Depuis quelques années, les caméras vidéo et surtout les appareils photos numériques ont connu une augmentation fulgurante de leur résolution, *i.e.* le nombre de pixels de leur image. Ceci a ainsi ouvert la possibilité d'augmenter l'angle de vue des appareils tout en conservant une résolution raisonnable dans toutes les régions de l'image. Ainsi, les objectifs grand angle et *fisheye* ont reçu une attention nouvelle de la part des chercheurs en vision par ordinateur. Par ailleurs, une nouvelle forme d'objectif appelé "catadioptrique omnidirectionnel", illustré à la figure 1.2c, a été inventée par Nayar en 1997 [98]. Plutôt que d'utiliser un système conventionnel de lentilles pour augmenter l'angle de vue de la caméra, la caméra est pointée vers une surface courbe choisie de telle sorte que l'angle de vue de la caméra est augmentée.

Par rapport aux caméras conventionnelles, les nouveaux systèmes omnidirectionnels possèdent des caractéristiques particulières :

leurs images sont déformées de façon non-linéaire, *i.e.* la projection d'une droite n'est pas une droite, en général, mais une courbe ;

le centre de projection effectif, ou point focal, est souvent déplacé à l'extérieur de l'appareil d'acquisition ;

il peut même y avoir plusieurs centres de projection effectifs (points de vue multiples, projection non-centrale).

Le centre focal effectif d'une caméra est l'endroit d'où les rayons d'échantillonage émanent de la caméra. Une caméra à points de vue multiples[3] possède plusieurs centres focaux effectifs, c'est-à-dire que les rayons d'échantillonnage ne s'intersectent pas tous au même endroit ou ne s'intersectent pas du tout. On parle souvent de projection non-centrale, par opposition aux caméras centrales, *i.e.* ayant un point de vue effectif unique[4]. Ce genre de projection peut être obtenue de façon artificielle,

[3] *Non Single ViewPoint* (NSVP), non-central projection

[4] *Single ViewPoint* (SVP), central projection

par exemple en combinant l'image de plusieurs caméras de positions différentes [109]. Une autre façon, plus intéressante, consiste à pointer une seule caméra vers un ou des miroirs courbés [50, 82, 99, 148]. Finalement, les *fisheye* peuvent parfois produire une projection non-centrale [173].

Les effets de distorsion complexifient les algorithmes de calibrage et d'auto-calibrage et rendent plus difficile l'automatisation complète du processus. Voici quelques raisons :

la détection de lignes (courbes de formes souvent inconnues), par transformée de Hough est plus difficile ou impossible ;

pour une projection non-centrale, la rectification d'image n'est pas possible ;

la mise en correspondance de points caractéristiques est moins fiable, car les invariants affines ne sont pas conservés ;

le suivi de points caractéristiques est moins fiable, surtout en bordure des images, là où les distorsions sont les plus fortes ;

les modèles de distorsion sont généralement non-linéaires, ce qui nécessite l'optimisation de fonctions de coût plutôt complexes nécessitant idéalement un estimé de départ assez précis des paramètres.

4.3 Caméras grand angle et fisheye

Dans cette section et les deux suivantes, nous décrivons les modèles de distorsion les plus répandus, en particulier, leurs avantages et leurs inconvénients. Lorsque applicable, nous assumons que le système de coordonnées de l'image est choisi de telle sorte que le centre de distorsion est à l'origine, tel que décrit §4.3.1. De plus, lorsque la fonction de distorsion L est donnée sans indices d (pour *distortion*) ou u (pour *undistortion*), cela signifie que le modèle peut, en principe, être utilisé comme fonction de distorsion autant que comme fonction inverse ou de rectification.

4.3.1 Distorsion radiale

Le modèle le plus répandu est le modèle de distorsion radiale polynômial combiné à une projection perspective conventionnelle [26]. Il s'agit d'une déformation radiale symétrique par rapport au centre de distorsion (*cf*.figure 4.1). Comme plusieurs modèles, il est possible de l'utiliser soit pour représenter la déformation, *i.e.* le passage de l'image perspective à l'image distordue, soit pour représenter la fonction de rectification, *i.e.* de l'image originale à l'image corrigée :

▷ *Modèle de distorsion :* un point 3D $\mathbf{P} \in \mathbb{P}^3$ est d'abord projeté dans l'image par une matrice de projection perspective M :

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \propto \mathsf{M}_{(3\times4)}\mathbf{P},$$

ce qui suppose, automatiquement que l'angle de vue de la caméra est plus petit que 180°. Pour appliquer la déformation image, il est pratique d'utiliser un système de coordonnée image dans lequel le centre de distorsion $\tilde{\mathbf{c}} \in \mathbb{R}^2$ est à l'origine. On obtient :

$$\tilde{\mathbf{p}}^u = (x, y)^\mathsf{T} - \tilde{\mathbf{c}}$$

où, $\tilde{\mathbf{p}}^u$ est la coordonnée du pixel non-distordu. Pour simplifier le reste des équations plus bas, la distance entre $\mathbf{p}^u$ et le centre de distorsion est définie par :

$$r^u = \|\tilde{\mathbf{p}}^u\|.$$

Comme le modèle est radial symétrique, la distorsion L^d est une fonction de r^u. La coordonnée de $\tilde{\mathbf{p}}^u$, après distorsion, est $\tilde{\mathbf{p}}^d$, donnée par l'équation :

$$\tilde{\mathbf{p}}^d = \mathrm{L}^d(r^u)\tilde{\mathbf{p}}^u \quad \text{où} \quad \mathrm{L}^d(r) = 1 + \sum_i^\infty k_i r^i, \tag{4.1}$$

où, k_i sont les paramètres du modèle. Il s'agit d'une fonction qui, comme énoncé par le théorème de Taylor, peut représenter toute fonction. Puisqu'un nombre fini de coefficients doit être choisi, L^d est un polynôme correspondant à une approximation de Taylor de la vraie fonction de distorsion. Généralement, des termes jusqu'à k_5 sont utilisés [26]. Finalement, la position du pixel distordu dans l'image originale est donnée par $\tilde{\mathbf{p}}^d + \tilde{\mathbf{c}}$.

▷ *Modèle inverse :* il est souvent plus commode d'utiliser un modèle inverse, particulièrement à l'intérieur d'un algorithme dont l'objectif est de rectifier les images. Lorsqu'utilisé à cette fin, le modèle polynômial déplace un pixel $\tilde{\mathbf{p}}^d$ de l'image originale à la position

$$\tilde{\mathbf{p}}^u = \mathrm{L}^u(r^d)\tilde{\mathbf{p}}^d \quad \text{où,} \quad \mathrm{L}^u(r) = 1 + \sum_i^\infty k_i' r^i \quad \text{et} \quad r^d = \|\tilde{\mathbf{p}}^d\|. \tag{4.2}$$

Dans le système de coordonnées de départ, la position finale du point rectifié est donnée par $\tilde{\mathbf{p}}^u + \mathbf{c}$. Un inconvénient du modèle polynômial est que son inversion analytique (le passage $r^u = \mathrm{L}^u(r^d) \leftrightarrow r^d = \mathrm{L}^d(r^u)$) n'est pas possible lorsque le nombre de coefficients devient trop élevé. Il faut alors employer des méthodes numériques. Ma *et al.* ont proposé d'utiliser des polynômes du deuxième degré par morceaux qui ont l'avantage de s'inverser de façon analytique [86].

4.3.2 Distorsion tangentielle

Lorsque les lentilles composants l'objectif de caméra ne sont pas parfaitement perpendiculaires par rapport à l'axe optique, on obtient un effet de distorsion tangentielle[5] [25, 34]. Comme son nom l'indique, ce déplacement est perpendiculaire à la direction radiale et est une fonction de la distance du point au centre de distorsion et de l'orientation de l'axe par rapport à un axe passant par le centre de distorsion

[5] *Decentering*

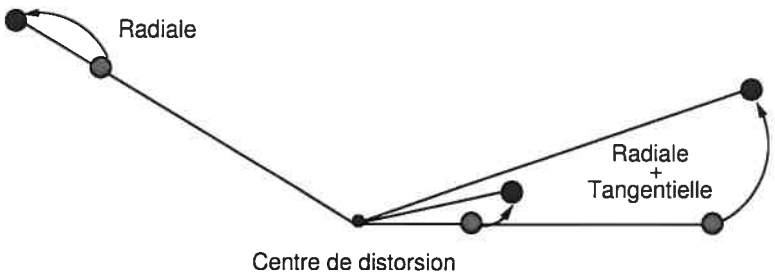


FIG. **4.1. Distorsions radiale et tangentielle. Pour un modèle correctif : en rouge (pâle), les points distordus et en bleu (foncé), les points corrigés (l'inverse pour un modèle de distorsion).**

$\mathbf{c}$ (par exemple la ligne $(0, 1, -c_2)^\mathsf{T}$). Ceci est illustré à la figure 4.1. Ce modèle de distorsion est pratiquement toujours combiné à un modèle de distorsion radiale. Les effets de la distorsion tangentielle sont souvent si peu important que le modèle n'est pas employé dans plusieurs applications [165].

Lorsque le modèle est utilisé pour la rectification, un pixel $\tilde{\mathbf{p}}^d$ est corrigé selon :

$$\begin{aligned} \tilde{\mathbf{p}}^u &= \tilde{\mathbf{p}}^d + \Delta T^u \left(r^d, \arctan(y, x)\right). \\ \Delta T^u(r, \phi) &= \begin{pmatrix} t_1 r^2 (1 + 2\cos^2(\phi)) + t_2 r^2 \sin(\phi)\cos(\phi) \\ t_2 r^2 (1 + 2\sin^2(\phi)) + t_1 r^2 \sin(\phi)\cos(\phi) \end{pmatrix} \end{aligned}$$

où, ΔT^u est la fonction de distorsion tangentielle de paramètres t_i.

4.3.3 Modèle division

Le modèle division est, tout comme le modèle polynômial, une approximation au premier degré de la vraie fonction de distorsion [39]. Ce modèle est obtenu en remplaçant L^u et L^d par :

$$L(r) = \frac{1}{1 + kr^2}.$$

dans (4.1) et (4.2). Il fut démontré par Strand et Hayman que d'après ce modèle, la projection d'une ligne dans l'image résulte toujours en un arc de cercle [138]. Ce-

pendant, tout cercle n'est pas nécessairement l'image d'une ligne [10]. Les résultats empiriques de Clauss et Fitzgibbon suggèrent que ce modèle n'est pas assez flexible pour modéliser de large distorsion [32]. Pour palier à cette lacune, Thirthala et Pollefeys proposent d'inclure un plus grand nombre de paramètres [156] :

$$L(r) = \frac{1}{1 + \sum_i kr^i}.$$

Finalement, nous avons montré dans nos travaux que ce modèle peut représenter des caméras ayant un angle de vue plus grand que 180° [151, 152]. En effet, ceci est possible en utilisant la fonction $L(r)$ comme coordonnée projective du point rectifié :

$$\mathbf{p}^u \propto \begin{pmatrix} \tilde{\mathbf{p}}^d \\ L^d(r^d) \end{pmatrix}$$

et en permettant à la fonction L^d d'être égale ou plus petite que zéro. Ce modèle est discuté en détails et illustré aux chapitres 7 et 9.

4.3.4 Modèle fisheye

Le modèle *fisheye* de Basu est basé sur le fait que ce type de caméras à une bonne résolution au centre laquelle diminue vers les bordures [15]. Ce modèle propose donc d'utiliser la fonction "log" pour représenter la distorsion :

$$L(r) = s \log(1 + kr^u)$$

où, les paramètres du modèle sont les scalaires s et k.

4.3.5 Modèle angle de Vue

Le modèle *angle de vue*[6] a été proposé par Devarnay et Faugeras [36]. La fonction de distorsion est :

$$L^d(r^u) = \frac{1}{k} \arctan(2r^u \tan k/2)$$

qui est facilement inversible :

$$L^u(r^d) = \frac{\tan(r^d k)}{2 \tan k/2}.$$

où, k est le paramètre de distorsion. Par ailleurs, les auteurs proposent de le combiner à une fonction polynômiale conventionnelle. Ce modèle a été testé au chapitre 7.

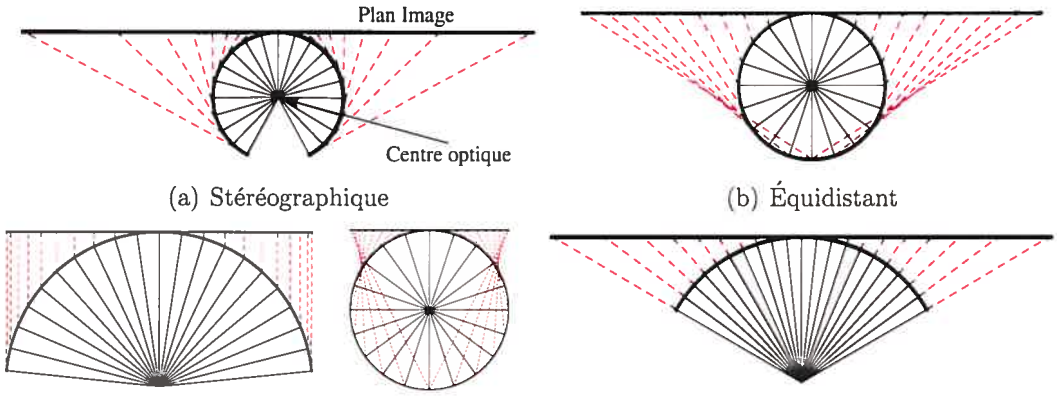
4.3.6 Modèle angulaire

Plutôt que d'établir une fonction de distorsion entre r^d et r^u, Fleck propose de l'établir entre r^d et l'angle θ entre le rayon d'échantillonage associé au pixel et l'axe optique [41]. Pour ce faire, l'auteur fait l'hypothèse courante que la distorsion engendre une déformation radiale symétrique de l'angle de vue de la caméra autour de l'axe optique de la caméra (*cf.*§4.5.3). Par conséquent, le centre de distorsion et le point principal sont alignés. Les fonctions de θ suivantes sont proposées :

- stéréographique $r^d(\theta) = f \tan \frac{\theta}{2}$,
- équidistant $r^d(\theta) = f\theta$,
- loi de sinus $r^d(\theta) = f \sin \theta$,
- angle equi-solide $r^d(\theta) = f \sin \frac{\theta}{2}$.
- perspectif $r^d(\theta) = f \tan \theta$,

où, f est la distance focale de la caméra. À noter que la dernière fonction correspond simplement au modèle sténopé. Ces fonctions sont illustrées à la figure 4.2.

[6] *Field of View*



(a) Stéréographique (b) Équidistant

(c) Loi de sinus (d) Angle equi-solide (e) Perspectif

FIG. **4.2. Modèle angulaire [41] : lignes noires (pleines), directions d'échantillonage des rayons de la caméra ; lignes rouges (pointillées), direction de projection vers le plan image.**

D'après [7], le modèle stéréographique est celui qui est le mieux adapté à un objectif *fisheye*. Kannala et Brandt proposent d'utiliser le même type de modèle, mais avec une fonction polynomial de θ [72] :

$$r^d(\theta) = \sum_i^I k_i \theta^{2i-1},$$

qu'ils combinent aussi à un modèle de distorsion tangentielle.

4.4 Caméras catadioptriques

Nous discutons ici des configurations de caméras catadioptriques dont la projection est centrale tel que décrit par Baker et Nayar [6]. Des configurations et modèles de projection non-centrales sont donnés §4.6.

▷ Parabolique–orthographique

L'une des configurations de caméra catadioptrique parmi les plus populaires est

la combinaison d'un miroir parabolique à une caméra orthographique (figure 4.3). La projection est centrale lorsque le plan image de la caméra est perpendiculaire à l'axe de révolution de la surface du miroir. Le modèle de caméra proposé par Padjla utilise une description explicite de la surface du miroir [107]. Celle-ci est donnée par l'équation paramétrique :

$$z(x,y) = \frac{x^2+y^2}{2a} - \frac{a}{2}$$

où, a défini la courbure de la surface. La projection d'un point $\tilde{\mathbf{P}} \in \mathbb{R}^3$ dans l'image passe par sa projection à la surface du miroir donnée par :

$$\tilde{\mathbf{Q}}^m = \lambda\tilde{\mathbf{Q}}, \quad \tilde{\mathbf{Q}} = \mathsf{R}(\tilde{\mathbf{P}} - \tilde{\mathbf{t}}), \quad \lambda = \frac{a(\tilde{Q}_3 + ||\tilde{\mathbf{Q}}||)}{\tilde{Q}_1^2 + \tilde{Q}_2^2}.$$

où R est une rotation 3D et $\mathbf{t} \in \mathbb{R}^3$ sont les paramètres externes de la caméra. Le point image est alors la projection orthographique de $\tilde{\mathbf{Q}}^m$ dans l'image, donnée par :

$$\mathbf{p} = \mathsf{K}\mathsf{R}_c \begin{pmatrix} \lambda \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tilde{\mathbf{Q}}^m \\ 1 \end{pmatrix}$$

où, R_c de taille (3×3) est une matrice de rotation 2D (en forme projective) du plan image autour de l'axe optique et, K est la matrice des paramètres internes de la caméra. Dans ce modèle, les matrices K et R_c, et le scalaire a constituent les paramètres internes de la caméra.

▷ **Hyperbolique–perspectif**

Un peu comme dans le cas parabolique, il est possible d'obtenir une projection centrale en utilisant un miroir hyperbolique. Les conditions sont toutefois un peu plus restreintes tel qu'illustré à la figure 4.4. L'axe optique d'une caméra perspective doit être aligné avec l'axe de révolution de la surface du miroir. De

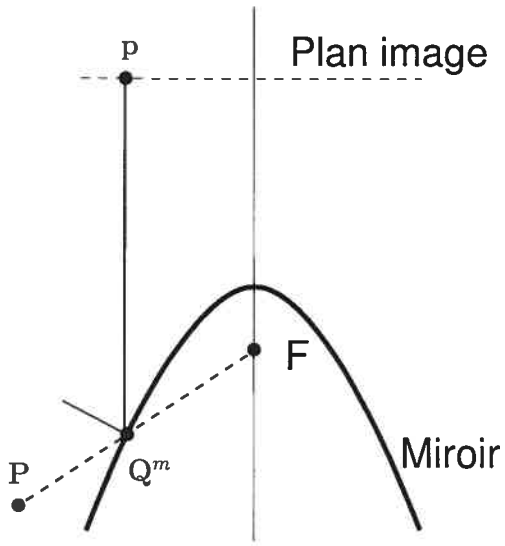


FIG. **4.3. Projection pour une caméra catadioptrique parabolique-orthographique.**

plus, le centre optique de la caméra doit être aligné avec le point focal F' situé à l'extérieur du miroir. Un modèle semblable à celui du miroir parabolique a aussi été présenté par Padjla [107].

▷ **Conique**

Il a été montré que la combinaison d'un miroir conique et d'une caméra sténopé ne permet pas une projection centrale [6]. En effet, pour obtenir une telle projection, il faut que le point de vue effectif de la caméra soit situé exactement à la pointe du cône, tel qu'illustré à la figure 4.5a. Le cône provocant l'occultation du centre optique, il est impossible d'obtenir une image dans une telle configuration. Néanmoins, Lin *et al.* proposent de modéliser la caméra avec un modèle de lentille mince (§2.5.1) et montrent dans quelles circonstances le point de vue est unique (figure 4.5b) [82]. La combinaison miroir–lentille permet de faire converger toute la lumière provenant d'un point 3D vers le même point image. Ainsi, même si le rayon $\mathbf{r}$ (figure 4.5b) passant par la pointe du cône

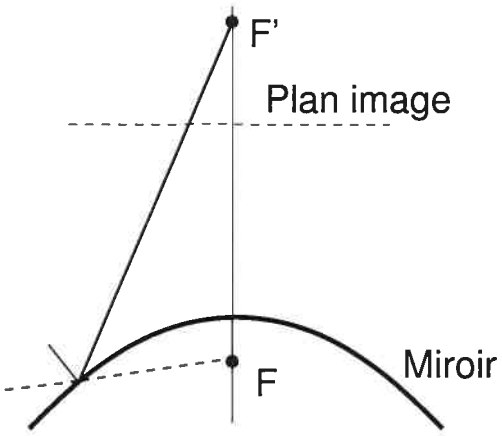


FIG. **4.4. Projection pour une caméra hyperbolique–perspectif.**

n'est pas réfléchi vers le point image **p**, les rayons secondaires contribuent à l'intensité du point image.

▷ **Sphérique**

Il n'est possible d'obtenir une projection centrale avec ce type de miroir que lorsque le centre optique de la caméra est exactement situé au centre du miroir. Ceci n'a aucune utilité pratique puisque la caméra ne voit que sa propre image. Lorsque la caméra est située à l'extérieur de la sphère, une projection non-centrale pouvant se décrire par une caustique est obtenue (détails §4.6.2).

▷ **Elliptique**

Tel qu'illustré à la figure 4.6, il est possible de construire une caméra à projection centrale en utilisant un miroir elliptique. Pour ce faire, il faut que la caméra soit située à un des points focaux du miroir, et pointée dans la direction du deuxième point focal.

▷ **Planaire**

Bien entendu, lorsqu'une caméra regarde au travers un miroir plan, la projection reste centrale. En pratique, ceci n'a pas tellement d'utilité puisque l'angle de vue de la caméra n'est pas affecté. Afin d'augmenter le champs de vue de la

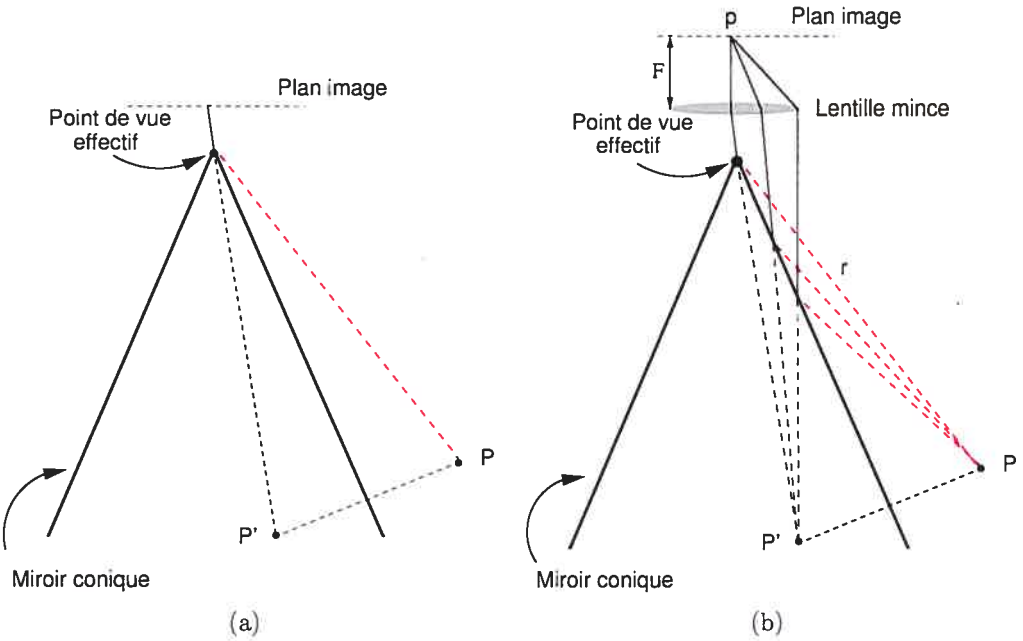


FIG. **4.5. (a) Utilisation d'un modèle sténopé pour une caméra catadioptrique ayant un miroir conique ; (b) La même situation lorsque la caméra est décrite selon un modèle à lentille mince.**

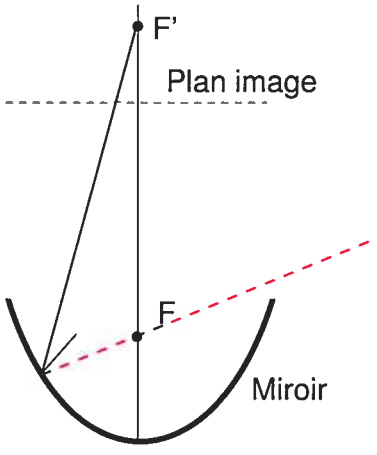


FIG. **4.6. Projection pour une caméra elliptique-perspectif.**

caméra, il faut utiliser plusieurs miroirs à différentes inclinaisons tel que proposé par Nalwa [97] ; cependant la projection n'est plus centrale.

Pour chacune des configurations précédentes, il est possible d'utiliser un modèle spécifique pour la décrire. Ceci comporte toutefois l'inconvénient que le choix du modèle et d'un algorithme de calibrage doit se faire doit se faire en fonction de cette configuration, laquelle n'est pas toujours connue. Dans ce cas, il faut détecter la configuration à l'intérieur même de la procédure de calibrage. Le prochain modèle offre l'une des premières solutions à ce problème, en proposant d'unifier certaines configurations à l'intérieur d'un seul et même modèle. D'autres solutions sont aussi discutées §§4.5 et 4.6.

4.4.1 Modèle de caméra catadioptrique centrale unifié

Pour simplifier la modélisation des caméras catadioptriques centrales, Geyer et Daniilidis proposent une approche permettant d'unifier les modèles de caméra parabolique–orthographique, hyperbolique–perspectif et perspectif [8, 9, 44, 45, 170]. Ce modèle de projection consiste en deux étapes (figure 4.7) :

- la projection d'un point 3D à la surface d'une sphère à partir du centre de projection effectif ;
- une seconde projection sur le plan image à partir d'un second centre de projection choisi en fonction du miroir et de la distance focale de la caméra.

Par exemple, à la figure 4.7, le point $\mathbf{P}$ est réfléchi à la surface du miroir au point $\mathbf{Q}^m$, puis de façon orthographique sur le plan image au point $\mathbf{p}$. Le modèle unifié effectue l'opération équivalente en projetant le point $\mathbf{P}$ à la surface de la sphère centrée en $\mathbf{F}$ (le centre de projection effectif de la caméra) au point $\mathbf{P}^s$, lequel est projeté à partir de $\mathbf{C}^s$ au point $\mathbf{p}$. Yinh et Hu, de leur côté, démontrent que ce modèle de projection peut aussi s'appliquer assez bien aux *fisheyes* [173].

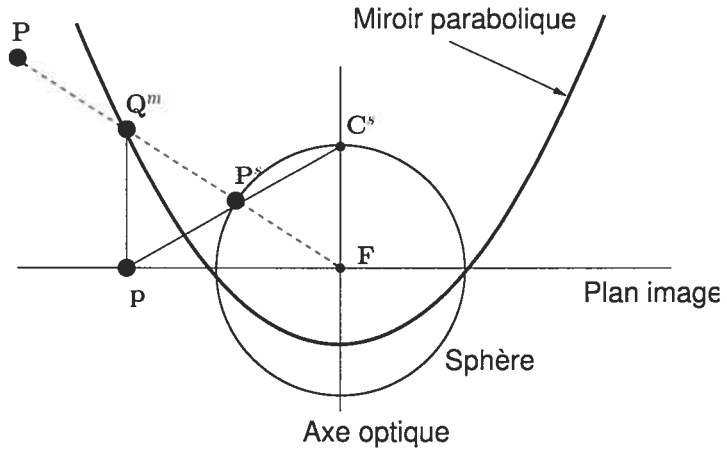


FIG. **4.7. Projection pour le modèle unifié des caméras catadioptriques centrales : cas parabolique-perspectif.**

4.5 Caméras centrales générales

Le modèle catadioptrique unifié offre une certaine flexibilité pour modéliser plusieurs type de caméras à la fois. Reste qu'il est assez restrictif et ne s'applique pas toujours. Nous présentons dans cette section plusieurs modèles de caméras centrales, qui ne sont pas spécifiques au type d'objectif de la caméra ou à la combinaison de miroir et de caméra.

4.5.1 Modèle rationnel

Clauss et Fitzgibbon proposent le modèle rationnel [32] :

$$\mathbf{p}^u \propto \mathsf{A}\chi(\tilde{\mathbf{p}}^d), \quad \chi(\mathbf{p}) = (p_1^2, p_1p_2, p_2^2, p_1, p_2, 1)^\mathsf{T}$$

où, $\mathsf{A} \in \mathbb{R}^{3\times 6}$ et la fonction χ effectue la transformation non-linéaire de l'image. On le nomme rationnel, car en coordonnées image, le point rectifié image $\tilde{\mathbf{p}}^u$ est le fruit d'une division entre deux polynômes ayant comme coefficients les éléments de $\tilde{\mathbf{p}}^d$. De plus, le modèle n'est pas nécessairement radial symétrique. Selon nous, il

est probablement possible de l'appliquer à des caméras omnidirectionnelles selon le même principe que le modèle division, *i.e.* si ${\mathbf{A}_3}^\top \chi(\tilde{\mathbf{p}}^d) < 0$ pour certains rayons et que les numérateurs sont toujours positifs. Mais cela n'a pas été testé par les auteurs. Une propriété importante de ce modèle est que la projection d'une droite de l'espace résulte toujours en une conique (*cf*.6.2.6).

4.5.2 Modèle rationnel radial symétrique

Ma *et al.* proposent un modèle rationnel radial symétrique apparent au modèle division [87] :

$$\mathrm{L}^d(r) = \frac{1 + k_1 r + k_2 r^2}{1 + k_3 r + k_4 r^2 + k_5 r^3}.$$

où, le nombre de paramètres k_i est choisi pour qu'une inversion analytique de la fonction L^d soit possible (quoique son expression n'est pas très simple).

4.5.3 Caméra centrale radiale symétrique générale

Les modèles de caméras radiales symétriques appliquent une transformation non-linéaire aux pixels de l'image en fonction de leur distance r^d (ou r^u) au centre de distorsion. Ces fonctions sont paramétriques et prennent toutes sortes de formes. Entre autres, elles associent r^d avec r^u ou bien r^d avec θ. Une fonction radiale symétrique générale effectue la même tâche en utilisant une fonction discrète. À chaque rayon r^d est associé un coefficient de distorsion différent. La distorsion est radiale, mais elle n'a pas de modèle. Il y a plusieurs façons d'exprimer ce paramétrage discret. En voici quelques unes :

- Thirthala et Pollefeys [157] :

$$r^d \leftrightarrow \theta_{r^d}$$

  où, θ_{r^d} est l'angle de vue associé aux pixels de rayons r^d dans l'image distordue. Ce modèle a été testé au chapitre 9 ;

- Hartley et Kang [61] :

$$\tilde{\mathbf{p}}^u = \lambda_{r^d} \tilde{\mathbf{p}}^d$$

où, λ_{r^d} est un facteur d'échelle non-paramétrique. Ce modèle a été testé au chapitre 5 ;

- Tardif *et al.* [152] :

$$\mathbf{p}^u \propto (\tilde{\mathbf{p}}^d, f_{r^d})^\mathsf{T}.$$

Les détails pour ce dernier modèle sont décrits §4.6.3. Notez que les deux derniers modèles sont très similaires, $\lambda_{r^d} = 1/f_{r^d}$;

- Une dernière possibilité pourrait être :

$$\tilde{\mathbf{p}}^u = \tilde{\mathbf{p}}^d + \lambda_{r^d}.$$

En principe, ces représentations sont toutes équivalentes ; mais pas en pratique. En particulier, celle de Hartley et Kang ne permet pas de représenter des caméras ayant un angle de vue plus grand que 180°. En effet, les pixels correspondant à des rayons d'angle de vue exactement de 180° devraient avoir $r^u = \infty$, donc $\lambda_{r^d} = \infty$. Par contre, en utilisant l'inverse tel que nous le faisons, f_{r^d} prend la valeur 0. Cette différence est analysée en détail §5.9.

4.6 Caméras non-centrales

Comme mentionné §4.2, les rayons d'échantillonage d'une caméra ne s'intersectent pas toujours en un seul centre de projection effectif. Nous présentons dans les sections suivantes quelques configurations et modèles de caméra pour les représenter.

4.6.1 Deux miroirs plans

L'une des premières configurations ayant une projection non-centrale est proposée par Gluckman et Nayar [50]. Leur objectif est de construire un système stéréo à l'aide d'une seule caméra. Pour ce faire, ils utilisent une caméra sténopé devant deux miroirs plans d'orientation différentes divisant l'image en deux parties. À chacune d'elles peut être associée une caméra sténopé virtuelle de centre de projection différent. L'orientation des miroirs est choisie de telle sorte que le champs de vision des caméras virtuelles est sensiblement le même. Le résultat est un système stéréo ne nécessitant aucune synchronisation lors de l'acquisition des images. Les auteurs proposent un modèle spécifique pour ce genre de configuration et une méthode de calibrage.

4.6.2 Caustique

Lorsqu'une caméra est pointée vers un miroir, le système d'acquisition est presque toujours non-central, sauf pour les situations décrites §4.4 [6]. Les catacaustiques permettent de représenter plusieurs situations courantes, par exemple, les caméras catadioptriques parabolique–perspectif, hyperbolique–orthographique et sphérique–perspectif [146].

Une catacaustique (ou simplement caustique) est une surface paramétrique définie par la position du centre de projection de la caméra (possiblement situé à l'infini) et la surface du miroir. Dans les cas les plus simples, la caustique est une surface de révolution. Nous ne décrivons que cette situation, aussi illustrée à la figure 4.8. Pour chaque point de la surface du miroir est associé un point de la caustique. Celui-ci, ainsi que la tangente à la surface dans la direction opposée de l'axe optique définissent un rayon d'échantillonage.

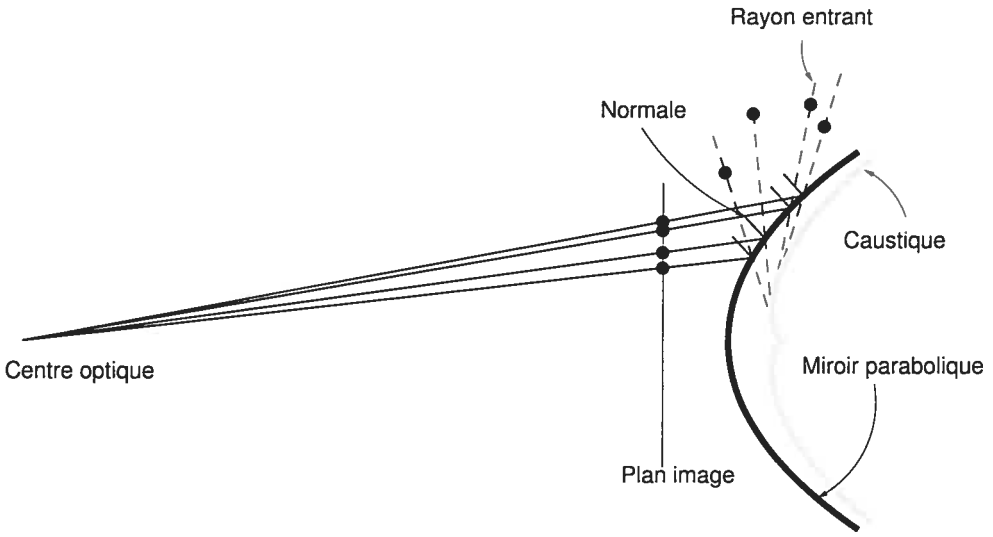


FIG. **4.8. Coupe transversale de la surface caustique d'une caméra catadioptrique-parabolique utilisant caméra perspective.**

4.6.3 Modèle radiale symétrique non-central

Au chapitre 5, nous introduisons un modèle radial symétrique général à points de vues multiples ainsi que trois méthodes de calibrage. À la différence des modèles à projection centrale, celui-ci inclut un déplacement, noté t_{r^d}, du centre de projection en fonction du rayon. Cela permet de modéliser un certain nombre de caméras non-centrales un peu comme le font les caustiques radiales symétriques, mais de façon non-paramétrique. Nous proposons une modélisation à l'aide d'un ensemble de cônes situés sur le même axe (l'axe optique de la caméra), mais pas nécessairement de même sommet, (*cf*.figure 4.9 pour une illustration du modèle central). Une façon efficace de les représenter est à l'aide d'une fonction de projection dont les paramètres varies en fonction du rayon de l'image :

$$\mathbf{p}^d \propto \begin{bmatrix} f_{r^d} & 0 & 0 \\ 0 & f_{r^d} & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathsf{R}[\mathtt{I}_{(3\times3)}| - \mathbf{C} + t_{r^d}\mathbf{R}_3]\mathbf{P} \tag{4.3}$$

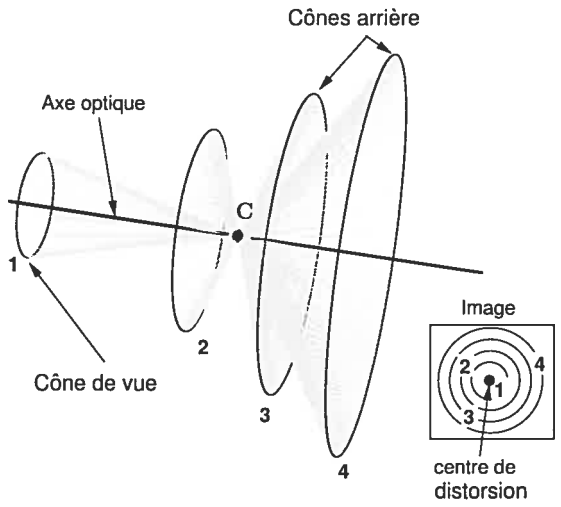


FIG. **4.9. Caméra radiale symétrique centrale.**

où, $\mathbf{P} \in \mathbb{P}^4$, R est une matrice de rotation 3D, et $\mathbf{C}$ est le centre de projection de la caméra. La variable f_{rd} est une focale variable en fonction du rayon, alors que t_{rd} est un déplacement le long de l'axe optique. À noter qu'en posant $t_{rd} = 0$, on obtient le modèle central discuté §4.5.3.

4.6.4 Caméra radiale 1D

Thirthala et Pollefeys présentent une modélisation permettant d'éliminer l'effet de la distorsion dans le modèle de projection [156, 157]. Pour ce faire, ils assument eux aussi que le centre de distorsion est aligné avec le point principal de la caméra. Dans ce cas, les lignes passant par l'origine (le centre de distorsion) sont de la forme $(a, b, 0)^\mathsf{T}$. On peut donc les exprimer par leurs deux premières coordonnées (a, b). Par ailleurs, lorsque la distorsion est radiale (mais pas nécessairement symétrique) les points distordus et rectifiés passent tous deux par la ligne :

$$\mathbf{l} \propto (\tilde{\mathbf{l}}^\mathsf{T}, 0)^\mathsf{T} \quad \text{où,} \quad \tilde{\mathbf{l}} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tilde{\mathbf{p}}^d.$$

Dans ce contexte, il est utile d'exprimer la projection d'un point 3D vers une ligne radiale :

$$\tilde{\mathbf{l}} \propto \mathsf{M}_{(2\times4)}\mathbf{P},\ \mathbf{P} \in \mathbb{P}^3, \tilde{\mathbf{l}} \in \mathbb{R}^2$$

où, M est une matrice de projection affine définie à un facteur d'échelle près et $\tilde{\mathbf{l}}$ est la ligne radiale. En modélisant les caméras de cette manière, il est possible de construire des algorithmes de reconstruction pour lesquels la distorsion radiale n'a pas d'effet (*cf.*§8.2.5).

4.6.5 Modèles complètement génériques

Dans ce type de modèle de caméra, chaque point de l'image est soit associé à un rayon d'échantillonage, ou soit associé à un cône d'une certaine orientation, position et ouverture [55, 109, 145]. La fonction d'échantillonage est donc complètement non-paramétrique. Ces modèles permettent en théorie de représenter pratiquement tous les systèmes d'acquisition d'images. En revanche, ils sont plus difficiles à calibrer ou à auto-calibrer, nécessitant un grand nombre d'images [112] ou des correspondances images denses [113].

Une variante est de simplement associer à chaque pixel $\tilde{\mathbf{p}}^d$, une position $\tilde{\mathbf{p}}^u$ dans l'image rectifiée [102].

4.7 Méthodes de calibrage

Dans cette section, nous résumons six algorithmes de calibrage plan, *i.e.* utilisant la vue d'un ou plusieurs plans de calibrage. La première est une méthode traditionnelle pour caméra sténopé, les trois suivantes sont pour des caméras centrales, et les deux dernières pour les caméras non-centrales. Dans ce qui suit, les points $\mathbf{p}_i^c$ et $\mathbf{p}_i^d$ correspondent à la $i^{\text{ième}}$ correspondance plan de calibrage–plan image ($\mathbf{p}_i^c \leftrightarrow \mathbf{p}_i^d$). Nous assumons que ces correspondances sont données par un algorithme automatique ou semi-automatique comme celui d'OpenCV [67].

Objectifs des méthodes

L'objectif des méthodes de calibrage est de calculer les paramètres internes, incluant ceux de distorsion, et les paramètres externes. Pour les caméras centrales, les méthodes se concentrent souvent sur l'estimation des paramètres de distorsion. Ceci permet de rectifier les images originales et d'employer une méthode de calibrage traditionnelle pour retrouver le autres paramètres internes et la position de la caméra.

4.7.1 Méthode de Zhang et Sturm

Nous présentons un résumé de la méthode de calibrage plan développée indépendemment par Zhang [181] et Sturm [143] pour les caméras sténopé. Rappelons que la matrice de projection d'une caméra est donnée par (en laissant tomber l'indice i) :

$$\mathbf{p}^u \propto \mathsf{M}\mathbf{P} = \mathsf{K}_{(3\times3)}\mathsf{R}_{(3\times3)}[\mathtt{I}_{(3\times3)}|\mathbf{t}]\mathbf{P},\ \mathbf{P} \in \mathbb{P}^3.$$

Lorsque les points 3D sont situés dans un plan, on peut modifier le système de coordonnées du monde pour que ces points soient situés dans le plan $Z = 0$. Ainsi, la projection peut être exprimée par un transfert projectif entre le plan de calibrage et le plan image :

$$\mathbf{p}^u \propto \mathsf{KR}[\mathtt{I}_{(3\times3)}|\mathbf{t}] \begin{pmatrix} P_1 \\ P_2 \\ 0 \\ P_4 \end{pmatrix} = \underbrace{\mathsf{KR}\begin{bmatrix} 1 & 0 & \\ 0 & 1 & \mathbf{t} \\ 0 & 0 & \end{bmatrix}}_{\mathsf{H}_{(3\times3)}} \mathbf{p}^c,\ \text{où},\quad \mathbf{p}^c \propto (P_1, P_2, P_4)^{\mathsf{T}}.$$

La méthode utilise un outil géométrique appelée *conique absolue*[7]. Il s'agit d'une conique de points complexes (voir équation ci-bas), située dans le plan infini. Un

[7] *Absolute conic*

point $(x, y, z, h)^\mathsf{T}$ à sa surface vérifie :

$$(x, y, z) \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = 0, \qquad h = 0.$$

La projection dans l'image des points à la surface de cette conique peut être représentée par une transformation projective indépendante de la translation de la caméra, puisque

$$\mathbf{p} \propto \mathsf{KR}[\mathtt{I}|\mathtt{t}]\begin{pmatrix} x & y & z & 0 \end{pmatrix}^\mathsf{T} = \mathsf{KR}\begin{pmatrix} x & y & z \end{pmatrix}^\mathsf{T}.$$

Par conséquent, le transfert de la conique absolue dans l'image s'effectue selon :

$$\omega \propto (\mathsf{KR})^{-\mathsf{T}} \mathtt{I}_{(3\times3)} (\mathsf{KR})^{-1} = \mathsf{K}^{-\mathsf{T}} \mathsf{R} \mathtt{I} \mathsf{R}^{-1} \mathsf{K}^{-1} = \mathsf{K}^{-\mathsf{T}} \mathsf{K}^{-1}$$

où, ω est nommé *image de la conique absolue*[8] (IAC). Par exemple, pour une matrice de paramètres internes de la forme :

$$\mathsf{K} = \begin{bmatrix} f & 0 & u \\ 0 & f\alpha & v \\ 0 & 0 & 1 \end{bmatrix}, \quad \text{nous avons} \quad \omega \propto \begin{bmatrix} \alpha^2 & 0 & -\alpha^2 u_0 \\ 0 & 1 & -v_0 \\ -\alpha^2 u_0 & -v_0 & f^2\alpha^2 + u_0^2\alpha^2 + v_0^2 \end{bmatrix}.$$

Le calibrage des paramètres internes de la caméra revient donc à calculer ω. Ceci est possible à partir d'homographies reliant le plan de calibrage et son image dans plusieurs vues, pouvant être calculées à partir de correspondances. Puisque que chaque homographie est de la forme $\mathsf{H} \propto \mathsf{KR} \begin{bmatrix} 1 & 0 & \\ 0 & 1 & \mathbf{t} \\ 0 & 0 & \end{bmatrix}$, on peut vérifier que chacune d'elles

[8] *Image of the absolute conic*

fournit deux contraintes linéaires sur ω :

$$\mathbf{H}^{1\mathsf{T}}\omega\mathbf{H}^{2} = 0, \text{ et } \mathbf{H}^{1\mathsf{T}}\omega\mathbf{H}^{1} - \mathbf{H}^{2\mathsf{T}}\omega\mathbf{H}^{2} = 0$$

où, $\mathbf{H}^{i}$ est la $i^{\text{ième}}$ colonne de H. Il faut donc plusieurs vues pour obtenir assez d'équations pour estimer ω. Après avoir estimé les paramètres internes, les paramètres externes pour chaque vue du plan peuvent être estimés en décomposant l'homographie.

Distorsion radiale

Cette méthode ne prévoit pas de façon directe d'estimer un modèle de distorsion. L'implantation d'OpenCV permet d'inclure quelques paramètres d'un modèle polynomial de distorsion radiale et tangentielle, mais un estimé de départ des paramètres est requis [67]. Ceux-ci sont utilisés pour rectifier les images auxquelles l'algorithme de calibrage est appliqué. La solution est raffinée par optimisation non-linéaire de l'erreur de reprojection.

4.7.2 Méthode pour modèle radial symétrique

La méthode de calibrage de Hartley et Kang permet d'abord d'estimer le centre de distorsion radiale ; puis, simultanément, un modèle non-paramétrique de distorsion radiale et les homographies de transfert des points du plan de calibrage vers les images rectifiées [61]. Le modèle non-paramétrique de distorsion peut s'écrire :

$$\mathbf{p}_i^d = \mathbf{c} + \lambda_i(\mathbf{p}_i^u - \mathbf{c}), \quad \mathbf{c} \in \mathbb{P}^2 \tag{4.4}$$

où, λ_i est un paramètre de distorsion uniquement associé au point $\mathbf{p}_i^d$ et $\mathbf{c}$ est le centre de distorsion. La première étape consiste à calculer du centre de distorsion. En multipliant l'équation précédente de chaque côté par $[\mathbf{c}]_\times$ et en notant que $\mathbf{p}^u \propto \mathsf{H}\mathbf{p}_i^c$

où, $\mathsf{H}_{(3\times3)}$ est l'homographie (inconnu) entre le plan de calibrage et le plan image rectifié, on obtient :

$$[\mathbf{c}]_\times \mathbf{p}_i^d = \lambda_i [\mathbf{c}]_\times \mathsf{H} \mathbf{p}_i^c.$$

Finalement, en multipliant par la gauche par $\mathbf{p}_i^{d\mathsf{T}}$ et on notant que $\mathbf{p}_i^{d\mathsf{T}}[\mathbf{c}]_\times \mathbf{p}_i^d = 0$, on détermine la *matrice fondamentale de distorsion radiale* F :

$$0 = \mathbf{p}_i^{d\mathsf{T}} \underbrace{\lambda_i [\mathbf{c}]_\times \mathsf{H}}_{\mathsf{F}_{(3\times3)}} \mathbf{p}_i^c$$

qui est de rang 2 comme la matrice fondamentale traditionnelle. Le noyau droit de F donne directement le centre de distorsion $\mathbf{c}$ et le noyau gauche correspond à l'intersection de l'axe optique de la caméra avec le plan de calibrage.

Une fois que $\mathbf{c}$ est estimé, il est possible de changer le système de coordonnées de l'image pour le mettre à l'origine : $\tilde{\mathbf{p}}_i^d \leftarrow \tilde{\mathbf{p}}_i^c - \tilde{\mathbf{c}}$ et $\mathbf{c} \leftarrow (0,0,1)^\mathsf{T}$. On peut alors calculer à nouveau $\mathsf{F} = [\mathbf{c}]_\times \mathsf{H}$, donnant directement les deux première rangées de $\mathsf{H}_{(3\times3)} = [\mathbf{F}_2 \ -\mathbf{F}_1 \ \mathbf{v}]^\mathsf{T}$, où, $\mathbf{v}$ est inconnu. L'étape finale consiste à estimer $\mathbf{v}$. Notons :

$$r_i^u = \|\mathbf{p}_i^u\| = \left\| \begin{bmatrix} \mathbf{H}_1^\mathsf{T} \\ \mathbf{H}_2^\mathsf{T} \end{bmatrix} \mathbf{p}_\mathbf{i}^\mathbf{c} / (\mathbf{v}^\mathsf{T} \mathbf{p}_\mathbf{i}^\mathbf{c}) \right\|, \quad r_i^d = \|\mathbf{p}_i^d\| \tag{4.5}$$

les distances par rapport au centre de distorsion pour un point rectifié et pour un point distordu, respectivement. En triant les points en ordre croissant de r_i^d ($r_i^d \leq r_{i+1}^d$), on obtient aussi la relation pour les rayons des points rectifiés : $r_i^u \leq r_{i+1}^u$. Pour retrouver $\mathbf{v}$, les auteurs choisissent de minimiser une quantité reliée à la variation totale des rayons rectifiés $\sum_i (r_i^u - r_{i+1}^u)$, qui peut être réécrite en utilisant (4.5) :

$$\sum_i \left((\mathbf{v}^\mathsf{T}\mathbf{q}_{i+1})(\mathbf{v}^\mathsf{T}\mathbf{q}_i)(r_i^u - r_{i+1}^u) \right) = \sum_i \left(\mathbf{v}^\mathsf{T}\mathbf{q}_{i+1} \left\| \begin{bmatrix} \mathbf{H}_1^\mathsf{T} \\ \mathbf{H}_2^\mathsf{T} \end{bmatrix} \mathbf{p}_i^c \right\| - \mathbf{v}^\mathsf{T}\mathbf{q}_i \left\| \begin{bmatrix} \mathbf{H}_1^\mathsf{T} \\ \mathbf{H}_2^\mathsf{T} \end{bmatrix} \mathbf{q}_{i+1}) \right\| \right),$$

qui permet de construire un système linéaire pour estimer $\mathbf{v}$ à partir des correspon-

dances. L'estimation de $\mathbf{v}$ permet l'estimation de $\mathbf{p}_i^u$ à partir de (4.5) et ensuite λ_i à partir de (4.4). Cette méthode a été testée au chapitre 5.

4.7.3 Méthode pour le modèle angulaire polynômial

Kannala et Brandt proposent un algorithme pour calibrer le modèle angulaire polynômial (*cf.*§4.3.6) [72]. À ce modèle, ils ajoutent des coefficients de distorsion tangentielle. Leur méthode nécessite un estimé de l'angle de vue et de la distance focale de la caméra, obtenus à partir des données du manufacturier de l'objectif de caméra. L'algorithme se résume en quelques étapes :

- estimation de k_1 et k_2 a partir d'un des modèles de Fleck [41] et les informations sur la focal f et l'angle de vue de la caméra ;
- initialisation des autres paramètres de distorsion à 0 ;
- rectification des points $\tilde{\mathbf{p}}_i^d$ pour obtenir une estimation[9] de $\mathbf{p}_i^u$;
- calcul d'une homographie entre les points $\mathbf{p}_i^u$ et $\mathbf{p}_i^c$;
- extraction des paramètres externes ;
- minimisation de l'erreur de reprojection pour tous les paramètres par une méthode itérative.

Cette méthode comporte une limitation, à savoir qu'un estimé des paramètres de la caméra doit être connu au départ et que l'optimisation du modèle se fait uniquement par minimisation non-linéaire. Cette méthode de calibrage n'est donc pas spécifique au modèle angulaire polynômial.

4.7.4 Méthode pour le modèle rationnel

Il s'agit d'une méthode linéaire permettant calibrer le modèle rationnel (*cf.*§4.5.1) à l'aide de points de correspondance avec une seule vue du plan de calibrage [32]. Il

[9] Le point $\mathbf{p}_i^u$ est en coordonnée projective parce que le modèle peut représenter des caméras omnidirectionnelles.

est possible de construire une relation linéaire :

$$\begin{aligned} \mathbf{p}_i^u &\propto \mathsf{H}_{(3\times3)}\mathbf{p}_i^c \\ \mathsf{A}_{(3\times6)}\chi(\tilde{\mathbf{p}}_i^d) &\propto \mathsf{H}\mathbf{p}_i^c \\ \mathsf{H}^{-1}\mathsf{A}\chi(\tilde{\mathbf{p}}_i^d) &\propto \mathbf{p}_i^c \\ [\mathbf{p}_i^c]_\times \underbrace{\mathsf{H}^{-1}\mathsf{A}}_{\mathsf{A}'_{(3\times6)}}\chi(\tilde{\mathbf{p}}_i^d) &= 0 \end{aligned}$$

où, H est la transformation entre le plan de calibrage et le plan image rectifié et A′ est la matrice A des coefficients de distorsion à une homographie H près. L'estimation de A′ à partir des correspondances permet de rectifier l'image, même si H est inconnue.

4.7.5 Méthode pour les caustiques

La méthode de Swaminathan *et al.* permet de calibrer la surface caustique pour une caméra non-centrale [148]. Elle utilise des correspondances entre deux vues d'un plan pour une caméra subissant une translation pure connue. Le problème est donc d'estimer les paramètres de la caustique et la position 3D des points de correspondance situés sur le plan. L'erreur de reprojection est minimisée par une méthode itérative.

4.7.6 Méthode pour le modèle générique

Sturm et Ramalingam proposent une méthode de calibrage de caméra générique nécessitant au moins trois vues d'un plan de calibrage (seulement deux si la projection est centrale) [112, 145]. Supposons un pixel image $\mathbf{p}^d$ pour lequel les trois correspondances $\mathbf{q}^1, ..., \mathbf{q}^3 \in \mathbb{P}^2$ avec le plan de calibrage sont obtenues pour différentes poses. En supposant que le système de coordonnées monde est aligné avec la vue du premier plan, on peut représenter toutes les coordonnées $\mathbf{q}^i$ par des coordonnées 3D $\mathbf{Q}^i \in \mathbb{P}^3$ dans le même système de coordonnées en leur appliquant une transformation eucli-

dienne (inconnue) :

$$\begin{aligned}
\mathbf{Q}^1 &= \begin{pmatrix} q_1^1 & q_2^1 & 0 & q_4^1 \end{pmatrix}^\mathsf{T} \\
\mathbf{Q}^2 &= \begin{bmatrix} \mathsf{R}^1_{(3\times3)} & \mathbf{t}^1 \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} \begin{pmatrix} q_1^2 & q_2^2 & 0 & q_4^2 \end{pmatrix}^\mathsf{T} \\
\mathbf{Q}^3 &= \begin{bmatrix} \mathsf{R}^2_{(3\times3)} & \mathbf{t}^2 \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} \begin{pmatrix} q_1^3 & q_2^3 & 0 & q_4^3 \end{pmatrix}^\mathsf{T}.
\end{aligned}$$

Autrement dit, $\mathbf{Q}^2$ et $\mathbf{Q}^3$ sont connus à une transformation euclidienne près. L'objectif de la méthode de calibrage est donc de retrouver les paramètres R^i et $\mathbf{t}^i$, *i.e.* la pose du plan de calibrage pour chaque vue. Après quoi ceci permet de joindre les trois points $\mathbf{Q}^i$ pour obtenir un rayon d'échantillonage associé au pixel. Pour le calcul de pose, on place les trois vecteurs côte à côte dans une matrice :

$$\mathsf{Q}_{(4\times3)} = \begin{bmatrix} \mathbf{Q}^1 & \mathbf{Q}^2 & \mathbf{Q}^3 \end{bmatrix}.$$

Comme cette matrice est de rang 3, chaque sous-matrice de taille (3×3) a un déterminant égal à zéro. Ces contraintes permettent de former un tenseur duquel les coefficients des R^i et $\mathbf{t}^i$ peuvent être extraits. La méthode est encore plus simple lorsque la caméra est centrale.

4.7.7 Méthodes avec des images de lignes

La discussion se rapportant à ces méthodes est présentée au chapitre 6 puisqu'elles sont reliées à une de nos contributions décrite au chapitre 7.

4.8 Contributions

Seules les méthodes génériques peuvent être appliquées autant aux caméras centrales que non-centrales. Les autres sont dédiés aux caméras centrales uniquement

(Hartley, Clauss et Fitzgibbon) ou spécifique à certaines configurations non-centrales (caustique). Les méthodes génériques ont un grand potentiel, mais elles nécessitent un grand nombre d'images pour des résultats fiables. Nos recherches ont porté sur l'élaboration d'un modèle et d'une méthode de calibrage applicable à une large classe de caméras incluant les centrales et non-centrales. Ceci est le sujet du prochain chapitre.

Nous y détaillons le modèle de distorsion radiale symétrique avec points de vue multiples. La fonction de distorsion et le déplacement du centre optique peuvent prendre des formes paramétrique ou non-paramétrique. Nous présentons ensuite trois nouvelles méthodes de calibrage plan de ce modèle, ne nécessitant aucune information concernant la caméra (angle de vue, catadioptrique ou non, centrale ou non). Dans le cas d'un modèle non-paramétrique, des correspondances denses entre le plan de calibrage et les images sont nécessaires. Ces correspondances denses sont obtenues par lumière structurée (le sujet des chapitres 10 et 11). Pour l'estimation d'un modèle paramétrique, il n'est pas nécessaire d'utiliser des correspondances denses comme c'est le cas des méthodes présentées plus haut.

Chapitre 5

CALIBRATION OF CAMERAS WITH RADIALLY SYMMETRIC DISTORTION

Cet article [151] a été publié comme l'indique la référence bibliographique.

Tardif J.-P., Sturm P., Calibration of Cameras with Radially Symmetric Distortion, dans Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras, (OMNIVIS'2005, en parralèle avec ICCV 2005), Chine, Octobre, pp. 44-51, 2005.

Nous présentons ici une version contenant de nouvelles contributions qui sont soumises à la revue Pattern Analysis and Machine Intelligence (PAMI).

Abstract

We present a theory and algorithms for plane-based calibration and pose recovery of general radially distorted cameras. By this we understand cameras that have a distortion center and an optical axis such that the projection rays of pixels lying on a circle centered on the distortion center, form a right *viewing cone* centered on the optical axis. The camera is said to have a singular viewpoint (SVP) if all such viewing cones have the same vertex (the optical center), otherwise we speak of NSVP cases where each viewing cone may have its own optical center on the optical axis. This model encompasses the classical radial distortion model [26], fisheyes, most central or non-central catadioptric cameras, and other cameras with radially symmetric caustics.

Calibration consists in the estimation of the distortion center, the opening angles of all viewing cones and their optical centers. We present two approaches of computing

a full calibration from dense correspondences of a single or multiple planes with known Euclidean structure. The first one is based on a geometric constraint linking viewing cones and their intersections (conic sections) with the calibration plane. We show that the conics generated by one particular camera can be represented by a family of curves whose parameters identify the extrinsics of the camera. The calibration of the camera corresponds to finding the parameters of that family. The second approach is akin to classical homography-based methods. First, partial homographies are estimated to recover the optical axis. Secondly, the position of the camera and the opening angle of the viewing cones are estimated. Experiments using simulated and a broad variety of real cameras show very good stability. Furthermore, we provide a comparison with Hartley-Kang's algorithm [61], which however can not handle such a broad variety of camera configurations, showing very similar performance.

5.1 Introduction

In the last few years, we have seen an increasing interest in non-conventional cameras and projection models, going beyond affine or perspective projection. There exists a large diversity of camera models, many of which specific to certain types of projections [82, 136], others applicable to families of cameras such as central catadioptric systems [6, 9, 40, 45]. All these models are described by a few intrinsic parameters, much like the classical pinhole model, possibly enhanced with radial or decentering distortion coefficients. Calibration methods exist for all these models, and they are usually tailor-made for them, *i.e.* can not be used for any other projection model [94]. Several works address the calibration problem from an opposite point of view, by adopting a very generic imaging model that incorporates most commonly used cameras [29, 53, 55, 145, 146].

In the most general case, cameras are modeled by attributing an individual projection ray to each pixel. Such a model is highly expressive, but it is difficult to

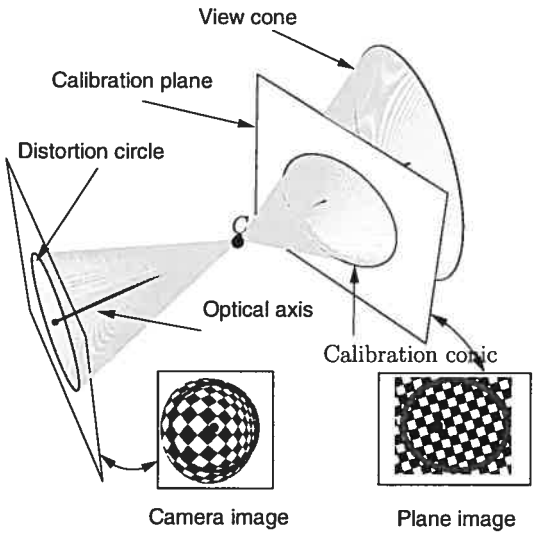


Figure 5.1. Our camera model (see text for explanations). The inlayed illustrations show the distortion center (in blue) and a distortion circle for a true image taken with a fisheye, and the corresponding calibration ellipse overlaid on a pattern shown on the calibration plane.

obtain a stable calibration of cameras with it, at least with few input images taken in an uncontrolled environment. Finally, many researchers have proposed a compromise between parametric and such generic models. They assume radial symmetry around the distortion center (often considered as coinciding with the principal point), but with a general distortion function [61, 151, 152, 156, 157].

In this paper, we adopt this type of model, which we find sufficiently general to model many common types of cameras, but with fewer parameters than the fully generic model, thus making calibration easy and stable. We model cameras using the notions of a **distortion center** in the image and an **optical axis** in 3D. For cameras with **radially symmetric distortion**, the projection rays associated with pixels lying on a same **distortion circle** centered on the distortion center, lie on a right **viewing cone** centered on the optical axis. Let us call the line spanned by a pixel

and the distortion center a **radial line** and the plane spanned by the projection ray associated with the pixel and the optical axis, a **radial plane**. In our camera model, we specify that angles between radial lines and associated radial planes are identical (otherwise, it would be a very uncommon camera). This radially symmetric camera model is illustrated in fig. 5.1. It encompasses many common camera models, such as pinhole (modulo aspect ratio and skew), the classical polynomial radial distortion model, fisheyes, or any catadioptric system whose mirror is a surface of revolution, and for which the optical axis of the perspective (or affine) camera looking at it is aligned with the mirror's revolution axis. Our model comprises **central** cameras (SVP), where all viewing cones have the same vertex (the **optical center**), but also **non-central** ones (NSVP), for which the viewing cones' vertices lie anywhere on the optical axis. In the latter case, we may speak of one optical center per viewing cone.

Problem statement. We want to calibrate cameras based on the above model from one or several images of a calibration plane in unknown positions. The problem of calibration from a planar scene of known geometry has been studied intensively in computer vision. Proposed approaches are usually based on constraints on the image of the absolute conic (IAC) [142, 181], and also the image of the circular points (ICP) [58, 59].

The input to the proposed calibration algorithms is the Euclidean structure of the plane(s) and a dense matching between the plane(s) and the camera image. From this, we compute, for all viewing cones, their focal length (equivalent to the opening angle). Our algorithms assume a known position of the distortion center, but we also show how to estimate it, using repeated calibrations for different candidates. Calibration also comprises a pose estimation problem: estimating the orientation of the optical axis (relative to a calibration plane) and the location of each viewing cone's vertex on it.

Organization. A geometric study of our model is presented in §5.2, together with our first calibration approach. The second approach, based on the estimation

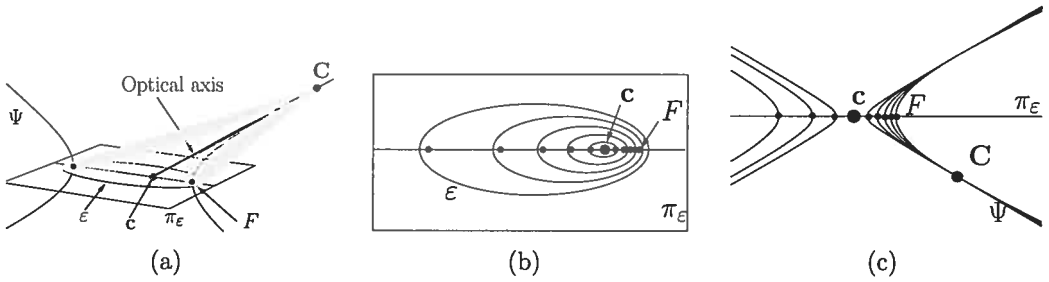


Figure 5.2. Illustrations of the geometry of viewing cones, calibration conics (here ellipses) and location of optical center in the SVP case. (a) Complete illustration for one viewing cone. (b) View of the calibration plane, showing many cones' calibration ellipses. Note that their major axes are collinear. (c) Side view of the viewpoint conics (hyperbolas in this case) associated with many calibration conics.

of a projection function, is described in §5.3. We then describe an algorithm for estimating the distortion center in §5.4. Several practical issues and experimental results, including a comparison with Hartley-Kang's methods [61], are presented in §5.5 and §5.6, respectively. Finally, we conclude in §5.7.

5.2 Geometry

5.2.1 One Distortion Circle in One Image of a Calibration Plane

Let us consider one distortion circle in the image plane. Its associated viewing cone cuts the calibration plane in a **calibration conic**. From the dense matches between image and calibration plane, we can compute this calibration conic (see §5.5 for more on the dense matching). This conic can be either an ellipse or a hyperbola (the parabola is only a theoretically interesting case). If we knew the position of the camera's optical center relative to the calibration plane, then we could directly compute the viewing cone of our distortion circle, i.e. the cone that has the optical center as vertex and that contains the above calibration conic. As mentioned, that

cone has several useful properties: its axis is the camera's optical axis and it is a right cone, *i.e.* rotationally symmetric with respect to its axis. From the cone, the focal length of the considered distortion circle can be easily computed: the cone's opening angle equals the field of view.

In practice, we do not know the optical center's position relative to the calibration plane. In the following, we show geometrical relations between the calibration conic, the optical center and the optical axis of the camera, that allow to compute the optical center. Recall that when talking about optical center, we mean the optical center per distortion circle; they all lie on the optical axis and in the SVP case, they are identical.

Without loss of generality, we assume that the calibration plane is the plane $Z = 0$, and that the calibration conic ε is given by the matrix $\varepsilon \propto \mathrm{diag}(a, b, -1)$ ($\propto$ represents equality up to scale), *i.e.* the X-axis is the conic's major axis. The type of the calibration conic ε depends on a and b as follows:

$$\begin{cases} b \geq a > 0 & \text{ellipse} \\ b < 0 \ \text{ and } \ |b| > a > 0 & \text{hyperbola.} \end{cases} \tag{5.1}$$

Our aim is to provide constraints on the position of the optical center, as well as on the orientation of the optical axis, from this conic. We first do this for the case of a calibration ellipse, then for the case of a hyperbola.

The case of calibration ellipses. Let us first state a well-known result. Consider a right cone whose vertex is a point with real coordinates, and its intersection with a plane. For now, we assume that the intersection is an ellipse (the case of the hyperbola will be discussed later). It is easy to prove that the orthogonal projection of the cone's vertex onto the plane, lies on the ellipse's major axis (*cf.*fig. 5.2(a)). This implies that the cone's vertex lies in the plane that is orthogonal to the ellipse's supporting plane and that contains its major axis.

For our problem, this means that the optical center must lie in the plane $Y = 0$ (since the ellipse lies in plane $Z = 0$ and has the X-axis as major axis). We may further constrain its position $\mathbf{C} = (X, 0, Z, 1)^\mathsf{T}$, as follows [18]. The cone with $\mathbf{C}$ as vertex and that contains the calibration ellipse ε, is given by:

$$\Lambda \propto \begin{bmatrix} a\,Z^2 & 0 & -a\,X\,Z & 0 \\ 0 & b\,Z^2 & 0 & 0 \\ -a\,X\,Z & 0 & a\,X^2 - 1 & Z \\ 0 & 0 & Z & -Z^2 \end{bmatrix}.$$

For this cone to be a right one, its upper left 3×3 matrix $\bar{\Lambda}$ must have a double eigenvalue. The three eigenvalues are:

$$b\,Z^2, \frac{a\,(X^2+Z^2) - 1 \pm \sqrt{4a\,Z^2 + (1-a\,(X^2+Z^2))^2}}{2}. \tag{5.2}$$

The second and third eigenvalues can not be equal for real values of X and Z (besides in the trivial case $X = Z = 0$ which is excluded since it would correspond to the optical center lying in the calibration plane). The first eigenvalue is equal to the third one if $Z = 0$ (excluded for the same reason as above) and to the second one if:

$$a\,b\,X^2 + b(a-b)\,Z^2 + (a-b) = 0. \tag{5.3}$$

This equation tells us that the optical center lies on a **viewpoint conic** in the plane $Y = 0$, given by the following matrix and the associated equation

$$\Psi \propto \begin{bmatrix} a\,b & & \\ & b(a-b) & \\ & & a-b \end{bmatrix}, \qquad \begin{pmatrix} X & Z & 1 \end{pmatrix} \Psi \begin{pmatrix} X \\ Z \\ 1 \end{pmatrix} = 0. \tag{5.4}$$

This is a hyperbola, due to $b \geq a > 0$ (cf. (5.1)); it is sketched in fig. 5.2(a). Furthermore, its asymptotes correspond to the directions of the two circular cylinders that contain the calibration ellipse.

Let us now consider the orientation of the optical axis. Due to (5.3), we have an optical center given by:

$$Z = \pm\sqrt{\frac{a\,b\,X^2 + a - b}{b(b-a)}}. \tag{5.5}$$

Here, we exclude the case $a = b$, which would correspond to the camera looking straight at the calibration plane.

The direction of the cone's axis is given by the eigenvector associated with the single eigenvalue of $\bar{\Lambda}$, *i.e.* the third one, augmented with a homogeneous coordinate 0:

$$\begin{pmatrix} \pm\sqrt{b(b-a)(a\,b\,X^2 + a - b)} \\ 0 \\ a\,b\,X \\ 0 \end{pmatrix}. \tag{5.6}$$

We now show that the cone's axis is identical with the tangent of the hyperbola Ψ in the optical center $\mathbf{C}$, which is given by the line (in the plane $Y = 0$):

$$\Psi \begin{pmatrix} X \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} a\,b\,X \\ \mp\sqrt{b(b-a)(a\,b\,X^2 + a - b)} \\ a - b \end{pmatrix}.$$

Its point at infinity is (still in the plane $Y = 0$):

$$\begin{pmatrix} \pm\sqrt{b(b-a)(a\,b\,X^2 + a - b)} \\ a\,b\,X \\ 0 \end{pmatrix}$$

i.e. it is identical with the point given in (5.6). Hence, for an optical center on Ψ, the optical axis is directly given by the associated tangent.

Coming back to (5.2), the eigenvalues of the cone can be used to compute the focal length f_d associated to a distortion circle of radius d. The value f_d/d is the tangent of half the opening angle of the viewing cone. Furthermore, it can be shown that $(f_d/d)^2$ equals the negative of the ratio of the double and the single eigenvalue of $\bar{\Lambda}$, *i.e.* the negative of the ratio of the first and the third eigenvalues given in (5.2). Using (5.5), we get:

$$\left(\frac{f_d}{d}\right)^2 = -\frac{b\,(abX^2 + a - b)}{a(a-b)}. \tag{5.7}$$

This relation will prove useful in §5.2.4.

The case of calibration hyperbolas. The case where the intersection between a cone and the calibration plane yields a hyperbola is accounted for automatically by the previous formulation. All above findings hold here, with the exception that the viewpoint conic Ψ is an ellipse. This case typically occurs with very wide angle cameras or when the angle between the camera's optical axis and the calibration plane is large.

5.2.2 Multiple Distortion Circles

So far, we have shown that for an individual distortion circle, the associated viewing cone can be determined from the associated calibration conic, up to 1 degree of freedom (location on the viewpoint conic Ψ and associated orientation of the optical axis). We now show how to get a unique solution, when considering several distortion circles simultaneously. Let us first note that calibration conics corresponding to different distortion circles are not independent: their major axes are collinear (*cf*.fig. 5.2(b)), even in the NSVP case. Their centers are not identical however, unless they are all circles, which can only happen when the camera looks straight at the calibration plane.

Let Ψ_d be the viewpoint conic associated with distortion circle (of radius) d, all Ψ_d being given in the same coordinate frame. In the case of a single viewpoint camera, the optical center must lie on all these conics. Furthermore, the optical axis is tangent to all of them. This implies that all conics touch each other (have a double intersection point) in the optical center. This is illustrated in figure 5.2(c). A naïve algorithm would compute the viewpoint conic for all calibration conics and seek their single intersection/contact point. However, very little noise can cause two viewpoint conics to have no real intersection point at all, instead of a double one.

Interestingly, this constraint gives a geometric explanation of the ambiguity, or correlation, between camera position and focal length that often occurs when calibrating a camera from a single view. Consider perfectly recovered viewpoint conics (such as in fig. 5.2), the ambiguity is observed as the area where the viewpoint conics are "very close" to each other. Clearly, a very low perturbation of the two curves can result in a significant error on the optical center and the focal length.

In the NSVP case, to each distortion circle and viewing cone corresponds a separate optical center. Hence, the viewpoint conics won't have a single contact point anymore. However, the optical axis is shared by all viewing cones. Hence, it is the single (in general) line that is tangent to all viewpoint conics. Furthermore, each contact point with the optical axis is the associated optical center.

5.2.3 Solving the Calibration and the Pose

In the following, we first propose a naïve calibration approach; it is not optimal though and a better one will be discussed in the next section. In the SVP case, one could solve for the calibration by first estimating the optical center (relative to the calibration plane) as the 3D point which is closest on average to all viewpoint conics (see below). Then, (5.7) can be used to compute the focal lengths for all distortion circles. In the NSVP case, a possibility would be to compute the optical axis: the line $\mathbf{L}$ that minimizes the sum of squared distances to the viewpoint conics.

SVP case: Computing the closest point to the viewpoint conics. Computing the orthogonal distance of a point to a general conic requires solving a fourth degree polynomial [179]. Using this to compute the closest point to our set of viewpoint conics is not very practical. Instead, we iteratively minimize a cost function subject to constraints. The closest point $\mathbf{q}$ is found by solving:

$$\min_{\mathbf{q},\mathbf{q}_d} \sum_d dist(\mathbf{q},\mathbf{q}_d)^2, \text{ subject to } \mathbf{q}_d^\top \Psi_d \mathbf{q}_d = 0,$$

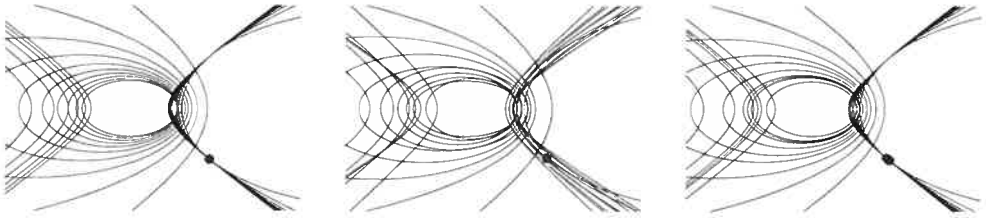
i.e. we also estimate one point per conic Ψ_d that will, after convergence, be the orthogonal projection of $\mathbf{q}$ on Ψ_d. Since the function and constraints are polynomial, this problem can be optimized using an algorithm relying on Cylindrical Algebraic Decomposition (CAD) which guarantees a global minimum [139]. Such an algorithm is available through the *Minimize* function of *Mathematica*. The problem can also be solved by standard non-linear optimization.

5.2.4 A formulation enforcing the constraints directly

In our first experiments, we found the previous approach to be unstable, even though good results could be obtained in some cases [151]. This is because the formulation has several drawbacks. First, although CAD optimization is algorithmic, it becomes computationally intractable when the number of viewpoint conics increases. Secondly, it appears that finding the closest point to the set of the recovered viewpoint conics is not the optimal criterion. As discussed above and shown in fig. 5.3, when the noise is large, the shape and position of the viewpoint conics may become very perturbed.

For an SVP camera, a better formulation would directly enforce that the viewpoint conics all touch in one location and have the same tangent at this point. Then, CAD optimization could be avoided.

Before giving our solution, we come back to our calibration conics. As mentioned above, they share an axis (in the absence of noise). We assume that their common



(a) (b) (c)

Figure 5.3. The effect on viewpoint conics of errors on the calibration conics (shown in the same plane for visualization). (a) Original configuration with the optical center. (b) Added noise to the calibration conics and resulting viewpoint conics. Obviously, finding the closest point to these curves will yield a large error. (c) Recovered optical center in dark/blue and corrected conics given by our approach, *cf.* §5.2.4.

axis can be estimated with high accuracy despite noise in the data, due to using many calibration conics. Once the conics' axis is estimated, it is convenient to change the coordinate system of the calibration plane such that the conics are aligned with the X-axis. Then, each one of them can be parameterized by its major and minor axis lengths b_d and a_d, as well as by a displacement k_d along the X-axis:

$$\varepsilon_d \propto \begin{bmatrix} a_d & 0 & -a_d\, k_d \\ 0 & b_d & 0 \\ -a_d k_d & 0 & a_d\, k_d^2 - 1 \end{bmatrix}. \tag{5.8}$$

The parameters a_d, b_d and k_d can be estimated much like in a classical conic fitting algorithm. We perform all subsequent computations with those axis-aligned conics. This parameterization guarantees that all viewpoint conics lie in the same plane

$(Y = 0)$, in which they can be expressed as:

$$\Psi_d \propto \begin{bmatrix} a_d\, b_d & 0 & -a_d\, b_d\, k_d \\ 0 & b_d\,(a_d - b_d) & 0 \\ -a_d\, b_d k_d & 0 & a_d\, b_d\, k_d^2 + a_d - b_d \end{bmatrix}. \tag{5.9}$$

However, this parameterization does not guarantee other properties given in §5.2.2, especially that these viewpoint conics all touch in a single point. To achieve this, we state a new result. For a central radially symmetric camera, the intersections of all its viewing cones with the calibration plane are calibration conics $\hat{\varepsilon}_d$ given by:

$$\hat{\varepsilon}_d \propto \begin{bmatrix} \gamma^2\rho_d & 0 & \epsilon\rho_d\gamma + \gamma \\ 0 & \delta^2\,(2\rho_d + 1) & 0 \\ \epsilon\rho_d\gamma + \gamma & 0 & \rho_d\epsilon^2 + 2\epsilon - 2 \end{bmatrix}, \tag{5.10}$$

where γ, ϵ and δ encode the external parameters of the camera, and ρ_d is a parameter for each image circle of radius d. For each calibration conic $\hat{\varepsilon}_d$, the corresponding viewpoint conic is given by:

$$\hat{\Psi}_d \propto \begin{bmatrix} \gamma & 0 & \epsilon + \frac{1}{\rho_d} \\ 0 & \frac{\gamma^2 - 2\delta^2 - \frac{\delta^2}{\rho_d}}{\gamma} & 0 \\ \epsilon + \frac{1}{\rho_d} & 0 & \frac{\gamma^2 + 2\delta^2(\epsilon - 1) + \delta^2\epsilon^2\rho_d}{\gamma\delta^2\rho_d} \end{bmatrix}. \tag{5.11}$$

The derivation and the proof of this formulation are presented in Appendix 5.8. Note that in practice, we are only interested in the estimation of $\hat{\varepsilon}_d$. Much like (5.7) can be obtained for Ψ_d, we can compute $(f_d/d)^2$ for $\hat{\Psi}_d$. After some algebraic manipulation using that result (not shown here) we substitute ρ_d in (5.9) and (5.10) with:

$$\rho_d = -\frac{\phi_d\gamma^2}{2\,(\gamma^2 - 2\delta^2)} - \frac{1}{2}. \tag{5.12}$$

Table 5.1. Computing the external parameters of the camera from the parameters

Information	Formula
X-coordinate of the optical center	$-\frac{\gamma}{\delta^2} - \frac{\epsilon}{\gamma} + \frac{2}{\gamma}$
Z-coordinate of the optical center	$-\sqrt{-\frac{\gamma^2-2\delta^2}{\delta^4}}$
Intersection of optical axis and calibration plane	$-\epsilon/\gamma$
Intersection of calibration plane and principal plane[1] (setting $\phi_d = 0$)	$y = (2-\epsilon)/\gamma$

[1] The principal plane refers to a plane passing by the optical center with its normal parallel to the optical axis

where ϕ_d is chosen so to be equal to $(f_d/d)^2$.

The formulas for estimating the external parameters of the camera are summarized in table 5.2.4. The position of the camera is obtained by solving:

$$(X, Z, 1)\, \Psi_i\, (X, Z, 1)^\top = (X, Z, 1)\, \Psi_j\, (X, Z, 1)^\top$$

for $(i \neq j)$. The value $-\frac{\epsilon}{\gamma}$ gives the X-coordinate of the intersection point of the optical axis with the calibration plane. Indeed, we can verify that

$$(-\epsilon/\gamma, 0, 1)\, \Psi_d\, (X, Z, 1)^\top = 0, \forall d$$

where $\Psi_d\, (X, Z, 1)^\top$ is the tangent to the viewpoint conic at the optical center, *i.e.* the optical axis. One may wonder why the extrinsics were not directly "encoded" in the parameterization. Indeed, such parameterizations exist and were investigated. However, they were abandoned because the resulting calibration conics $\hat{\varepsilon}_d$ were not as "simple" as the one we propose. When a calibration algorithm could be deduced from any of these formulations, it was less stable than the one we show next.

With this global formulation, all calibration conics $\hat{\varepsilon}_d$ should be fitted a the same time. However, this task is difficult since the function is non-linear.

Perhaps surprisingly, there exists an analytic solution to computing the extrinsics

parameters once the values of a_d, b_d and k_d for all calibration conics are estimated. The position of the camera can be estimated, while ignoring the focal length at each circle. We exploit the fact that without noise $\forall d : \varepsilon_d = \hat{\varepsilon}_d$, *i.e.* that the unconstrained and constrained formulation should give identical results[1]. We solve this equation for each parameter ϵ, γ and ϕ_d and obtain:

$$\phi_d = \frac{2b_d\,(b_d - a_d)}{\delta^2 a_d} - 1 \tag{5.13}$$

$$\tilde{\epsilon}_d = -\frac{\left(\sqrt{\gamma^2 + a_d}\right)}{\sqrt{a_d}} - \gamma k_d + 1 \tag{5.14}$$

$$\tilde{\gamma}_d = \pm\frac{\delta\sqrt{a_d}\sqrt{\delta^2 + 2b_d}}{b_d}, \tag{5.15}$$

for each calibration conic. Note that the two last equations involve only the extrinsics parameters. We rename them with a $\tilde{}$ and a d subscript because, in the presence of noise, they are different for each calibration conic. However, all of them should be very close. This can be exploited to estimate δ by minimizing the variance of $\forall d : \tilde{\gamma}_d$. In practice, we prefer a minimization over $\tilde{\gamma}_d^2$, since it eliminates the square root in the expression. This results in solving:

$$\begin{aligned} \delta^2 &= \arg\min_{\delta} \sigma^2, \text{where} \\ \sigma^2 &= \sum_d (\mu - \tilde{\gamma}_d^2)^2 \end{aligned} \tag{5.16}$$

where μ is the mean. The expression σ^2 is a polynomial of degree 4 in δ^2, *i.e.* it admits only three extrema in δ^2. One of them corresponds to $\delta = 0$ ans is a maximum, and the two others are minima with identical absolute value but opposite sign. The one larger than zero is our solution for δ^2 and δ can always be taken positive. Given this value, we can estimate the other parameters with $\epsilon = \text{Mean}(\tilde{\epsilon}_d)$ and $\gamma = \pm\text{Mean}(\tilde{\gamma}_d)$

[1] We scale ε_d and $\hat{\varepsilon}_d$ such that they have unity at the upper-left coordinate.

using (5.14) and (5.15). This sign ambiguity will be resolved latter. In the presence of noise, we perform a final optimization:

$$\arg\min_{\delta,\epsilon,\gamma} \sum_d (\epsilon - \tilde{\epsilon}_d)^2 + (\gamma - \tilde{\gamma}_d)^2 \tag{5.17}$$

using a standard Gauss-Newton algorithm. Here, it is important to mention that the latter optimization is unnecessary if the noise is small. Most importantly, it can be completely avoided in one case. To see how, we come back to the meaning of the parameter ϵ. From the equation in table 5.2.4, we see immediately that it is the intersection of the optical axis on the calibration plane (modulo the $1/\gamma$ factor) w.r.t. to a given point on th X-axis. However, this intersection point can be easily obtained: it is the point on the calibration plane that matches the distortion center (*cf*.the matching described in §5.5). In this case, we may fix the coordinate system to have this point at the origin and set ϵ to zero. All the previous equations are greatly simplified and the optimization (5.17) can be avoided.

Once the external parameters of the camera are known, that is, the value of δ, ϵ and γ, we are seeking the intrinsics ϕ_d. Although (5.13) could be used, a better solution is to find $\hat{\varepsilon}_d$ that best fits the original data points. Given the external parameters, fitting the conic with a least square algebraic cost function is straightforward and only involves the variable ϕ_d. It is a second degree polynomial. We try both signs for γ and keep the one that best fits the calibration conics.

Many calibration planes. If many calibration planes are available, the focal length of each circle must be consistent over all the views. One could average the recovered focal lengths over each view. Even better, calibration conics for different planes can be fitted at the same time in a global Linear Least Square problem where ϕ_d is the only variable. The function to minimize is again a second degree polynomial.

In the following, this calibration approach is referred to as the Right Cone Constraint method, `RCC`.

INPUT: Dense plane–image correspondences
ALGORITHM:
For each calibration plane v:

- Estimate individually every calibration conic ε_d.
- Compute the average major axis $\mathbf{m}$ and select a point, denoted $\mathbf{g}$, on the axis.
- Change the coordinate system of the original data to set $\mathbf{m}$ on the X-axis and $\mathbf{g}$ at the origin.
- Estimate the new ε_d of the form: $\propto \begin{bmatrix} a & 0 & c \\ 0 & b & 0 \\ c & 0 & d \end{bmatrix}$.
- Solve (5.16).
- Minimize (5.17) and obtain $\epsilon_v, \gamma_v, \delta_v$.

Compute, for all distortion circles d, the ϕ_d using (5.10).

Figure 5.4. Complete algorithm for the RCC calibration approach.

5.2.5 NSVP extension

Our previous SVP model can be generalized allowing an NSVP. Du to lack of space, this will be made available in a technical report. Our tests demonstrated that such a parameterization is not that useful in practice. Indeed, the parameters can only be recovered by means of optimization that would require initial estimates of the parameters, which are difficult to obtain. The algorithms presented in the next section though, provide an efficient solution even in the NSVP case.

5.3 Homography-based Calibration

The RCC approach of the previous section relies on two steps: estimation of the pose of each camera w.r.t. each calibration plane, then estimation of the calibration asso-

ciated to each distortion circle. One drawback is that the pose estimation becomes unstable as the camera becomes near fronto parallel to the calibration plane. In this section, we consider two algorithms that are not subject to this problem. Furthermore, the second algorithm (*cf*.5.3.3) can estimate either a non-parametric or a polynomial for the focal lengths, thus avoiding dense correspondences.

5.3.1 Model parameterization

It is possible to consider the distortion circles and associated viewing cones as individual *perspective cameras*, with different focal lengths but identical principal points [151, 152]. In the SVP case, extrinsic parameters of all these cameras are identical, whereas in the NSVP case, they all share the same orientation and the optical centers are merely displaced along the optical axis.

Let us consider the distortion circle of radius d and one image of a calibration plane. From point correspondences between pixels on this circle and points on the calibration plane (on the calibration conic), we can compute a plane-to-image homography H_d. For simplicity, let us assume that image coordinates have been translated to put the distortion center at the origin. The homography can then be decomposed such that:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \propto \mathsf{H}_d \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathsf{K}_d \mathsf{R} \begin{bmatrix} 1 & 0 & \\ 0 & 1 & -\mathbf{t} + t_d \mathbf{r}_3 \\ 0 & 0 & \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{5.18}$$

where (x, y) is a calibration point, (u, v) a pixel on the distortion circle, and R and $\mathbf{t}$, a rotation matrix and translation vector representing camera pose (same for all d). The scalar t_d allows to model translational displacement of individual viewing cones along the optical axis (given by $\mathbf{r}_3^\mathsf{T}$, the third row of R), which is needed for NSVP cameras. For SVP cameras, we set $t_d = 0$ for all d.

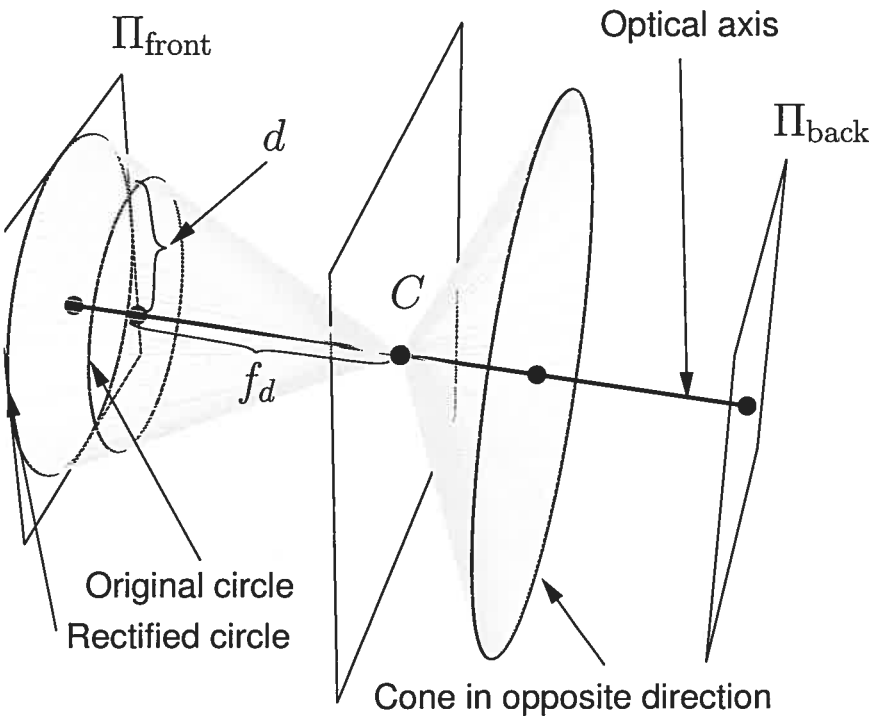


Figure 5.5. Viewing cones can also be seen as individual perspective cameras with different focal length. A rectified image can be obtained by projecting the distortion circles (which lie in different planes) on one plane Π_{front} (or Π_{back} for a field of view larger than $180°$).

As for K_d, it is a calibration matrix[2] $\text{diag}(f_d, f_d, 1)$, where f_d is the focal length associated with the considered distortion circle. We may interpret the relation between d and f_d as a distortion function applied to a perspective camera whose undistorted image plane is π_{front} (*cf*.fig. 5.5).

Note that this parameterization only accounts for viewing cones with field of view smaller than 180°. Larger fields of view can be modeled by adding a reflection to the

[2] As mentioned in the introduction, this model does not include a skew between pixel axes or an aspect ratio different from 1. Also, it assumes that the distortion center is at the principal point. Generalizing this would be straightforward though.

rotational component of the pose, $\mathsf{R}' = \mathrm{diag}(1, 1, -1)\mathsf{R}$, and a corresponding image plane π_{back}. Equivalently, one may describe these cones as cameras with negative focal length as presented in [152].

5.3.2 Unconstrained approach

In [151], an approach based on recovering the *Image of the Absolute Conic*, associated to each distortion circle, from one or many images of a calibration plane, is introduced. Once the internal parameters are recovered, the camera external parameters are estimated using the approach presented in [142, 143, 181]. The weakness of this approach is that since the calibration is individually performed on each distortion circle, it does not enforce SVP or NSVP constraints. However, it is useful for estimating the distortion center as described in §5.4. This algorithm will be referred to "`IAC`".

5.3.3 Constrained approach

We present another homography-based approach that can enforce the constraints directly. It is related to Hartley-Kang's approach, but it handles omnidirectional cameras as well as the NSVP extension.

SVP case, one camera. In the SVP case, t_d is zero for all d. It follows that:

$$\mathsf{K}_d^{-1}\mathbf{q} \propto \begin{bmatrix} 1 & & \\ & 1 & \\ & & f_d \end{bmatrix} \mathbf{q} \propto \mathsf{R} \underbrace{\begin{bmatrix} 1 & 0 & \\ 0 & 1 & -\mathbf{t} \\ 0 & 0 & \end{bmatrix}}_{\mathsf{M}} \mathbf{Q} \tag{5.19}$$

holds up to scale. If we divide the first by the second coordinates of both sides, we

obtain an equation which is independent from the distortion circle d:

$$\frac{q_1}{q_2} = \frac{(\mathsf{MQ})_1}{(\mathsf{MQ})_2}.$$

We finally obtain a linear equation on the six coefficients in the first two rows of the pose matrix M:

$$\begin{aligned} & q_1 \left(M_{21}Q_1 + M_{22}Q_2 + M_{23}Q_3\right) - \\ & q_2 \left(M_{11}Q_1 + M_{12}Q_2 + M_{13}Q_3\right) \;=\; 0. \end{aligned} \tag{5.20}$$

The above equation being linear homogeneous, we may use all point correspondences between the calibration plane and the image to estimate the first two rows of M, up to scale. The pose, *i.e.* R and $\mathbf{t}$, can be partially estimated from M. There exists a scalar λ such that:

$$\begin{bmatrix} R_{11} & R_{12} & -\mathbf{r}_1^\mathsf{T}\mathbf{t} \\ R_{21} & R_{22} & -\mathbf{r}_2^\mathsf{T}\mathbf{t} \end{bmatrix} = \lambda \bar{\mathsf{M}}$$

where $\mathbf{r}_i^\mathsf{T}$ is the i^{th} row of R and $\bar{\mathsf{M}}$ the upper 2×3 part of M. It leads to the following observations:

- the rotation matrix R can be estimated up to 4 solutions, from the left 2×2 part of $\bar{\mathsf{M}}$. They differ by a choice of sign for columns and rows of the solution: if R is one valid solution, then the other three are given by:

  $$\mathsf{DR} \quad \mathsf{RD} \quad \mathsf{DRD}$$

  with $\mathsf{D} = \mathrm{diag}(-1, -1, 1)$. Distinguishing the valid rotation matrix will be done later when estimating the full camera position and internal parameters;

- the intersection $\mathbf{t}_0$ between the optical axis of the camera and the calibration plane can be recovered as the nullspace of $\bar{\mathsf{M}}$, since it is the point on the plane

projecting onto the origin. Hence, the translation vector $\mathbf{t}$ can be estimated up to one degree of freedom, a displacement μ along the optical axis:

$$\mathbf{t} = \mathbf{t}_0 + \mu \mathbf{r}_3.$$

Estimating μ can be done simultaneously with estimating the focal lengths of all distortion circles. By inserting the solution for $\mathbf{t}$ into (5.19), we obtain:

$$\mathbf{q} \propto \begin{bmatrix} f_d & & \\ & f_d & \\ & & 1 \end{bmatrix} \mathsf{R} \begin{bmatrix} 1 & 0 & \\ 0 & 1 & -\mathbf{t}_0 - \mu \mathbf{r}_3 \\ 0 & 0 & \end{bmatrix} \mathbf{Q}$$

Due to the orthonormality of R, we get:

$$\mathbf{q} \propto \begin{bmatrix} f_d & & \\ & f_d & \\ & & 1 \end{bmatrix} \left(\mathsf{R} \underbrace{\begin{bmatrix} 1 & 0 & \\ 0 & 1 & -\mathbf{t}_0 \\ 0 & 0 & \end{bmatrix}}_{\mathsf{T}_0} - \begin{bmatrix} 0 & & \\ & 0 & \\ & & \mu \end{bmatrix} \right) \mathbf{Q}$$

Let us denote:

$$\mathbf{S} = \mathsf{R}\mathsf{T}_0\mathbf{Q}$$

which is a known vector. We thus can write the above equation as:

$$\mathbf{q} \propto \begin{bmatrix} f_d S_1 \\ f_d S_2 \\ S_3 - \mu Q_3 \end{bmatrix} = \begin{bmatrix} S_1 & & \\ & S_2 & \\ & & Q_3 \end{bmatrix} \begin{pmatrix} f_d \\ f_d \\ S_3/Q_3 - \mu \end{pmatrix}$$

and obtain the following set of linear equations on the unknowns f_d and μ:

$$[\mathbf{q}]_\times \begin{bmatrix} S_1 & & \\ & S_2 & \\ & & -Q_3 \end{bmatrix} \begin{pmatrix} f_d \\ f_d \\ \mu - S_3/Q_3 \end{pmatrix} = \mathbf{0}$$

or:

$$[\mathbf{q}]_\times \begin{bmatrix} S_1 & \\ S_2 & \\ & Q_3 \end{bmatrix} \begin{pmatrix} f_d \\ \mu \end{pmatrix} = [\mathbf{q}]_\times \begin{pmatrix} 0 \\ 0 \\ -S_3 \end{pmatrix}. \tag{5.21}$$

Note that the equation system is non-homogeneous, *i.e.* the solution for the f_d and μ is computed exactly, not only up to scale. All point correspondences, from all distortion circles, can be used simultaneously: each correspondence contributes to estimating μ and to the focal length of the distortion circle it belongs to. The third equation in (5.21) is of the form $\begin{pmatrix} * & 0 \end{pmatrix} \begin{pmatrix} f_d & \mu \end{pmatrix}^\top = 0$, where $*$ is zero in the noise free case. With noise, we found this equation to bias f_d towards small values and thus do not use this equation. Using the other equations, we have overall a linear system of size $2n \times (D+1)$, where n is the number of point correspondences and D the number of distortion circles.

This assumed a known rotation. All rotation matrices among the four possibilities shown above, give the same solution for f_d and μ, up to different signs. Since all values should be positive, the correct rotation is found easily.

Using many camera positions. The above equations can be used in a slightly modified form to simultaneously use many planes positions. Let v be the index for each view out of V. Each associated 2×3 partial homography M_v can be computed using (5.20). Then $\mathbf{t}_{v0}$ and R_v can be estimated individually from each of them. Finally, (5.21) is extended naturally to account for many views by simultaneously solving for all displacements μ_v and focal lengths f_d. The resulting equation system is of size $2n \times (D+V)$.

NSVP case, many views. In the NSVP case, the previous algorithm can be applied nearly without modification. Let us consider the form of the plane-to-image homography for an individual distortion circle, given in equation (5.18), for the NSVP case. In comparison to the SVP case (equation (5.19)), there is an additional term ν_d for the position of the displaced optical center on the optical axis:

$$\begin{aligned} \mathbf{q} \;\propto\; \mathsf{H}_{vd}\mathbf{Q} &= \mathsf{K}_d\,\mathsf{R}_v \begin{bmatrix} 1 & 0 & \\ 0 & 1 & -(\mathbf{t}_{v0} + (\mu_v + \nu_d)\mathbf{r}_{v3}) \\ 0 & 0 & \end{bmatrix} \mathbf{Q} \\ &= \mathsf{K}_d \left(\mathsf{R}_v\mathsf{T}_{v0} - \begin{bmatrix} 0 & & \\ & 0 & \\ & & \mu_v + \nu_d \end{bmatrix} \right) \mathbf{Q}. \end{aligned} \tag{5.22}$$

Note that ν_d is the same for all views. Since K_d is a diagonal matrix, the first two coordinates of the right hand side do not depend on μ_v and ν_d. Hence, like in the SVP case, (5.20) can be used to estimate linearly the first two rows of the pose matrix M_v, using all point correspondences, for all distortion circles. The pose parameters R_v and $\mathbf{t}_v$ can also be extracted in the same manner, with $\mathbf{t}_v$ being obtained up to a displacement along the optical axis $\mathbf{r}_{v3}$.

We now consider how to estimate the focal lengths f_d and the optical center positions $\mu_v + \nu_d$. The equations are identical to those in the SVP case, with the difference that μ_v has to be replaced by $\mu_v + \nu_d$, *i.e.* is not the same for all distortion circles. The set of μ_v and ν_d is an overparameterization, since subtracting a value from all μ_v and adding it to all ν_d leaves (5.22) unchanged. Hence, we may set one of them to any fixed value. The equation system is thus of size $2n \times (2D - 1 + V)$.

Polynomial model. Calibrating a general model requires many data points to obtain precise and accurate results, i.e. to avoid over-fitting. Otherwise, relying on a more restricted model may give better results. In [152], a polynomial model was used

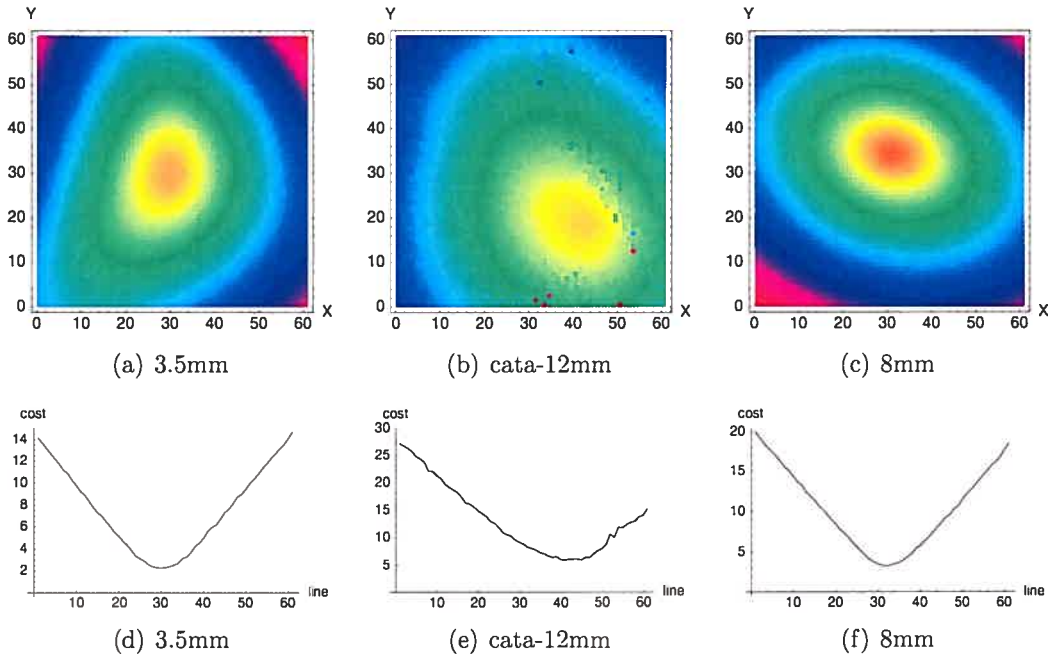
to represent the relationship between focal length and radius d. Our formulation can also be trivially modified to use polynomials for f_d as well as ν_d. For NSVP cameras, it turns out that in practice, estimating the fully general model can be unstable because of the focal length–displacement ambiguity mentioned in §5.2.2. The result is that both f_d and ν_d are not smooth functions for small radius values d. Instead of relying on smoothing terms (whose weight are difficult to set) to circumvents this, we prefer using polynomials.

The above method is referred to as "HB" in the rest of the paper.

5.4 Computing the Distortion Center

Until now, we have assumed, for both algorithms, that the distortion center was known; this information was used to select the distortion circles. Remind that the distortion center is also the principal point of the camera in our model. Tests with noiseless simulated data showed that the calibration may be quite sensitive to a bad choice of distortion center; indeed, as for real cameras, using the image center as an approximation was not satisfying in general. Hence, the distortion center should be estimated as part of the calibration process. Note that using Hartley-Kang's algorithm is not satisfactory in general (see discussion below and §5.9). The sensitivity of calibration we observed in simulations suggests that it should be possible to estimate the distortion center rather reliably, which was confirmed in practice.

Algorithm. We used the following heuristics to define an optimization criterion for the distortion center. Let us apply the IAC approach of §5.3.2 with several images as input. The plane-based calibration for each distortion circle is then capable of estimating a principal point, besides the focal length f_d. It seems plausible that the better the assumed distortion center was, the closer the estimated principal points will be to it. Since plane-based calibration is applied on images centered on the assumed distortion center, we can consider the average distance of the estimated principal



(a) 3.5mm (b) cata-12mm (c) 8mm

(d) 3.5mm (e) cata-12mm (f) 8mm

Figure 5.6. Plots of the goodness measure for the distortion center, obtained for three tested lens (*cf.*§5.6). (a,c,e) 60×60 grid around the image center (yellow meaning smaller). (b,d,f) One slice per plot, through the respective minimum.

points (one per distortion circle) as a measure for the goodness of the center.

Figure 5.6 shows the values of this measure, computed for distortion center positions on a 60×60 grid around the image center, for real cameras. The shape of the cost surface indicates that we can find the optimum distortion center using a simple steepest descent type method. We implemented such an approach that accurately finds the distortion center within a couple of minutes of computation. Note that the second column of figure 5.6 shows that, although the principal points used to plot it were computed individually per distortion circle, they are very densely clustered (average distance to assumed distortion center of less than 3 pixels). This suggests a high stability of the calibration.

Discussion. Compared to other approaches, our optimization criterion is chosen to

find the best optical axis. Thus, the distortion center is very apparent to a principal point in a pinhole camera. This formulation is also used in [45, 156, 157]. Our criterion is different from the one in image-based distortion functions, where the distortion center (together with the distortion function) is chosen to maximize the linearity of rectified line images [36, 61, 152, 181]. However, the latter is not optimal when the camera is NSVP since image rectification is not possible. For this reason, the distortion center computation will not be included in the comparison with Hartley-Kang's approach.

5.5 *Practical Issues*

5.5.1 Dense Camera–Plane Correspondences

The easiest approach we found to get dense correspondences between the calibration plane and the camera is to use a flat screen. We used a simple coded structured light algorithm [122], which consists in displaying a sequence of patterns of horizontal and vertical black and white stripes of varying thickness on the screen to encode the position of each screen pixel (*cf*.fig. 5.7). Then, for each camera pixel, we identify the corresponding position on the calibration plane by decoding the observed intensities in each pattern. When performed in a controlled environment (low-constant ambient lighting, screen of high contrast and resolution), the accuracy of such a method is reasonably good (around ± 2 pixel of error on average). Since the points located on the distortion circles are given in floating point coordinates, we compute their correspondences by a weighted sum of the correspondences recovered for the four closest image pixels. Besides allowing dense correspondences with the calibration plane, these approaches render trivial the problem of recovering the structure of the calibration plane. This is as opposed to using calibration grid images where grid points must be automatically extracted and identified. This is especially difficult when the distortion is very large.

(a) (b) (c)

Figure 5.7. Projected patterns for correspondences are horizontal and vertical black and white stripes. Images taken with (a) the Goyo 3.5mm, (b) catadioptric, and (c) paracatadioptric camera (cata-12mm).

5.5.2 Omnidirectional Cameras

There are several issues worth mentioning for omnidirectional cameras. If the field of view is larger than 180°, then some distortion circles will have viewing cones that actually approach planes. For the RCC approach, this means that fitting the calibration conic may become unstable. These cases can be detected as the ones whose correspondences on the calibration plane are close to collinear. They can then be discarded from the actual calibration procedure and we may attribute them an unknown focal length. However, if one is able to find the radius for which the correspondences are collinear, the radius d where ϕ_d equals 0 is known. This is a very strong constraint on the system that should be enforced.

In the case of the homography-based algorithm that uses all matches simultaneously, no special attention is needed. The only difference concerns the determination of the correct rotation matrix among the four possibilities. Instead of making sure that all f_d and μ are positive, we only require f_{d_0} and μ to be positive; other focal lengths f_d may indeed be negative.

5.5.3 Non-linear optimization

Because f_d is a function of the radius in the distorted image, it is not straightforward to perform the projection of a 3D point into the image (as opposed to the backprojection of image points to 3D). This means that (5.18) can not be used directly to perform a non-linear optimization of the calibration unknowns. Since the focal length function can not be inverted when it crosses zero, it is preferable to define the distortion in terms of view angle θ_d. As seen in figure 5.13(b), this function is generally simple, so easily invertible. Given this function, the radius of the image of a 3D point is computed from the angle between the optical axis and the line spanned by the optical center and the 3D point. The non-linear optimization is then performed with θ_d instead of f_d.

The NSVP case can not be handled as simply, since there is no single optical center relative to which to compute the angle θ. In this case, we use (5.18) and estimate a radius d_i associated to each 3D–2D correspondence, along with the other parameters. This yields a sparse non-linear problem.

We may also enforce monotonicity on f_d and θ_d. For example, this can be done approximatively by adding terms to the cost function, of the form:

$$\left(|f_d - f_{d+s}| - (f_d - f_{d+s})\right)^2, \; s > 0$$

which is a quadratic penalty if the constraints are not enforced, but gives 0 otherwise.

5.6 Experiments

We tested our approaches using different types of cameras with simulated and real data. They were also compared to Hartley-Kang's algorithm [61] (referred to as "HK"). Since the model they use is not identical to ours, the comparison is limited to certain aspects. These differences are the topic of Appendix 5.9.

5.6.1 Simulation

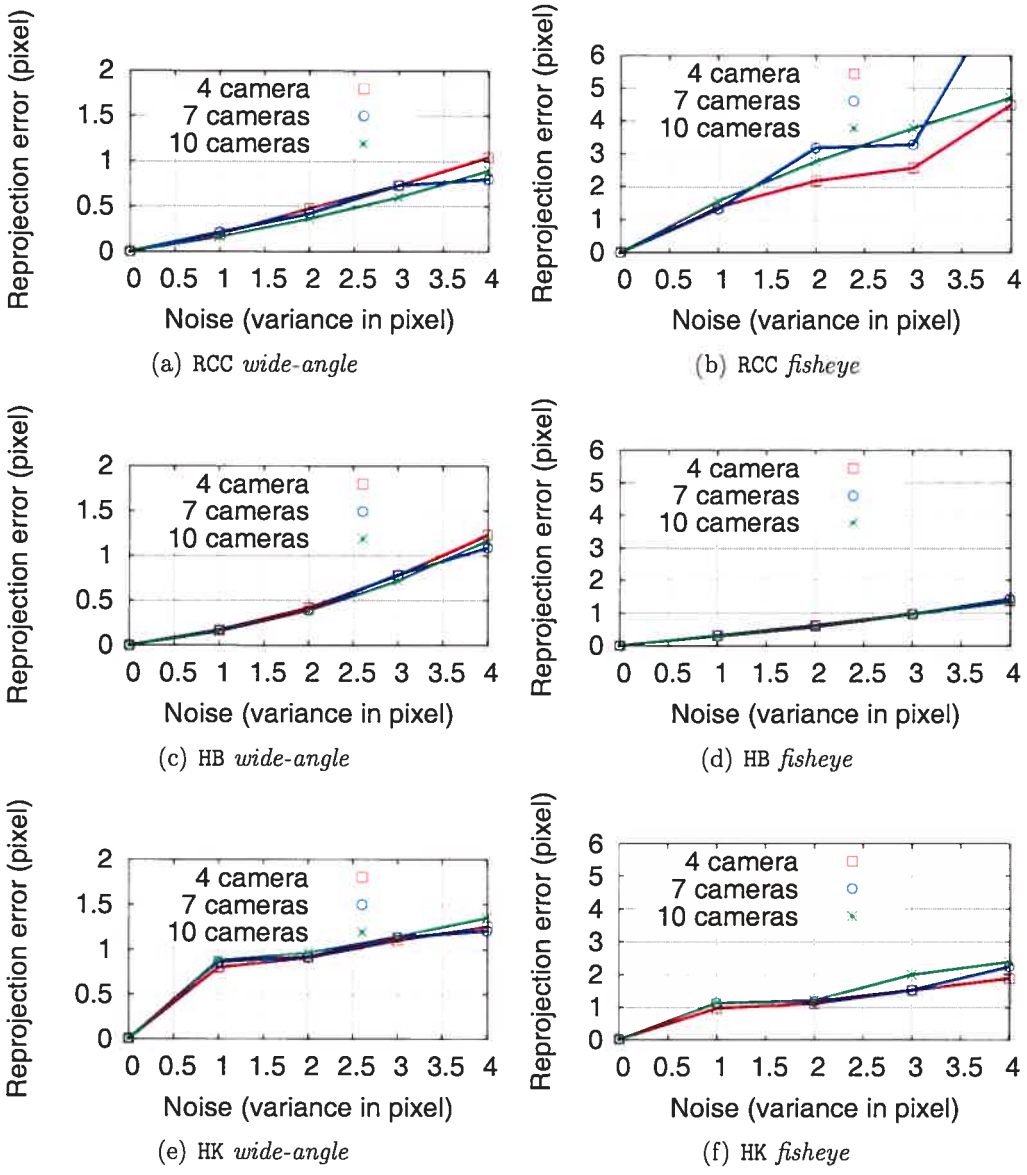
SVP cameras. We simulated two types of cameras: wide-angle with small radial distortion and fisheyes with large distortion[3]. The focal length (and distortion) function of each type of cameras was built randomly via monotonically decreasing polynomials of fifth degree. We used an image size comparable to our own camera: 1 Mega pixel (see real data). In terms of focal function, the wide-angle cameras were apparent to our 15mm (fig. 5.13) and we used f_d with $f_0 = 1000 \pm 200$ pixels. The simulated fisheyes were analogous to our 3.5mm, so we used $f_0 = 400 \pm 100$ pixels. In these tests, we assumed a known distortion center and also made sure the camera was never placed in a near fronto-parallel position w.r.t. the calibration plane.

We compared the reprojection error, the error on the pose and on the calibration. We define the latter as the average difference between the recovered focal function f_d and the ground truth. Our tests showed that all of them are highly related. The reprojection errors for the three algorithms w.r.t. noise and the number of used cameras are shown in figure 5.8. We added Gaussian noise of standard deviation up to 4 pixels to the original data, which consisted of 50 points per distortion circle (which is rather small compared to the several hundred usually available from a structured light dense mapping).

In general, all three algorithms performed similarly. The RCC did not perform as well as the other ones for the fisheye camera because the data points were not uniformly distributed around the distortion circles. Hence, the pose estimation was not as stable. Otherwise, like with the wide-angle camera, the results are comparable to the other algorithms.

NSVP cameras. We performed an in-depth analysis of the performance of our algorithms for NSVP cameras. Three aspects were considered. First, how well the displacement along the optical axis can be recovered under noise. Second, is the

[3] We did not compare catadioptric cameras with the HK approach because the comparison would have been unfair.



(a) RCC *wide-angle* (b) RCC *fisheye*

(c) HB *wide-angle* (d) HB *fisheye*

(e) HK *wide-angle* (f) HK *fisheye*

Figure 5.8. Reprojection errors for the simulated *wide-angle* and *fisheye* cameras.

linear NSVP algorithm useful in practice? Third, does the NSVP model overfit when the camera is actually SVP? We performed our tests using the homography based approach because it is the only one that naturally enforces both SVP and NSVP constraints. Two approaches were tested:

1) Initialization using linear calibration based on an SVP assumption, followed by non-linear optimization of the NSVP model.

2) Initialization as well as optimization using the NSVP model.

The tests were performed on simulated SVP and NSVP catadioptric cameras with viewpoints moving along the optical axis (fig. 5.9(a,d)). We used 10 different positions for the plane and added Gaussian noise of standard deviation 1 pixel to the data. Three hundred points per image were used and polynomial models for the calibration were used. A typical behavior of the two approaches is shown in figure 5.9. They lead to the following observations:

- For an SVP camera, both optimizations should lead to similar results (negligible NSVP);

- For an NSVP camera, enforcing an SVP yields a biased focal length function (*cf*.fig. 5.9(f)). In some cases, this can be satisfying in terms of reprojection error, like for one our real cameras (*cf*.5.12);

- When the model is refined to include t_d, the optimization might not converge to a satisfying minimum;

- If the camera is NSVP, the second approach should perform better, but only if the noise is sufficiently low (*cf*.fig. 5.9(d)). Otherwise it can result into worse result because solving with (5.22) is not as stable as with (5.19).

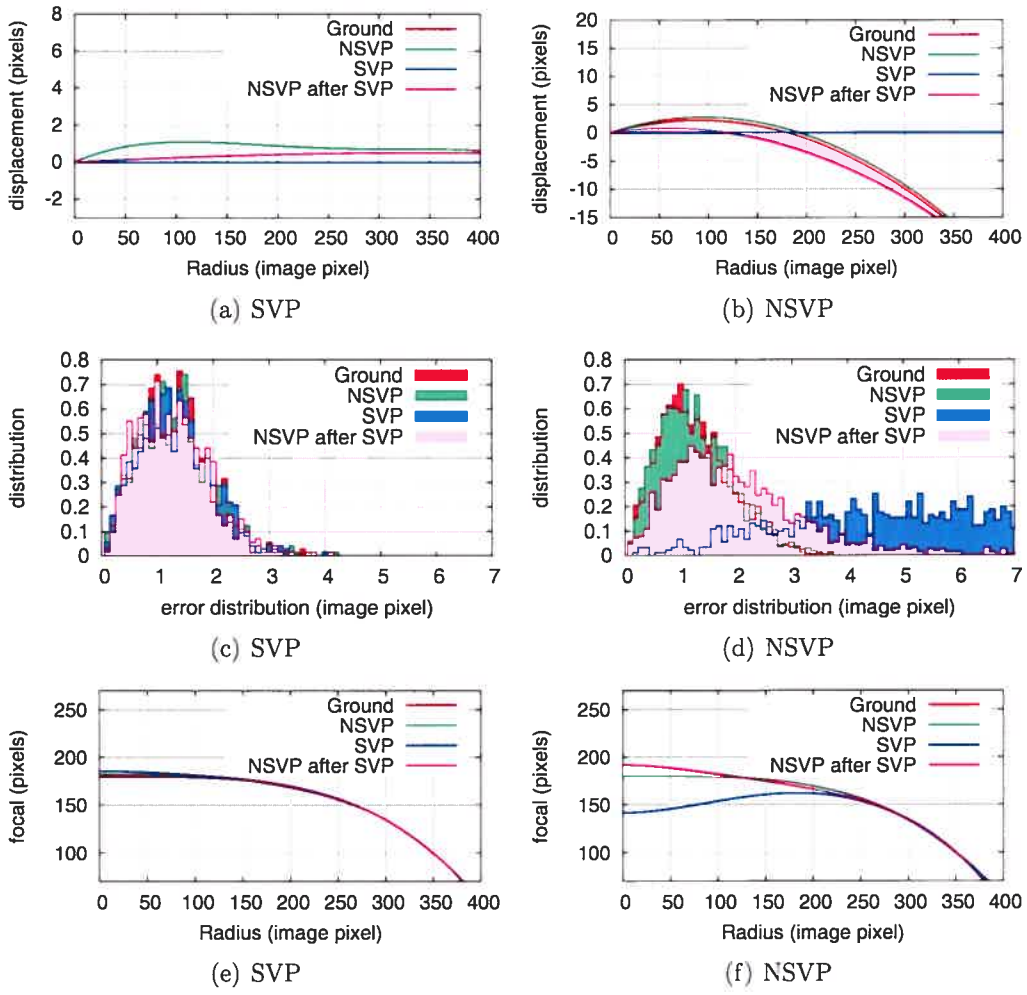


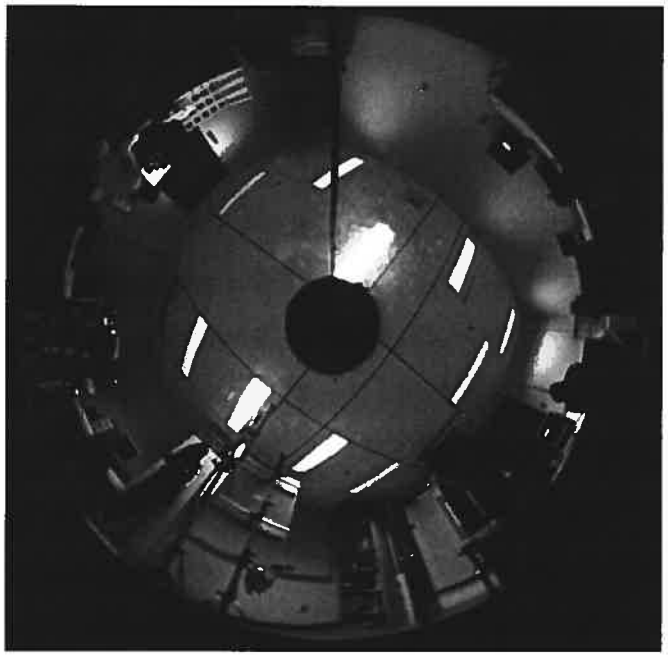
Figure 5.9. The two optimization algorithms (see text for details) with simulated data. (a)+(b) The recovered displacement t_d. (c)+(d) The error distribution. (e)+(f) The focal length functions compared to the ground truth.

5.6.2 Real data

Several camera configurations were tested. First, a CCTV Basler A201bc was combined with a fisheye Goyo 3.5mm lens, to an 8mm Cosmicar lens with small distortion and to a RemoteReality catadioptric lens combined with a 12.5mm Cosmicar lens (referred to as "cata-12mm"). Secondly, a Canon SLR was combined with a fisheye 15mm lens and to a 0-360 catadioptric lens combined with a 30mm lens (referred to as "cata-30mm"). In all cases, the calibration plane of known Euclidean structure was a 20 inch LCD screen. The number of calibration views was between 8 and 10 for the different experiments.

Figure 5.13(a) gives the computed focal length of the 15mm, 3.5mm, cata-12mm and cata-30mm, w.r.t. the distance d to the distortion center, using all methods. These are the functions that were recovered without further optimization based on the reprojection error. Table 5.2 shows the average reprojection errors of the three algorithms for all cameras. In many cases, the cameras could be calibrated from a single image of the screen (*cf*.fig. 5.11 for the RCC), although in general, we recommend using at least five images for good stability. Remind that our structured light based matching provides a large number of correspondences. The catadioptric cameras were calibrated with all the approaches (except for Hartley-Kang's approach where only the portion of the image corresponding to forward cones was used) with very similar results (*cf*.fig. 5.13). The RCC gave very accurate results, however only with a limited number of planes. The difficulties arose when not enough data were available to allow a good fitting of the calibration conics. We handled this by dropping these planes and use only the other ones.

If discrete values for f_d are computed, instead of e.g. a polynomial function, then only a subset of distortion circles are used for calibration; others can then be extrapolated or interpolated from a polynomial fitting of the data. Let us define this polynomial p; from the camera model, it is best to ensure that its derivative at 0



(a)

(b)

Figure 5.10. Image rectification for the Basler camera with the 12.5mm lens and the RemoteReality lens (cata-12mm). (a) Original image. (b) Rectified image for a rotated view.

Table 5.2. Comparison of the average reprojection errors. NL refers to non-linear optimization. For the catadioptric cameras, we only used distortion circles corresponding to forward looking viewing cones, to be able to compare with Hartley-Kang's approach.

	Algorithms					
Cameras	RCC	RCC+NL	HB	HB+NL	HK	HK+NL
cata-12mm	4.76	1.33	1.31	0.97	5.28	0.89
cata-30mm	—	—	1.42	1.33	—	—
3.5mm	6.89	1.15	1.46	1.06	2.53	1.16
8mm	10.74	3.01	6.64	3.08	10.92	3.10
Canon 15mm	0.56	0.48	0.51	0.48	0.51	0.48

(corresponding to the distortion center) is 0. This constraint is due to the symmetry of the distortion model. Another criterion is that the function should be monotonically decreasing. This last constraint is not directly enforced in our algorithms. However, this didn't seem to be an issue in our tests. In practice, polynomials of degree 5 appeared to be sufficient. To handle the case of omnidirectional cameras more appropriately, the interpolation is carried out with the view angle instead of the focal length. In this case, a monotonically increasing polynomial passing through 0 can also be fitted (see fig. 5.13(b)).

Both catadioptric cameras cata-12mm and cata-30mm are typical examples of configurations yielding multiple viewpoints. Indeed, both mirrors are parabolic and the mounted lenses are perspective [146]. However, only the second one was found to be NSVP (*cf*.fig. 5.12 and table 5.3). We conjecture that although our 12.5mm camera is not orthographic, it has a field of view sufficiently small to provide a locus of viewpoints very close to a single effective viewpoint. To verify our hypothesis, it would be useful to perform the test with more specialized equipment like in [146].

Evaluating the results based on the reprojection error can lead to biased conclusions in the case of a generic model. Indeed, the model offers more freedom which

Table 5.3. Comparison of the average reprojection error for different constraint of the viewpoint. 'L' refers to the linear algorithms and 'NL' to non-linear optimization of the parameters.

	Algorithms-Constraints				
Cameras	SVP-L	SVP-NL	NSVP after SVP-NL	NSVP-L	NSVP-NL
cata-12mm	1.65	1.08	1.06	1.58	1.12
cata-30mm	2.22	1.93	1.73	1.42	1.33

Table 5.4. Result for pose estimation. The camera was moved to three positions with known relative motion. Coefficients p_{ij} and a_{ij} denote the distance (in centimeters) and relative angle (in degrees) between camera positions i and j.

	Position			Angle		
Algorithms	p_{01}	p_{12}	p_{02}	a_{01}	a_{12}	a_{02}
Ground truth	5	5	10	0°	0°	0°
RCC	4.94	5.00	9.93	0.83°	0.1°	0.92°
HB	4.90	4.94	9.85	0.79°	0.79°	1.6°
HK	4.90	4.96	9.86	0.84°	0.68°	1.51°

allows to fit the data better. Meaningful quantitative results were obtained for the Goyo 3.5mm lens, using a pose estimation procedure. Using a translation stage, the camera was moved to three positions with known relative motion (no rotation, known translation). Using the calibration information (obtained using other images), the pose of the camera relative to the calibration plane was computed for all three positions. From this, the relative motions were computed and compared to the ground truth. The results presented in table 5.4 show a good stability for all methods.

Images from three panoramic cameras were rectified based on the calibration results (*cf*.fig. 5.15 and 5.10(a,b)). For the wide-angle Goyo lens and the cata-12mm, the results seem to be very good, even towards the image borders (*cf*.the

inset images in fig. 5.15(b) and 5.10(b)).

Finally, a home-made catadioptric device built from a Fujinon 12.5mm lens pointed at a roughly spherical mirror was tested (*cf*.fig. 5.14). Although its radial configuration was not perfect, the distortion center could be found and a satisfying calibration could be obtained with our methods. The "HB" approach gave the best results because it could take advantage of up to eight images, which is more robust to the imperfect configuration of the camera. The rectification is surprisingly good for a large part of the image, especially around the borders (*cf*.fig. 5.15(a,c)). The remaining distortions in the center were found to be caused by a small bump on the "mirror's" surface.

5.7 Summary and Conclusion

We have proposed new calibration approaches for a camera model that may be a good compromise between flexibility and stability for many camera types, especially wide-angle ones. Previous work showed that the RCC approach might have a limited practical usability because of stability issues. This was because only one calibration plane could be used directly and because camera position was recovered in two steps: conic fitting and finding the closest point to a set of viewpoint conics. Both issues were addressed in this paper and we showed that the use of the RCC approach is very well suited when performed with only few camera poses. We put a lot of confidence in our homography-based approach. It can be adapted to using a polynomial distortion model and extended to NSVP configurations. This allows to perform calibration without a dense plane-to-image matching, unlike the previous approach.

Our experiments also showed that Hartley-Kang's approach gives very good results with the benefit that it has an elegant solution for the distortion center estimation. However, it cannot deal with NSVP and/or omnidirectional cameras which is one of the main goals of this work.

5.8 Appendix: Derivation of (5.10)

We want to derive a formulation for a family of conics that touch in one point and have identical tangents at this point. We proceed by construction, starting from a simplified family which we extend. This family is the one with the X-axis as major axis, going through $(1,1)$ with slope 1, given by:

$$\begin{bmatrix} \rho & 0 & 1 \\ 0 & -\rho-1 & 0 \\ 1 & 0 & -1 \end{bmatrix}, \quad \rho \geq 1/2 \tag{5.23}$$

To represent any family of conics tangent at their intersection point, with the X-axis as their major axis, we only need to operate a simple transformation to the above family, namely a change of scale in X and Y, and a shift in X:

$$\begin{bmatrix} \gamma & 0 & \epsilon \\ 0 & \delta & 0 \\ 0 & 0 & 1 \end{bmatrix}^{\top} \Psi \begin{bmatrix} \gamma & 0 & \epsilon \\ 0 & \delta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.24}$$

where γ and δ are positive.

We are not going to prove this, as we are really interested in the corresponding families of conics in the calibration plane. Converting the above yields more complicated matrices than directly applying our transformation to the basic family of calibration conics. Hence, we are going to do just that, and prove that in so doing, we only obtain families of calibration conics whose corresponding viewpoint conics are tangent at their contact point, with the X-axis as their major axis, and that we obtain all such families. The basic family of conics is the one corresponding to the

family:

$$\begin{bmatrix} \rho & 0 & 1 \\ 0 & 2\rho+1 & 0 \\ 1 & 0 & -2 \end{bmatrix}. \tag{5.25}$$

Applying the transformation of (5.24) to (5.25), we get:

$$\begin{bmatrix} \rho\gamma^2 & 0 & \gamma(\rho\epsilon+1) \\ 0 & (2\rho+1)\delta^2 & 0 \\ \gamma(\rho\epsilon+1) & 0 & \rho\epsilon^2+2\epsilon-2 \end{bmatrix}.$$

which is identical to (5.10). The corresponding family of viewpoint conics for fixed values of γ, δ, ϵ is given in (5.11). We can check directly that these conics go through the point $(X,Z) = \left(-\frac{\gamma}{\delta^2} - \frac{\epsilon}{\gamma} + \frac{2}{\gamma}, \sqrt{\frac{2\delta^2-\gamma^2}{\delta^4}}\right)$, with slope $m = \frac{\gamma}{\sqrt{2\delta^2-\gamma^2}}$.

Conversely, we would like to express the parameters in terms of m, $Z > 0$ and any X. To simplify the rest, we let $\alpha = \gamma/\delta$, so $m = \frac{\alpha}{\sqrt{2-\alpha^2}}$. Then, we have:

$$\begin{aligned} \alpha &= \frac{\sqrt{2m^2}}{\sqrt{m^2+1}} = \sqrt{2 - \frac{2}{m^2+1}} \\ \delta &= \frac{\mathrm{Z}}{\sqrt{2-\alpha^2}}, \quad \epsilon = -\alpha^2 - X\delta\alpha + 2, \end{aligned}$$

where α is always between 0 and $\sqrt{2}$. We immediately obtain the family of conics tangent at (X,Z) with slope m.

5.9 *Appendix: Hartley and Kang's approach*

The camera model used in Hartley-Kang's approach for radial distortion is not exactly the same as ours. A first important distinction is that it does not enforce a distortion center aligned with the principal point. On the other hand, the algorithm can not work for catadioptric cameras with a view angle larger than 180°. A main reason is that for those cameras, if the rectification is modeled as an image displacement, it is

not a monotonic function [152]. Indeed, the displacement tends to infinity when the view angle gets close to 180°. It then shifts to minus infinity and slowly tends to 0. This behavior breaks the assumption underlying the approach when computing the distortion function.

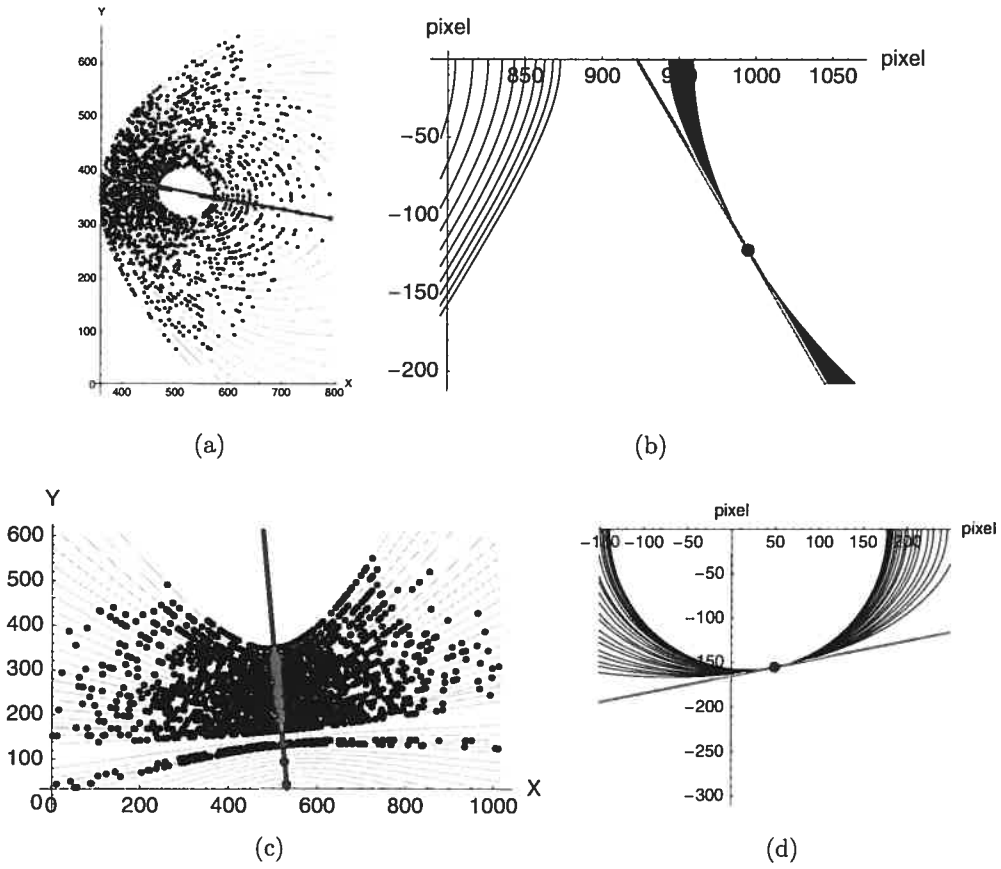
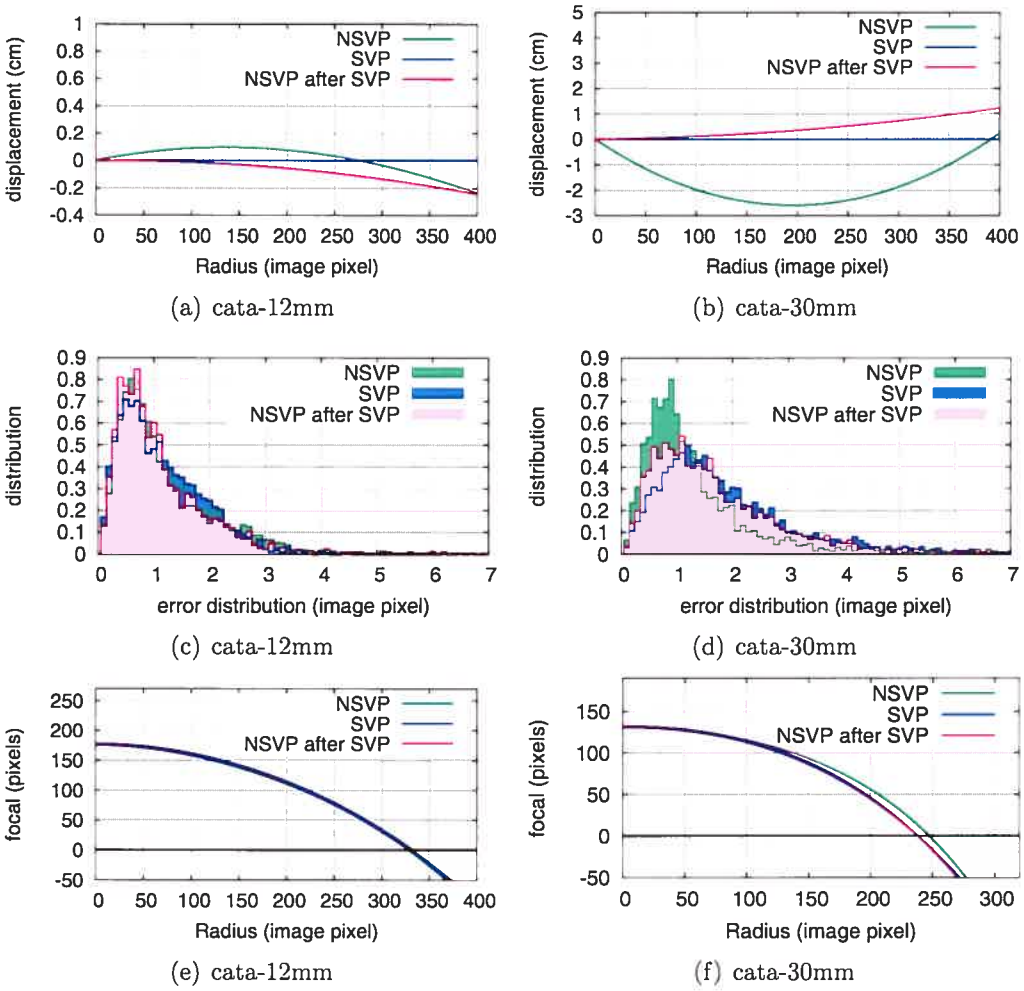


Figure 5.11. Calibration with the RCC approach. (a) **Fitted ellipses for the Goyo 3.5mm lens and** (b) **Corresponding hyperbolas, computed intersection and optical axis (gray line).** (c) **For the cata-12mm camera, the intersection between the calibration plane and the cones yielded ellipses and hyperbolas, constraining the viewpoint to lie respectively on hyperbolas and ellipses.** (d) **Intersection of the viewpoint conics.**

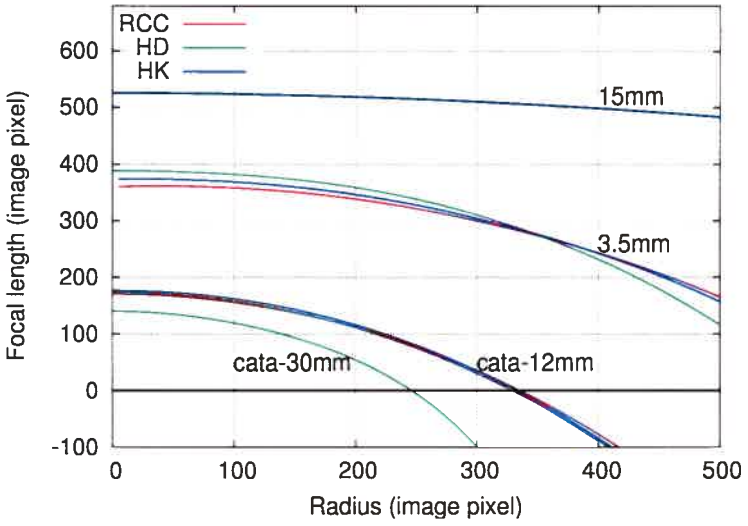


(a) cata-12mm (b) cata-30mm

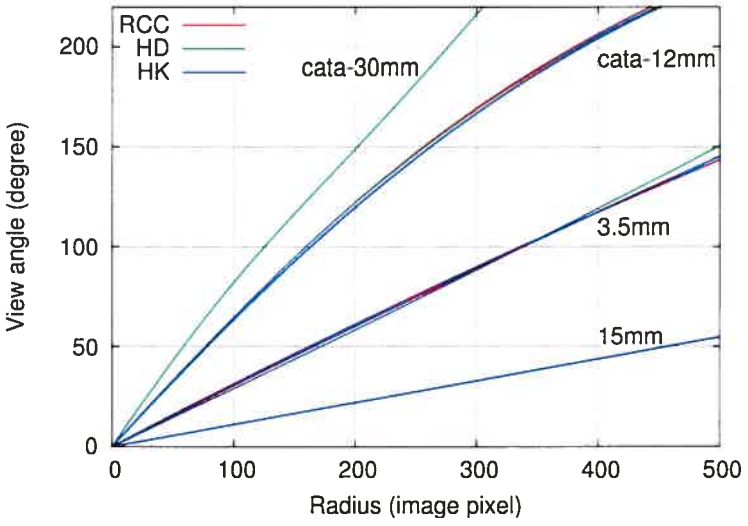
(c) cata-12mm (d) cata-30mm

(e) cata-12mm (f) cata-30mm

Figure 5.12. The two optimization algorithms (see text for details) with our catadioptric cameras. (a)+(b) **The recovered displacement** t_d, (c)+(d) **The distribution of reprojection errors.** (e)+(f) **The recovered focal length functions.**



(a)



(b)

Figure 5.13. (a) **Recovered focal length (in pixels) for the three algorithms (after polynomial fitting of the data).** (b) **Recovered view angle (in degrees). These calibration curves were obtained without optimization based on the reprojection error. Performing such an optimization lead to very similar results in general. Observe that for the catadioptric cameras there are negative focal lengths, meaning that their view angle is larger than** $180°$.

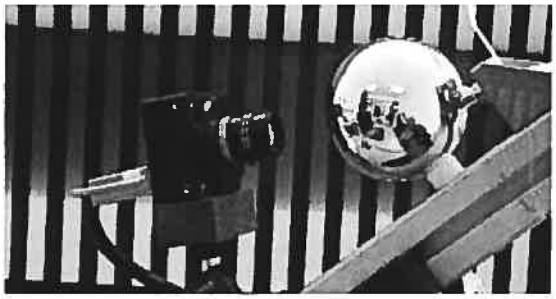


Figure 5.14. Home-made catadioptric camera built from a Basler A201bc camera with a Fujinon 12.5mm lens pointed at a Christmas ornament representing a roughly spherical mirror.

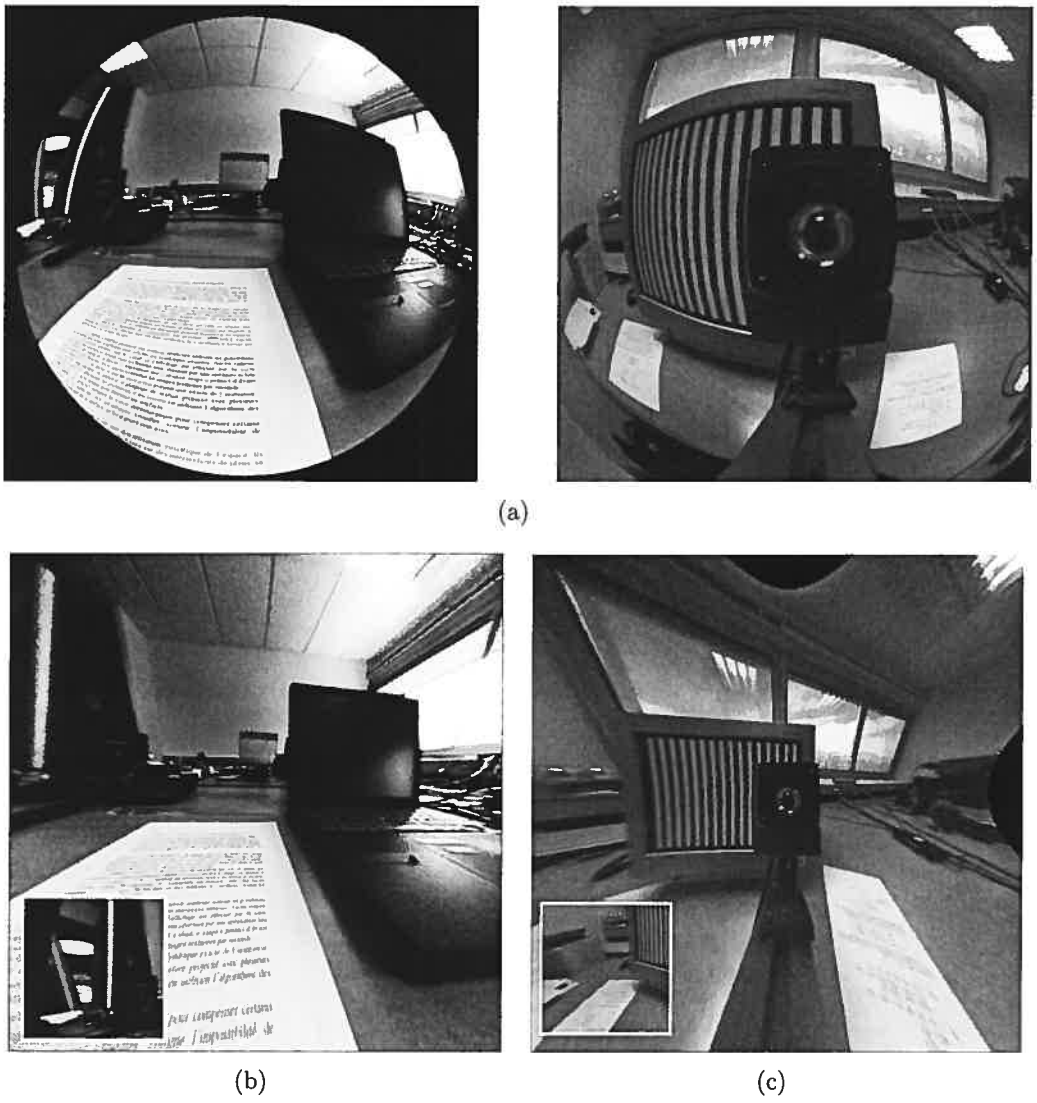


Figure 5.15. Image rectification. (a) Original images. (b) Rectified image for the Goyo 3.5mm. (c) Rectified image for the home-made catadioptric camera. Small inset images show rectification of the border regions.

Chapitre 6

CALIBRAGE ET AUTO-CALIBRAGE DE CAMÉRAS OMNIDIRECTIONNELLES À L'AIDE D'IMAGES DE LIGNES

6.1 Introduction

Dans ce chapitre, nous ne nous intéressons qu'aux caméras à projection centrale. Par conséquent, l'objectif de toute méthode de calibrage ou d'auto-calibrage est celui de corriger les images de telle sorte que la projection de toute droite 3D résulte en une droite.

À partir de ce constat, il est naturel de construire des algorithmes[1] utilisant l'image de telles courbes. L'origine de ces images de lignes permet alors de préciser le type de méthode. Si les images de lignes sont obtenues à l'aide d'un objet connu, on parle alors d'une méthode de calibrage. À l'opposé, si les courbes sont obtenues dans des images naturelles, il s'agit d'une méthode d'auto-calibrage. Idéalement, la détection de ces courbes devraient se faire de façon complètement automatique. Au chapitre 7, nous proposons une solution basée sur le suivi de points saillants ne nécessitant pas explicitement d'images de lignes.

Organisation

Nous décrivons tout d'abord quelques méthodes de calibrage ou auto-calibrage (§6.2). Ensuite, §6.3, nous donnons un aperçu de notre méthode qui est présentée au chapitre 7.

[1] *Plumbline methods*

6.2 Méthodes de calibrage ou d'auto-calibrage

Notation

Dans toutes les méthodes, la projection d'une ligne (distordue) dans une image est notée $\mathcal{L}^d$ et elle représente un ensemble des points $\mathbf{p}^d$. Nous utilisons parfois l'indice j lorsqu'il y a plusieurs lignes.

6.2.1 Méthode basée sur les serpents

Kang fut le premier à proposer une approche utilisant des images de lignes [71]. Elle utilise les *serpents*[2], *i.e.* un ensemble de pixels connectés entre eux, pour la détection de courbes dans les images. À chaque serpent est associée une fonction d'énergie qui est minimisée lorsque celui-ci est aligné avec un contour. Kang propose d'ajouter à cette fonction d'énergie une pénalité associée au modèle de distorsion radiale : que le serpent, après rectification, soit une droite. L'algorithme procède de manière itérative pour estimer des serpents dans des images et les paramètres de la fonction de distorsion.

6.2.2 Méthode pour le modèle angle de vue

Devernay et Faugeras proposent une méthode permettant d'auto-calibrer le modèle angle de vue qui peut être combiné à un modèle polynômial (*cf.*§4.3.5) [36]. Elle procède par optimisation non-linéaire d'un critère de linéarité de points rectifiés provenant de l'image d'une ligne. Cependant, leur approche ne nécessite pas un paramétrage explicite de la ligne rectifiée, *i.e.* seuls les paramètres de la fonction de distorsion sont inconnus, simplifiant du coup l'initialisation de l'algorithme. Pour les points rectifiés $\tilde{\mathbf{p}}_i^u = (x_i, y_i)^\mathsf{T}$ provenant de l'image de la même ligne 3D, nous avons

[2] *Snakes*

que

$$\psi^2 = a\sin^2\phi - 2|b||\sin\phi|\cos\phi + c\cos^2\phi$$

où,

$$a = \sum_i x_i^2 - \frac{1}{n}\left(\sum_i x_i\right)^2, \quad b = \sum_i x_i y_i - \frac{1}{n}\sum_i x_i \sum_i y_i, \quad c = \sum_i y_i^2 - \frac{1}{n}\left(\sum_i y_i\right)^2,$$

$$\alpha = a - c, \quad \beta = \frac{\alpha}{2\sqrt{\alpha^2 + 4b^2}}, \quad |sin\phi| = \sqrt{1/2 - \beta}, \quad \cos\phi = \sqrt{1/2 - \beta}.$$

est égale à zéro si tous les points sont colinéaires. Bien sûr, ce sont les paramètres de distorsion permettant de rectifier les points $\tilde{\mathbf{p}}_i^d$ qui sont recherchés. Il suffit donc de remplacer $(x_i, y_i)^\mathsf{T}$ par $\mathrm{L}^u(\tilde{\mathbf{p}}_i^d)$ dans ψ^2 et de minimiser la somme pour toutes les lignes :

$$\arg\min_{\mathbf{K}} \sum_j \psi_j^2$$

où, j est l'indice associé à différentes lignes et $\mathbf{K}$ est le vecteur des paramètres de la fonction L^u. Une heuristique de détection de courbes dans les images, combinées à un méthode itérative robuste permet le calcul des paramètres de distorsion.

Cette approche s'applique à n'importe quel modèle de distorsion du moment que les courbes sont rectifiées dans l'image. Pour cette raison elle ne fonctionne pas pour les caméras omnidirectionnelles. Nous avons testé cette méthode aux chapitre 7.

6.2.3 Méthode optimale par les distorsions radiale et tangentielle

Swaminathan et Nayar proposent d'utiliser un critère d'erreur géométrique dans l'image original, plutôt que dans l'image rectifiée [147]. En principe, la rectification d'un point $\mathbf{p}_i^d$ devrait se retrouver sur une ligne, notée $\mathbf{l}^u$, *i.e.* $\mathbf{p}_i^u = \mathrm{L}_u(\mathbf{p}_i^d)$ et $\mathbf{l}^{u\mathsf{T}}\mathbf{p}_i^u = 0$. À cause de l'erreur sur la détection de la courbe, une approche classique consiste à minimiser la distance au carré du point à la ligne : $d^2(\mathbf{l}^u, \mathbf{p}_i^d)$. En pratique, il est préférable de minimiser un critère dans l'image de départ, là où l'erreur de détection

des courbes est produites. Pour ce faire, chaque point $\mathbf{p}_i^d$ est associé à un point $\tilde{\mathbf{p}}_i^e$ situé sur la ligne l^u. On cherche ce point et cette ligne de telle sorte que le point distordu $\mathrm{L}_d(\tilde{\mathbf{p}}_i^e)$ est le plus proche possible de $\tilde{\mathbf{p}}_i^d$. Pour l'ensemble des points sur $\mathcal{L}^d$, ceci revient à minimiser :

$$\arg \min_{\mathbf{K},\mathrm{l}^u,\mathbf{p}_i^e} \sum_i^n \|\tilde{\mathbf{p}}_i^d - \mathrm{L}_d(\tilde{\mathbf{p}}_i^e)\|, \quad \text{tel que} \quad \mathrm{l}^{u\mathsf{T}}\mathbf{p}_i^e = 0$$

avec une méthode itérative avec contraintes linéaires §3.4.7. On peut bien sûr employer plusieurs lignes à la fois. Pour diviser le nombre de paramètres par 2, les auteurs proposent aussi d'utiliser un déplacement radial seulement, *i.e.* remplacer $\tilde{\mathbf{p}}_i^e$ par $\lambda_i \tilde{\mathbf{p}}_i^d$.

6.2.4 Méthodes pour le modèle catadioptrique unifié

Geyer et Daniilidis, et Barreto *et al.* proposent des méthodes d'auto-calibrage du modèle catadioptrique unifié (*cf.*§4.4.1) [11, 45]. Ces méthodes comportent deux étapes et sont basées sur le fait que la projection d'une ligne dans l'image donne toujours une conique. D'abord, les paramètres d'une conique sont estimés pour chaque courbe $\mathcal{L}_j^d$. Ceux-ci permettent ensuite d'estimer les paramètres internes du modèle. Par exemple, dans le cas d'une caméra parabolique–orthographique, les courbes $\mathcal{L}_j^d$ sont nécessairement des arcs de cercle. Notons $(u_j, v_j)^\mathsf{T}$ et r_j, respectivement, le centre et le rayon de chacun de ces cercles. On peut retrouver les paramètres internes de la caméra, *i.e.* le centre de distorsion $(c_x, c_y)^\mathsf{T}$ et la focale f en solutionnant :

$$\arg \min_{c_x,c_y,f} \sum_j \left((u_j - c_x)^2 + (v_j - c_y)^2 + 4f^2 - r_j^2\right).$$

Des algorithmes similaires existent pour d'autres configurations, incluant les caméras *fisheye* [173].

6.2.5 Méthode pour le modèle angulaire polynômial

Ying propose plusieurs méthodes d'auto-calibrage basées sur des images de lignes [172, 177] ou de sphères [174, 175, 176]. L'une d'entre elles permet de calibrer le modèle angulaire polynômial (*cf*.4.3.6) sans minimiser directement l'erreur de correction des lignes dans l'image rectifiée, comme proposé par Devarnay et Faugeras (*cf*.§6.2.2). À la place, l'image est rectifiée sur une sphère faisant en sorte que le critère de rectification s'appuie sur le fait que les points d'une ligne doivent être alignés sur un plan passant par l'origine de la sphère. Pour initialiser le centre de distorsion et le ratio d'aspect de la caméra, Ying propose d'utiliser la bordure elliptique de l'image (voir figure 1.1). Pour la distorsion, il emploie une méthode similaire à Kannala et Brandt, utilisant les spécifications du fabriquant de l'objectif de caméra [72].

6.2.6 Méthodes pour le modèle rationnel

Clauss et Fitzgibbon proposent une méthode d'auto-calibrage pour le modèle rationnel [33]. Comme pour les méthodes s'appliquant au modèle catadioptrique unifié, celle-ci exploite la propriété voulant que la projection d'une droite donne nécessairement une conique. Rappelons que pour ce modèle, un point est rectifié selon l'équation $\mathbf{p}^u = \mathsf{A}_{(3\times 6)}\chi(\tilde{\mathbf{p}}^d)$. L'ensemble des points rectifiés sur un même droite vérifie donc que :

$$\underbrace{\mathbf{l}^\mathsf{T}\mathsf{A}}_{\mathbf{C}^\mathsf{T}}\chi(\tilde{\mathbf{p}}^d) = 0, \tag{6.1}$$

où, l est la ligne rectifiée. Il s'agit aussi de l'équation implicite d'une conique, où $\mathbf{C}$ est un vecteur des paramètres, qui peut être être estimée à partir des points de la courbe. Par construction, nous avons que

$$\mathbf{C} = \mathsf{A}^\mathsf{T}\mathbf{l},$$

ce qui peut être généralisé pour le cas à plusieurs images de lignes :

$$\underbrace{\begin{bmatrix} \mathbf{C}^1 & \dots & \mathbf{C}^n \end{bmatrix}}_{\mathsf{C}_{(6\times n)}} = \mathsf{A} \underbrace{\begin{bmatrix} \mathbf{l}^1 & \dots & \mathbf{l}^n \end{bmatrix}}_{\mathsf{L}_{(3\times n)}}$$

où, n est le nombre de coniques et C est de rang trois lorsque les coniques sont estimées sans erreur. Puisque C contient toujours des imprécisions, A et $\mathbf{l}^i$ sont estimés de façon à minimiser le critère algébrique $\|\mathsf{C} - \mathsf{AL}\|_F^2$. Il s'agit d'un problème classique de factorisation de matrice (pleine) se résolvant à l'aide de la SVD (*cf*.3.5.1).

6.3 Contributions

Les principales faiblesses ou inconvénients des méthodes décrites ci-haut sont de trois types :

- elles nécessitent l'optimisation d'une fonction non-linéaire à partir d'aucune ou d'une très approximative solution initiale ;
- elles requièrent de long segments de ligne afin de permettre une estimation précise de coniques ;
- elles sont mal adaptées aux caméras omnidirectionnelles, assumant que les pixels rectifiés sont situés dans le plan image, *i.e.* dans $\mathbb{R}^2$.

Au prochain chapitre, nous proposons une approche qui remédie à ces lacunes. En faisant l'hypothèse que le centre de distorsion est connu de façon approximative (une hypothèse raisonnable très courante), nous montrons qu'il est possible d'auto-calibrer des caméras centrales ayant des distorsions radiales symétriques en utilisant une fonction paramétrique ou discrète (*cf*.§4.5.3).

Chapitre 7

SELF-CALIBRATION OF A GENERAL RADIALLY SYMMETRIC DISTORTION MODEL

Cet article [152] a été publié comme l'indique la référence bibliographique.
With kind permission of Springer Science and Business Media.

Tardif J.-P., Sturm P., Roy S., Self-calibration of a general radially symmetric distortion model, dans European Conference on Computer Vision (ECCV), Graz, Autriche, pp. 186-199, Mai 2006.

Abstract

We present a new approach for self-calibrating the distortion function and the distortion center of cameras with general radially symmetric distortion. In contrast to most current models, we propose a model encompassing fisheye lenses as well as catadioptric cameras with a view angle larger than 180°.

Rather than representing distortion as an image displacement, we model it as a varying focal length, which is a function of the distance to the distortion center. This function can be discretized, acting as a general model, or represented with e.g. a polynomial expression.

We present two flexible approaches for calibrating the distortion function. The first one is a plumbline-type method; images of line patterns are used to formulate linear constraints on the distortion function parameters. This linear system can be solved up to an unknown scale factor (a global focal length), which is sufficient for image rectification. The second approach is based on the first one and performs self-calibration from images of a textured planar object of unknown structure. We also

show that by restricting the camera motion, self-calibration is possible from images of a completely unknown, non-planar scene.

The analysis of rectified images, obtained using the computed distortion functions, shows very good results compared to other approaches and models, even those relying on non-linear optimization.

7.1 Introduction

Most theoretical advances in geometric computer vision make use of the pin-hole camera model. One benefit of such a model is the linearity of the projection which simplifies multi-view constraints and other structure-from-motion computations. Unfortunately in many cases, this model is a poor representation of how the camera samples the world, especially when dealing with wide angle cameras where radial distortion usually occurs. In addition to these cameras, catadioptric devices (i.e. cameras pointed at a mirror) also admit a very large field of view. Their image distortion can also be seen as a type of radial distortion, although, in general, it cannot be modeled with traditional models. This is because the view angle of these cameras can be larger than 180°, which is not compatible with the usual *image-displacement* approach. The effect of radial distortion is that straight lines in the scene are not in general projected onto straight lines in the image, contrary to pin-hole cameras. Many calibration algorithms can deal with distortion, but they are usually tailor-made for specific distortion models and involve non-linear optimization.

In this paper, we introduce a general distortion model, whose main feature is to consider radially symmetric distortion. More precisely, we make the following assumptions on the camera projection function:

- the aspect ratio is 1,

- the distortion center is aligned with the principal point[1],
- the projection function is radially symmetric (around the distortion center),
- the projection is central, i.e. projection rays pass through a single (effective) optical center.

Given the quality of camera hardware manufacturing, it is common practice to assume an aspect ratio of 1. As for the second and third assumptions, they are made to ensure our model is consistent with both catadioptric devices and regular fisheye cameras. Finally, a central projection is assumed for simplicity even for very large field of view cameras [6, 173] in which a non-single viewpoint might be induced by the lens [21], or by a misaligned mirror [146].

Our full camera model consists therefore of the position of the distortion center and the actual distortion function that maps distance from the distortion center to focal length. This model, together with the above assumptions, fully represents a camera projection function. It is a good compromise between traditional low-parametric camera models and fully general ones, modeling one projection ray per pixel [54, 145], in terms of modeling power and ease and stability of calibration. The model is indeed general enough to represent cameras of different types and with very different view angles.

Problem statement. In this paper, we intend to solve the proposed model relying on images of collinear points in space. Our algorithm makes no assumption on the distortion function and on the distortion center position. Only a rough initial value of the latter is needed.

Organization. A short review of the most popular distortion models is presented in the first section. The model we adopt is presented in §7.3. In §7.4 we propose a

[1] We will see that this constraint may be dropped in some cases.

plumbline method for calibrating our model using images of collinear points. Based on this, we propose a plane-based self-calibration approach, in §7.5. Finally, the performance of our methods is analyzed and compared to another similar approach [36].

7.2. Related Work

As the field of view of a camera lens increases, the distortion occurring in the captured images becomes more and more important. Traditionally, researchers have sought new models with more degrees of freedom and complexity. These models include the traditional polynomial model [60] (which can be combined with a field of view model (FOV) [36]), division [39] and rational [40]. Most of the time the models are calibrated using non-linear optimization of either a full projection model from points located on a calibration object [181] or a homography mapping from a planar grid [40]. Recent papers have also shown that radial distortion models can be calibrated linearly from a calibration grid [61] of by feature point matching between images [39, 40, 156, 157].

Other approaches focus only on calibrating the distortion function by imposing either that a straight line in space should appear straight in the image [26, 36] or that spherical objects should appear circular [136].

The afore mentioned models all apply to cameras with a field of view smaller than 180° since the distortion is *image-based*. They fail to handle data captured by a camera with a view angle larger than 180°, typical for catadioptric devices. Different models and algorithm have been specifically designed to address these cases [44, 94] and their parameters have an explicit geometric interpretation rather than expressing distortion directly.

Finally, only few attempts were made to find models able to deal with dioptric systems (including radial distortion) and catadioptric ones [9, 173]. The model we propose fits in this category with the benefit that its distortion function can be

general.

7.3. Camera Model

We describe the camera model that corresponds to the assumptions explained in the introduction. Consider a camera with canonical orientation, i.e. the optical axis is aligned with the Z-axis and image x and y-axes are parallel to world X and Y-axes respectively. Our camera model is then fully described by the position of a distortion center $(c_x, c_y)^\top$ and a distortion "function" $f : \mathcal{R} \rightarrow \mathcal{R}$, such that an image point $(x, y)^\top$ is back-projected to a 3D line spanned by the optical center and the point at infinity with coordinates:

$$\left[x - c_x, \;\; y - c_y, \;\; f(r), \;\; 0\right]^\top, \; r = \sqrt{(x - c_x)^2 + (y - c_y)^2}$$

The distortion function (it should actually be called "undistortion function", but we did not find this very elegant) can for example be chosen as a polynomial with even powers of r, in which case we have the division model, as used in [39, 156]. The model also subsumes fisheye models [41, 128] and cameras of the 'unified central catadioptric model' [44].

In this paper, we use two representations for the distortion function. The first one is a polynomial of a degree d to be fixed, like in the division model, however including odd powers:

$$f(r) = \sum_{i=0}^{d} \lambda_i r^i. \tag{7.1}$$

The second one is a discrete representation, consisting of a lookup table of the distortion function values at a set of discrete values for r (in practice, we use one sample per step of one pixel). We denote these values as:

$$f(r) = f_r. \tag{7.2}$$

Note that a constant function f allows the representation of a pinhole camera with f's value as focal length. From the above back-projection equation, it is easy to deduce equations for distortion correction, also called rectification in the sequel. This can for example be done by re-projecting the points at infinity of projection rays into a pinhole camera with the same optical center and orientation as the original camera. As for the intrinsic parameters of the (virtual) pinhole camera, we usually also adopt an aspect ratio of 1 and zero skew; if the distortion center is to have the same coordinates in the rectified image as in the original one, and if g denotes the rectified image's focal length, then the homogeneous coordinates of the rectified point are:

$$\begin{bmatrix} g & 0 & c_x \\ 0 & g & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - c_x \\ y - c_y \\ f(r) \end{bmatrix}.$$

In the following, we introduce a few geometric notions that will be used in this paper. A **distortion circle** is a circle in the image, centered in the distortion center. Projection rays of points lying on a distortion circle span an associated **viewing cone** in space. In our model, all cones have the same axis (the optical axis) and vertex (the optical center).

Each cone can actually be understood as an individual pinhole camera, with $f(r)$ as focal length (r being the distortion circle's radius). Geometrically, this is equivalent to virtually moving the image plane along the optical axis, according to the distortion function. This situation is depicted in fig. 7.1. In the case of a camera with a view angle larger than 180°, the focal length becomes equal or smaller than zero. In the zero case, the cone is actually the **principal plane**, i.e. the plane containing the optical center and that is perpendicular to the optical axis. Let us call the associated distortion circle **principal distortion circle**. A negative $f(r)$ is equivalent to a camera with positive focal length, looking backward and whose image is mirrored in x and y. Typical situations for rectification are depicted in fig. 7.2.

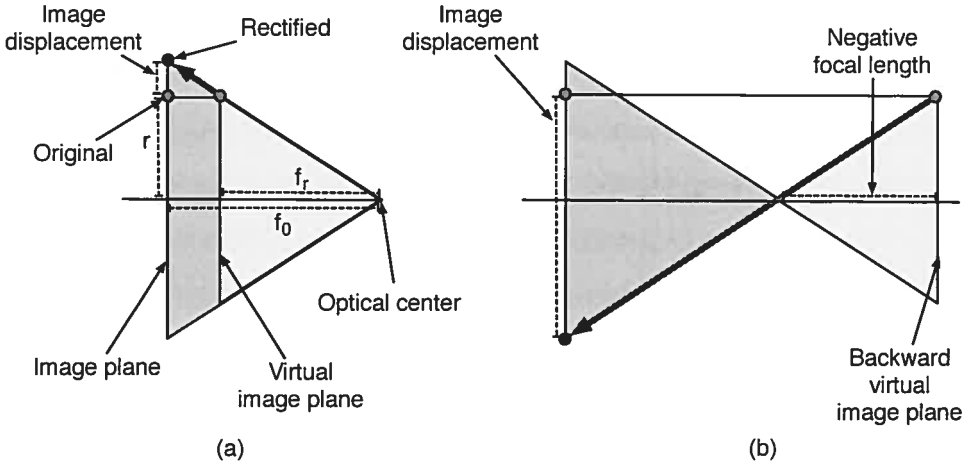


Figure 7.1. Distortion circles are associated with cones in space. Theoretically, any point of the image can be projected into a single plane. a) Pixel from a cone looking forward, b) one from a cone looking backward.

Rectification for cameras with a view angle larger than 180° cannot be done as usual: the above rectification operation is no longer a bijection (two points in the original image may be mapped to the same location in the rectified one) and points on the principal distortion circle are mapped to points at infinity (fig. 7.2b). It is still possible to rectify individual parts of the image correctly, by giving the virtual pinhole camera a limited field of view and allowing it to rotate relative to the true camera.

7.4. Plumbline Calibration

In this section, we show that the distortion function f and the distortion center can be recovered linearly from the images of lines (straight edges) or points that are collinear in space. This is thus akin to the classical plumbline calibration technique [26, 36].

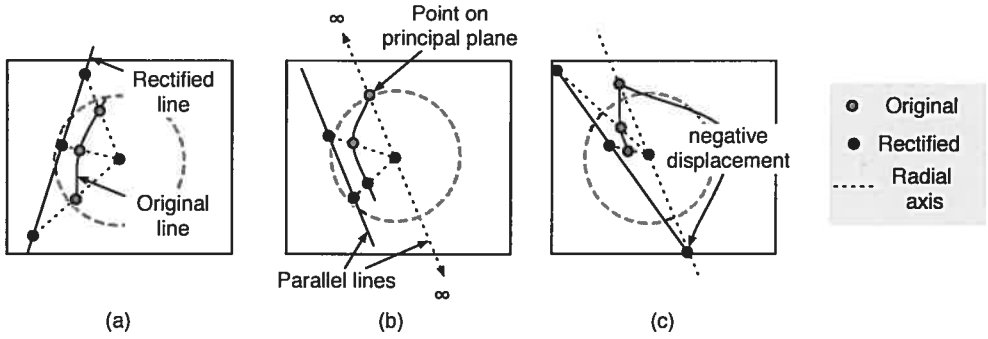


Figure 7.2. Situations where three points are rectified into collinear positions. a) Three points corresponding to forward cones. b) One point located on principal distortion circle, i.e. scene point on principal plane. c) Two points on forward cones and one on a backward cone.

7.4.1. Calibration of Distortion Function

We obtain linear constraints on the distortion function as follows. Consider the images of three collinear points, $\mathbf{p}_i = (x_i, y_i)^\top$. For now, let us assume that the distortion center is known and that the image coordinate system is centered in this point. Hence, $r_i = ||(x_i, y_i)||$ is the distance of a point from the distortion center. Provided that these points should be collinear once rectified, we know that:

$$\begin{vmatrix} x_0 & x_1 & x_2 \\ y_0 & y_1 & y_2 \\ f(r_0) & f(r_1) & f(r_2) \end{vmatrix} = 0 \tag{7.3}$$

which can be written explicitly as a linear constraint on the $f(r_i)$'s:

$$f(r_0)\begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} + f(r_1)\begin{vmatrix} x_2 & x_0 \\ y_2 & y_0 \end{vmatrix} + f(r_2)\begin{vmatrix} x_0 & x_1 \\ y_0 & y_1 \end{vmatrix} = 0. \tag{7.4}$$

If f is of the form (7.1) or (7.2), then this equation gives a linear constraint on its parameters λ_i respectively f_r.

Constraints can be accumulated from all possible triplets of points that are projections of collinear points in space. We thus obtain a linear equation system of the form $\mathbf{Ax} = \mathbf{0}$, where $\mathbf{x}$ contains the parameters of f (the λ_i's or the f_r's). Note that constraints from triplets where two or all three image points lie close to one another are not very useful and hence can be neglected in order to reduce the number of equations. Solving this system to least squares yields parameters that maximize the collinearity of the rectified points[2]. Note that the equation system is homogeneous, i.e. the distortion parameters are only estimated up to scale. This is natural, as explained below; a unique solution can be guaranteed by setting $\lambda_0 = 1$ as is usually done for the division model, or by setting one f_r to a fixed value.

7.4.2. Calibration of Distortion Center

So far, we have assumed the knowledge of the distortion center. In this section, we show how it can be estimated as well, in addition to the actual distortion function. A first idea is to sample likely positions of the distortion center, e.g. consider a regular grid of points in a circular region in the image center, and compute the distortion function for each of them using the above method. We then keep the point yielding the smallest residual of the linear equation system as the estimated distortion center. This approach is simple and not very elegant, but is fully justified and works well in practice. Its downside is that the computation time is proportional to the number of sampled points.

Therefore, we investigate a local optimization procedure, as opposed to the above brute force one. Let (c_x, c_y) be the unknown distortion center. Equation (7.3) now

[2] However, it is not optimal in terms of geometric distance.

becomes:

$$\begin{vmatrix} x_0 - c_x & x_1 - c_x & x_2 - c_x \\ y_0 - c_y & y_1 - c_y & y_2 - c_y \\ f(\left\| \begin{bmatrix} x_0 - c_x \\ y_0 - c_y \end{bmatrix} \right\|) & f(\left\| \begin{bmatrix} x_1 - c_x \\ y_1 - c_y \end{bmatrix} \right\|) & f(\left\| \begin{bmatrix} x_2 - c_x \\ y_2 - c_y \end{bmatrix} \right\|) \end{vmatrix} = 0. \qquad (7.5)$$

First, this constraint cannot be used directly for the discretized version of the distortion function. Second, if we use the polynomial model, the constraint is highly non-linear in the coordinates of the distortion center.

We thus consider an approximation of (7.5): we assume that a current estimate of the distortion center is not too far away from the true position ($||(c_x, c_y)||$ is small), so that f can be approximated with $(c_x, c_y) = \mathbf{0}$ and

$$f(\left\| \begin{bmatrix} x \\ y \end{bmatrix} \right\|) \approx f(\left\| \begin{bmatrix} x - c_x \\ y - c_y \end{bmatrix} \right\|).$$

Equation (7.5) simplifies to:

$$\begin{vmatrix} x_0 - c_x & x_1 - c_x & x_2 - c_x \\ y_0 - c_y & y_1 - c_y & y_2 - c_y \\ f(\left\| \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \right\|) & f(\left\| \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \right\|) & f(\left\| \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} \right\|) \end{vmatrix} = 0 \qquad (7.6)$$

which is linear in c_x and c_y. Once again, combining many constraints leads to an over-determined linear equation system. The recovered distortion center may not be optimal because the points are expressed relative to the approximate center and because of the simplification of (7.5). Hoping that the previous assumptions are applicable, this new center should nevertheless improve our rectification. This estimation is used in a local optimization scheme of alternation type:

0. Initialize the distortion center with e.g. the center of the image.

1. Fix the distortion center and compute the distortion function (§7.4.1).

2. Fix the distortion function and update the distortion center (§7.4.2).

3. Go to step 1, unless convergence is observed.

Alternatively to using the least-squares cost function based on the algebraic distance (7.3), we also consider a more geometric cost function to judge convergence in step 3. Consider a set of image points belonging to a line image. From the current values of distortion center and function, we compute their projection rays and fit a plane as follows: determine the plane that contains the optical center and that minimizes the sum of (squared) angles with projection rays. The residual squared angles, summed over all line images, give the alternative cost function.

7.4.3. Discussion

The estimation of distortion center and function is based on an algebraic distance expressing collinearity of rectified image points. Better would be of course to use a geometric distance in the original images; this is possible but rather involved and is left for future work.

We briefly describe what the calibration of the distortion function amounts to, in terms of full metric calibration. First, recall that the distortion function can be computed up to scale only from our input (see §7.4.1). This is natural: if we have a distortion function that satisfies all collinearity constraints, then multiplying it by a scale factor results in a distortion function that satisfies them as well. This ambiguity means that once the distortion function is computed (up to scale) and the image rectified, the camera can be considered as equivalent to a pinhole camera with unknown focal length, with the difference that the field of view is potentially larger than 180°. Any existing focal length calibration or self-calibration algorithm

designed for pinhole cameras can be applied to obtain a full metric calibration. A direct application of such algorithms can probably use only features that lie inside the principal distortion circle, but it should be possible to adapt them so as to use even fields of view larger than 180°. At this step, the second assumption of §7.1 can also be relaxed if desired: a full pinhole model, i.e. not only focal length, can in principle be estimated from rectified images.

7.5. Self-Calibration

We now develop a plane-based self-calibration approach that is based on the plumbline technique of the previous section. Consider that the camera acquires two images of a textured plane with otherwise unknown structure. We suppose that we can match the two images densely; the matching does not actually need to be perfectly dense, but assuming it simplifies the following explanations. This is discussed below in more details.

We now describe how dense matches between two images of a planar scene allow the generation of line images and hence to apply the plumbline technique. Consider any radial line (line going through the distortion center) in the first image; the projection rays associated with the points on that line are necessarily coplanar according to our camera model. Therefore, the scene points that are observed along that radial line must be collinear: they lie on the intersection of the plane of projection rays, with the scene plane. Due to the dense matching, we know the projections of these collinear scene points in the second image. By considering dense matches of points along n radial lines in one image, we thus obtain n line images in the other image, and vice versa. In addition, these line images usually extend across a large part of the image, bringing about strong constraints.

We now simply stack all plumbline constraints (7.4) for all pairs of images, and solve for the distortion parameters as in §7.4. Here, we have assumed the knowledge of the distortion center (in order to define radial lines); the distortion center can of

course also be estimated, using e.g. the exhaustive approach of §7.4.2. Moreover, the input, once rectified, can be given to a classical plane-based self-calibration algorithm to obtain a full metric calibration, using e.g. [163].

Dense Matching. Dense matching can be achieved rather straightforwardly. If the camera acquires a continuous image sequence, most existing optical flow algorithms can be applied for successive frames and their results propagated in order to obtain a dense matching between two images with a substantial motion between them. In addition, the fact that a planar scene is observed eliminates the occlusion problem. If the scene is not sufficiently textured, but only allows to extract and track sparse interest points, then we proceed as follows. We extract dominant lines in each image using a Hough transform of the extracted interest points, and only keep the lines passing near the current distortion center estimate. These are almost radial lines. This is illustrated in fig. 7.3, and an example is shown in fig. 7.4a,b. The rest of the self-calibration is as above.

Constrained Camera Motions. Another way to obtain line images without the need for linear features in the scene is to acquire images under constrained camera motions. A first possibility is to carry out pure rotations about the optical center, as suggested also by [156]. The scene can then be assimilated to a plane, and the above self-calibration method can be directly applied. A second possibility is to perform pure translations (with e.g. a tripod) and to track image points across several images. In this case, any point track constitutes a line image (an example is shown in fig. 7.4c,d).

7.6. Results and Analysis

We tested our algorithm with data acquired from real and simulated cameras. An 8.0 mm lens, a 3.5mm fisheye lens and a para-catadioptric camera were used. We also simulated ten cameras featuring distortions from small to very large.

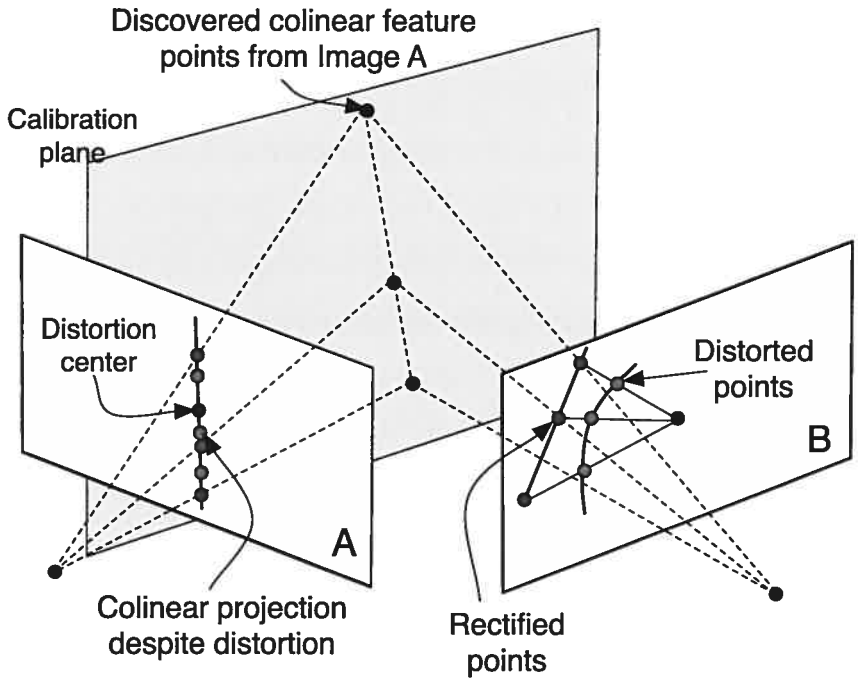
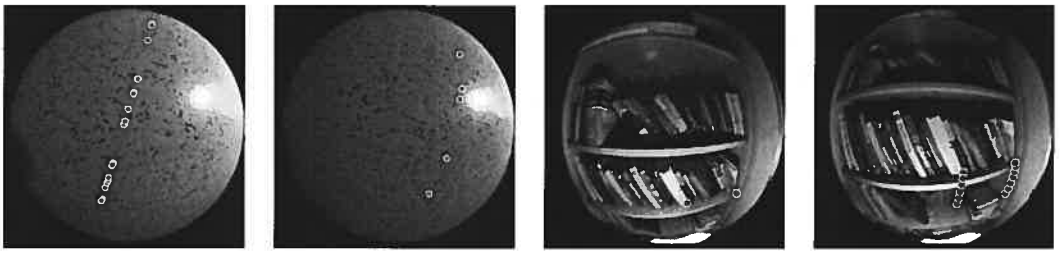
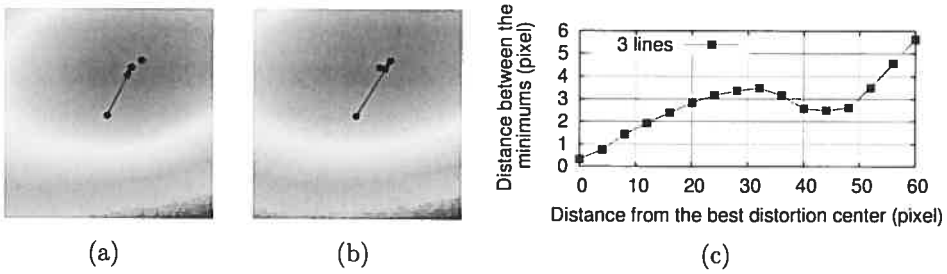


Figure 7.3. Dense matching between two images to extract radial lines.



(a) (b) (c) (d)

Figure 7.4. (a)+(b) Two images of a planar scene. a) shows interest points lying on a radial line in the first image and b) corresponding points in the second image. (c)+(d) Two images of a general scene, taken with pure translation. c) shows two interest points in the first image and d) their paths, accumulated in the last image.



(a) (b) (c)

Figure 7.5. Plots of cost functions and optimization paths associated with (a) eq. (7.5) and (b) eq. (7.6). (c) Distance between minima of these two cost functions, with respect to distance of current estimate of distortion center from optimal one. Data from the 3.5mm fisheye lens.

7.6.1. Convergence Analysis of the Distortion Center Detection

Two aspects of convergence of the plumbline method were evaluated. First, evaluating if the minimization of the constraints given by (7.6) instead of (7.5) leads to similar results. This is not critical though, as the path of the optimizer needs not be the same to ensure convergence. On the other hand, if the paths are similar, it suggests that the convergence pace is not penalized too much with the simplified cost function. We proceeded as follows. For samples of distortion center positions in a box around the initial position, we computed the two cost functions and found their minima (c.f. fig. 7.5a,b). We see that the functions' general shapes are almost identical, as well the positions of their respective minima. Another evaluation consists in initializing the distortion center randomly around the optimal one and finding the minima of the two cost functions. Figure 7.5c shows the average distance between these minima, as a function of the distance of the given distortion center from the optimal one. It is generally small, suggesting that both cost functions may lead to similar optimization paths.

Secondly, the overall convergence was tested with simulated and real data. In the first case, three criteria were considered: the number of line images given as input,

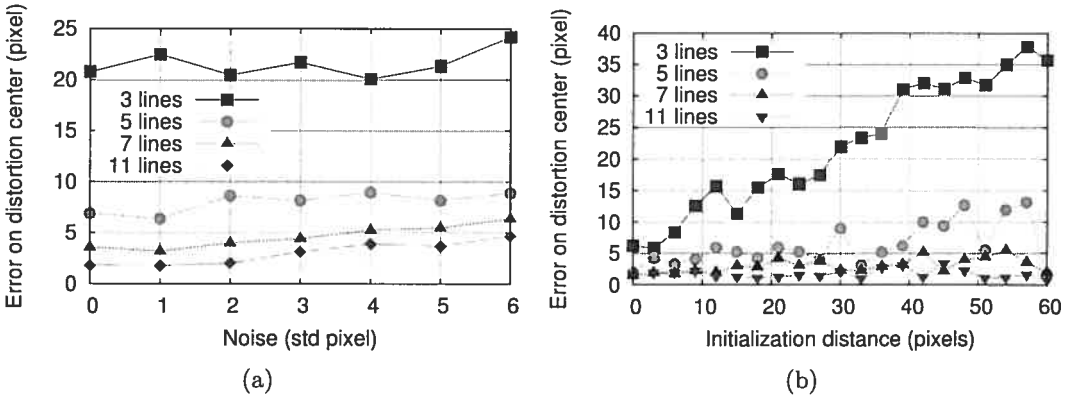


Figure 7.6. Precision of the recovered distortion center on simulated data w.r.t. a) noise and number of lines, b) number of lines and initialization distance.

the amount of noise added to the data and the distance of the given initial distortion center from the true one. For each simulated camera, up to 11 line segments were generated randomly, Gaussian noise of standard deviation 0 to 6 pixels was added to image point coordinates and these were then quantized to pixel precision. For every camera, 50 initial values for the distortion center were randomly chosen in a circle of 60 pixels radius around the true position (for images of size 1000×1000) and given as input to the algorithm. This a realistic test considering that for our real cameras, we found that the estimated distortion center converged to around 30 pixels from the initial value (image center) in the worst case. The results in fig. 7.6 show that the number of lines has a much larger impact on the quality of the recovered distortion center than the noise and the initialization distance. This is especially true when the number of line is larger than 7.

7.6.2. Plumbline Calibration

We acquired images of lines with our real cameras, calibrated the distortion and then performed rectification. Once again, we tested the convergence and also the quality of the rectification by checking the collinearity of rectified line images. Convergence was

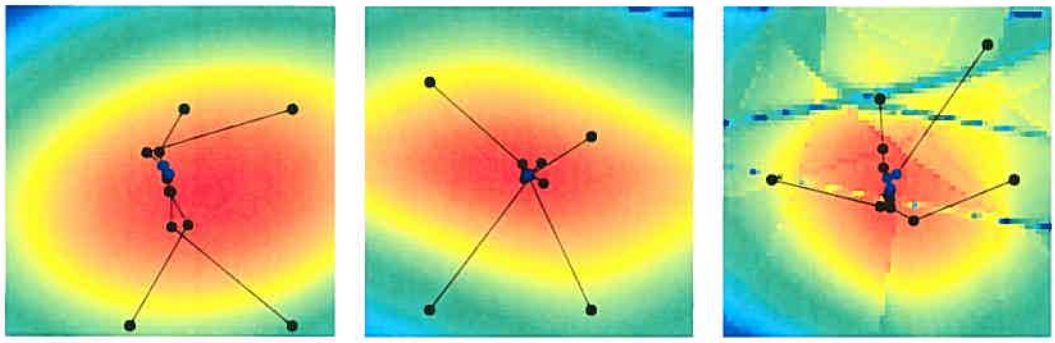


Figure 7.7. Convergence examples of the algorithm for a) the 8.0 mm, b) the 3.5 mm fisheye, c) the para-catadioptric. The density plots show the value of the cost function explained at the end of §7.4.2, with f computed using distortion center positions (c_x, c_y) in a box of 60×60 pixels around the final distortion centers. In dark-green, different initializations of the algorithm; in black, the centers at each step of the algorithm; in purple, the final centers.

never found to be an issue, especially for the two dioptric lenses (cf. fig. 7.7). Even with a really bad initialization of the distortion center, resulting in a poor initial estimate of the distortion function, the algorithm converged surprisingly fast (fig. 7.9). The distortion functions for our real cameras are shown in fig. 7.8 as well as rectified images in fig. 7.10 (images not used for the calibration). We compared our approach with the one presented in [36], run on the same data. Since that approach performs non-linear optimization, it can easily incorporate different distortion models. Results for different models are shown in table 7.1; we initialized the distortion centers with the one that was estimated with our approach and the distortion function as a constant.

Details are given in fig. 7.11 for the catadioptric cameras. We observe that a polynomial function did not give satisfying results. Using higher degrees (up to 10) and changing the distortion function did not give much better results. On the other hand, we see that a division function is very well suited to model the distortion in

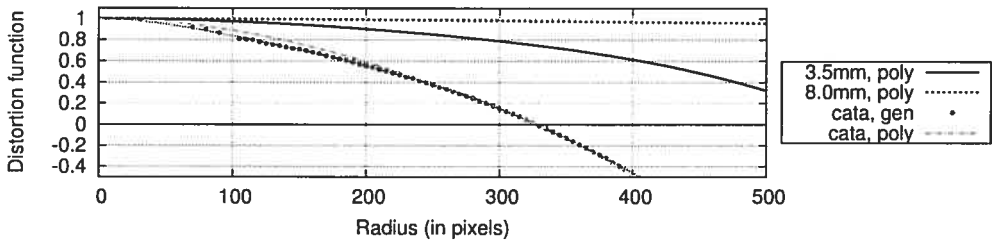


Figure 7.8. Calibrated distortion functions for our real cameras. *poly* **refers to (7.1) and** *gen* **to (7.2). For the 8.0 and 3.5mm, both representations lead to virtually identical results (details at table 7.1).**

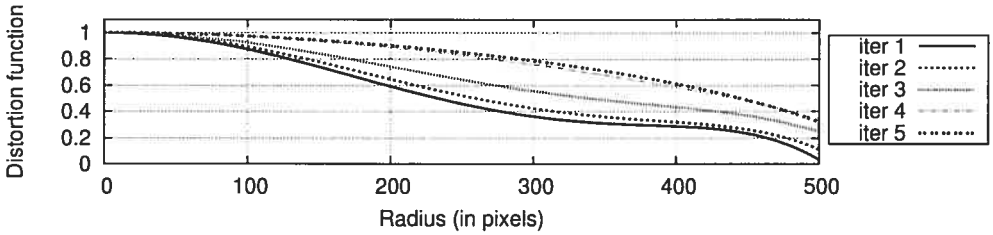


Figure 7.9. The distortion function of the fisheye lens, at different iterations of the calibration algorithm for an initial center very far from the true position (200,400). The final estimate of (512,523) was found in only 5 iterations (image of size 1000×1000 **pixels). Subsequent steps were only minor improvements.**

the image.

7.6.3. Self-Calibration from Real Sequences

Two sequences were tested. In the first one, points were tracked from a flat surface (our laboratory floor) with a hand-held camera. In the second case, a tripod was used and the camera was translated in constant direction. Overall, the results were satisfying although not as precise as with the direct plumbline technique using images of actual linear features. Results are summarized in table 7.2; values shown were computed like explained in table 7.1 and using images of actual lines. The distortion center detection was also not as precise. The algorithm converged as usual, but

Table 7.1. Results using our models and algorithm (first two rows) and other models and the non-linear algorithm of [36]. Shown values refer to residual distances for fitting lines to rectified points (average and worst case). The rectified images were scaled to have the same size as the original. For the catadioptric camera, our approach used all the points, whereas the others used only the points corresponding to forward viewing cones (they failed otherwise). "—" means the algorithm did not converge without careful initialization or gave very bad results.

Models and rectifying equations	8mm		3.5mm		catadioptric	
Discrete model of (7.2)	0.16	1.03	**0.35**	**3.7**	**0.51**	**7.6**
Model of (7.1) with $d = 6$	0.16	1.12	0.35	5.5	**0.47**	**6.3**
6^{th} order polynomial $\mathbf{p}(1 + \lambda_1\|\|\mathbf{p}\|\| + ... + \lambda_6\|\|\mathbf{p}\|\|^6)$	0.16	1.08	0.42	7.0	1.5	14.4
6^{th} order division (non-linear)	0.16	1.08	0.36	5.6	—	—
FOV-model [36]: $\mathbf{p}\frac{\tan(\omega\|\|\mathbf{p}\|\|)}{2\tan(\frac{\omega}{2})\|\|\mathbf{p}\|\|}$	0.23	4.86	0.54	7.9	—	—
FOV-model + 2^{nd} order polynomial	0.16	1.06	0.37	6.1	—	—

not exactly to the best distortion center. In fact, it was much closer to the image center. This is explained by the fact that towards the image border, features are much more difficult to track: they are smaller and blurry. In this case, they are usually dropped by the tracking algorithm resulting in less data for large radiuses, where the distortion is the worst. Consequently, the distortion is a little bit under-evaluated and the distortion center less well constrained.

7.7. Conclusion

We presented flexible calibration methods for a general model for radial distortion, one plumbline type method and one for plane-based self-calibration. The methods were applied for simulated and real images of different cameras (fisheye and catadioptric). Results are satisfying, in terms of convergence basin and speed, precision as well as accuracy.

Table 7.2. Results for the 3.5mm fisheye with data from real sequences (fig. 7.4).

Models	plane		translation	
Discrete model of (7.2)	0.68	8.05	0.55	7.0
Model of (7.1) with $d = 6$	0.58	9.7	0.85	14.6

The most closely related works are [156, 157]. There, elegant though rather more involved procedures are proposed. These start with an even more general camera model than here, that does not enforce radial symmetry; only after computing and exploiting multi-view relations for that model, radial symmetry is enforced in order to compute distortion parameters. Our methods are much simpler to implement, use radial symmetry directly and can work with fewer images (two for plane-based self-calibration). Future work will mainly concern improving the tracking for the self-calibration method and investigating the optimization of reprojection based cost functions.

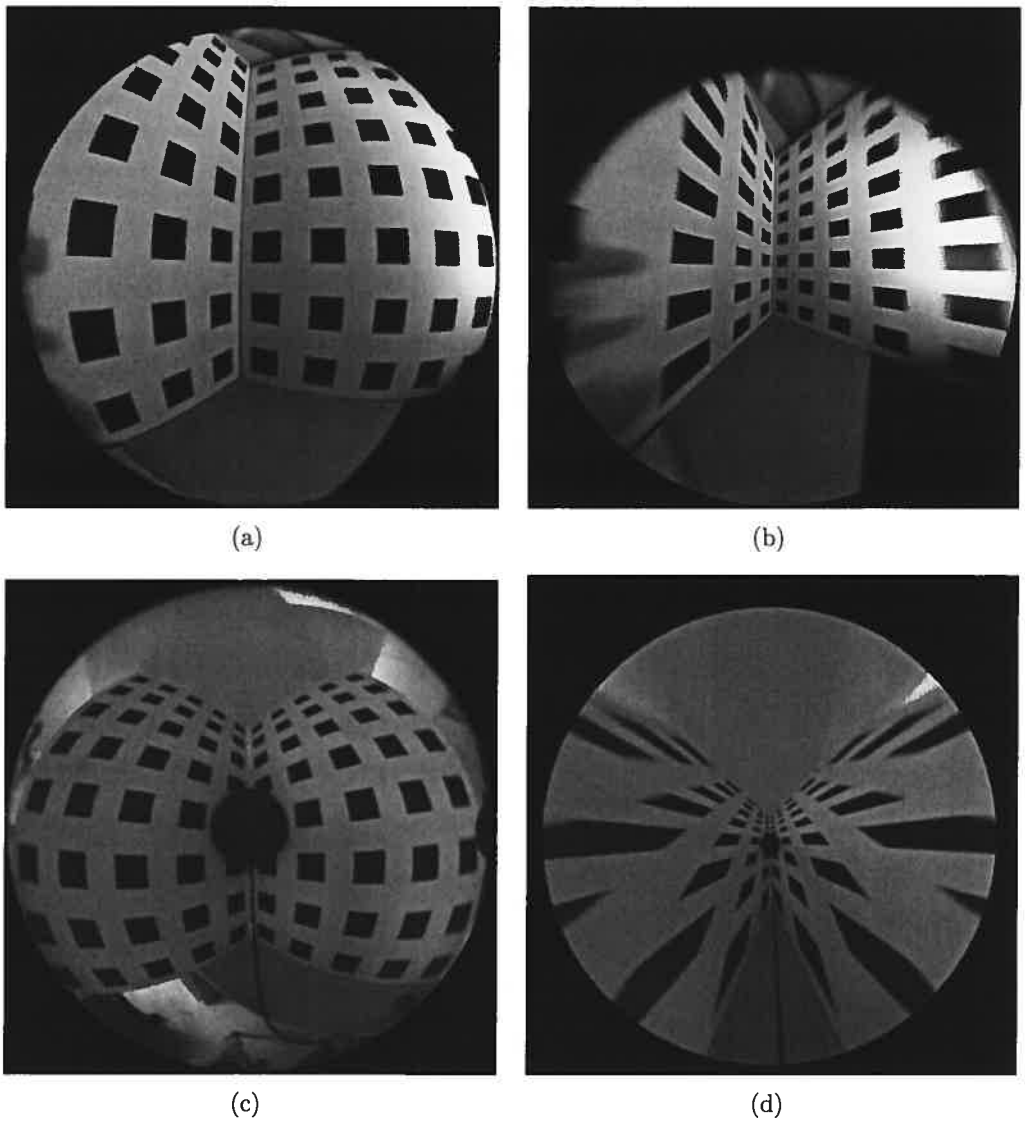
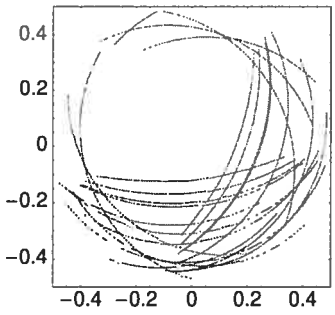
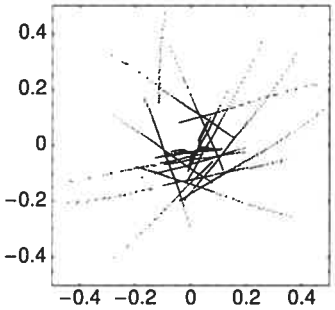


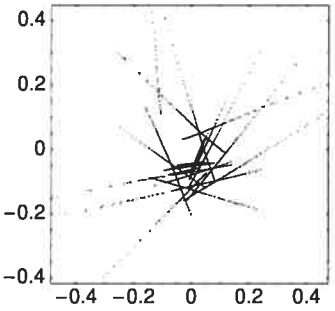
Figure 7.10. Rectification examples. a,b) A 3.5mm fisheye original and rectified images. c,d) a catadioptric image. The radius of the principal distortion circle was estimated as 329 pixels, so circles of radius 0 to 320 pixels were rectified.



(a)



(b)



(c)

Figure 7.11. Line images used as input of the algorithms and rectification results for the catadioptric camera (with principal distortion circle found at 329 pixels). a) Input (only shown for radius smaller than 315 pixels), b) Rectification with a traditional model of degree 6 (model as in third row of table 7.1), c) with the polynomial distortion function (7.1) and $d = 6$ (the discrete model of (7.2) gave almost identical results).

Chapitre 8

MÉTHODE D'AUTO-CALIBRAGE DE DISTORSION RADIALE PAR POINTS DE CORRESPONDANCE

8.1 Introduction

Les méthodes d'auto-calibrage de distorsion d'image basées sur la rectification d'images de lignes distordues ont l'avantage de fonctionner, en principe, avec une seule image. En pratique, ces méthodes fonctionnent rarement de façon complètement automatique parce que la détection et l'extraction des courbes est un problème difficile. De plus, elles sont surtout limitées aux environnements urbains où un grand nombre de droites sont présentes.

Dans ce chapitre, nous présentons des méthodes utilisant de correspondances de points d'intérêt (ou points saillants) entre deux images ou plus. Ces correspondances sont généralement obtenues à partir de "descripteurs" de points, invariants au point de vue de la caméra et assez fiables, même dans le cas d'images très distordues [83, 95]. Par rapport aux méthodes par lignes, l'utilisation de correspondances est plus difficile, puisque la relation entre les points implique autant les paramètres externes (homographie, matrice fondamentale ou autre tenseur d'appariement) que les paramètres internes.

Organisation

Plusieurs méthodes d'auto-calibrage par correspondances sont présentées §8.2. Leur présentation est suivie d'un résumé de nos contributions sur ce sujet §8.3.

8.2 *Auto-calibrage de distorsion radiale avec correspondances*

Nous utilisons les vecteurs $\mathbf{p}_i$ et $\mathbf{q}_i$ pour décrire une correspondance entre deux images et laissons souvent tomber l'indice i. Toutes les méthodes supposent que le centre de distorsion est connu *a priori*.

8.2.1 Méthode pour le modèle division à un paramètre (même caméra)

Fitzgibbon développe une méthode numérique d'estimation simultanée de la matrice fondamentale entre deux images distordues prises par la même caméra et du modèle division à un seul paramètre [39]. Dans ce contexte, un pixel est rectifié selon :

$$\mathbf{p}^u \propto \begin{pmatrix} \tilde{\mathbf{p}}^d \\ 1 \end{pmatrix} + k \begin{pmatrix} 0 \\ 0 \\ r^d \end{pmatrix}$$

où, k est le paramètre de distorsion. Après manipulation algébrique de la relation épipolaire $\mathbf{p}^{u\mathsf{T}}\mathsf{F}\mathbf{q}^u = 0$ on obtient

$$(\mathbf{D}_1 + k\mathbf{D}_2 + k^2\mathbf{D}_3)\mathbf{f} = 0$$

où, $\mathbf{f}^\mathsf{T} = (\mathbf{F}^{1\mathsf{T}}, \mathbf{F}^{2\mathsf{T}}, \mathbf{F}^{3\mathsf{T}})$ et les vecteurs $\mathbf{D}_i \in \mathbb{R}^9$ sont constitués d'éléments de $\tilde{\mathbf{p}}^d$, $\tilde{\mathbf{q}}^d$ et k. À partir de plusieurs correspondances, estimer $\mathbf{f}$ et k est un problème de valeurs propres quadratiques[1] (QEP) pour lequel il existe des solutions numériques [158]. Cet algorithme nécessite un minimum de neuf points de correspondances entre les deux images (huit ddl pour F sans la contrainte de rang et un ddl pour la distorsion). Kukelova et Pajdla présentent un algorithme permettant de forcer la contrainte de rang lorsque le nombre de correspondances est exactement de huit [76].

[1] *Quadratic Eigenvalue Problem*

8.2.2 Méthode pour le modèle division à un paramètre (caméras différentes)

Barreto and Daniilidis proposent une deuxième solution pour l'estimation du modèle division à un seul paramètre [10]. À la différence de celle de Fitzgibbon, elle a l'avantage d'être applicable à des caméras ayant des fonctions de distorsion différentes. Un pixel $\mathbf{p}^d = (x, y, z)^\mathsf{T}$ est rectifiée selon :

$$\mathbf{p}^u \propto \left(xz, yz, z^2 + k_p \|(x, y)\|^2\right)^\mathsf{T}$$

où, k_p est le paramètre de distorsion associé à la première image. En plaçant la coordonnées homogène au début du vecteur, $\mathbf{p}^u$ peut se réécrire à l'aide de coordonnées augmentés[2]

$$\mathbf{p}^u \propto \underbrace{\begin{bmatrix} k_p & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\Delta_{k_p}} \underbrace{\begin{pmatrix} \|(x, y)\|^2 \\ xy \\ yz \\ z^2 \end{pmatrix}}_{\rho(\mathbf{p}^d)},$$

où, ρ est la fonction d'augmentation. Ceci permet de construire la matrice fondamentale :

$$\mathbf{p}^{u\mathsf{T}} \mathsf{F} \mathbf{q}^u = \rho(\mathbf{p}^d)^\mathsf{T} \underbrace{\Delta_{k_p}^\mathsf{T} \mathsf{F} \Delta_{k_q}}_{\mathcal{F}} \rho(\mathbf{q}^d)$$

où, F est la matrice fondamentale entre les deux vues rectifiées et $\mathcal{F}$, elle aussi de rang 2, est la relation épipolaire des coordonnées distordues, donnée par

$$\mathcal{F} = \begin{bmatrix} k_p k_q f_{33} & k_q f_{31} & k_q f_{32} & k_q f_{33} \\ k_p f_{13} & f_{11} & f_{12} & f_{13} \\ k_p f_{23} & f_{21} & f_{22} & f_{23} \\ k_p f_{33} & f_{31} & f_{32} & f_{33} \end{bmatrix},$$

[2] *Lifted coordinate*

k_p est le facteur de distorsion associé aux points de la première image et k_q celui associé aux points de la deuxième. En forçant la contrainte de rang *a posteriori*, $\mathcal{F}$ peut être estimé à partir d'au moins 15 correspondances. Ensuite, le reste des contraintes sur la forme de $\mathcal{F}$ sont forcées par optimisation non-linéaire.

8.2.3 Matrices fondamentales mixtes

Faisant suite aux travaux de Geyer et Daniilidis sur la matrice fondamentale pour les caméras catadioptriques parabolique–orthographique [48, 49], Sturm montre qu'il est possible de construire des relations bilinéaire, trilinéaire et quadrilinéaire entre les images de caméras perspective, affine ou parabolique–orthographique [144]. Par exemple, il existe une matrice fondamentale de taille (3×4) entre des caméras perspective ou affine avec une caméra parabolique–orthographique.

Ces résultats sont généralisés par Barreto et Daniilidis en utilisant les vecteurs de Veronese[3] [13] :

$$\Gamma(\mathbf{p},\mathbf{p}) = \begin{pmatrix} x^2 & y^2 & z^2 & xz & yz & xy \end{pmatrix}^\mathsf{T}, \quad \text{pour} \quad \mathbf{p} = \begin{pmatrix} x & y & z \end{pmatrix}^\mathsf{T}.$$

Les relations épipolaires sont construites en utilisant $\Gamma(\mathbf{p}^d,\mathbf{p}^d)$, résultant en des matrices fondamentales de tailles (6×6) entre caméras catadioptriques parabolique et hyperbolique, parabolique et parabolique, et aussi entre *fisheye* et catadioptrique. Finalement, une matrice de taille (3×6) permet l'appariement d'une image distordue avec une image de caméra perspective, les coordonnées des points de cette dernière n'étant pas augmentées.

8.2.4 Méthode pour le modèle rationnel

Pour le modèle rationnel, Clauss et Fitzgibbon proposent une matrice fondamentale augmentée apparentée à celle de Barreto et Daniilidis (*cf.*§4.5.1) [32]. Elle est

[3] *Veronese map*

obtenue en remplaçant $\mathbf{p}^u$ et $\mathbf{q}^u$, respectivement, par $\chi(\tilde{\mathbf{p}}^d)$ et $\chi(\tilde{\mathbf{q}}^d)$ dans la relation épipolaire classique :

$$\begin{aligned} \mathbf{p}^{u\mathsf{T}}\mathsf{F}\mathbf{q}^u &= \\ \chi(\tilde{\mathbf{p}}^d)^\mathsf{T} \underbrace{\mathsf{A}^\mathsf{T}\mathsf{F}\mathsf{A}}_{\mathcal{F}_{(6\times6)}} \chi(\tilde{\mathbf{q}}^d) &= 0 \end{aligned}$$

où, $\mathcal{F}$, de rang 2, est la matrice fondamentale radiale de laquelle A et F doivent être extraits. Notez que pour utiliser des caméras ayant des paramètres internes différents il faudrait remplacer une des deux matrices A par une matrice différente A′. Ce cas n'est pas abordé par les auteurs et il semble que la difficulté soit reliée à la décomposition de $\mathcal{F}$ en A, A′ et F.

8.2.5 Méthodes pour caméra radiale 1D

Ces méthodes consistent à construire des tenseurs d'appariement entre des vues de caméras radiales 1D (*cf.*§4.6.4) [156, 157]. Pour des caméras centrales en rotation pure, un tenseur trifocal (trois vues) est suffisant, alors qu'un tenseur quadrifocal (quatre vues) permet l'auto-calibrage de caméras possiblement non-centrales sans restriction sur leur position et orientation. Rappelons que l'équation de projection de ces caméras est :

$$\lambda\tilde{\mathbf{l}} = \mathsf{M}_{(2\times4)}\mathbf{P},\ \mathbf{P} \in \mathbb{P}^3, \tilde{\mathbf{l}} \in \mathbb{R}^2$$

où, λ est un facteur d'échelle inconnu. Cette équation signifie que la projection des points est associée à une ligne radiale dans l'image. Il est possible de formuler des tenseurs d'appariement entre différentes vues. En effet, pour quatre vues d'un même

point $\mathbf{P}$, on peut construire le système :

$$\underbrace{\begin{bmatrix} \mathsf{M} & \tilde{\mathsf{I}} & 0 & 0 & 0 \\ \mathsf{M}' & 0 & \tilde{\mathsf{I}}' & 0 & 0 \\ \mathsf{M}'' & 0 & 0 & \tilde{\mathsf{I}}'' & 0 \\ \mathsf{M}''' & 0 & 0 & 0 & \tilde{\mathsf{I}}''' \end{bmatrix}}_{\mathcal{M}_{(8\times 8)}} \begin{pmatrix} \mathbf{P} \\ -\lambda \\ -\lambda' \\ -\lambda'' \\ -\lambda''' \end{pmatrix} = \mathbf{0}.$$

ce qui signifie que $\mathcal{M}$ possède un noyau droit et donc que $\det(\mathcal{M}) = 0$. L'expansion de ce déterminant permet de construire une contrainte quadrilinéaire :

$$\sum_{i}^{2}\sum_{j}^{2}\sum_{k}^{2}\sum_{l}^{2} \tilde{l}_i\tilde{l}_j\tilde{l}_k\tilde{l}_l Q_{ijkl} = 0$$

où, Q_{ijkl} est un élément du tenseur quadrifocal radial Q de taille $(2 \times 2 \times 2 \times 2)$ similaire, mais plus petit, que le tenseur quadrifocal classique [160].

En pratique, le cas de caméras dont les axes optiques s'intersectent à l'origine est encore plus simple, puisque $\mathsf{M}(0,0,0,1)^{\mathsf{T}} = (0,0)^{\mathsf{T}}$. Ainsi, on peut simplifier l'équation de projection :

$$\tilde{\mathsf{I}} \propto \mathsf{M}_{(2\times 3)}\tilde{\mathbf{P}},\ \tilde{\mathbf{P}} \in \mathbb{R}^3, \tilde{\mathsf{I}} \in \mathbb{R}^2.$$

où, $\tilde{\mathbf{P}}$ est essentiellement une direction. Comme ci-haut, on peut construire une relation linéaire tensorielle pour la correspondance d'une ligne radiale dans trois vues. On obtient un tenseur trifocal T de taille $(2 \times 2 \times 2)$. À partir des tenseurs Q ou T, il est possible de récupérer les matrices de caméra radiale 1D et d'effectuer une reconstruction projective de la scène. Celle-ci est suffisante pour retrouver les paramètres d'un modèle de distorsion division polynomial [156]. Mieux encore, elle peut être mise à niveau à une reconstruction métrique permettant l'auto-calibrage complet du modèle radial 1D de chaque caméra de façon non-paramétrique. Ces deux méthodes ont été testées au chapitre 9.

8.2.6 Méthode de Ramalingam et Sturm

Ramalingam et Sturm présentent une méthode d'auto-calibrage pour le modèle de caméra centrale radiale symétrique non-paramétrique (*cf.*§4.5.3) [114] à partir de deux vues. Leur méthode requiert des correspondances denses entre deux images ; chaque cercle de distorsion doit contenir cinq correspondances avec la deuxième image. Leur méthode procède en deux étapes trop complexes pour être décrites en détails dans le présent document.

8.2.7 Méthode le modèle de caustique

La méthode de Micusik et Pajdla utilise un modèle de caustique similaire à celui introduit par Swaminathan *et al.* [146] pour effectuer une reconstruction 3D métrique à l'aide caméras catadioptrique non-centrales [94]. Une première reconstruction est obtenue en utilisant une approximation centrale du modèle de projection. De façon similaire à Fitzgibbon (*cf.*§8.2.1) [39], l'estimation de la courbure du miroir et des matrices fondamentales entre les différentes vues est possible en solutionnant un problème quadratique de valeurs singulières (QEP). Finalement, le modèle non-central est utilisé pour raffiner la reconstruction par ajustement de faisceaux [164].

8.2.8 Méthode non-paramétrique de Nistér et al.

Nous présentons le résultat principal de la méthode non-paramétrique pour les caméras centrales, introduite par Nistér *et al.* [102]. Ici, une fonction de rectification non-paramétrique f_u (*cf.*4.6.5)

$$\tilde{\mathbf{p}}^u = f_u(\mathbf{p}^d)$$

peut être calculée lorsqu'une caméra effectue trois mouvements instantanés de rotation pure. La position de rectification $\mathbf{p}^u$ d'un pixel $\mathbf{p}^d$ est estimée à partir des vecteurs de flot instantanés $\mathbf{v}_i, i = 1, ..., 3$ situés à ce point pour chaque mouvement.

La base de la méthode est la relation suivante :

$$\mathsf{J}_{(3\times2)}\mathsf{V}_{(2\times3)} = -[\mathbf{p}^u]_\times$$

où, $\mathsf{V} = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3]$ contient les trois vecteurs de flots et $\mathsf{J} = \frac{\partial f_u(\mathbf{x})}{\partial \mathbf{p}^d}$, *i.e.* la matrice (inconnue) des dérivées partielles. Puisque $\mathbf{p}^u$ est dans le noyau droit de $-[\mathbf{p}^u]_\times$, il est aussi dans celui de V et donc aussi dans celui de JV. Par conséquent, on peut estimer $\mathbf{p}^u$ par le noyau droit de V.

8.3 Contributions

Le chapitre 9 porte sur une méthode d'auto-calibrage plan de distorsion radiale symétrique paramétrique ou non-paramétrique, à partir de deux vues et sans utiliser de correspondances denses. En effet, aucune méthode ne permet d'effectuer cette tâche : les méthodes de Thirthala et Pollefeys requiert au minimum trois ou quatre vues, et la méthode de Ramalingam et Sturm nécessite des correspondances denses.

La difficulté du problème provient du fait que l'estimation simultanée des paramètres de distorsion ainsi que de l'homographie reliant les deux vues rectifiées est un problème bilinéaire non-convexe. Nous proposons une approximation convexe sous forme d'un problème quadratique avec contraintes d'inégalité (*cf.*§§3.3.1 et 3.4.8).

Chapitre 9

PLANE-BASED SELF-CALIBRATION OF RADIAL DISTORTION

Cet article [153] a été publié comme l'indique la référence bibliographique.

Tardif J.P., Sturm P., Roy S. Plane based self-calibration of radial distortion. à paraître dans IEEE International Conference on Computer Vision (ICCV), Rio de Janeiro, Brésil, 2007.

Abstract

We present an algorithm for plane-based self-calibration of cameras with radially symmetric distortions given a set of sparse feature matches in at least two views. The projection function of such cameras can be seen as a projection with a pinhole camera, followed by a non-parametric displacement of the image points in the direction of the distortion center. The displacement is a function of the points' distance to the center. Thus, the generated distortion is radially symmetric. Regular cameras, fish-eyes as well as the most popular central catadioptric devices can be described by such a model.

Our approach recovers a distortion function consistent with all the views, or estimates one for each view if they are taken by different cameras. We consider a least squares algebraic solution for computing the homography between two views that is valid for rectified (undistorted) point correspondences. We observe that the terms of the function are bilinear in the unknowns of the homography and the distortion coefficient associated to each point. Our contribution is to approximate this non-convex

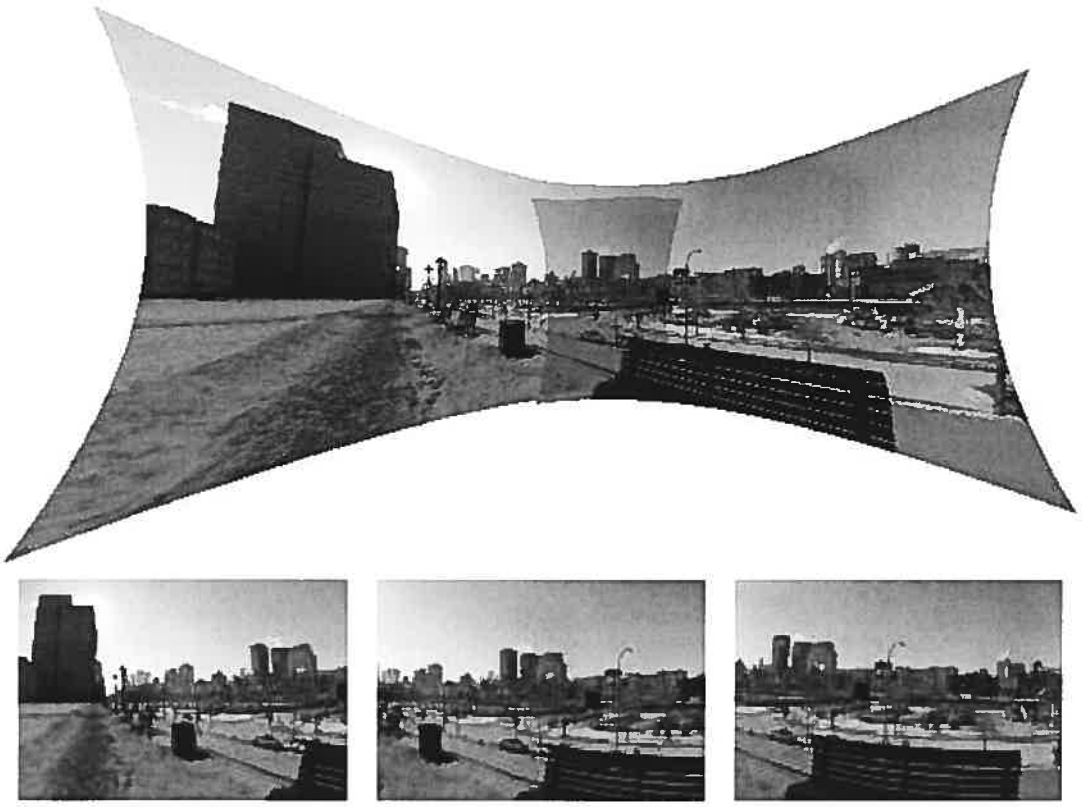


Figure 9.1. Top: **Planar stitching with automatic distortion correction. Red/dark overlay shows the mosaic's portion common to all images.** Bottom: **Original images taken with a wide-angle lens.**

problem by a convex one. To do so, we replace the bilinear terms by a set of new variables and obtain a linear least squares problem. We show that like the distortion coefficients, these variables are subject to monotonicity constraints. Thus, the approximate problem is a convex quadratic program. We show that solving it is sufficient for accurately estimating the distortion parameters. We validate our approach on simulated data as well as on fish-eye and catadioptric cameras. We also compare our solution to three state-of-the-art algorithms and show similar performance.

Cameras with general distortions include pinhole, fish-eyes and many single view-

point catadioptric devices. The projection function of such cameras can be seen as the projection from a (central) perspective camera, followed by a non-parametric displacement of the imaged point in the direction of the distortion center. This displacement induces a non-linear distortion of the image, *i.e.* straight lines in space are not imaged as straight lines unless passing through the distortion center. A very common assumption about the distortion is its radial symmetry [12, 39, 46, 61, 72, 114, 156, 157, 173]. It implies that the image displacement of a point is a function of its distance to the distortion center (irrespective of radial orientation). Two other assumptions are the alignment of the distortion center with the principal point and a unit aspect ratio. These allow an interpretation of the distortion function not only in terms of image displacement, but in terms of viewing angle with respect to the optical axis. It is an effective way of including cameras that have a field of view larger than 180°.

Problem statement. We present an approach for self-calibrating this general distortion model under the above assumptions. The algorithm uses sparse point matches from at least two views of a plane of unknown geometry, or equivalently, from cameras with coinciding optical centers observing a general scene. We assume *known distortion centers* for the considered cameras.

Organization. In the next section, we present the most important related work and discuss the main differences with ours. Then, the camera/distortion model we use is reviewed in §9.2. Our main contribution is given in §9.3 to §9.5. Our experiments and results are discussed in §9.7, followed by the conclusion in §9.8.

Notation. Matrices are in *sans-serif*, *e.g.* M and vectors in bold, *e.g.* $\mathbf{v}$. We use homogeneous coordinates unless otherwise stated; a bar indicates a vector containing affine coordinates, *e.g.* $\bar{\mathbf{p}}$.

9.1 Previous work

Traditionally, radial distortions have been treated as a lens aberration to be corrected in the image. Very wide angle lenses had limited applications because of the cameras' small image resolution. In recent years, this limitation has been overcome. It has led to different omnidirectional devices such as fish-eye and catadioptric cameras that can capture large portions of a scene with a high level of detail. In these cameras however, radial distortion is no longer an aberration, but the result of particular designs to increase their field of view.

For these new devices, new camera models are needed. They can be divided into two classes: parametric and non-parametric. Typically, parametric models have been designed for specific acquisition devices. Examples of such models are the classical polynomial model [132], the division model [39], the field of view model [36], the rational model [40], stereographic projection [41] and the unified catadioptric model [46]. A recent tendency has been to apply models originally designed for specific types of cameras to others. It was shown that the unified catadioptric model could also be applied to regular fish-eyes [12, 173] and that the polynomial division model could represent catadioptric cameras [152]. Non-parametric camera models take an opposite point of view. In their most general form, each pixel is associated to one sampling ray [56, 102, 113, 145]. Some researchers have also proposed compromises between parametric and fully general models [61, 72, 152]. The one we use fits into this category. It assumes radial symmetry around a distortion center, but no parametric function is used to describe the distortion.

Self-calibration of cameras is the problem of estimating the cameras' internal and external parameters without using objects of known geometry. One must make assumptions about the camera's internal parameters, *e.g.* constant parameters or unit aspect ratio, or about its external parameters, *e.g.* pure rotation or translation. This paper considers another common assumption, that the observed scene is planar,

which, in the context of our approach, is equivalent to seeing a general scene from a purely rotating camera. We show that a general radially symmetric distortion function can be estimated using two or more images. The proposed algorithms can be used in the context of 3D reconstruction and image stitching, *i.e.* mosaic building.

The characteristics of the most closely related self-calibration algorithms are summarized in table 9.1. We discuss the difference with the works closest to ours, that of Thirthala and Pollefeys [156] and that of Tardif *et al.* [152]. An empirical comparison is also given in §9.7. The model proposed in [156] will be discussed below. Thirthala and Pollefeys propose a tensor method to perform a projective 2D reconstruction of the scene. Given the reconstruction, the distortion parameters can be recovered. Feature matches in three images of a planar scene are required to compute a trifocal tensor. In [157], this approach is generalized to a general scene and even non-central cameras by using a quadrifocal tensor. In some applications, matching pixels between three views can be unwieldy. A typical situation is in mosaic stitching, as seen in figure 9.1, where the portion of the region common to all images is very small.

In [152], a plumbline approach is proposed. It can be extended to use dense point features by discovering collinear points. This is the most important weakness of the approach, making it difficult to refine the estimate of the distortion center. Furthermore, it is assumed that many points have (approximately) equal radii, i.e. distances from the distortion center, which simplifies the problem, or a polynomial division model is used directly.

9.2 Camera model

Our camera model follows the one of Tardif *et al.* [152] and Thirthala and Pollefeys [156]. The discrete representation of the distortion function is also related to the calibration algorithm of Hartley and Kang [61]. We give a brief summary of the geometry of such cameras. There are two complementary ways to describe their sampling

Table 9.1. Summary of different radial distortion self-calibration approaches. 'Para' refers to parametric distortion model, 'mixed' to algorithms that handle images with different distortions, 'dense' to methods requiring dense matches and $*$ indicates the methods included in our experimental comparison in §9.7.

References	non-para.	para.	mixed	dense
Barreto *et al.* [12]		×	×	
Claus *et al.* [40]		×		
Fitzgibbon [39]		×		
Geyer *et al.* [47]		×		
Sturm [144]		×	×	
Ramalingam *et al.* [114]	×			×
Tardif *et al.* [152]*	×	×	×	×
Thirthala *et al.* [156]*		×	×	
Thirthala *et al.* [157]*	×		×	
Zhang [180]		×	×	
Ours	×	×	×	

function in the case of radial symmetry and a single effective viewpoint. Keeping in mind both representations gives the intuition behind the algebraic derivation we use.

The description below refers to figure 9.2. The representation relies on the concept of **radial 1D camera** [156]. We consider one image point to which is associated a **radial line** $\mathbf{l}_i$ joining the point and the distortion center. Let us define the plane Π_i formed by this radial line and the optical axis of the camera. Since the deviation from the pinhole model occurs along the radial line, the sampling ray associated to the point must be located somewhere in this plane. The benefit of modeling the whole camera with a set of such sampling planes is that this circumvents the effect of the distortion of the camera. On the other hand, the distortion parameters (as well as the associated constraints like radial symmetry) cannot be recovered in a single step.

A second, more restrictive, description is possible in terms of **distortion circles** centered in the distortion center [152]. Each circle of radius r_j is associated to a right **viewing cone** C_j in space. These cones are centered in the optical axis and have the camera's optical center as vertex. The **distortion function** relates their

opening angle to the radius of the associated distortion circle. An equivalent and more convenient representation however is by using a single pinhole camera for each r_j. For a camera in canonical position, the projection function associated to a point (x, y) is given by:

$$\begin{bmatrix} f_r & 0 & c_x & 0 \\ 0 & f_r & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad r = \|(x - c_x, y - c_y)\|, \tag{9.1}$$

where (c_x, c_y) is the position of the distortion center. Thus, the image distortion is described as a dependency of the focal length on the radius, or distance to the distortion center. A zero focal length corresponds to a **viewing plane** Π_p, *i.e.* a cone with opening angle equal to 180°. A negative focal length models a backward looking cone such as C_3. It comes up when the viewing angle is larger than 180° like for some catadioptric or fish-eye cameras. Assuming the distortion center at the origin, the point can be "undistorted" by dividing it by f_r, or equivalently by letting $(x, y, f_r)^\mathsf{T}$ be its rectified homogeneous coordinates.[1] Note that for perspective rectification, it is sufficient to recover the set of f_r up to scale. For the most common cameras, f_r is a smooth monotonically decreasing function. This guarantees that the field of view is strictly monotonically increasing with respect to the radius. In the following, we will call the f_r sometimes focal lengths, sometimes simply distortion coefficients.

9.3 Projective point transfer

We are given a set of correspondences between two images. Our goal is to recover the distortion coefficient associated with the radius of each of the points of the two images. Once these are recovered, it is reasonable to assume that the distortion coefficients for other values of the radius can be computed by interpolation, *e.g.* using a polynomial.

[1] However we will refer to f_r as a distortion coefficient.

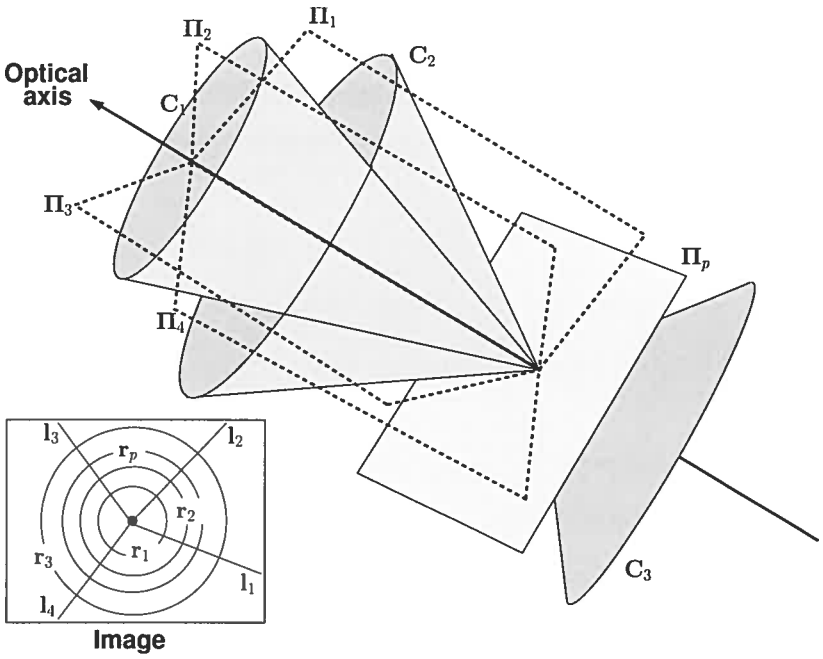


Figure 9.2. Illustration of our camera model. See text for details.

We assume that the distortion center is known. Thus, we can change the coordinate system of our images so the origin coincides with this center. If both cameras see an image of a plane, matching points $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$, once rectified, will obey the classical homography image transfer:

$$\begin{pmatrix} \bar{\mathbf{p}} \\ g \end{pmatrix} \propto \mathsf{H} \begin{pmatrix} \bar{\mathbf{q}} \\ f \end{pmatrix}, \text{ with } \bar{\mathbf{p}} = \begin{pmatrix} x \\ y \end{pmatrix} \text{ and } \bar{\mathbf{q}} = \begin{pmatrix} u \\ v \end{pmatrix}, \tag{9.2}$$

where g and f are the distortion coefficients associated to $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$. Note that this is valid even when f and/or g are zero or negative.

It is well known that two points of $\mathbb{P}^2$ are identical if their cross product vanishes. This is typically used to linearly estimate the homography between two views using point correspondences [60]. However, in our case, this yields two trilinear and one

bilinear equations in the elements of H and the distortion coefficients:

$$\begin{pmatrix} x & y & g \end{pmatrix}^{\mathsf{T}} \times \left(\mathsf{H} \begin{pmatrix} u & v & f \end{pmatrix}^{\mathsf{T}} \right) = \begin{pmatrix} uyh_{31} + vyh_{32} + fyh_{33} - guh_{21} - gvh_{22} - fgh_{23} \\ guh_{11} + gvh_{12} + fgh_{13} - uxh_{31} - vxh_{32} - fxh_{33} \\ uxh_{21} + vxh_{22} + fxh_{23} - uyh_{11} - vyh_{12} - fyh_{13} \end{pmatrix} = \mathbf{0}. \tag{9.3}$$

For reasons that will become clear shortly, we use only the third equation. This is equivalent to considering the first camera as a radial 1D and effectively eliminating the distortion coefficient g. Indeed, we can define the radial line $\mathbf{l} = (0, 0, 1)^{\mathsf{T}} \times (x, y, 1)^{\mathsf{T}} = (-y, x, 0)^{\mathsf{T}}$ and verify that $\left(\bar{\mathbf{p}}^{\mathsf{T}}, f\right) \mathbf{l} = (x, y, g)(-y, x, 0)^{\mathsf{T}} = 0$. Hence, we have:

$$\mathbf{l}^{\mathsf{T}} \begin{pmatrix} \bar{\mathbf{p}} \\ g \end{pmatrix} = \mathbf{l}^{\mathsf{T}} \mathsf{H} \begin{pmatrix} \bar{\mathbf{q}} \\ f \end{pmatrix} = uxh_{21} + vxh_{22} + fxh_{23} - uyh_{11} - vyh_{12} - fyh_{13} = 0. \tag{9.4}$$

Note that g does not appear in the equation anymore. Neither do many parameters of H. This is not critical however, since recovering the homography between the two views can be done *a posteriori*. One could formulate self-calibration as a least squares problem, *i.e.* as the minimization of the sum of squares of the term (9.4) over the available point correspondences. This is a non-convex problem since (9.4) is bilinear in f, h_{13} and h_{23}. In the next section, we show how this sum of squares can be approximated by a convex quadratic program with inequality constraints.

9.4 A convex approximation

In this section, we describe our approximation scheme and provide some theoretical insight to justify our approach in §9.4.1. We assume that both h_{13} and h_{23} are non-zero and fix the scale of H by setting $h_{13} = 1$. Degenerate cases are discussed in

§9.4.2. Thus, (9.4) simplifies to:

$$uxh_{21} + vxh_{22} + fxh_{23} - uyh_{11} - vyh_{12} - fy = 0. \tag{9.5}$$

Let us replace the only remaining bilinear term fh_{23} by a new variable: $fh_{23} \rightarrow \alpha$. This gives:

$$uxh_{21} + vxh_{22} + x\alpha - uyh_{11} - vyh_{12} - fy = 0, \tag{9.6}$$

$$\text{subject to } \alpha = fh_{23}. \tag{9.7}$$

With n point correspondences, we get n linear equations and n bilinear constraints involving $h_{11}, h_{12}, h_{21}, h_{22}$ and[2] $f_i, \alpha_i, i...n$. So far, we thus have a linear least squares problem with bilinear constraints, which is still non-convex. Our approximation is to replace the bilinear constraints with monotonicity constraints on the f_i and α_i. Let us reorder our correspondence indices i so the $\bar{\mathbf{q}}_i$ are in ascending order of their distance to the origin. We observe that the monotonicity constraint on the f_i also applies to the α_i since they are equal to the f_i up to a scale h_{23}. Since the sign of h_{23} is unknown, we have either $\alpha_i \geq \alpha_{i+1}$ or $\alpha_i \leq \alpha_{i+1}$ for all i. Combining all equations

[2] To simplify notations, i in f_i is a point index and not the radius as in f_r above.

in matrix form, we get:

$$\underbrace{\begin{bmatrix} y_1\bar{\mathbf{q}}_1^\top & -x_1\bar{\mathbf{q}}_1^\top & \bar{\mathbf{p}}_1^\top\mathsf{D} & & \\ \vdots & \vdots & & \ddots & \\ y_n\bar{\mathbf{q}}_n^\top & -x_n\bar{\mathbf{q}}_n^\top & & & \bar{\mathbf{p}}_n^\top\mathsf{D} \end{bmatrix}}_{\mathsf{A}} \underbrace{\begin{pmatrix} h_{11} \\ h_{12} \\ h_{21} \\ h_{22} \\ \alpha_1 \\ f_1 \\ \vdots \\ \alpha_n \\ f_n \end{pmatrix}}_{\mathbf{x}} = \mathbf{0} \tag{9.8}$$

$$\begin{aligned} \text{subject to} \quad & f_1 \geq f_2 \geq \ldots \geq f_n \\ \text{and either} \quad & \alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_n \\ \text{or} \quad & \alpha_1 \leq \alpha_2 \leq \ldots \leq \alpha_n \end{aligned} \tag{9.9}$$

where $\mathsf{D} = \mathrm{diag}(-1, 1)$. We assume the general case of sparse matches, where none of the points $\bar{\mathbf{q}}_i$ share the same radius. Consequently, A has more columns than rows and (9.8) is underconstrained. However, enforcing the monotonicity constraints provides sufficient constraints as explained in §9.4.1. We compute:

$$\min_{\mathbf{x}} \|\mathsf{A}\mathbf{x}\|^2 \tag{9.10}$$

subject to (9.9) and $f_j = 1$.

The index j is a choice from any of the n points. The constraint $f_j = 1$ avoids the trivial solution $\mathbf{x} = \mathbf{0}$ and also fixes the overall scale of the distortion coefficients.[3] The system (9.10) represents a sparse convex quadratic program since $\mathsf{A}^\mathsf{T}\mathsf{A}$ is positive semi-definite. The optimization of this problem is relatively easy using a modern numerical package. We used the Matlab CVX interface [52] to SeDuMi [140] which implements a sparse interior point method. The choice between $\alpha_i \geq \alpha_{i+1}$ and $\alpha_i \leq \alpha_{i+1}$ depends on the sign of h_{23}. Naturally, it is not known *a priori*. We thus minimize both systems and keep the solution giving the smallest residual error.

In general, solving this problem yields a satisfying distortion function. However, the constraints in (9.7) are not enforced and, under noise, each α_i/f_i will give a slightly different value. One can estimate h_{23} as the average of these ratios. However, we prefer to use the median as it is more robust to errors. Once h_{23} is estimated, (9.5) becomes linear in the f_i and the other entries of H. We can re-estimate them with a system similar to (9.10) with inequality constraints for the f_i only.

One could perform non-linear optimization using (9.3) or another meaningful geometric error such as the reprojection error. Note however that the distortion function is not invertible if the distortion coefficients are (close) to zero. In this case, the reprojection error in the original images cannot be computed and we are stuck with using the rectified pixel coordinates. It is preferable to use the projective angle between a point and its transferred point from the other image to perform the optimization.

Finally, full self-calibration of the camera can be done with already known techniques for plane-based self-calibration [163] or for a purely rotating camera [93, 129]. Note that the recovered principal point of the camera need not be identical to the distortion center. Then, a full 3D reconstruction of the points on the plane (or the plane at infinity) can be performed, followed by Euclidean bundle adjustment.

[3] Two views of a plane do not allow self-calibration of the 'absolute' focal length.

9.4.1 Justification

Let us first consider the case where some of the considered points have the same radius. In this case, the number of variables f_i and α_i goes down. With enough points with equal radius, the system becomes overconstrained. In the absence of noise, its solution is clearly the one we are seeking, and as such, it satisfies the constraints (9.7). With noise, points with equal radius trivially have the same ratio α/f.

In practice, interest points may have similar radius, but usually never exactly identical ones. With the knowledge that the f is smooth and monotonically decreasing, adding constraints (9.9) provides a reasonable approximation to the above overconstrained situation.

9.4.2 Degenerate cases

A first degenerate case occurs when either h_{13} or h_{23} are (very close to) zero. In this case (9.5) simplifies to an equation linear in f and the parameters of H. We now discuss the case where both h_{13} and h_{23} are zero. A first occurrence of this degeneracy is when the camera performs a *pure rotation* around its optical axis. Note, although the distortion is observable in the image, it is not in terms of point transfer. That is, an homography (precisely an image rotation) may satisfy the point transfer for unrectified images.

A more interesting case occurs in *plane-based* self-calibration. In this case, $h_{13} = h_{23} = 0$ implies that $\begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\mathsf{T} \propto \mathsf{H}\begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\mathsf{T}$ i.e. that the centers of distortion are matching points with respect to the plane homography. Hence, the optical axes intersect in a point on the scene plane. The converse is also true: if the optical axes intersect in the scene plane then $h_{13} = h_{23} = 0$. One observes that (9.5) simplifies to a linear relation in the upper-left 2×2 elements of H. This implies that H can be estimated up to 3 degrees of freedom. However, it can be shown that no constraint

can be obtained on f and g, thus self-calibration is not possible in this case.

9.5 Regularization

9.5.1 Intervals for monotonicity constraints

Under noise, the monotonicity constraints of (9.9) can result in instability for the optimization problem. Intuitively, this happens when the distance between two points, say $\bar{\mathbf{q}}_i$ and $\bar{\mathbf{q}}_j$, is smaller than the noise in their coordinates. The effect of the monotonicity constraints on the results is that "stairs" can appear for f_i and α_i. We thus examined different regularization schemes. For instance, replacing $f_i \geq f_j$ with $f_i \geq f_j - \beta_i, \beta_i \geq 0$ and minimizing these new variables as part of the problem. But this did not resolve the issue and increased the computational burden.

We propose a simpler solution: the idea is to replace the hard monotonicity constraints with interval constraints. The intervals are defined using points with closest absolute difference of radius above a certain threshold ϵ. Formally, the interval for the coefficient f_i corresponding to point $\bar{\mathbf{q}}_i$ is $f_k \geq f_i \geq f_j$, with k the largest index such that $\|\bar{\mathbf{q}}_i\| - \|\bar{\mathbf{q}}_k\| \geq \epsilon$ and with j the smallest index such that $\|\bar{\mathbf{q}}_j\| - \|\bar{\mathbf{q}}_i\| \geq \epsilon$. The same is applied to the α_i. A rule of thumb for selecting ϵ is to set it larger than the maximal error of the point transfer. In practice, we set it to 10 image pixels in all our tests.

9.5.2 Polynomials and robust computation

Our approach can be easily modified to directly fit a parametric distortion function instead of recovering a general one. Our tests suggest however that doing so is not as accurate as performing the fitting of the model on the recovered f_i in a second step (see results in §9.7). Nevertheless, the computation time is significantly reduced, which proves very useful as explained below.

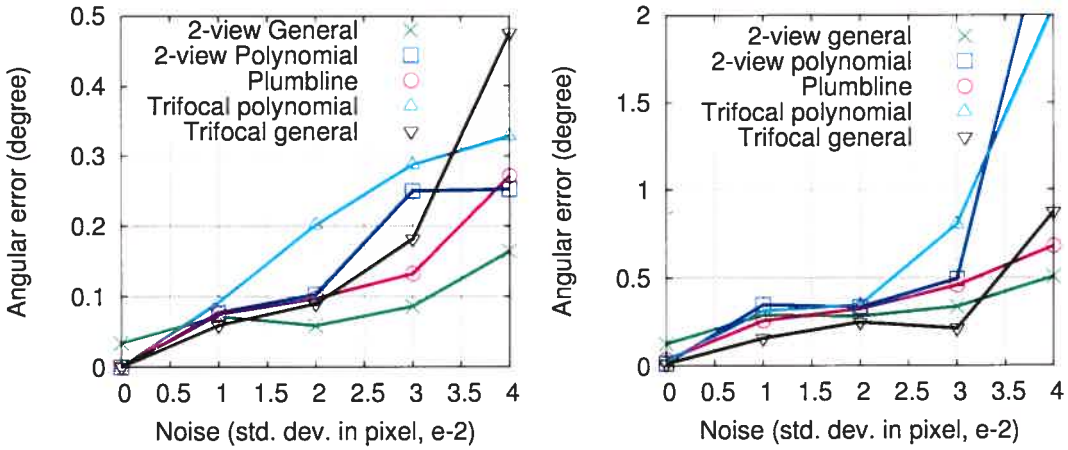
The parametric function can take any form as long as it is linear in its parameters.

A typical example is a polynomial. In this case, the relaxation is performed by replacing f_i and α_i by two polynomials $f(r_i) = 1 + \sum_{j=2} \lambda_j r_i^j$ and $\alpha(r_i) = \gamma_0 + \sum_{j=2} \gamma_j r_i^j$ with $r_i = \|\bar{\mathbf{q}}_i\|$ and modify (9.8) accordingly. Constraint (9.7) requires that the coefficients of the two polynomials be equal up to a scaling factor h_{23}. Similarly as before, this is replaced by monotonicity constraints on both polynomials with respect to the radii of the considered image points. Thus, using polynomials also implies solving a convex quadratic program. Once again, one can estimate h_{23} as the median of the ratios $\alpha(r_i)/f(r_i)$ at every point and use it to re-estimate $f(r_i)$ and $h_{11}, h_{12}, h_{21}, h_{22}$.

A very important difference with using a discrete function is that $\mathsf{A}^\mathsf{T}\mathsf{A}$ is positive definite. Dropping the constraints and given a sufficient number of points, we obtain an over-constrained linear least square problem. In fact, without noise and given that the model is appropriate, the minimum of this problem will automatically respect the constraints. It is natural to ask whether solving this linear problem would also be sufficient under noise. Our tests suggest that it is indeed a reasonable compromise that reduces significantly the computation time. We could successfully use this fast approximation inside a robust algorithm based on random sampling, *e.g.* RANSAC.

9.6 The case of multiple views

When many image pairs between two different cameras are available or when both images were taken from the same one, it is useful to estimate all 2-view relations in a single problem. The benefit is that the distortion coefficients of the features from all images can be combined. To do so, one sorts the distortion coefficients of all the views and applies constraints such as in (9.9) or intervals as explained in §9.5.1. Let $\mathsf{H}^1, ..., \mathsf{H}^V$ be the homographies. The only issue concerns the sign of the h_{23}^v which induces a choice of constraints in (9.9). Remind that for one image pair, both positive and negative values of h_{23}^v must be tested. With V relations, this yields 2^V



(a) Fish-eye (b) Catadioptric

Figure 9.3. Comparison of the algorithms with respect to noise.

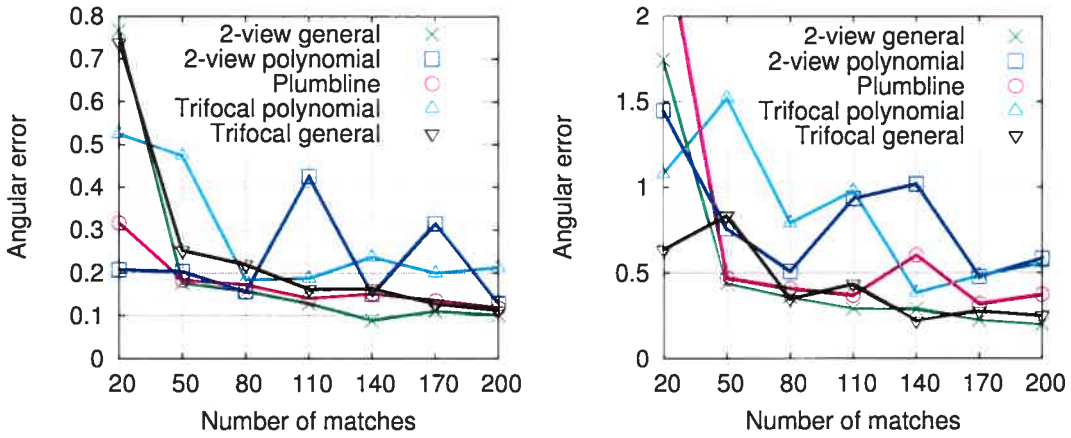
minimization to solve. A simpler method is to individually solve each homography in order to get the sign of h^v_{23}. The joint minimization is solved only once with the obtained signs.

9.7 Experiments

We compared our approach on simulated datasets with both Thirthala-Pollefeys trifocal tensor methods [156, 157] (referred to as 'trifocal polynomial' and 'trifocal general') and the plumbline method of Tardif *et al.* [152]. For our method, we recovered both (discretely sampled) general distortion functions and polynomials of degree four. In the case of the general function, the interpolation was done by fitting such a polynomial to the estimated f_i.

9.7.1 Simulation

We simulated several fish-eye and catadioptric lenses. Their distortion functions were randomly generated with monotonically decreasing polynomials. Catadioptric cameras had larger distortions than the fish-eyes and we also made sure their viewing



(a) Fish-eye (b) Catadioptric

Figure 9.4. Comparison with respect to the number of features.

angle was larger than 180°. Note that the tested algorithms take slightly different input. For our methods and the trifocal tensor based ones, we generated respectively 2-view and 3-view correspondences. The data for the plumbline method was generated from the projection of points located on 3D lines. After the distortion function was recovered, we assumed that the other internal parameters could be recovered with a conventional self-calibration algorithm. The error was computed by measuring the angle between the real back-projection rays and the one from the estimated models, for 100 newly generated image points.

Two experiments tested the sensitivity to noise and to the size of the dataset. The first one, illustrated in figure 9.3, used 100 features per view (2-view or 3-view correspondences, or points from 3D lines), with added Gaussian noise of varying level. The second one, illustrated in figure 9.4, used a fixed noise level of $\sigma = 3 \times 10^{-2}$ with different number of features. Note that larger error for catadioptric cameras is expected since their field of view is larger. Our method provides accurate and stable results and even outperforms the others in many tests, although using only two views. A result of our approximation scheme is that the error is not exactly zero in the absence of noise. However, it remains very low even with high noise. Another im-

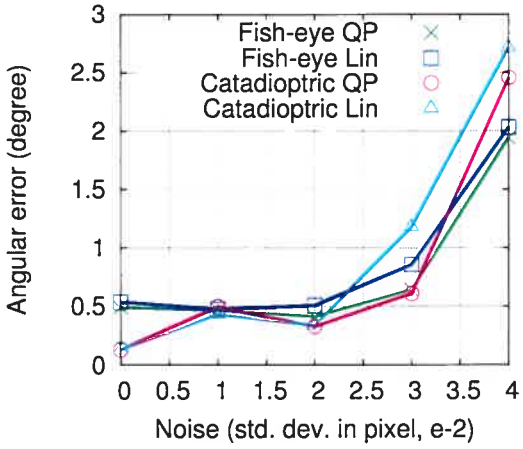


Figure 9.5. Computation using a polynomial function and the minimum number of required matches. Comparison between incorporating monotonicity constraints (QP) or not (Lin) for a fish-eye lens and a catadioptric device.

portant conclusion is that solving a general distortion function is better than directly recovering the polynomial, except when only a small number of correspondences are available.

Another experiment refers to §9.5.2: we compared the linear and constrained quadratic formulation for solving a third degree polynomial distortion function using 9 image correspondences (the minimal requirement being $8\frac{1}{2}$). Our results, illustrated in figure 9.5, strongly suggest that this approximation does not substantially worsen the results.

9.7.2 Real images

Several real cameras were tested. In each case, between 400 and 700 correspondences were automatically found (except in one case) between two images and outliers were removed with RANSAC using the linear formulation for estimating a third degree polynomial.

Figure 9.1 shows a 3-view mosaic built using a 15mm wide angle lens mounted

on a Canon SLR. Pairwise correspondences were extracted and a global distortion function was recovered by estimating the homographies relating the views in both directions.

We also calibrated three omnidirectional cameras: a Nikon 8mm fish-eye lens mounted on Nikon Coolpix camera, and two different catadioptric cameras. In all three cases, two images under pure rotation were used to recover the distortion function, followed by Euclidean upgrade using the method proposed in [129]. The recovered distortion functions of the first two cameras are shown in figure 9.6. In all cases, the two images could be combined together in undistorted cubemaps (*cf*.figures 9.7, 9.8 and 9.9). Non-linear optimization was not required to obtain very good results on image rectification, but was used for accurate stitching of the images.

Finally, our second catadioptric camera was also fully calibrated from two images of a plane (*cf*.figure 9.10). In this case, we had to manually remove the correspondences outside of the plane since the camera displacement was not large enough to disambiguate perfectly between pure rotation and planar self-calibration. Image rectification is very good, but not as good as in the pure rotation case. Indeed, features from the plane can cover at most half of each image and are more difficult to obtain near borders.

9.8 Conclusion

We have demonstrated a practical solution to plane-based self-calibration of radially symmetric distortion. Our contribution is that it can use sparse point matches from only two views. Our method provides accurate and stable results based on a convex approximation of an initially non-convex problem.

Our future work will be focused on applying this idea to perform two-view self-calibration under general motion and scene structure.

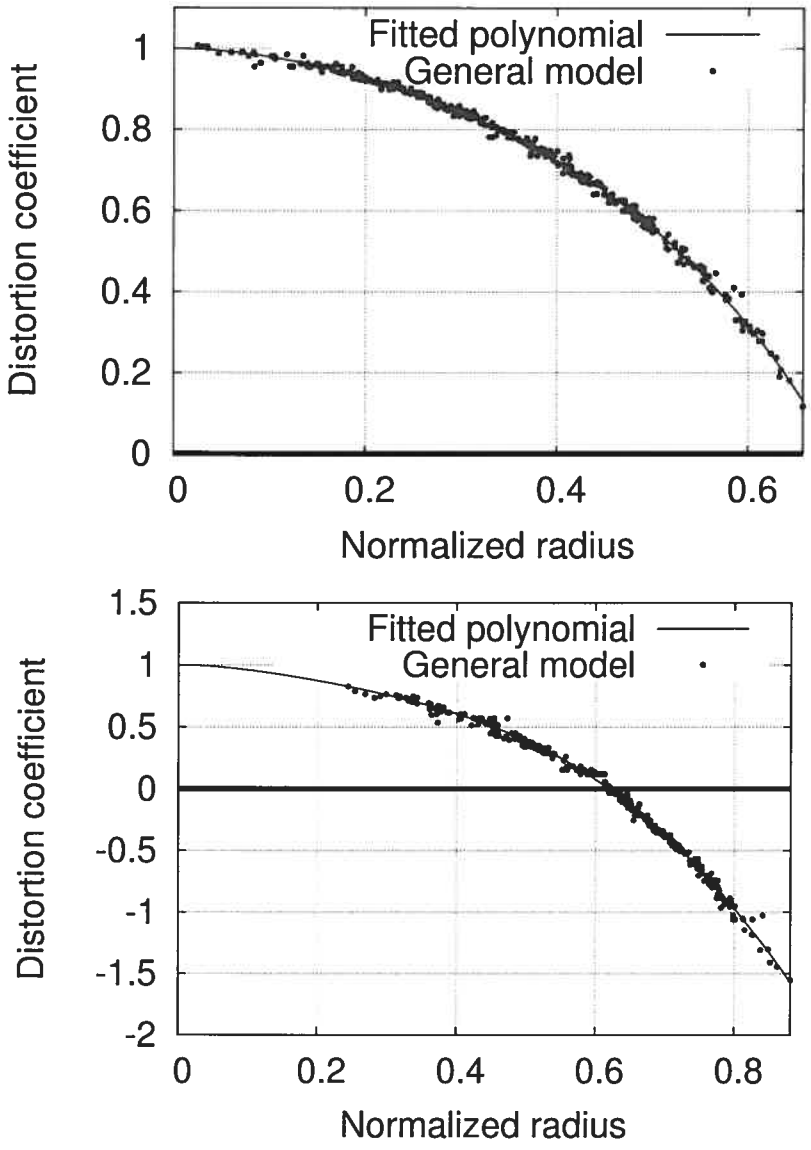


Figure 9.6. Distortion function for left: fish-eye (see also figure 9.7) right: first catadioptric camera (see also figure 9.8).

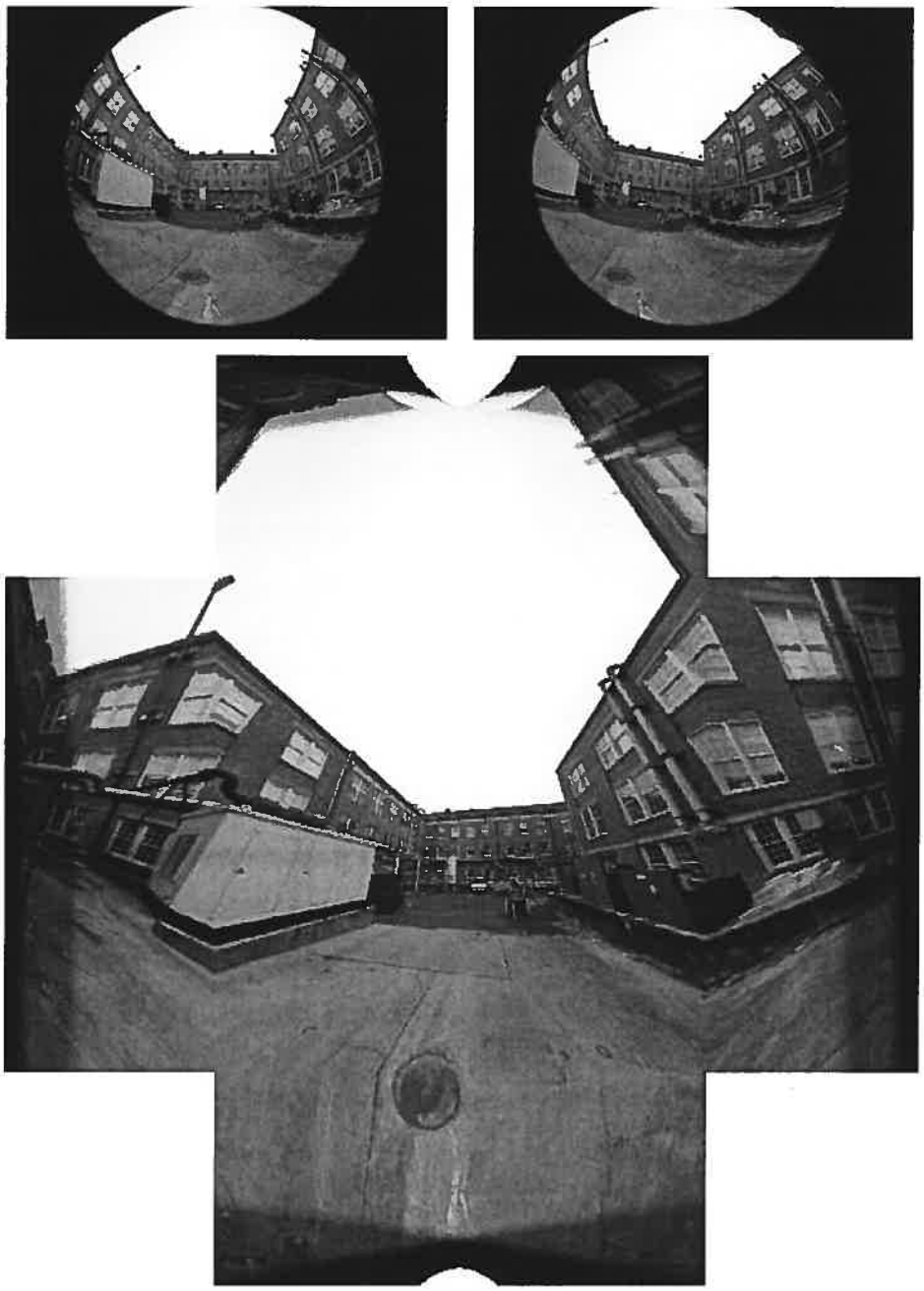


Figure 9.7. Fish-eye images. Top: **Two of the images under pure rotation.** Bottom: **The images combined together in a cubemap.**

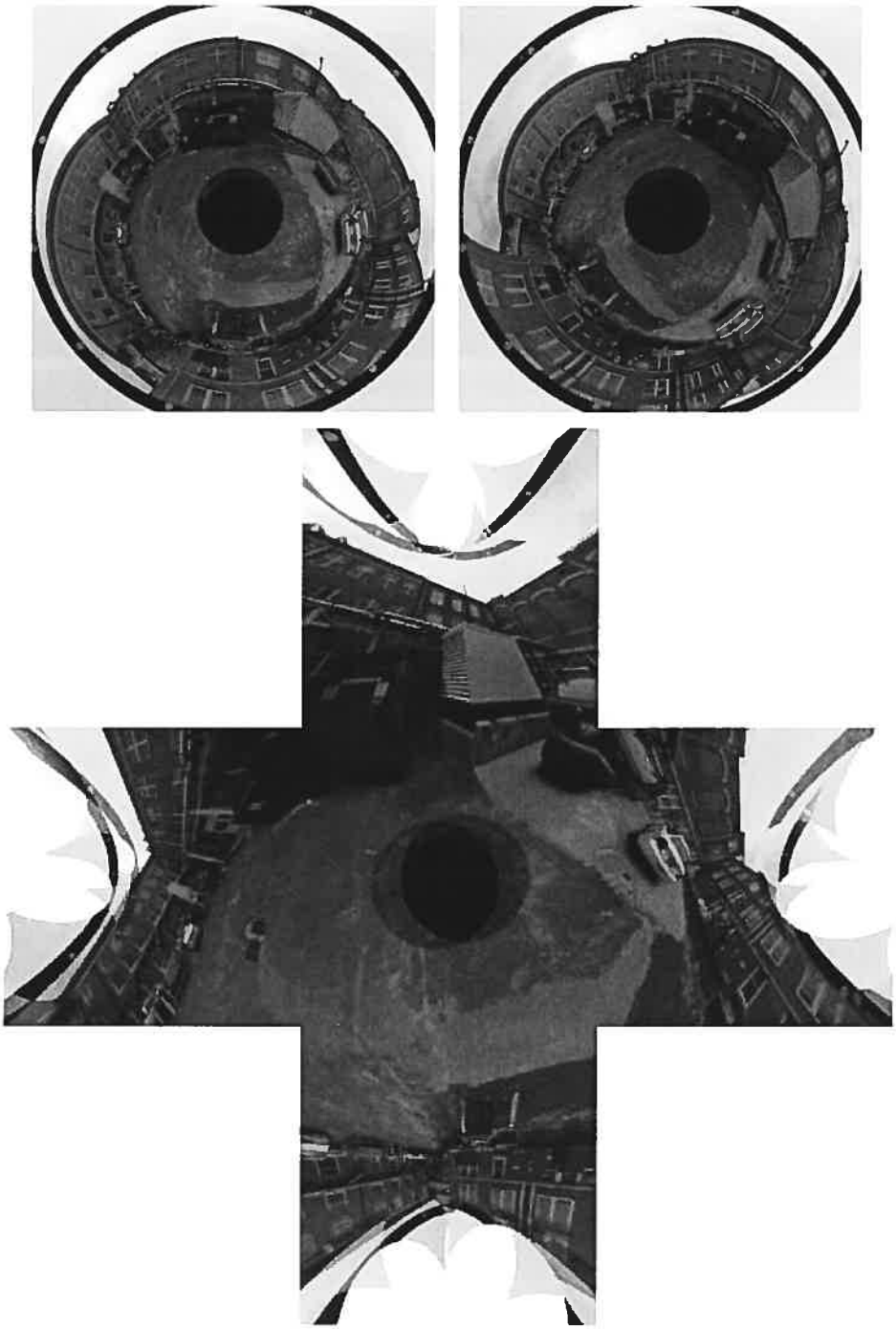


Figure 9.8. First catadioptric camera. Top: Two images under pure rotation. Bottom: The images combined together in a cubemap.

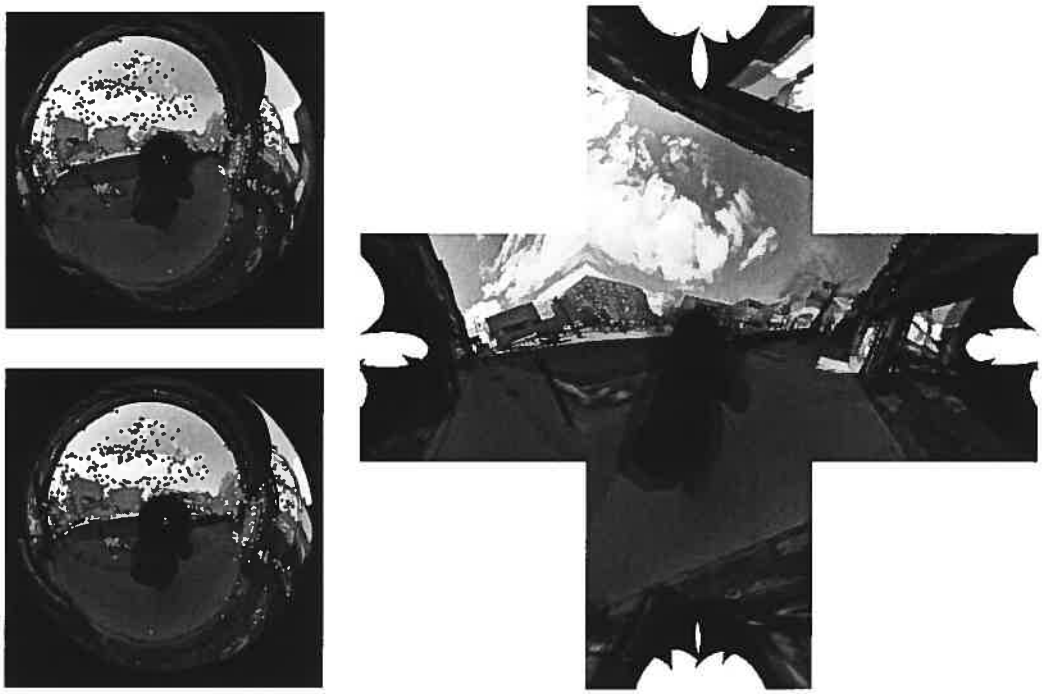


Figure 9.9. Second catadioptric camera. Top: **two images under pure rotation.** Bottom: **The images combined together in a cubemap.**

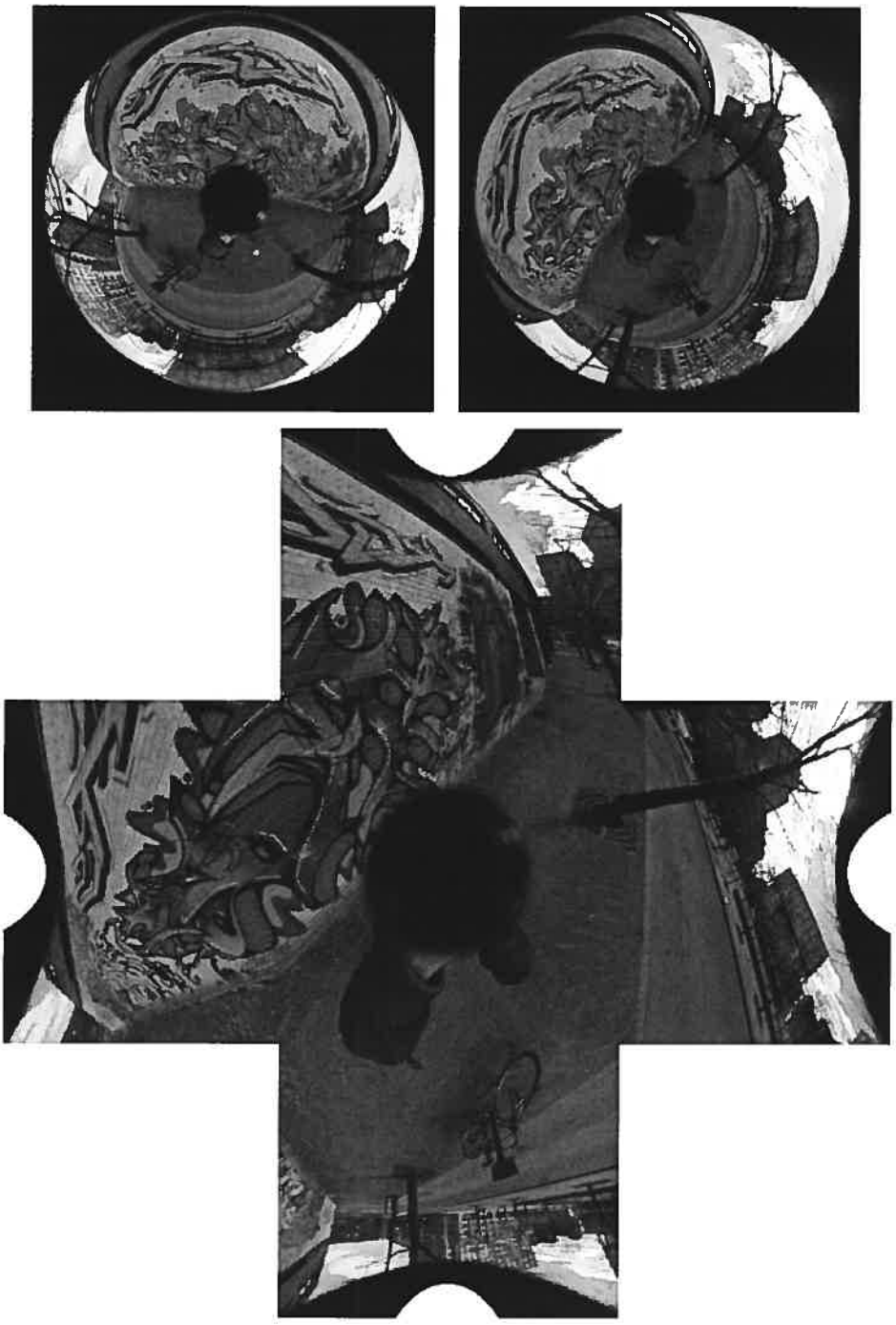


Figure 9.10. Plane-based self-calibration of the second catadioptric camera. Top: **The original images.** Bottom: **Right image rectified.**

Chapitre 10

MÉTHODES DE MISE EN CORRESPONDANCES DENSES PAR LUMIÈRE STRUCTURÉE

10.1 Introduction

La vision stéréo est l'une des méthodes parmi les plus populaires pour effectuer des reconstructions 3D. Elle est basée sur l'utilisation d'au moins deux caméras entre lesquelles des correspondances denses (la projection d'un même point 3D dans les images) sont obtenues. Pour chaque correspondance, il est possible d'effectuer une triangulation : les rayons d'échantillonage dans chaque caméra sont intersectés pour obtenir une position 3D. La reconstruction est donc possible lorsque les caméras sont calibrées, et lorsque le problème de mise en correspondance est résolu. Établir des correspondances (de façon passive) entre deux images ou plus est un problème difficile, entre autres, en raison des occultations, *i.e.* lorsqu'un point n'est pas vu dans une image, ou encore en raison du manque de texture à la surface de l'objet. Par conséquent, les méthodes nécessitent une certaine forme de régularisation résultant en des problèmes d'optimisation qui sont difficiles à résoudre [126].

La reconstruction 3D active permet de remédier à cette difficulté en remplaçant une des caméras par un projecteur multimédia. Ce dernier projette des motifs sur la scène, permettant d'encoder la position de chacun des pixels de son image. Observés dans les images de la caméra, ces motifs doivent être décodés pour permettre de retrouver automatiquement la correspondance caméra–projecteur (figure 10.1). Dans cette méthode, le projecteur agit comme une caméra, permettant la triangulation des points 3D.

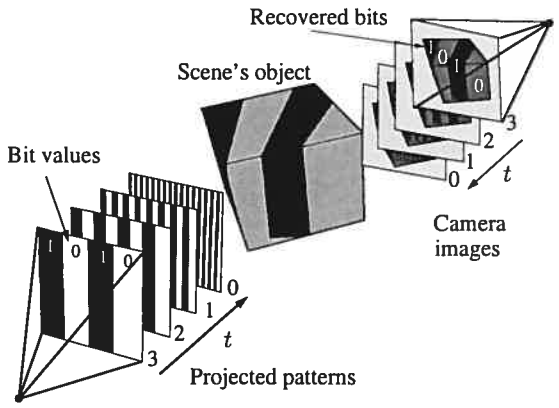


FIG. **10.1. Principe de mise en correspondances denses par lumière structurée. Image tirée de [150].**

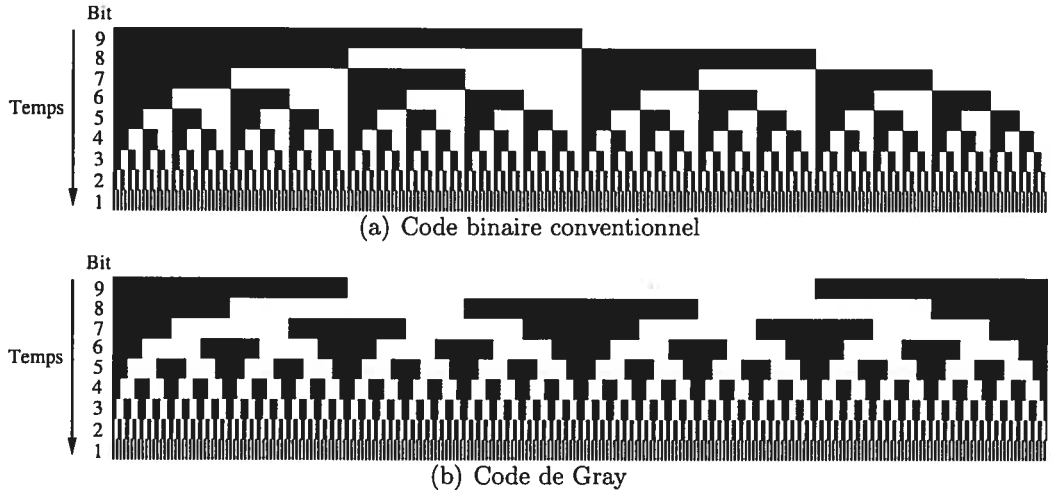
Organisation

Nous présentons, §10.2, quelques approches de mise en correspondances denses à l'aide de la lumière structurée. Nous discutons ensuite, §10.3, d'une application au problème de correction d'images dans un système multi-projecteurs. Finalement, nous donnons un aperçu des nos contributions, présentées en détails au chapitre 11.

10.2 Méthodes de mise en correspondances denses par lumière structurée

Une revue exhaustive de ce sujet est donnée par Salvi *et al.* [123]. Nous en donnerons ici qu'un bref aperçu.

En pratique, la triangulation d'un point 3D ne nécessite que la correspondance entre un point de l'image de la caméra avec une ligne du projecteur (en s'assurant que l'orientation de la ligne est bien choisie) : cette dernière est donnée par l'intersection du rayon d'échantillonage de la caméra avec le plan passant par le centre optique du projecteur et la ligne de l'image. Pour cette raison, le problème est souvent simplifié à une mise en correspondance 1D. Dans ce qui suit, nous utilisons le terme *étiquettes*



(a) Code binaire conventionnel

(b) Code de Gray

FIG. **10.2. Deux types de codage temporel sur neuf bits.**

pour parler des couleurs ou degré d'intensité que chaque couleur peut prendre.

10.2.1 Méthodes temporelles

Les méthodes temporelles consistent en la projection d'une séquence de motifs, chacun d'eux encodant une partie des coordonnées du projecteur. Elles font donc l'hypothèse que la scène reconstruite est statique. Les deux types de motif les plus courants sont :

▷ **Code binaire**

Ils consistent en deux couleurs ou étiquettes au maximum, typiquement noir et blanc, chacune d'elle interprétée comme la valeur de 0 ou 1. Dans le cas le plus simple, le nombre de motifs est donné par le nombre de bits nécessaires pour représenter les coordonnées des pixels. Dans chaque motif i, la couleur des pixels est choisie en fonction de la valeur du bit i de leur coordonnée. Comme illustré à la figure 10.2a, ceci résulte en des motifs de bandes de plus en plus fines pour les bits de poids faible.

Pour chaque image d'un motif prise par la caméra, il faut effectuer une *bina-*

FIG. **10.3. Un motif constitué d'un profil gaussien, typiquement utilisé dans les méthodes à décalage de motif.**

risation, *i.e.* distinguer les pixels blancs des noirs à l'aide d'un ou plusieurs seuils. Ce processus permet la mise en correspondance avec une précision de l'ordre du pixel. Une plus grande précision est toutefois possible en analysant le profil du passage d'une couleur à l'autre, permettant de détecter la bordure des bandes avec une précision sous-pixel. La binarisation est plus difficile à mesure que l'image se dégrade, à cause de la présence de bruit ou d'une diminution de contraste. Cet effet est encore plus important dans les régions de l'image correspondant au passage d'une couleur à l'autre. Tel qu'expliqué §11.3, les codes binaires conventionnels sont assez sensibles à ce phénomène et il est préférable d'utiliser un type d'encodage appelé code de Gray, illustré à la figure 10.2b [66].

▷ **Décalage de motif**[1]

Ce type de méthode consiste à utiliser une série de motifs (*e.g.* figure 10.3) identiques mais décalés d'un certain nombre de pixels les uns par rapport aux autres. Cette technique permet de retrouver les correspondances avec une précision sous-pixel. Elle ne permet cependant que de retrouver leur bits de poids faibles. Pour récupérer les autres, il est possible d'utiliser la cohérence spatiale des correspondances. Une autre approche consiste à les récupérer en utilisant d'autres motifs comme les codes de Gray.

10.2.2 Méthodes spatiales

L'objectif des méthodes spatiales est d'utiliser un seul motif, permettant l'acquisition d'objets déformés ou en mouvement au cours d'une séquence d'images. À

[1] *Phase shift*

FIG. **10.4. Motif de couleurs basé sur une séquence de De Bruijn.**

l'opposé des méthodes temporelles, celles-ci supposent que la surface des objets est relativement lisse. Pour être robuste, elles doivent utiliser le moins d'étiquettes possibles. Pour cette raison, le code de chaque pixel est déterminé par son étiquette, mais aussi celle de ses voisins. Voici deux types de code fréquemment utilisé.

▷ **Codification de De Bruijn**

Une k-séquence de De Bruijn d'ordre n est une séquence de nombres $d_0, ..., d_{k^n-1}$ où chaque élément est pris parmi un choix de k étiquettes, et pour laquelle toute sous-séquence de longueur n n'apparaît qu'une seule fois. Par exemple, une séquence avec $k = 5$ et $n = 3$ est [121] :

$$00010020030040110120130140210220230240310203303404104204304411$$
$$12113114122123124132133134142143144222322423323424324433343444. \quad (10.1)$$

Ce type de séquence permet de construire un motif de bandes. Idéalement, le choix de couleur associée à chaque étiquette devrait faire en sorte qu'elles sont faciles à distinguer les unes par rapport aux autres [106, 182]. Un exemple basé sur la séquence (10.1) est donné à la figure 10.4. Le décodage consiste donc à analyser la couleur du pixel et les $(k-1)$ couleurs retrouvées dans ces pixels voisins.

▷ **M-array**

Un *M-array* est une matrice M où la valeur de chaque élément est choisie parmi les étiquettes 0 à $k-1$, et qui a les propriétés voulant que :

- chaque sous-matrice de taille $(n \times m)$ n'apparaît qu'une seule fois ;
- toute matrice de taille $(n \times m)$, sauf la matrice $0_{(n \times m)}$ est une sous-matrice de M.

FIG. **10.5. Bande horizontale avec encodage direct basé sur l'intensité. Chaque bande est d'intensité différente.**

Elles sont très similaires aux séquences de De Bruijn, mais en deux dimensions, et sont plus difficiles à générer [96]. Un M-*array* peut donc être utilisé pour générer une grille ou un motif de points, où la couleur de chaque élément est choisie en fonction de la valeur correspondante dans la matrice.

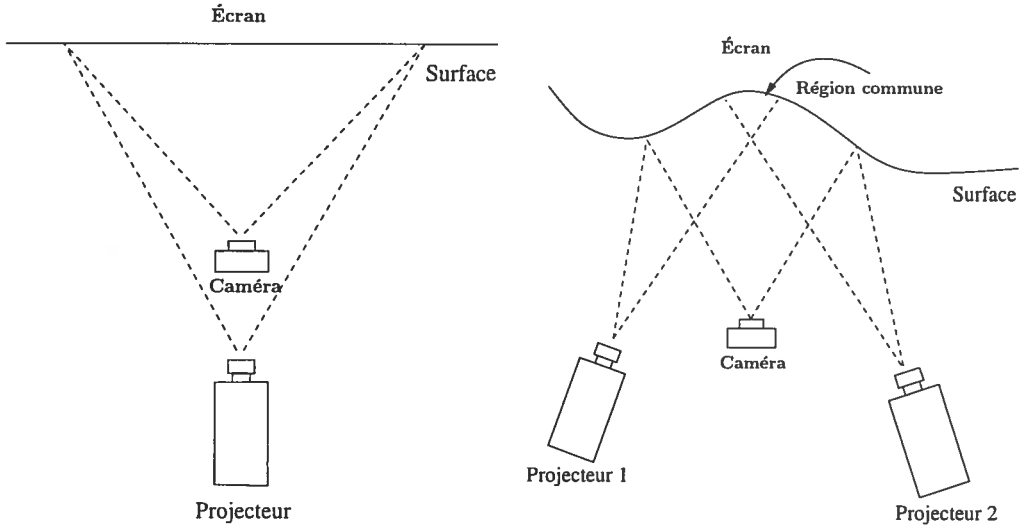
10.2.3 Méthodes directes

L'encodage direct ne nécessite aucune information spatiale ou temporelle. Il est particulièrement bien adapté aux scènes dynamiques avec des discontinuités. Les coordonnées d'un pixel du projecteur sont encodées de façon unique à partir de sa couleur ou intensité, comme illustré à la figure 10.5. Il y a donc autant d'étiquettes que de codes. Par conséquent, ces méthodes sont sujettes à plusieurs difficultés. Elles sont très sensibles au bruit et requiert un calibrage très précis des courbes de réponse du projecteur et de la caméra. De plus, leurs applications sont surtout limitées aux objets très pâles.

10.3 Multi-projecteurs

Nos contributions du chapitre 11 sont grandement reliées au problème d'alignement d'images dans un système multi-projecteurs. Comme ce sujet est hors du propos de ce document, nous ne donnons ici qu'une très brève description des problématiques rencontrées dans ce type de système.

La plupart des systèmes multi-projecteurs servent à afficher des images de très haute résolution sur une surface planaire ou courbée. Chaque projecteur couvre une partie de l'écran et les bordures de leurs images sont généralement superposées. Pour



(a) Montage idéal (b) Système multi-projecteurs

FIG. **10.6. Gauche) Configuration observateur–écran-projecteur idéale, droit) configuration réelle nécessitant l'alignement d'images.**

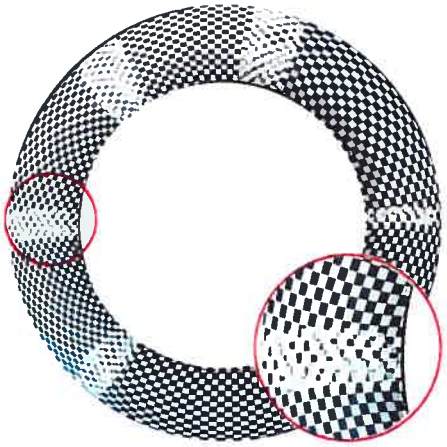
projeter une certaine image sur l'écran, il faut donc que chaque projecteur connaisse la partie de l'image qu'il doit projeter et comment la transformer pour qu'aux yeux des observateurs, l'image apparaisse sans déformation. De plus, il faut que l'intensité de chaque pixel de l'image des projecteurs soit modifiée pour que les observateurs distinguent le moins possible les régions couvertes par différents projecteurs.

Nous portons ici notre attention particulièrement sur les aspects géométriques des systèmes multi-projecteurs dans le cas de surfaces non-planaires. La plupart du temps, de tels systèmes requièrent un modèle tridimensionnel de l'écran, la position des projecteurs et leur paramètres internes [115]. Dans le cadre de travaux précédents cette thèse, nous avons développé un prototype utilisant une approche originale pour l'alignement des images sans calibrage des projecteurs ou reconstruction explicite de l'écran [155]. Cette idée a été reprise pour le développment du système multi-projecteurs LightTWIST [119, 149].

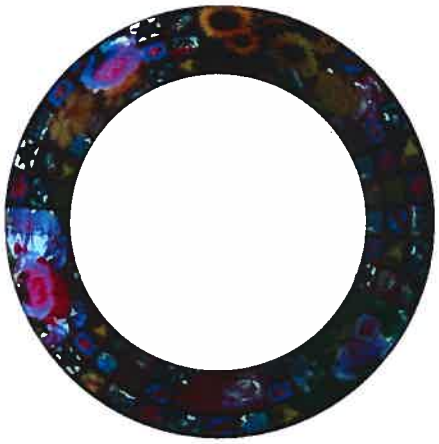
Cette méthode modélise le point de vue d'un observateur à l'aide d'une caméra

(a) Avant correction. Caméra placée au centre de l'écran, encerclée en rouge.

(b) Après correction



(c) Avant correction



(d) Après correction

FIG. **10.7. (a)-(b) Image de l'écran vue de l'extérieur de l'écran cylindrique ; (c)-(d) Point de vue de la caméra.**

pointée vers l'écran. Dans une situation idéale, comme illustrée à la figure 10.6a, un seul projecteur est placé le plus près possible de l'observateur et un écran planaire et perpendiculaire à l'orientation de l'observateur est utilisé. L'image projetée par le projecteur correspond donc à celle observée par la caméra. La configuration d'un système multi-projecteur est plutôt apparentée à celle illustrée à la figure 10.6b. L'approche proposée consiste à obtenir des correspondances denses entre les projecteurs et la caméra. Celles-ci permettent de générer automatiquement une image

corrigée pour le point de vue de l'observateur. La figure 10.7 montre un système multi-projecteurs panoramique utilisant un écran cylindrique. Pour observer entièrement l'écran, une caméra catadioptrique est placée au milieu du cylindre. Les figures 10.7a et 10.7b présentent une vue extérieure de l'écran, respectivement, pour des images non-corrigées d'un échiquier et pour une image corrigée. Les figures 10.7c et 10.7d montrent les deux mêmes situations du point de vue de la caméra.

10.4 Contributions

Les méthodes de mise en correspondances denses supposent en général un ratio de pixel d'environ 1 entre le projecteur et la caméra. Dans un système multi-projecteur comme celui-ci, à chaque pixel de la caméra correspond un assez grand nombre de pixels de chaque projecteur (jusqu'à 25). De plus, il y a souvent peu de contrôle sur l'environnement dans lequel est utilisé le système [149]. Les méthodes traditionnelles sont assez mal adaptées à ces circonstances.

Au chapitre 11, nous proposons une approche de mise en correspondance basée sur les codes de Gray. Plutôt que de trouver le code dans chaque pixel par binarisation, nous calculons une probabilité associée à chacun des codes possibles. Pour réduire les erreurs potentielles, nous tenons compte de la cohérence spatiale des codes d'un pixel à l'autre, c'est-à-dire que le code d'un voisin est généralement assez proche, sauf en présence de discontinuité de profondeur. Nous proposons une formulation probabiliste markovienne très simple que nous résolvons par une méthode itérative : (*Iterated Conditional Mode*). Nos résultats empiriques suggèrent que cette approche est supérieure à la simple binarisation et plus précise qu'une méthode de lissage conventionnelle.

Chapitre 11

A MRF FORMULATION FOR CODED STRUCTURED LIGHT

Cet article [150] a été publié comme l'indique la référence bibliographique.

Tardif, J-P, Roy, S., MRF formulation for coded strctured light, dans International Conference on 3D Digital Imaging and Modeling (3DIM), Ottawa, Canada, pp. 22-29, 2005.

Abstract

Multimedia projectors and cameras make possible the use of structured light to solve problems such as 3D reconstruction, disparity map computation and camera or projector calibration. Each projector displays patterns over a scene viewed by a camera, thereby allowing automatic computation of camera-projector pixel correspondences. This paper introduces a new algorithm to establish this correspondence in difficult cases of image acquisition. A probabilistic model formulated as a Markov Random Field uses the stripe images to find the most likely correspondences in the presence of noise. Our model is specially tailored to handle the unfavorable projector-camera pixel ratios that occur in multiple-projector single-camera setups. For the case where more than one camera is used, we propose a robust approach to establish correspondences between the cameras and compute an accurate disparity map. To conduct experiments, a ground truth was first reconstructed from a high quality acquisition. Various degradations were applied to the pattern images which were then solved using our method. The results were compared to the ground truth for error analysis and

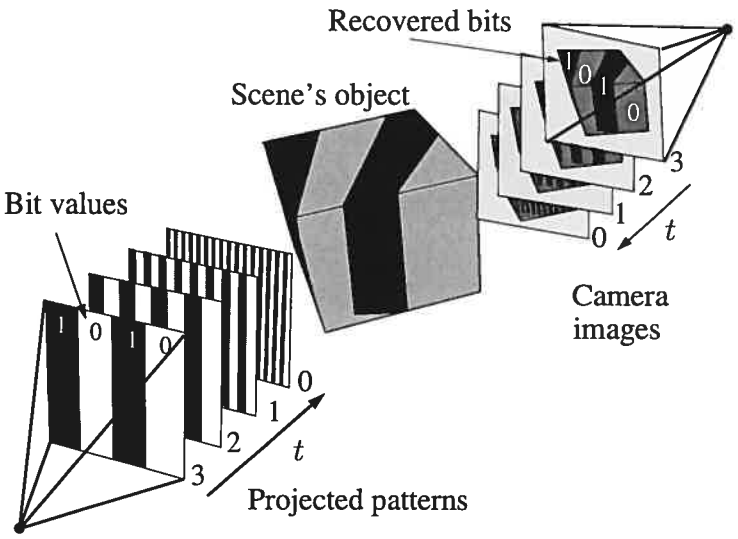


Figure 11.1. Projected patterns for bits $3, 2, 1, 0$ in x direction for $T = 4$. Inverse patterns and those used for the y direction are not shown.

showed very good performances, even near depth discontinuities.

11.1 Introduction

Coded structured light is an active computer vision method employing multimedia projectors and cameras to solve problems such as camera or projector calibration [115], 3D reconstruction [65, 75, 117, 120] and disparity maps computation [127]. It encodes the position of each projector pixel with one or many patterns projected over some surface imaged by a camera. Those images are combined to recover the code and thus pixel correspondences between the camera and the projector. Many kinds of structured-light systems have been described, and a good overview is presented in [16, 122]. Errors in the correspondences occur from noise in the images and, in some cases, ambiguities in the patterns themselves. Indeed, solution usually make a compromise between the quality of the correspondences and the acquisition time

(related to the number of patterns) [30, 64, 105]. The use of many cameras can also increase the pattern decoding robustness [92].

Most industrial scanners require a controlled environment to work properly. In some cases, the nature of the scene imposes a hostile environment which makes the scanning much more difficult. Also, today's systems usually assume that the camera-projector pixel ratio (the number of pixels of the projector seen by only one pixel of the camera) is around one. But this is not always true. For instance, structured light based multiple-projector systems, in which the pixel ratio decreases as the number of projectors increases, were demonstrated [111, 155]. Also, as these systems become less costly and more widely available, support for poor camera quality and bad environment is needed. A more robust structured light approach should be developed for those situations.

This paper introduces a new algorithm to establish correspondence in difficult cases of image acquisition. A probabilistic model formulated as a Markov Random Field uses stripe images to estimate the most likely correspondences in the presence of noise. Our model is specially well-suited to handle the unfavorable projector-camera pixel ratios that occur in multiple-projector single-camera setups. Other degradations can be caused by low contrast due to strong ambient light, high image noise from low quality cameras, and also weak projector lamps or large scanning distances. The probabilistic nature of these degradations justifies the use of such a model.

Generally, the evaluation of the correspondence accuracy requires the use of calibrated camera, projector and reference object. To avoid this task, we used multiple uncalibrated cameras to compute disparity maps. These maps can be used directly to evaluate performance and could be used to compare to passive stereo methods [126]. We propose a robust approach to compute the disparity from many camera-projector correspondence maps. In our experiments, a ground truth disparity map was first reconstructed from a high quality acquisition. Various degradations were applied to the pattern images which were then processed using our method. The results were

compared to the ground truth for error analysis.

The article proceeds as follows: first we introduce the structured-light approach we chose; next we introduce the model for code correction; then we show how to build multiple-camera disparity maps; finally, some results are shown and discussed.

11.2 System overview

To illustrate how a Markovian model can be used to achieve code correction, we present a complete structured light system. We deliberately chose a simple system in order to demonstrate more clearly the effectiveness of our reconstruction model, but it should also be directly applicable to other systems. In our case, the projected patterns are horizontal and vertical black and white stripes to allow an arbitrary projector/camera configuration. Inverse patterns are also used to increase the robustness of the decoding. This decoding defines the initial measurements for the camera-projector correspondences. In our model, they are the most likely values. However, errors can occur, so we show how to compute a confidence value associated to every bit. It is used to define a Markov Random Field for which the most likely configuration can be determined using the Iterated Conditional Mode (ICM) algorithm.

11.3 Complete structured light system

The complete encoding of a pixel position is done using multiple patterns. To simplify notation in this paper, we assume the projector image is square and its width is representable with T bits. A position (x, y) is encoded independently in each dimension so we illustrate our method using a single coordinate, say the x one.

In order to define our structured light patterns, we define the binary encoding of a pixel α of the projector as:

$$P(\alpha) = \overline{\alpha_s} \oplus \alpha_s$$

where α_s is the binary encoding of α, $\overline{\alpha_s}$ the binary complement of α_s and $\oplus$ the concatenation operator. We also define $P^t(\alpha)$ as the t^{th} bit of $P(\alpha)$. The color of α in the pattern t is white if $P^t(\alpha)$ is 1 and black if it is 0. This encoding corresponds the projection of patterns as shown in figure 11.1 and the binary complement formalizes the use of inverse patterns to increase robustness in the decoding process (section 11.3.1).

Subtraction of an image from its inverse and thresholding is the basic way to discover the value of a bit. Unfortunately, uncertainties occur when the difference becomes very small. This typically happens when the contrast in the images is low or when a border between stripes is projected onto a single pixel. This basic encoding has the drawback of keeping many stripe borders aligned which can make many bits uncertain in a single code. To correct this, we rely on Gray encoding [168] which minimizes the encoding's bitwise difference between spatial neighbors in the projector image. For example, if we consider two projector pixels having x coordinates 127 and 128. The use of Gray encoding changes their binary representations from 0111111 and 1000000 to 0100000 and 1100000, thus reducing the number of stripe borders located between these pixels from 8 to 1. The encoding becomes:

$$P'(\alpha) = \overline{G(\alpha_s)} \oplus G(\alpha_s)$$

where G is defined as:

$$G(\alpha_s) = \alpha_s \ \mathbf{xor} \ (\alpha_s >> 1)$$

and $>>$ is the *right bit shift.*

11.3.1 Pattern decoding

This step builds the first estimate of the correspondence of each camera pixel to a projector pixel and computes a confidence measure based on the observed pixelwise

contrast.

For each camera pixel β and each pattern t, an intensity $I^t(\beta)$ is measured. We define the image contrast as:

$$\delta(\beta) = \max_t \left(I^t(\beta)\right) - \min_t \left(I^t(\beta)\right).$$

The Gray code correspondence can then be recovered with a simple image difference:

$$C(\beta) = \bigoplus_{t=T-1}^{0} \text{Bin}\left(I^t(\beta) - I^{t+T}(\beta)\right)$$

where

$$\text{Bin}(a) = \begin{cases} 1 & \text{if } a > 0 \\ 0 & \text{otherwise} \end{cases}$$

(the decreasing index of the concatenation simply reflects the fact that the most significant bits appear first). Note that no thresholding is done at this step. However, a threshold is used to build a confidence mask for the measurement:

$$H_\beta = \left\{ \text{Conf}\left(I^t(\beta) - I^{t+T}(\beta), \beta\right) \right\}_{t=T-1}^{0}$$

where

$$\text{Conf}(d, \beta) = \begin{cases} 1 & \text{if } |d| > K_c\delta(\beta) \text{ and } \delta(\beta) > K_r \\ \frac{1}{2} & \text{otherwise.} \end{cases} \tag{11.1}$$

H_β^t is the probability we determine for the t^{th} bit of the Gray code of β's projector x-coordinate to be equal to $C^t(\beta)$, the t^{th} bit of $C(\beta)$. The constants $K_c = \frac{1}{2}$ and $K_r = 15$ are conservative enough to insure that the contrast is sufficiently high. These values rarely need to be changed in practice. A confidence value can be seen as a probability that the measurement can be trusted. Indeed, a value of 1 means that the bit was *unambiguously* recovered and a value of $\frac{1}{2}$ means total uncertainty. Finally,

the number of 1's in H_β indicates the overall quality of the code recovered for pixel β. Accordingly, the Boolean value:

$$C_v(\beta) = \begin{cases} 1 \text{ if the \# of 1's in } H_\beta \geq T - \max(\log_2 \rho, 0) - 1 \\ 0 \text{ otherwise} \end{cases}$$

where ρ is an estimate of the smallest pixel ratio between the projector and its image in the camera and "-1" is plainly a margin of error. This function determines if β was sufficiently illuminated by the projector.

The value of $C(\beta)$ is the most likely value of the projector correspondence of β. In ideal situations, this value is close to being exact, but most of the time many errors occur. The next section explains how the codes can be corrected.

11.3.2 A Markovian model for code correction

The low confidence in certain bits of a correspondence can result from two factors. The first is the projection of a border onto a pixel. Even though this occurs more frequently for low order bits, it can actually happen at any level. Fortunately, even if high order bits weren't recovered for a pixel, there is a good chance they were for its neighbors. When all of them have the same value for some bit, chances are this is the right value for the current pixel too.

The second factor is related to the pixel ratio between the camera and the projector. For similar resolutions, if the area covered by the projector is smaller than that covered by the camera, low order bits cannot be recovered. In this case, the neighbors are of no help. However, a hypothesis can be made, that the object surface is locally smooth, and thus the codes as well. In most cases, with the use of Gray codes, only a small number of bits of a correspondence will be uncertain. The scheme we present tries to find the code that best satisfies these assumptions. The Markovian approach is known to be well adapted to solve this type of problems.

We represent the camera image by a graph $G = (B, N)$. A site $\beta \in B$ is a pixel with a value of $C_v(\beta)$ equal to 1. Each site's neighborhood $\mathcal{N}_\beta$ is the usual 8-neighborhood (possibly consisting of less than 8 elements). The labels are the 2^T possible values of the coordinate in the projector image. When a value is associated to each site, the Markovian field M is in a configuration m whose probability is function of the measured code C. Given its computed value c, we are looking for the most likely value of M. In the following, we use C^t_β for $C^t(\beta)$ to increase the equations compactness. We have:

$$\begin{aligned} P(M = m \mid C = c) &\propto P(C = c \mid M = m) \cdot P(M = m) \\ &\propto \left(\prod_{\beta \in B} P(C_\beta = c_\beta \mid M_\beta = m_\beta) \right) \cdot P(M = m) \end{aligned}$$

(if we suppose pixel- and bitwise independence)

$$\begin{aligned} P(M = m \mid C = c) &\propto \left(\prod_{\beta \in B} \prod_{t=0}^{T-1} P(C^t_\beta = c^t_\beta \mid M^t_\beta = m^t_\beta) \right) \cdot P(M = m) \\ &\propto \left(\prod_{\beta \in B} \prod_{t=0}^{T-1} P(C^t_\beta = c^t_\beta \mid M^t_\beta = m^t_\beta) \right) \prod_{\substack{\beta_1 \in B \\ \beta_2 \in \mathcal{N}_{\beta_1}}} e^{-\xi V(m_{\beta_1}, m_{\beta_2})} \end{aligned}$$

where V is the smoothing cost function and ξ a smoothing factor. Taking minus the log, this is equivalent to minimizing directly the cost function:

$$-\sum_{\beta \in B} \sum_{t=0}^{T-1} \log P(C^t_\beta = c^t_\beta \mid M^t_\beta = m^t_\beta) + \xi \sum_{\substack{\beta_1 \in B \\ \beta_2 \in \mathcal{N}_{\beta_1}}} V(m_{\beta_1}, m_{\beta_2}) \qquad (11.2)$$

w.r.t. m. The value of $P(C^t_\beta = c^t_\beta \mid M^t_\beta = m^t_\beta)$ is modeled using the confidence that

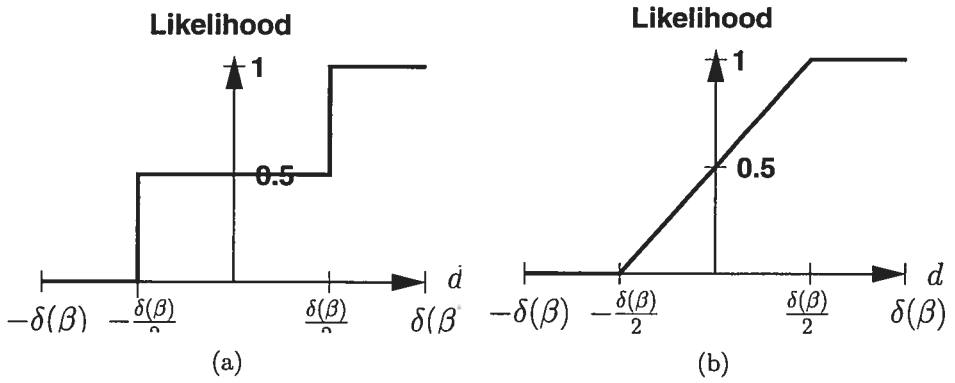


Figure 11.2. Likelihood that a recovered bit is 1 as a function of the intensity difference d of the two corresponding patterns, based on a) eq. 11.1 and b) eq. 11.4.

was recovered previously. This can be expressed as:

$$P(C_\beta^t = c_\beta^t \mid M_\beta^t = m_\beta^t) \propto \begin{cases} H_\beta^t & \text{if } c_\beta^t = m_\beta^t \\ 1 - H_\beta^t & \text{otherwise.} \end{cases} \tag{11.3}$$

The corresponding likelihood of getting a value of 1 as a function of the intensity difference is shown in figure 11.2a.

Defining $V(\beta_1, \beta_2) = |G^{-1}(\beta_1) - G^{-1}(\beta_2)|$, where G^{-1} converts a Gray code to its real value, is a logical choice. The effect of the smoothing term is that it favors codes that are in between those of the neighbors (figure 11.3). Unfortunately, it is not clear what PDF corresponds to this relation. .

Another definition clarifies the effect of the confidence and matching cost functions. For a given pixel β, a code ν is said to be *compatible* if it is identical to $\mu = C(\beta)$ for the bits *unambiguously* recovered. This can be expressed as:

$$\text{Comp}(\beta, \mu, \nu) = \begin{cases} 1 & \text{if } \forall t \text{ such that } H_\beta^t = 1 : \mu^t = \nu^t \\ 0 & \text{otherwise.} \end{cases}$$

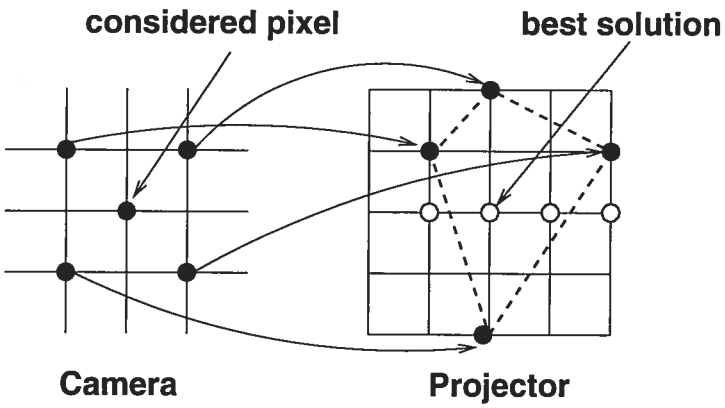


Figure 11.3. For a pixel whose x projector correspondence has uncertain low-order bits 0 and 1, four solutions are possible (empty circles). According to our model, the most likely code minimizes the average distance to its neighbors (only the 4-neighborhood is shown for clarity).

From eq. 11.1 and 11.3, we see that all compatible codes have equal matching costs and that all the others have probability 0. This means that the correct code must be compatible. In practice, this function can be used for the optimization to reject labels without computing the cost function. Also, note that when using eq. 11.1 in the MRF, multiplying ξ by any positive constant does not alter the amount of smoothing.

More sophisticated approaches can be used. Indeed, even when a certain bit of a code has low confidence, the value found by image difference is still more likely than its complement. A simple model for this is illustrated in figure 11.2b. The closer to zero this difference is, the more ambiguous the value becomes. The confidence function related to figure 11.2b is:

$$\mathrm{Conf}(d,\beta) = \begin{cases} 1 & \text{if } \delta(\beta) > K_\tau \text{ and } |d| > K_c\delta(\beta) \\ \frac{1}{2} & \text{if } \delta(\beta) \leq K_\tau \\ \frac{|d|+K_c\delta(\beta)}{2K_c\delta(\beta)} & \text{otherwise} \end{cases} \tag{11.4}$$

where K_c and K_r are the same as in equation 11.1. Finally, another confidence measure could be used without resorting to thresholding; all labels would have non zero probabilities. Unfortunately, minimizing the corresponding function is far too complex in practice.

Cost functions such as eq. 11.2 are generally not too difficult to minimize globally and efficiently. However, in our experiments, the image resolution and the number of labels are overwhelming. Resolution by ICM yields convincing results within a reasonable computational time (see section 11.7 for details), even though only a local minimum is found.

11.4 Projector-to-camera correspondences

The projector-to-camera correspondence map is used for image construction in a multiple projector system [155] and also for disparity map construction (section 11.5). It is built by inverting the camera-to-projector mapping obtained by structured light. This inverse function is not easily determined. In our experiments, we were using a camera and a projector with similar resolutions. Since the surface illuminated by the projector was contained inside the camera image, only a sampling of all the codes could be achieved. In the projector domain, this means that not all projector pixels have a corresponding camera pixel, which creates holes in the inverse map. One way to fill these holes is to use interpolation. One must find a scheme with a solid geometrical interpretation that performs well in terms of accuracy and execution time.

A first scheme uses homography-based interpolation. This assumes that camera and projector models are linear for small areas of the image, and that the object surface can be approximated locally by a small planar patch. The camera image is divided into 4-pixel patches reprojected in the projector image onto the correspondence points. Then, if some projector pixel is located inside this patch, its value is

calculated from a homography defined with the four corners of the patch.

Another scheme presented in [155] uses triangular patches with bilinear interpolation. The geometric interpretation is less intuitive, but it has been used successfully in applications where small inaccuracy could be tolerated. This scheme is specially useful when a fast implementation over a GPU is necessary.

The result of the inversion is that for each pixel α of the projector, a corresponding camera pixel $C^{-1}(\alpha)$ is defined. When a reconstruction is very well recovered, the relation $C(\beta) = C^{-1}(C(\beta))$ should be valid for all pixels β illuminated by the projector. This does not occur in practice, because of errors in the reconstruction and because the pixel ratio can be smaller than one.

11.5 Disparity map construction

Without calibration of the cameras and projectors, it is impossible to achieve a full 3D reconstruction of the scene. However, it is possible to build simple disparity maps [127].

The basic approach for disparity map construction is to use some camera as the main view. A pixel β of this camera and its correspondence β' in another camera j both have the same x and y projector codes. Their image distance is the disparity. Estimating camera-camera correspondences amounts to locating common projector codes. For known epipolar geometry, this is done with a linear search, otherwise, a 2D search is needed. This process can be very long for large images. Also, when the pixel ratio is smaller than one or when a lot of errors occur in the correspondences, some codes are not present in the other images. Figure 11.4 illustrates the percentage of pixels in an image that have an exact correspondence pixel in another image, as a function of the camera-projector pixel ratio. As expected, almost all codes are available for pixel ratios larger than 2, but become scarce for low ratios. In this case, direct estimation of disparity is unusable.

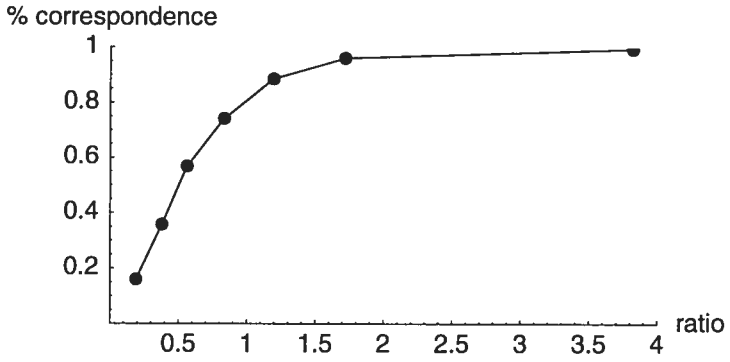


Figure 11.4. Percentage of exact correspondences between two cameras w.r.t. pixel ratio between the projector and its image in the cameras.

A more robust approach is to use the inverse correspondence map presented in section 11.4. Let us define the correspondence of projector pixel α in camera j as $C_j^{-1}(\alpha)$. Therefore, the pixel in camera j corresponding to pixel β of camera 0 is simply $C_j^{-1}(C_0(\beta))$. Because the inverse mapping function is interpolated, it allows code interpolation. For perfectly reconstructed scenes, $C_0(\beta) = C_j(C_j^{-1}(C_0(\beta)))$ should be true.

For many cameras located on a single baseline without any rotation, we compute the disparity of a pixel β of camera 0 w.r.t. camera j as:

$$D_{0,j}(\beta) = \frac{1}{n}\,\mathrm{dist}(0,j)\sum_{i=1}^{n}\frac{\left\|\beta - C_i^{-1}(C_0(\beta))\right\|}{\mathrm{dist}(0,i)}$$

where $\mathrm{dist}(0,j)$ is the distance between the two cameras' optical centers. A correspondence for one camera is not used when the error between correspondence codes is too large. In our experiments, we rejected a correspondence when the distance was above 2 pixels. Rejection of correspondences for all cameras results in unknown disparity for this pixel.

11.6 Validation

Validation of a structured light system is difficult and sometimes involves precise setup and calibration. A full 3D reconstruction of a perfectly known scene can be used for error analysis. In the context of poor image acquisition, we propose a different approach to test the quality of the correspondences. A scene containing two parallel and overlapping planes was carefully reconstructed in a controlled environment (constant ambient lighting) with a powerful XGA 1024×768 projector of 2000 lumens and a low-noise Basler A201bc cctv 1008×1018 camera. Each plane featured a checkerboard texture of varying colors so the contrast in the images is not uniform. A disparity map was computed using one camera moved to six locations on a single baseline. Its accuracy was high enough to be considered as our ground truth. Pictures of the two planes, the disparity map and one sample slice are shown in figure 11.7.

In previous experiments, we had observe that our algorithm performs much better than classical approaches in the presence of high noise. We also wanted to show that significant improvements could be achieved in better, more realistic conditions. In order to do this, we measured the noise induced in the pattern images when compression is turned on, as it commonly is on low quality cameras (cf. figure 11.5). Than in our tests, we corrupted the images with Gaussian noise of mean 0 and standard deviation 2, a smaller value than the previous measurement. In addition, we gradually reduced the images contrast by K percent (cf. figure 11.6).

Four algorithms were tested with gradually decreasing contrast to increase correspondence errors. We recovered the codes with each of them and built the disparity maps. Codes and maps were compared to the ground truth. The first algorithm (labeled P) consisted solely in the recovery of the codes without any further correction. In the second, labeled Q, we used a simple low-pass filtering of the codes. A 5×5 filter was empirically determined to be a good compromise between smoothing and discontinuity preservation. Finally for the third and fourth, respectively labeled M_1 and

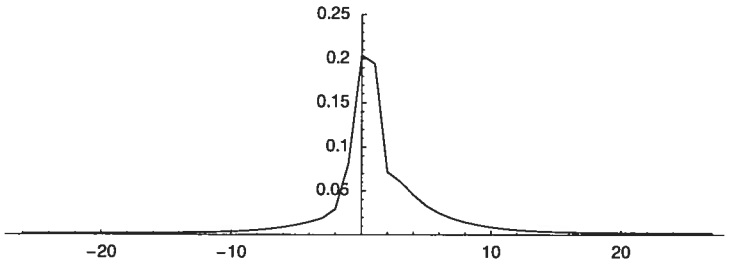
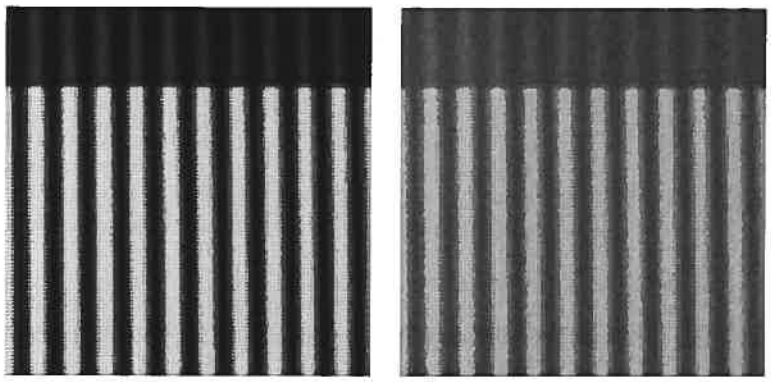


Figure 11.5. Histogram of the noise in the pattern images compressed with MJPEG (compression ratio around 10:1). The variance of the noise is higher than that of the noise we added to test our algorithms.

M_2, we tested our proposed Markovian approach with the matching cost functions of eq. 11.1 and eq. 11.4. We used the ICM algorithm to perform the optimization, moving randomly from one pixel to another [81]. An operation consists of computing the cost function for every possible code for a given pixel and then select the best value. One iteration contains a number of operations equal to the number of pixels in the image. The configuration for which the cost function was minimum was kept and the process stopped after 7 consecutive iterations without improvement.

11.7 Results

Figure 11.8 gives the average error in pixels in the disparity maps and the recovered codes of each algorithm, as a function of the contrast reduction K. This error is computed as the euclidean distance between a recovered code and the ground truth, meaning that errors for high-order bits are worst than that for low-order bits. For the disparity error (figure 11.8b), the only pixels considered are those for which a disparity value was recovered. Figure 11.8c illustrates the number of pixels kept for different values of K. We observed that the Markovian approach is always superior to raw codes (P) and filtering (Q), particularly when the decoded patterns have a lot of errors. Moreover, the number of rejected pixels is always smaller. It also came as



(a) (b)

Figure 11.6. a) Zoom on a section of a non-corrupted pattern. b) Same image with a 41% image contrast reduction and added noise.

a surprise that the simpler model M_1 performs as well as M_2, and sometimes better. For the latter, finding a good solution is more difficult when the codes are highly corrupted, especially near discontinuities (figure 11.10). Moreover, each iteration for algorithm M_1 takes about 5 seconds, but takes more than twice as much for M_2, as the likelihood cannot be precomputed because of memory limitations. For our tests, convergence of ICM takes between 20 and 40 iterations, and similarly for M_1 and M_2. The parameter $\xi > 0$ has no influence on the solution when using M_1, but has a big impact for M_2. This is illustrated in figure 11.9, in which we also observe that M_2 never yields significantly better results as M_1. A value ξ equal to 0 is equivalent to no code correction. As ξ gets larger, the solution for eq. 11.4 gradually converges to the one obtained with eq. 11.1. We observe that for a value above 0.02, the error is close to stable, and above 0.05, the solutions are exactly the same. In our tests, we used a value $\xi = 0.04$.

Figures 11.8 shows that even a small amount of error in the codes (11.8a) can result in large differences in the disparity maps (11.8b). These errors increase the number of rejected correspondences (11.8c), yielding holes in the maps, as illustrated

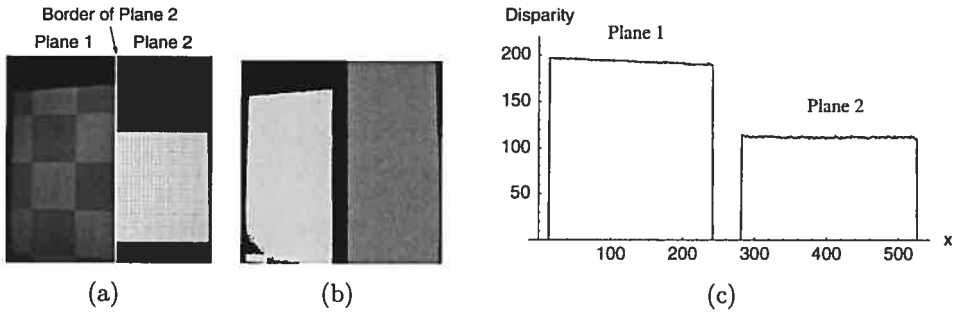


Figure 11.7. a) Image of the two planes. b) Disparity map reconstructed for the left camera. c) One slice of the disparity map.

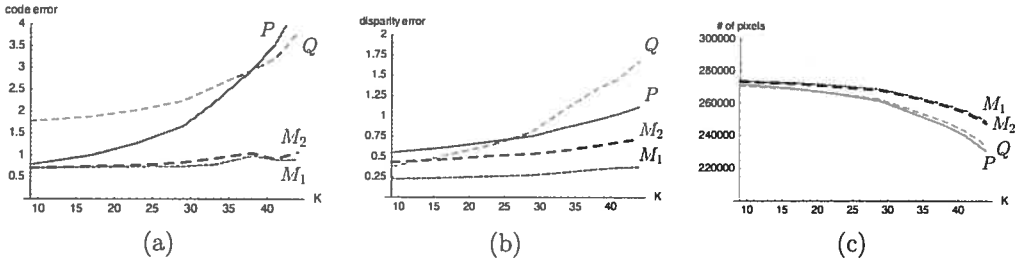


Figure 11.8. Performance as a function of contrast reduction K. a) Mean error in the recovered codes. b) Mean error in pixels of the disparity maps. c) Number of pixels with a recovered disparity.

in figures 11.12 and 11.11 .

The results of the filtering algorithm (Q) are surprising. Bad outliers occur frequently, and since they are not corrected in any way, they introduce very large disparity errors which appear as spikes in figure 11.12b. Consequently, filtering is suitable only in the absence of error in high order bits. In fact, the filtering mechanism, unless combined with a robust pixel selection, always propagates the error of a pixel to its neighbors instead of correcting it.

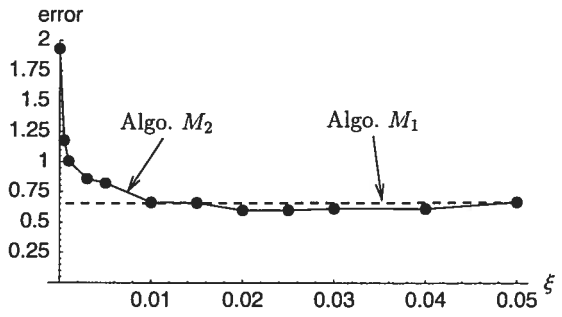
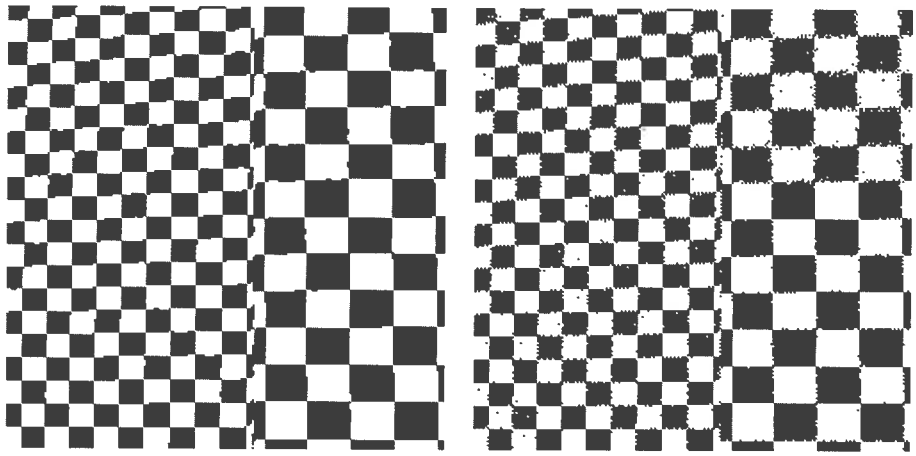


Figure 11.9. Mean error in the codes w.r.t. the smoothing parameter ξ. For $\xi = 0$, no correction is made to the codes. For $\xi \geq 0.05$, the choice of matching cost function makes virtually no difference.

11.8 Conclusion

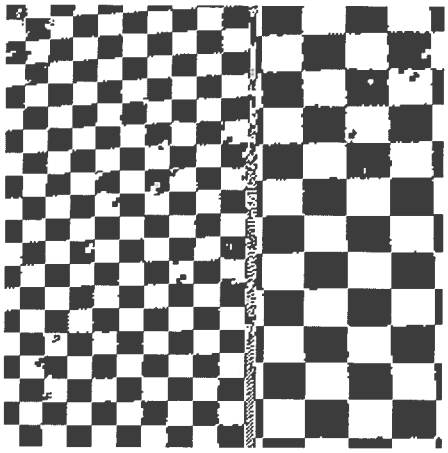
This paper presented a Markovian approach to coded structured light reconstruction. We consider it is one step toward widespread use of structured light in uncontrolled environment with commonly available equipment. It performs more robustly than conventional methods and recovers accurate depth discontinuities. It was used successfully to calibrate a large multi-projector screens used in a public performance [149].

In the future, a more physically plausible model could be investigated, but the increased computational burden might prove unsurmountable.

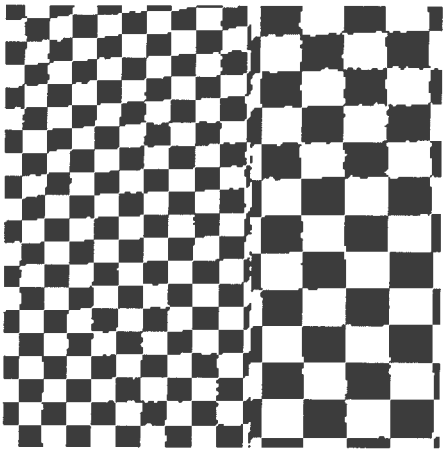


(a) Ground truth

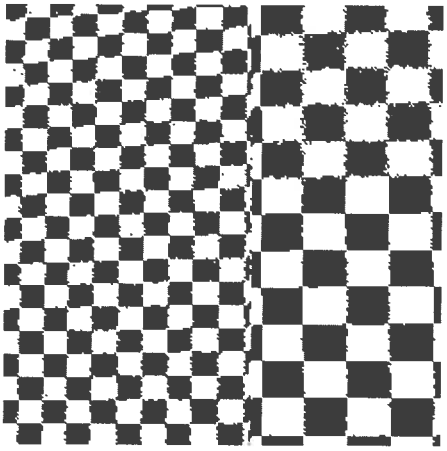
(b) Algo. P



(c) Algo. Q

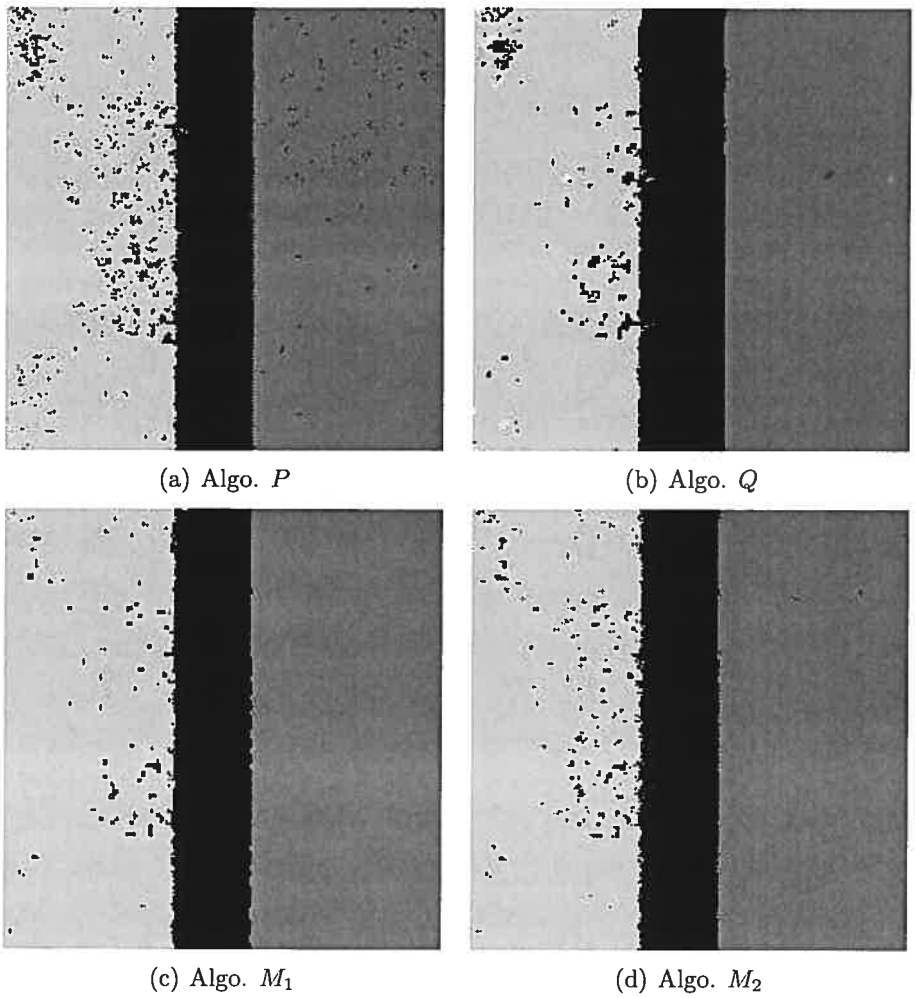


(d) Algo. M_1



(e) Algo. M_2

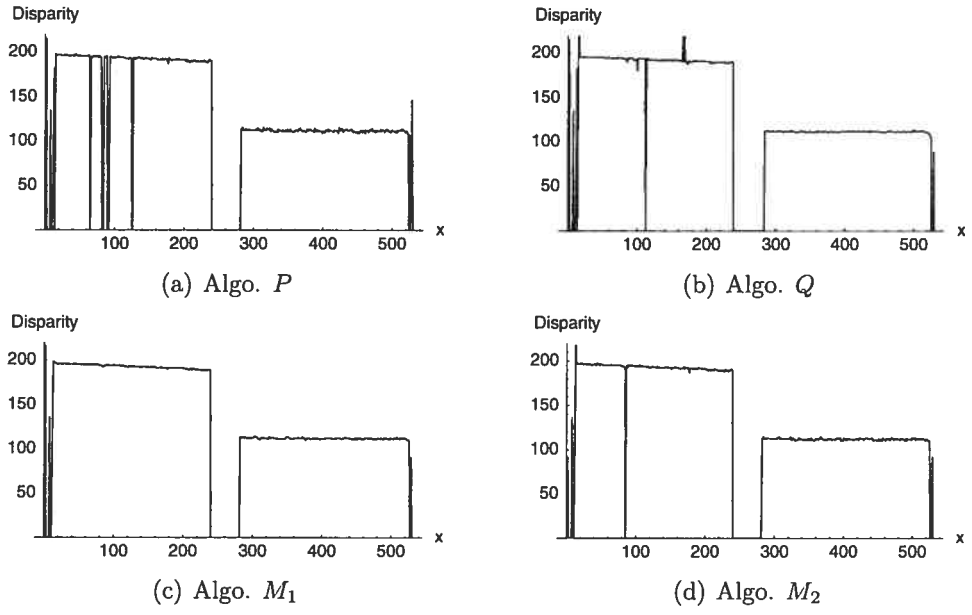
Figure 11.10. Recovered codes for contrast $K = 41\%$ textured with a checkerboard image.



(a) Algo. P (b) Algo. Q

(c) Algo. M_1 (d) Algo. M_2

Figure 11.11. Zoom on a section of the disparity maps computed with contrast $K = 30\%$. Black represents pixels with no recovered disparity.



(a) Algo. P (b) Algo. Q

(c) Algo. M_1 (d) Algo. M_2

Figure 11.12. Horizontal slices of the disparity maps of figure 11.11. Pixels with unrecovered disparities are set to 0.

Chapitre 12

RECONSTRUCTION 3D PASSIVE PAR FACTORISATION DE MATRICE AVEC DONNÉES MANQUANTES

12.1 Introduction

Il existe plusieurs problèmes importants en vision par ordinateur qui peuvent être formulés comme celui d'approcher une matrice de données par une matrice d'un certain rang :

- reconstruction 3D affine et perspective de scène statique [141, 159]
- reconstruction 3D affine d'objet déformable [24],
- segmentation de mouvement [167],
- estimation de pose multi-caméras–multi-plans [143],
- reconstruction par changement d'illumination [62],
- auto-calibrage de distortion radiale [33, 114].

D'autres applications sont aussi retrouvées en apprentissage par ordinateur[1] [77, 104, 135]. Tel que discuté §3.5.1, ce problème est équivalent à trouver directement une factorisation pour la matrice approchée. Dans plusieurs circonstances, les facteurs retrouvés sont sujets à des contraintes, par exemple, facteurs non-négatifs ou rangée contrainte à une certaine valeur. Très souvent, la matrice de mesure est incomplète ce qui rend le problème encore plus difficile à résoudre.

Dans cette thèse, nous nous intéressons à deux sortes de factorisation de matrice avec données manquantes : sans contrainte et de tout rang, de rang 4 avec la dernière rangée du deuxième facteur forcée à 1. La factorisation sans contrainte concerne la

[1] *Machine learning*

plupart des problèmes de vision, alors que le deuxième concerne la reconstruction 3D affine. Dans ce chapitre, nous nous intéressons surtout au problème de factorisation dans le contexte de la reconstruction 3D multi-vues non-calibrée. L'entrée des algorithmes consiste en des correspondances de points provenant d'une séquence d'images. Celles-ci sont obtenues à l'aide d'un algorithme de suivi de points [84] ou un algorithme de mise en correspondance automatique [17, 83]. La reconstruction 3D consiste donc à retrouver les matrices de projection expliquant le mouvement de caméra, et les points 3D associés aux correspondances images.

Organisation

Tout d'abord, §12.2, nous décrivons des méthodes de factorisation de matrice dense pour la reconstruction 3D dans le cas des modèles affine et perspectif. Nous décrivons ensuite, §12.3, plusieurs approches de factorisation de matrice avec données manquantes. Nous concluons ce chapitre, §12.4, avec un résumé de nos contributions décrites en détails aux chapitres 13 et 14.

12.2 Méthodes de factorisation de matrice pleine

Nous avons vu §3.5.1 que le problème de trouver une matrice AB qui minimise la fonction

$$\left\|\mathsf{M}_{(n\times m)} - \mathsf{A}_{(n\times r)}\mathsf{B}_{(r\times m)}\right\|_F^2 \tag{12.1}$$

peut être résolu de façon efficace à l'aide la SVD de M. Souvent, certaines contraintes sur A et B doivent aussi être forcées. Par exemple, en reconstruction 3D affine, on peut formuler le problème de telle sorte que la dernière rangée de B doit être constituée de 1 [125]. Dans plusieurs problèmes d'apprentissage, on impose parfois que A et B soient positifs [77].

L'ambiguïté de la factorisation fait en sorte que la fonction possède une infinité de minima globaux. Il importe aussi de comprendre la distinction entre ces différentes

solutions. Par exemple, en reconstruction 3D affine, les facteurs correspondent aux matrices de projection et aux points 3D, et leur estimation revient à effectuer une reconstruction 3D non-calibrée. L'ambiguïté de factorisation permet de transformer cette reconstruction à l'aide d'une matrice affine. À partir de certaines hypothèses sur les paramètres internes des caméras de la séquence, il est possible de restreindre le domaine des solutions valides. Ceci est le rôle de l'auto-calibrage de caméra, permettant le passage d'une reconstruction affine à une reconstruction métrique (*cf.*§2.2) [57, 110]. Deux hypothèses typiques à cet égard consistent à supposer que les paramètres internes sont constants au long de la séquence et que le ratio d'aspect est égale à 1. L'ensemble des solutions peut alors s'exprimer à partir d'une solution satisfaisant cette contrainte, suivie d'une transformation métrique de la reconstruction.

Notation

Dans ce chapitre, au lieu de M, A et B, nous utilisons parfois les lettres $\mathcal{M}$, $\mathcal{P}$, $\mathcal{Q}$. Cette notation est introduite §12.2.1 et est utilisée lorsque les algorithmes sont spécifiques aux problèmes de reconstruction 3D affine et perspective. De plus, pour la description des algorithmes itératifs, nous ajoutons aux matrices un exposant entre parenthèse, *e.g.* (k), ce qui signifie la solution à l'itération k.

12.2.1 Reconstruction affine

La solution originale de factorisation dans le contexte de reconstruction pour le modèle affine a été proposée par Tomassi et Kanade en 1992 [159]. Dans ce contexte, la matrice $\mathcal{M}$ est de la forme :

$$\mathcal{M}_{(2P\times m)} = \begin{bmatrix} \mathbf{m}_1^1 & \dots & \mathbf{m}_1^m \\ \vdots & & \vdots \\ \mathbf{m}_P^1 & \dots & \mathbf{m}_P^m \end{bmatrix} = \mathcal{P}_{(2P\times 3)}\mathcal{Q}_{(3\times m)} + \mathcal{T}_{(2P\times 1)}\mathbf{1}_{(1\times m)}^{\mathsf{T}}$$

où, m est le nombre de trajectoires de points de saillants, P est le nombre d'images, $\mathcal{P}$ représente les matrices de projection de caméra (sans la translation, *cf*.2.2.1) superposées les unes sur les autres, $\mathcal{T}$ représente les vecteurs de translation eux aussi superposés, et Q représente les points 3D euclidien de la reconstruction mis côte à côte. Ainsi, chaque élément de $\mathcal{M}$ est le résultat d'une projection affine :

$$\tilde{\mathbf{m}}^j_{p(2\times1)} = \mathsf{P}_{p(2\times3)}\tilde{\mathbf{Q}}^j_{(3\times1)} + \mathbf{t}_{p(2\times1)} \tag{12.2}$$

où, p est la $p^{\text{ième}}$ image de la séquence. Lorsque M est pleine, cela signifie que les points 3D sont visibles dans toutes les caméras. Une solution valide pour le vecteur des translation est donnée par :

$$\mathcal{M}\boldsymbol{\mu}_m, \quad \text{où,} \quad \boldsymbol{\mu}_m = \frac{1}{m}\mathbf{1}_{(m\times1)}, \tag{12.3}$$

i.e. $\boldsymbol{\mu}_m$ permet de calculer le vecteur comprenant la moyenne de chaque rangée d'une matrice de m colonnes. On soustrait ensuite la composante translationelle des caméras pour obtenir une nouvelle matrice de rang 3 :

$$\bar{\mathcal{M}} = \mathcal{M} - \mathcal{M}\boldsymbol{\mu}_m\mathbf{1}^{\mathsf{T}}_{(1\times m)}. \tag{12.4}$$

En présence de bruit dans les données, $\bar{\mathcal{M}}$ doit être approché par une matrice de rang 3, ce qui est équivalent à solutionner :

$$\min_{\mathcal{P},\mathcal{Q}} \left\| \left(\bar{\mathcal{M}}_{(2P\times m)} - \mathcal{P}_{(2P\times3)}\mathcal{Q}_{(3\times m)}\right) \right\|^2_F, \tag{12.5}$$

à l'aide de sa SVD. À noter que la solution retrouvée est optimale en assumant un modèle de bruit gaussien, puisqu'elle minimise l'erreur de reprojection.

12.2.2 Factorisation avec incertitude

Il est possible d'effectuer la factorisation affine précédente lorsque des mesures d'incertitude sur les données sont connues. Ces mesures peuvent être obtenues à partir d'information sur la qualité des correspondances entre deux images [83]. Nous présentons un très bref résumé de l'approche d'Anandan et Irani qui permet d'approcher l'erreur optimale pour un modèle de bruit anisotropique [4].

Un modèle de bruit est isotropique lorsque l'erreur en X et Y est non-corrélée, ce qui permet de la représenter par une loi Normale 2D. Ainsi, à chaque point saillant j de la séquence, pour chaque image p, correspond un modèle de bruit sur la mesure $\mathbf{e}_j^p \sim N(0, {\sigma_{pj}}^2 \mathtt{I}_{2\times 2})$, qui n'est pas nécessairement constant au long de séquence. La mesure d'un point saillant est donc donnée par la projection d'un point 3D, à laquelle est additionnée l'erreur de mesure :

$$\mathbf{m}^j_{p(2\times 1)} = (\mathtt{P}_{p(2\times 3)}\mathbf{Q}^j_{(3\times 1)} + \mathbf{t}_{p(2\times 1)}) + \tilde{\mathbf{e}}^p_j.$$

Dans le cas simplifié où le modèle d'erreur varie à chaque point mais est constant au cours la séquence, on peut remplacer σ_{pj} par σ_j, et le problème peut s'écrire :

$$\arg\min_{\mathcal{P},\mathcal{Q}} \sum_p \sum_j \frac{\tilde{\mathbf{e}}_j^{p\top}\tilde{\mathbf{e}}_j^p}{\sigma_p^2} = \arg\min_{\mathcal{P},\mathcal{Q}} \left\| \mathrm{diag}(\sigma_1, ..., \sigma_m)(\bar{\mathcal{M}} - \mathcal{P}\mathcal{Q}) \right\|_F^2$$

ce qui peut se résoudre, encore une fois, à l'aide de la SVD.

Si l'incertitude est corrélée en X et Y, le modèle de bruit doit être représenté par une matrice de covariance : $\tilde{\mathbf{e}}_j^p \sim N(0, \Sigma pj)$. Dans ce cas, la distance de Mahalanobis doit plutôt être minimisée. Nous cherchons :

$$\arg\min_{\mathcal{P},\mathcal{Q}} \sum_p \sum_j \tilde{\mathbf{e}}_j^{p\top} \Sigma pj^{-1} \tilde{\mathbf{e}}_j^p = \left\| (\bar{\mathcal{M}} - \mathcal{P}\mathcal{Q}) \right\|^2_{\Sigma pj}, \tag{12.6}$$

où, $\| \ \|_{\Sigma pj}$ signifie la distance de Mahalanobis. Ce problème n'admet pas de solution

analytique. Seules des solutions numériques itératives ou approchées existes, comme celle proposée par Anandan et Irani. Leur solution consiste à reformuler le problème (12.6) comme un problème de factorisation au sens de la norme de Frobenius sujette à certaines contraintes. Grossièrement, la solution approchée consiste en deux étapes : factoriser $\bar{\mathcal{M}}$ avec la norme de Frobenius et forcer les contraintes *a posteriori*.

12.2.3 Passage affine à perspectif

L'utilisation du modèle de caméra affine résulte en un problème bilinéaire de factorisation de matrice. Cependant, ce modèle de caméra n'est pas toujours une approximation valide et le modèle perspectif est préférable. Le problème résultant est cependant beaucoup plus difficile à résoudre. Plusieurs travaux proposent d'initialiser la reconstruction perspective au moyen d'une reconstruction affine, laquelle est ensuite raffinée de façon itérative [31, 57, 178]. Cette approche est raisonnable lorsque la caméra sténopé possède un angle de vue assez petit.

12.2.4 Factorisation pour le cas perspectif

Sturm et Triggs proposent une méthode de reconstruction pour le modèle perspectif, laquelle est basée sur la factorisation de matrice [141]. Dans ce contexte, l'équation de projection est :

$$\lambda_p^j \begin{pmatrix} \tilde{\mathbf{m}}_p^j \\ 1 \end{pmatrix} = \mathsf{P}_{p(3\times 4)} \mathbf{Q}_{(4\times 1)}^j \tag{12.7}$$

où, $\tilde{\mathbf{m}}_p^j \in \mathbb{R}^2$ représente la coordonnée du point dans l'image et λ_p^j, appelé *profondeur projective*, est un facteur d'échelle relié à la profondeur du point 3D. En supposant que la profondeur projective de tout point soit connue *a priori*, il est possible de réécrire le problème de reconstruction comme un problème de factorisation. En présence de bruit, on cherche

$$\arg\min_{\mathcal{P},\mathcal{Q}} \left\| \mathcal{M}_{(3P\times m)} - \mathcal{P}_{(3P\times 4)} \mathcal{Q}_{(4\times m)} \right\|_F^2, \tag{12.8}$$

où, $\mathcal{M}$ est constitué des trajectoires de points saillants en coordonnées projectives, chacun d'eux étant multiplié par sa profondeur projective à chaque image, et $\mathcal{P}$ est constitué des matrices de projection empilées les unes sur les autres. Contrairement à la reconstruction affine, ce problème ne minimise qu'une erreur algébrique, non pas l'erreur de reprojection.

Leur contribution est surtout une méthode permettant l'estimation des profondeurs projectives à partir de matrices fondamentales entre les vues. Leur méthode fonctionne de façon séquentielle, c'est-à-dire que les profondeurs projectives dans les deux premières vues sont d'abord estimées, puis, celles pour la vue n sont estimées à partir de celles dans la vue $n-1$.

Martinec et Pajdla proposent une méthode globale, mais approximative, d'estimation des profondeurs projectives [89], où les matrices fondamentales entre chaque paire de vues peuvent être utilisées.

Une dernière solution consiste à initialiser toutes les profondeurs projectives à 1, ce qui revient à faire l'hypothèse que la profondeur des objets demeure (approximativement) constante au cours de la séquence [60]. Finalement, certains travaux proposent de raffiner de façon itérative les profondeurs projectives et la reconstruction 3D après une première factorisation [88, 103, 161].

12.3 Méthode de factorisation de matrice avec données manquantes

Tel que décrit §3.5.2, le problème de factorisation d'une matrice avec données manquantes

$$\arg\min_{\mathsf{A},\mathsf{B}} \left\| \mathsf{W}_{(n\times m)} \odot \left(\mathsf{M}_{(n\times m)} - \mathsf{A}_{(n\times r)}\mathsf{B}_{(r\times m)} \right) \right\|_F^2 , \tag{12.9}$$

est plus difficile que celui de factoriser une matrice pleine, car il n'admet pas de solution analytique. Même sans la présence de bruit, il s'agit d'une problème difficile. Un exemple typique de matrice avec données manquantes est donné à la figure 12.1.

Il existe trois grandes classes de méthodes de factorisation. La première classe est

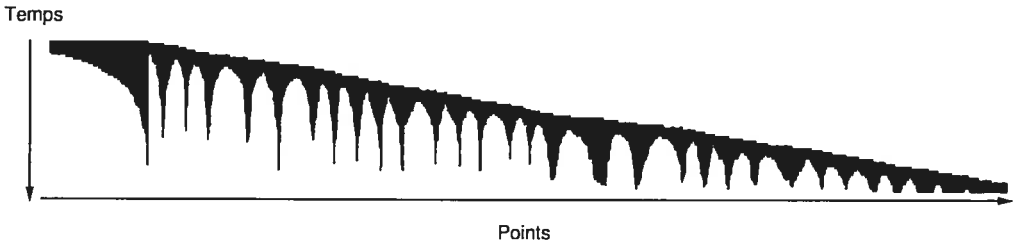


FIG. **12.1. Matrice de taille** (66×2483) **avec 96% de données manquantes. Les données disponibles sont en noir.**

constituée de méthodes itératives minimisant l'erreur de factorisation soit par des méthodes directes ou de façon alternée (*i.e.* fixer un des deux facteurs et raffiner le deuxième, jusqu'à convergence). Elles souffrent de deux inconvénients : elles sont assez lentes et convergent le plus souvent vers un minimum local [28]. La difficulté provient surtout du fait que les méthodes sont presque toujours initialisées à l'aide d'une solution aléatoire qui est automatiquement assez éloignées des minima globaux. Une description détaillée de ces méthodes est présentée à la référence [27].

La deuxième classe regroupe les méthodes hiérarchiques [14, 37, 101]. Ces dernières ont surtout été proposées pour fonctionner dans le contexte de la reconstruction 3D multi-vues, mais elles pourraient facilement être adaptées pour le problème général de factorisation. Nous les décrivons ici pour le contexte de reconstruction 3D. Une méthode hiérarchique procède en choisissant des sous-blocs complets de $\mathcal{M}$ de façon à avoir un certain chevauchement entre eux. Chacun d'eux est factorisé en utilisant la SVD donnant des reconstruction partielles. Ces reconstructions sont ensuite combinées entre elles de manière hiérarchique. La difficulté provient de l'ambiguïté de factorisation, *i.e.* chacune des reconstructions se trouve dans un système de coordonnées différent, ce qui requiert de les aligner les unes par rapport aux autres, et du fait que l'ordre dans lequel les reconstructions sont combinées influence la vitesse du processus et la qualité du résultat final.

La troisième classe regroupe les méthodes *batch*. L'objectif général de ces méthodes

est d'approcher le problème non-convexe original en séparant le calcul des deux facteurs en deux étapes. La première consiste à utiliser des sous-blocs complets de M desquels peuvent être extraites des contraintes. Celles-ci permettent l'estimation d'un des deux facteurs sous forme d'un problème convexe. La deuxième étape, plus facile, consiste en l'estimation du deuxième facteur à partir du premier.

Dans le reste de cette section, nous résumons huit méthodes de factorisation de matrice avec données manquantes.

12.3.1 PowerFactorization

L'algorithme *PowerFactorization* est l'une des méthodes itératives les plus simples [125]. Il s'agit d'une méthode d'alternation entre A et B avec l'ajout d'une étape d'orthonormalisation des colonnes de B à chaque itération. À l'itération k, l'algorithme effectue les trois opérations suivantes :

- $\mathsf{A}^{(k)} = \arg\min_{\mathsf{A}} \left\| \mathsf{W} \odot \left(\mathsf{M} - \mathsf{A}\mathsf{B}^{(k-1)} \right) \right\|$,
- $\mathsf{B}^{(k)} = \arg\min_{\mathsf{B}} \left\| \mathsf{W} \odot \left(\mathsf{M} - \mathsf{A}^{(k)}\mathsf{B} \right) \right\|$,
- Normalisation des colonnes de $\mathsf{B}^{(k)}$ par décomposition QR.

Il a été démontré que dans le cas de données complètes, cet algorithme itératif converge très rapidement vers un minimum global. Ce n'est plus le cas lorsque des données sont manquantes. Nos expériences et celles de Buchanan et Fitzgibbon [28] suggèrent en effet que la probabilité que l'algorithme converge vers un minimum global diminue rapidement à mesure que le pourcentage de données manquantes dans M augmente (*cf.*chapitres 13 et 14).

12.3.2 Approche Expectation-Maximization *(EM)*

Srebro et Jaakkola proposent une méthode itérative basée sur l'algorithme EM [35, 134]. Dans ce contexte, les paramètres observés sont les données connues de M, alors que les paramètres non-observées sont les données manquantes. Notons X, une matrice

de données correspondant aux données manquantes de M. Le processus consiste à alterner les étapes **E** et **M**, où l'itération k consiste en les étapes :

- $\mathbf{E} : \mathsf{M}' = (\mathsf{W} \odot \mathsf{M}) + \left((1 - \mathsf{W}) \odot \mathsf{X}^{(k-1)}\right)$
- $\mathbf{M} : \mathsf{X}^{(k)} = \mathbf{AR}_r(\mathsf{M}')$

où, M' est une matrice pleine et $\mathbf{AR}_r(\mathsf{M})$ est l'approximation de rang r de M selon un critère des moindres carrés.

Pour améliorer la convergence, les auteurs proposent de modifier les premières itérations pour que l'approximation de M' soit effectuée avec un rang plus grand que r, et graduellement diminué au cours du processus. Les étapes de l'algorithme modifié sont :

- $\mathbf{E} : \mathsf{M}' = (\mathsf{W} \odot \mathsf{M}) + \left((1 - \mathsf{W}) \odot \mathsf{X}^{(k-1)}\right)$
- $\mathbf{M} : \mathsf{X}^{(k)} = \mathbf{AR}_d(\mathsf{M}')$
- $d = \begin{cases} d - 1 & \text{si} \quad d > r \\ r & \text{sinon} \end{cases}$

où, d est choisi au départ selon $min(m, n)$, le minimum entre le nombre de rangées et le nombre de colonnes de M.

12.3.3 Factorisation robuste par re-pondération

AAnaes *et al.* proposent une approche itérative de factorisation robuste [1]. À la différence d'Anandan et Irani (§12.2.2), l'incertitude associée à chaque point de la séquence n'est pas donnée au départ. Elle est connue indirectement après une procédure de factorisation utilisant une fonction de coût robuste (fonction de Hubert ou moindre carré tronqué). En pratique, la minimisation de telles fonctions est difficile et nécessite l'utilisation de méthodes itératives. L'une d'entre elles est appelée *Moindres carrés re-pondérés de façon itérative*[2]. À la différence du problème classique de factorisation pondéré (*cf*.3.5.2), les valeurs des éléments de W varie au

[2] *Iterative Reweithed Least Squares* (IRLS)

cours du processus. Elles sont choisies afin que le problème de factorisation au sens des moindres carrés revienne à appliquer la fonction robuste à chaque élément de M. Comme ce poids est inconnu au départ, il est initialisé à $W_{ij} = 1$ si M_{ij} est un donnée connue et $W_{ij} = 0$ sinon. À chaque itération, W est gardé constant, permettant l'estimation de A et B à l'aide d'une méthode itérative. Ensuite, W est mis à jour en fonction du résidus de la solution courante, de sorte que le coût de factorisation correspond à celui de la fonction robuste. Les étapes suivantes sont effectuées à chaque itération :

- $\mathsf{A}^{(k)} = \arg\min_{\mathsf{A}} \left\| \mathsf{W}^{(k-1)} \odot \left(\mathsf{M} - \mathsf{A}\mathsf{B}^{(k-1)} \right) \right\|$,
- $\mathsf{B}^{(k)} = \arg\min_{\mathsf{B}} \left\| \mathsf{W}^{(k-1)} \odot \left(\mathsf{M} - \mathsf{A}^{(k)}\mathsf{B} \right) \right\|$,
- Ajuster les valeurs de W selon : $W_{ij}^{(k)} = \begin{cases} 1 & \text{si } \|e_{ij}\| \leq k \\ \|E/e_{ij}\| & \text{sinon} \end{cases}$

où, e_{ij} est l'erreur sur chaque élément de la solution courante et E est choisi par l'usager en fonction du bruit appréhendé dans les images. Il est bien connu que ce genre d'approche itérative ne converge pas nécessairement vers un minimum global de la fonction robuste. En pratique, un bon estimé de départ améliore grandement la probabilité de converger vers un minimum global. Une partie de nos contributions données au chapitre 14 porte sur ce sujet.

12.3.4 Minimisation non-linéaire

Buchanan et Fitzgibbon étudient l'utilisation de méthodes de minimisation non-linéaire comme Newton et la descente de gradient pour résoudre (12.9) [27, 28]. Leurs expériences suggèrent que l'algorithme *Newton pondéré*[3] est la méthode itérative qui converge le plus souvent au minimum global. Par ailleurs, ils proposent aussi des méthodes hybrides combinant alternation et méthode de Newton. Elles sont plus rapides et donnent des résultats similaires. Nous avons testé l'algorithme de Newton

[3] *Damped Newton*

pondéré aux chapitres 13 et 14.

12.3.5 Factorisation batch *pour le modèle perspectif*

La première méthode *batch* a été proposée par Triggs en 1997 pour le modèle perspectif [162]. Avant de l'aborder, il importe d'introduire une notation qui nous sera utile aussi dans les chapitres 13 et 14. Une matrice Π_j est une matrice diagonale de taille $(n \times n)$ amputée de ses rangés de telle sorte que $\Pi_j \mathcal{M}$ choisit les rangées de $\mathcal{M}$ correspondant aux points de l'image j de la séquence :

$$\hat{\mathcal{M}} = \Pi_j \mathcal{M} = \begin{bmatrix} \tilde{\mathbf{m}}_j^1 & \dots & \tilde{\mathbf{m}}_j^m \end{bmatrix}.$$

De façon similaire, on peut sélectionner les données des vues i et j en construisant une matrice Π_{ij}. Cette matrice de sélection peut aussi être appliquée aux matrices de projections, comme par exemple :

$$\begin{bmatrix} \mathsf{P}_i \\ \mathsf{P}_j \end{bmatrix} = \Pi_{ij} \mathcal{P}.$$

Triggs montre qu'à partir d'une matrice fondamentale F_{ji} entre deux vues j et i, il existe une relation linéaire

$$\mathsf{F}_{ji}\mathsf{P}_i + [\mathbf{o}_{ij}]_\times \mathsf{P}_j = \mathsf{F}_{ji}\Pi_i \mathcal{P} + [\mathbf{o}_{ij}]_\times \Pi_j \mathcal{P} = 0$$

où, $\mathbf{o}_{ij}$ est un épipôle. Celle-ci est appelée *contrainte de fermeture*[4]. Cependant, elle n'est valide que si les facteurs d'échelle de F_{ji} et $\mathbf{o}_{ij}$ sont choisis pour que :

$$\forall p : \mathsf{F}_{ji}(\lambda_{ip}\mathbf{m}_p^j) + [\mathbf{o}_{ij}]_\times(\lambda_p^j \mathbf{m}_p^j) = 0.$$

[4] *Closure constraint*

En d'autres mots, l'estimation des profondeurs projectives, tel que discutée §12.2.4, est encore nécessaire. Une contrainte similaire est aussi dérivée pour le tenseur trifocal. L'estimation de $\mathcal{P}$ est possible en combinant des contraintes reliant toutes les vues (en pratique, on minimise aux moindres carrés la somme des contraintes).

12.3.6 Factorisation batch *pour le modèle affine*

Guilbert *et al.* développent une méthode *batch* pour le modèle affine qui, bien sûr, ne nécessite par l'estimation de profondeur projectives [57]. Pour le modèle affine, la relation épipolaire a une forme simplifiée

$$\tilde{\mathbf{Q}}^{\mathsf{T}} \underbrace{\mathsf{P}_i^{\mathsf{T}} \mathsf{F}_{ji} \mathsf{P}_j}_{\mathsf{S}} \tilde{\mathbf{Q}} = 0, \quad \text{où,} \quad \mathsf{F}_{ij} \propto \begin{bmatrix} 0 & 0 & a \\ 0 & 0 & b \\ c & d & e \end{bmatrix},$$

où, $\mathbf{Q}$ est un point 3D euclidien et S est symétrique gauche[5] de la forme $\begin{bmatrix} 0_{(3\times3)} & \mathbf{a} \\ -\mathbf{a}^{\mathsf{T}} & 0 \end{bmatrix}$. Celle-ci permet de dériver quatre contraintes affines de fermeture sur les éléments de $\mathcal{P}$

$$\underbrace{\begin{pmatrix} a & b & c & d \end{pmatrix}}_{\mathbf{D}^{\mathsf{T}}} \begin{bmatrix} \mathsf{P}_i \\ \mathsf{P}_j \end{bmatrix} = \mathbf{D}^{\mathsf{T}} \Pi_{ij} \mathcal{P} = \begin{pmatrix} 0 & 0 & 0 & -e \end{pmatrix}$$

permettant d'estimer la trajectoire de la caméra de façon similaire à la méthode de Triggs (§12.3.5), Nous avons testé cette méthode au chapitre 14.

12.3.7 Factorisation batch *de Martinec et al.*

Martinec et Pajdla proposent une méthode de factorisation assez similaire à notre contribution du chapitre 14 [89]. Les différences entre leur approche et la nôtre y sont discutées en détails et une comparaison empirique est donnée. Les deux mêmes auteurs

[5] *Skew symmetric*

présentent aussi des variantes de leur approche pour le modèle perspectif calibré mais qui ne sont pas des méthodes de factorisation à proprement parler [90, 91].

12.3.8 Alternation perspective

Mahamud *et al.* proposent une méthode de factorisation alternée pour le modèle perspectif [88]. Pour éviter l'estimation des profondeurs projectives, l'algorithme procède en alternant la résection de caméra[6] et la triangulation des points 3D selon un critère algébrique [60]. Chaque itération revient à résoudre pour chaque image p :

$$\mathsf{P}_p^{(k)} = \arg\min_{\mathsf{P}} \sum_j \left\| w_p^j [\mathbf{m}_p^j]_\times \mathsf{P} \mathbf{Q}_j^{(k-1)} \right\|^2, \quad \text{tel que} \quad \|\mathsf{P}\|_F = 1$$

puis, pour tous les points 3D, résoudre :

$$\mathbf{Q}_j^{(k)} = \arg\min_{\mathsf{Q}} \sum_p \left\| w_p^j [\mathbf{m}_p^j]_\times \mathsf{P}_p^{(k)} \mathbf{Q} \right\|^2, \quad \text{tel que} \quad \|\mathsf{Q}\|_2 = 1$$

où, les points images $\mathbf{m}_p^j$ sont ici en coordonnées projectives et les contraintes servent à éviter la solution triviale $\mathsf{P}_p^{(k)} = 0, \mathbf{Q}_j^{(k)} = \mathbf{0}$. Cependant, l'espace des solutions pouvant être donné par cet algorithme n'est pas compacte, *i.e.* certaines solutions sont physiquement impossibles. D'après les auteurs, l'algorithme ne converge jamais vers une telle solution. Nous avons toutefois observé le contraire, lorsque la solution initiale est choisie de manière aléatoire. Cet algorithme est donc seulement utile pour raffiner une solution donnée par un algorithme de factorisation classique. Nous avons utilisé cet algorithme pour raffiner nos reconstruction 3D perspective au chapitre 14.

[6] Déterminer la position des caméras à partir de correspondances entre les points image et les points 3D.

12.4 Contributions

Les méthodes de factorisation *batch* ont été introduites dans le contexte de la reconstruction non-calibrée. Cependant, elles n'ont jamais été testées dans un cadre plus général. La dernière étude comparative des méthodes de factorisation avec données manquantes, présentée par Buchanan et Fitzgibbon, n'impliquent que les méthodes itératives. Notre première contribution, présentée au chapitre 13, est une comparaison empirique entre l'utilisation de méthodes *batch* et l'utilisation de solutions aléatoires pour l'initialisation de méthodes itératives. Nos résultats suggèrent que les méthodes *batch* améliorent significativement la probabilité de converger vers un minimum global.

De plus, nous généralisons les contraintes de fermeture de Triggs et de Guilbert *et al.* , pour le problème de factorisation général. Nous introduisons aussi de nouvelles *contraintes de base*[7] et montrons que la combinaison des deux types de contrainte améliore la qualité des résultats.

Au chapitre 14, nous étudions les deux types de contrainte pour le problème de reconstruction, en portant une attention particulière au modèle affine. Nous montrons que les contraintes de fermeture et de base peuvent être utilisées pour estimer non seulement les matrices de caméra (comme Triggs et Guilbert *et al.*), mais aussi la structure 3D. Nous montrons aussi que l'estimation robuste de chacune des contraintes permet l'estimation robuste des facteurs. Finalement, notre contribution la plus importante, est sans doute de montrer de façon expérimentale que l'utilisation des contraintes de base est supérieure à celle de contraintes de fermeture.

---

[7] *Basis constraints*

Chapitre 13

BATCH ALGORITHMS FOR MATRIX FACTORIZATION WITH MISSING DATA

Tardif J.P., Bartoli A., Roy S. Batch Algorithms for Matrix Factorization with Missing Data soumis à la conférence Conference on Neural Information Processing Systems (NIPS)

La réponse est attendue pour le 5 septembre 2007.

Abstract

Matrix factorization with missing data is a nonlinear optimization problem for which there is no algorithm guaranteed to find the optimal solution. This is typically solved by launching a some iterative Nonlinear Least Squares solvers from multiple random or ad hoc starting points. This is unreliable and time consuming. We propose a class of algorithms which find suboptimal factors in two rounds of convergent optimization, typically Linear Least Squares. They build on a recently proposed scheme of finding complete blocks in the data matrix from which constraints on one of the factors are formed. We generalize the original Closure Constraints and propose dual constraints we call Basis Constraints. A careful study of the two optimization rounds is conducted, and a method for finding the complete blocks is provided. Extensive experimental results show that our algorithms usually find close to optimal factors, while being extremely fast compared to standard Nonlinear Least Squares optimization. They allow the latter to reliably converge to the optimal solution while saving a great deal of computational time compared to using multiple starting points.

13.1 Introduction

Matrix factorization is an essential tool for solving several machine learning and computer vision problems. These include linear dimensionality reduction, Principal Component Analysis, Structure-from-Motion [141, 159], non-rigid 3D reconstruction [24], illumination based reconstruction [62] and motion segmentation [167]. When no data is missing and no constraint must be imposed on the factors, an efficient algorithm based on Singular Value Decomposition (SVD) can be used. However, missing data are unavoidable in many real-life situations, and it makes factorization much more difficult.

In this paper, we present batch methods for the factorization of a matrix with missing data. We use complete blocks of the data matrix to compute *closure* and (reduced) *basis* constraints on the first factor. They make possible the estimation of the factors trough Linear Least Squares (LLS). We describe three different algorithms : one using only closure, another one using only basis and, finally, one combining both. Our results suggest that this latter solution is generally more stable. The whole process (including block search) is performed within seconds of computation. The solution given by these algorithms can then be used as a starting point for iterative algorithms. We show that the initial results given by the batch methods are so close to the global minimum that a simple alternation [125] approach will converge to (or very close to) the global minimal more than 50 percents of the runs (see §13.5). Furthermore, the overall computation is done within a few minutes. Iterative algorithms are usually initialized using random solutions. However, simple alternation algorithms seldom converge to the global minimum. To achieve a better convergence rate, more sophisticated approaches, such as Damped Newton, must be used. They require several minutes or hours of computation and do not achieve this kind of performance [28].

Organization. The problem statement and related work are discussed in §§13.2 and 13.3. We describe our algorithms in §13.4. Closures and basis constraints are respectively derived in §§13.4.1 and 13.4.2. In §13.4.4, we describe a simple and efficient algorithm for finding complete blocks in the data matrix. Finally, experiments are discussed in §13.5 followed by the conclusion in §13.6.

13.2 Problem statement

Notation. Matrices are in *sans-serif*, *e.g.* M. Vectors are in bold, *e.g.* $\mathbf{v}$. The matrix operator $\odot$ is the Hadamard element-wise product and $\|\mathsf{M}\|$ is the Frobenius norm of M.

Problem statement. The factorization of a matrix M with missing data is formulated as the problem of finding a binary weighted approximation of M with the closest matrix AB of rank r :

$$\min_{\mathsf{A},\mathsf{B}} \|\mathsf{W}_{(n\times m)} \odot \left(\mathsf{M}_{(n\times m)} - \mathsf{A}_{(n\times r)}\mathsf{B}_{(r\times m)}\right)\|^2, \tag{13.1}$$

possibly with constraints on A and B. We call M the data matrix. Zero entries in the binary matrix W indicate missing data in M. The factorization is unique only up to the gauge (the coordinate frame) : A and B can be replaced respectively with AC and $\mathsf{C}^{-1}\mathsf{B}$ where C is any full rank $(r \times r)$ matrix representing the gauge ambiguity, as long as the constraints on the factors are still satisfied, without changing the factorization error. It is well known that when all data are available and when A et B are unconstrained, an optimal solution can be obtained from the SVD of M. Let $\mathsf{M} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$ be the SVD of M. One possible solution is $\mathsf{A} = \mathsf{U}(\Sigma')^{1/2}$ and $\mathsf{B} = (\Sigma')^{1/2}\mathsf{V}^\mathsf{T}$, where Σ' corresponds to the r first rows and columns of Σ.

13.3 Related Work

Algorithms for matrix factorization can be divided into two main categories : iterative and batch. Iterative techniques are the most popular. They use either an alternation scheme[1] [77, 104, 125] or minimize directly the factorization error by non-linear optimization [28]. In many cases, they are easy to implement and can handle various cost functions and constraints. However, their convergence can be quite slow, especially when initialized far from a minimum. This is a typical situation since a common practice is to initialize the algorithms with a random starting point. More importantly, this increases the chances of falling into a local minimum. For that reason, one usually performs several runs with different starting points.

Batch or direct methods aim at providing a good starting point to iterative approaches. Instead of minimizing directly the factorization error, they proceed into two (convex) steps that approximate the factorization error. Firstly, they use the data matrix to get constraints on one of the two factors. Secondly, they estimate the second factor based on the estimate obtained for the first one, much like an alternation approach, but without iterations. So far, it seems that these approaches have only been used in the context of Structure-from-Motion (SFM) [57, 89, 162]. Indeed, this problem can often be expressed in terms of rank matrix factorization [141, 159]. The data matrix consists of feature coordinates from images, rescaled by their projective depth. Because of occlusions, this matrix often has a lot of missing data. The data matrix needs to be factored into the camera projection matrices stacked on top of another (A) and the 3D coordinates of the features (B). For the case of point features, A and B are rank 4.

This paper brings several contributions. Firstly, we provide a formulation of the constraints that is not only applicable to the SFM problem, but to all unconstrained

[1] Alternation scheme refers to freezing one factor, estimating the other one and iterating until convergence.

factorization problems. We show that altough using basis constraints lead to very large Linear Least Squares problem, they can be estimated efficiently by taking advantage of the block structure of the matrix. We also derive a new constraint called *reduced basis* constraint. Using it instead of regular *basis* constraints slightly reduce the quality of the solution. However, it simplifies the implementation of the algorithm, as well as further increase the computation speed. Finally, we propose to combine together closure and basis constraint. This only increase the computation time by a fraction, but gives more stable results on average.

13.4 Batch Factorization

13.4.1 Closure Constraints

Closure Constraints were introduced in the vision community by Triggs [162] as a solution for the perspective Structure-from-Motion problem. They provide an efficient way of estimating the camera motion without the structure. Below, we give a general formulation of this constraint.

Let $\hat{\mathsf{M}}$ be a complete block of M, *i.e.* without missing data. Selecting a subset of rows is done by multiplying to the left by some row-amputated block-diagonal matrix Π. Selecting a subset of columns is done similarly by multiplying to the right by Γ, an identity matrix amputated of some of its columns, yielding :

$$\hat{\mathsf{M}}_{(\hat{n}\times\hat{m})} \stackrel{\text{def}}{=} \Pi\mathsf{M}\Gamma. \tag{13.2}$$

The SVD $\hat{\mathsf{M}} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$ can be used to compute optimal rank-r factors given by *e.g.* the leading r columns of U and r rows of $\Sigma\mathsf{V}^{\mathsf{T}}$. The first factor gives a basis for $\hat{\mathsf{M}}$, while the remaining columns of U form a basis for the left kernel or 'r-dimensionality closure[2]' of $\hat{\mathsf{M}}$, denoted $\hat{\mathcal{N}}$. This kernel only exists if $\hat{\mathsf{M}}$ has more than r rows. We

[2] This is the original term used by Triggs [162].

have :

$$\hat{\mathcal{N}}^{\mathsf{T}}\hat{\mathsf{M}} = \hat{\mathcal{N}}^{\mathsf{T}}\Pi\mathsf{A}\mathsf{B}\Gamma = 0. \tag{13.3}$$

When the data is corrupted by noise, $\hat{\mathcal{N}}^{\mathsf{T}}$ is the best approximation for the left kernel since it minimizes $\|\hat{\mathcal{N}}^{\mathsf{T}}\hat{\mathsf{M}}\|$. Furthermore, since both $\Pi\mathsf{A}$ and $\mathsf{B}\Gamma$ are at most rank r, any element in the left kernel of $\hat{\mathcal{N}}^{\mathsf{T}}$ lies in the left kernel of $\Pi\mathsf{A}$. This is the *Closure Constraint* on A given by :

$$\mathcal{N}^{\mathsf{T}}\mathsf{A} = 0 \quad \text{with} \quad \mathcal{N}^{\mathsf{T}} = \hat{\mathcal{N}}^{\mathsf{T}}\Pi.$$

We note that the sparsity of $\mathcal{N}$ is directly related to the block size. In practice, choosing small to medium size blocks ensures that $\mathcal{N}$ is highly sparse (see §13.4.4).

Solving through homogeneous Linear Least Squares. The closure constraint is the basis for a simple algorithm for estimating A using many blocks of the data matrix. The closure constraints with matrices $\mathcal{N}_1, \ldots, \mathcal{N}_l$ gives A, up to the $r \times r$ gauge ambiguity. Because of noise $\mathcal{N}^{\mathsf{T}}\mathsf{A}$ does not completely vanish and we seek the best solution for A in terms of Least Squares :

$$\arg\min_{\mathsf{A}} \|\mathsf{N}\mathsf{A}\|^2 \text{ s. t. } \operatorname{rank}(\mathsf{A}) = r \quad \text{with} \quad \mathsf{N}^{\mathsf{T}} = (\mathcal{N}_1 \ \cdots \ \mathcal{N}_l). \tag{13.4}$$

A solution to this problem is provided by taking the last r columns of V where[3] $\mathsf{N} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$ is the SVD of N. In general however, N is very sparse so one can use more efficient solutions. The first one is to find the eigenvectors of $\mathsf{N}^{\mathsf{T}}\mathsf{N}$ with an algorithm based on *Implicitly Restarted Arnoldi Methods* [5, 78], as proposed by Martinec *et al.* [89]. Such an algorithm is available in the ARPACK library and provided by the *eigs* function in MATLAB.

[3] Thanks to the gauge ambiguity, A can be made column orthonormal without loss of generality. Consider for instance the QR factorization $\mathsf{A} = \mathsf{Q}\mathsf{R}$, we may use $\mathsf{A}' = \mathsf{Q}$ and $\mathsf{B}' = \mathsf{R}\mathsf{B}$, with $\mathsf{A}\mathsf{B} = \mathsf{A}'\mathsf{B}'$.

Solving through affine Linear Least Squares. A second solution that closely approximates (13.4), but which is more effective, is to transform the homogeneous linear problem into an affine LLS problem (*i.e.* with a non-vanishing right hand side). Indeed, one can fix an $r \times r$ block of A (possibly non-contiguous) to some random rank r matrix such as the identity matrix $\mathrm{I}_{r\times r}$. By removing the corresponding columns of N, we obtain a full rank design matrix. Denote ρ a set consisting of r indices smaller or equal to n. Our new LLS problem is :

$$\arg \min_{\mathsf{A}_{\{1\ldots n\}\setminus\rho}} \|\mathsf{N}^{\{1\ldots n\}\setminus\rho}\mathsf{A}_{\{1\ldots n\}\setminus\rho} + \mathsf{N}^{\rho}\mathsf{A}_{\rho}\|^2 \tag{13.5}$$

where N^{ρ} are the columns of N with indices in ρ, A_{ρ} are the rows of A with indices in ρ and '$\setminus$' is the set difference. Choosing A_{ρ} fixes the gauge. Note that the rank constraint on matrix A is not necessary with this affine formulation. Finally, if required, the columns of A are made orthonormal, using *e.g.* its QR decomposition.

System (13.5) is a sparse Linear Least Squares problem that can be solved using sparse QR factorization. Without noise, this gives the true solution, thanks to the ambiguity of the factorization. Indeed, the choice we make for A_{ρ} is unimportant because it only consists of fixing the ambiguity of the factorization. As a result, $\mathsf{A}_{\{1\ldots n\}\setminus\rho}$ is full rank. With noise, the choice ρ of columns and the gauge induce slightly different solutions since this corresponds to assuming that A_{ρ} is noise-free. One may try a few ones and keep the solution that minimizes (13.4). In the following, this algorithm is referred to 'CC'.

13.4.2 Basis constraints

Closure constraints use only the left kernel of $\hat{\mathsf{M}}$. The basis of $\hat{\mathsf{M}}$ given by the r leading columns of U (recall the $\hat{\mathsf{M}} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$ is an SVD) is dropped. We propose constraints dual to the closures. Denote $\bar{\mathsf{U}}$, the r leading columns of U. There exists

an $r \times r$ full rank alignment matrix Z such that :

$$\Pi A = \bar{U} Z. \tag{13.6}$$

We call this equation a *Basis Constraint*. In other words, $\bar{U}$ is a basis of a portion of A corresponding to block $\hat{M}$. We give two ways of solving A using this constraint. The first one eliminates the aligning matrices Z, while the second one draws on the special sparse block structure of the problem.

Solving through the Reduced Basis Constraints. Denote Π^i some r rows of Π and Π^j the other ones. Assume for notational simplicity, and without loss of generality, that Π^i simply is the r first rows of Π. One can similarly split the rows of $\bar{U}$ in U^i and U^j. Expanding (13.6), we get :

$$\begin{pmatrix} \Pi^i \\ \Pi^j \end{pmatrix} A = \begin{pmatrix} U^i \\ U^j \end{pmatrix} Z = \begin{pmatrix} U^i \\ U^j \end{pmatrix} U^{i-1} \underbrace{U^i Z}_{Z'} = \begin{pmatrix} I \\ U^j U^{i-1} \end{pmatrix} Z'.$$

From this, we eliminate matrix Z and obtain the *Reduced Basis Constraint* :

$$(\Pi^j - U^j U^{i-1} \Pi^i) A = 0.$$

A similar constraint is used in [91] for the special case where M has $2r$ rows and contains rotation matrices. This equation above is the general form. This approach has the same problem as the closure constraint in that it requires to fix the gauge, which may change the result.

Solving through block arrowhead matrix inversion. This approach is based on the full basis constraints. Solving for A provided that all the bases are aligned to

be in the same coordinate system amounts to minimizing :

$$\sum_{k=1}^{l} \|\bar{\mathsf{U}}_k \mathsf{Z}_k - \Pi_k \mathsf{A}\|^2 = \sum_{k=1}^{l} \left\| \begin{pmatrix} \Pi_k & -\bar{\mathsf{U}}_k \end{pmatrix} \begin{pmatrix} \mathsf{A} \\ \mathsf{Z}_k \end{pmatrix} \right\|^2, \tag{13.7}$$

which can be rewritten as :

$$\min_{\mathsf{A},\mathsf{Z}_1,\ldots,\mathsf{Z}_l} \left\| \underbrace{\begin{pmatrix} \Pi_1 & -\bar{\mathsf{U}}_1 & & 0 \\ \vdots & & \ddots & \\ \Pi_l & 0 & & -\bar{\mathsf{U}}_l \end{pmatrix}}_{\mathcal{D}} \begin{pmatrix} \mathsf{A} \\ \mathsf{Z}_1 \\ \vdots \\ \mathsf{Z}_l \end{pmatrix} \right\|^2 \text{ s. t. } \operatorname{rank}(\mathsf{A}) = r. \tag{13.8}$$

Note that even though the number of non-zero elements in $\mathcal{D}$ increases linearly with the number of constraints, its size, on the other hand, increases quadratically. Fortunately, matrix $\mathcal{D}$ need not be explicitly constructed at any time with the solution below.

Once again, (13.8) can be minimized as a sparse homogeneous system or by transforming it to a sparse affine LLS. This latter solution is particularly efficient because we can take advantage of the block structure of $\mathcal{D}$. The idea is to choose the gauge by fixing one of the aligning transformations, by *e.g.* setting $\mathsf{Z}_l = \mathrm{I}$. Let D_1 be the leading columns of $\mathcal{D}$ composed of the l Π_i, D_2 the next columns with the $(l-1)$ first $-\bar{\mathsf{U}}_i$ on its diagonal and $\mathcal{Z}$ the $(l-1)$ first Z_i. Then (13.8) is rewritten :

$$\arg\min_{\mathsf{A},\mathcal{Z}} \left\| \begin{pmatrix} \mathsf{D}_1 & \mathsf{D}_2 \end{pmatrix} \begin{pmatrix} \mathsf{A} \\ \mathcal{Z} \end{pmatrix} - \begin{pmatrix} 0 \\ \mathsf{B} \end{pmatrix} \right\|^2 \quad \text{with} \quad \mathsf{B} = \bar{\mathsf{U}}_l \mathsf{Z}_l,$$

which amounts to solving the normal equations :

$$\underbrace{\begin{pmatrix} \mathsf{D}_1^\mathsf{T}\mathsf{D}_1 & \mathsf{D}_1^\mathsf{T}\mathsf{D}_2 \\ \mathsf{D}_2^\mathsf{T}\mathsf{D}_1 & \mathsf{D}_2^\mathsf{T}\mathsf{D}_2 \end{pmatrix}}_{\mathsf{D}} \begin{pmatrix} \mathsf{A} \\ \mathcal{Z} \end{pmatrix} = \begin{pmatrix} 0 \\ \mathsf{D}_2^\mathsf{T}\mathsf{B} \end{pmatrix}. \tag{13.9}$$

Matrix D has an arrowhead shape. This kind of matrices frequently appears, *e.g.* in Orthogonal Distance Regression problems [19]. The solution for A can thus be very easily obtained by eliminating Z in the two coupled matrix equations, giving :

$$\left(\mathsf{D}_1^\mathsf{T}\mathsf{D}_1 - \mathsf{D}_2^\mathsf{T}\mathsf{D}_1\left(\mathsf{D}_2^\mathsf{T}\mathsf{D}_2\right)^{-1}\mathsf{D}_1^\mathsf{T}\mathsf{D}_2\right)\mathsf{A} = -\mathsf{D}_2^\mathsf{T}\mathsf{D}_1\left(\mathsf{D}_2^\mathsf{T}\mathsf{D}_2\right)^{-1}\mathsf{D}_2^\mathsf{T}\mathsf{B},$$

that can be used to solve A. The speed up is obtained by efficiently inverting the block diagonal matrix $\left(\mathsf{D}_2^\mathsf{T}\mathsf{D}_2\right)^{-1}$ blockwise. In practice, we do not need to solve for $\mathcal{Z}$, so there is no need for back-substitution of A in (13.9). We will later refer to this algorithm as 'BC'.

13.4.3 Combining Closure and Basis

It is natural to ask whether the information from both closures and basis constraints could be combined together. It turns out this is easy. Indeed, systems (13.4) and (13.8) are both homogeneous in A and some unknown alignment matrices Z_i. Thus, they can be easily combined together :

$$\min_{\mathsf{A},\mathsf{Z}_1,\ldots,\mathsf{Z}_l} \left\| \begin{pmatrix} \Pi_1 & -\bar{\mathsf{U}}_1 & & 0 \\ \vdots & & \ddots & \\ \Pi_l & 0 & & -\bar{\mathsf{U}}_l \\ \hat{\mathcal{N}}_1^\mathsf{T}\Pi_1 & 0 & \ldots & 0 \\ \vdots & \vdots & & \vdots \\ \hat{\mathcal{N}}_l^\mathsf{T}\Pi_l & 0 & \ldots & 0 \end{pmatrix} \begin{pmatrix} \mathsf{A} \\ \mathsf{Z}_1 \\ \vdots \\ \mathsf{Z}_l \end{pmatrix} \right\|^2 \quad \text{s. t. } \operatorname{rank}(\mathsf{A}) = r. \tag{13.10}$$

This is a LLS problem of the same type of (13.8), so it can be minimized efficiently as previously described. This algorithm will be referred as 'CBCC' in the experiments.

13.4.4 Finding Complete Blocks

Finding complete blocks in the data matrix is a key step in our algorithms. Algorithm 1 gives the pseudo-code for our simple randomized block search. Remember that a block does not need to be made of contiguous columns and rows. Even tough the algorithm is very simple, it can find enough blocks in the data matrix very quickly. The input of the algorithm are our data matrices M and W, the rank r of the factorization and λ a threshold for the minimum number of constraints associated to each row. A key feature of our algorithm is to distribute the constraint amongst the rows as evenly as possible. The idea is to sequentially scan the columns of the matrix. For each column, the non-empty rows are candidates for building a block. We randomly select between $r + 1$ and $3r$ rows and check whether these rows contains enough non empty columns to create a block. The $3r$ limit ensures the algorithm can build blocks easily. To accelerate the search and to evenly distribute the constraint among the rows, we count the constraint associated to each rows (in **rCount**). Thus, the rows with smallest count are first considered to build the next block. Blocks are added until all rows have a sufficient number of constraints. There is a chance a block will be selected twice, but in practice, the chances are low and the effect is negligible.

13.5 Experiments

Simulation. We compared our three algorithms of §§13.4.1,13.4.2,13.4.3, respectively named CC, BC and CCBC, on simulated data. We also tested the iterative algorithms PowerFactorization (PF) [125] and Damped Newton [28]. We used a 300×300 matrices with more than 95% of missing data. We ran each algorithm 30 times with

Algorithm 1 Algorithm for block search in M

input : $\mathsf{W}_{n\times m}$ $\mathsf{M}_{n\times m}$ r λ {minimal constraints associated to each row}

1: **loop**
2: **rCount**(*all rows*) $\leftarrow$ **0** {counter for blocks associated to the rows }
3: **for** $c \leftarrow 1$ to m **do**
4: $\mathbf{R} \leftarrow$ set of all rows s. t. $\mathsf{W}(\mathbf{R}, c) = 1$
5: $k \leftarrow$ random() s. t. $r < k \leq 3r$
6: $\mathbf{R} \leftarrow$ subset of k elements of $\mathbf{R}$, the ones with smallest **rCount**
7: **if rCount**$(\mathbf{R}) \geq \lambda$ **then**
8: continue {the rows $\mathbf{R}$ are used in enough blocks}
9: **end if**
10: $\mathbf{C} \leftarrow$ set of columns s. t. $\mathsf{W}(\mathbf{R}, \mathbf{C}) == \mathbf{1}$
11: **if** $|\mathbf{C}| \geq r$ **then**
12: append block $\mathsf{M}(\mathbf{R}, \mathbf{C})$ to **ouput**
13: **rCount**$(\mathbf{R}) \leftarrow$ **rCount**$(\mathbf{R}) + \mathbf{1}$
14: **end if**
15: **if rCount**(*all rows*) $\geq \lambda$ **then**
16: return
17: **end if**
18: **end for**
19: **end loop**

increasing noise level. The minimal, average and maximal rms error over all tests are shown in figure 13.1a. Notice that the iterative approaches do not appear in the result, since they reached 1000 iterations without converging close to a global minimum. Our three batch algorithms provide very low error with similar behavior on average. However, the slightly better error of CBCC suggests that it is more stable. We performed a second comparison, this time only, on the batch algorithms. We fixed the noise level to 4e-2 and varied the number of constraints. The results are shown in figure 13.1b. We observe that with only 3 constraints associated to each row, the algorithms are a somewhat unstable, especially the one based only closure (CC). For the rest larger number of constraints, using only basis (BC) seem to be better on average. On the other hand, combining both closures and basis (CCBC) provides

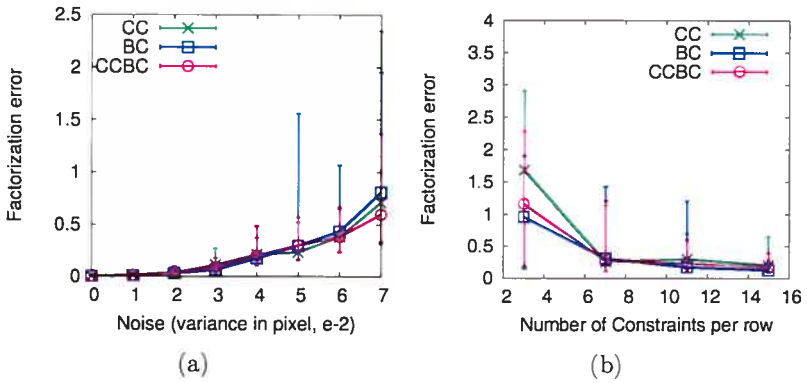


FIG. **13.1. Comparison on simulated data for the unconstrained factorization. a) All the algorithms with increasing noise level, b) All batch methods with an increasing number of constraint associated to the rows of the data matrix.**

better worst cases.

Real data. We tested our batch algorithms on the three examples of Buchanan *et al.* [28]. The problems are : a rank-4 Structure-from-Motion (the Dinosaur sequence, 72×319 matrix with 28% of known data), a rank-4 varying illumination based reconstruction of a face (20×2944 matrix with 58% known data), and, finally, a rank-6 non-rigid model reconstruction (the Giraffes sequence 240×167 matrix with 70% known data). For each problem, the rms global minimum is known.The results are summarized in figures 13.2 and 13.3. We show the rms error of all the runs for the batch algorithms, PowerFactorization initialized with random values (PF) and also with the solutions provided by the batch methods ($\cdot$+PF). The results for other more sophisticated algorithms such as Damped Newton or Hybrid methods combining Newton with alternation are given in [28].

We observe that as less data is available, the performance of our batch methods increase compared to iterative algorithms. It is especially good when combined to PowerFactorization. The global minimum is found most of the time and the computation time is below a few minutes. In the Dinosaur and Face examples, all runs converged

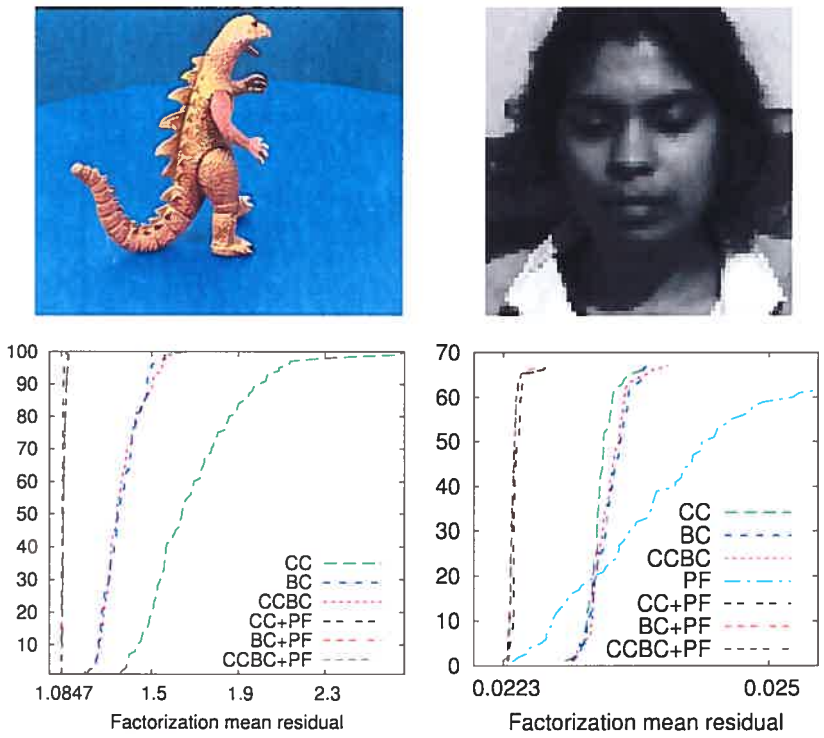


FIG. **13.2. Results for the structure from motion (left) and illumination based reconstruction (right). Top) Frame from each sequence. Bottom) Cumulative histograms of the rms error for all runs of the algorithms (run count vs rms error). Global minima are known to** 1.0847 **and** 0.0223 **respectively.**

to the global minimum or very close. On the other hand, Damped Newton, the best iterative algorithm, only achieved similar convergence 35 runs over a thousand and 160 over 500, respectively (see [28]). Furthermore, the batch methods always provide their solution in at most in a few seconds. Damped Newton takes minutes and even hours depending on the number of iterations. PowerFactorization is faster and takes only a few minutes of computation, but when initialized with random starting points, it fails to find the global minimum most of the time in all but the Giraffe sequence.

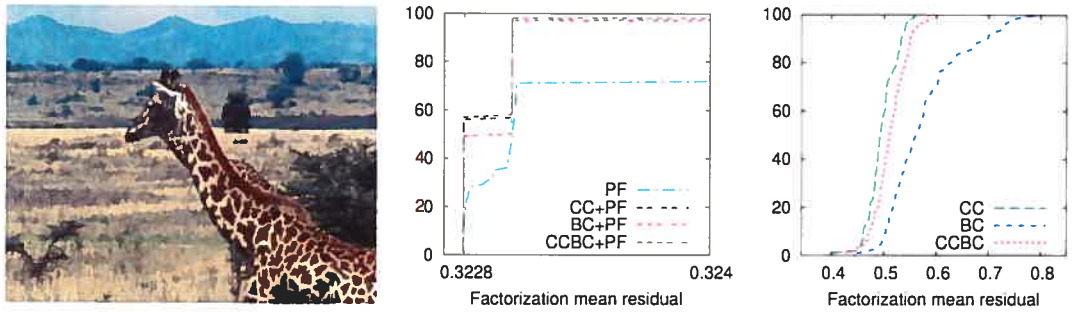


FIG. **13.3. Results for the non-rigid transformation tracking.** Left) **Sample input image.** Middle) **Cumulative histogram of the rms of the runs (run count vs rms error) near the global minimum.** Right) **Cumulative histogram for the batch algorithms without subsequent refinement with PowerFactorization. Global minimum is known to be** 0.3228.

13.6 Conclusion

Matrix factorization with missing data is a useful tool in several research fields, usually solved through iterative Nonlinear Least Squares minimization with multiple random starting points. The algorithms we propose are inspired from the Closure Constraint, which was derived by Triggs in 1996 in the context of Structure-from-Motion. We generalize the original ad hoc Closure Constraint to arbitrary matrices. We show that there exist a dual constraint we call the Basis Constraint. Effective matrix factorization algorithms are reported, based on the previously mentioned constraints, including a combination of both.

Experimental results demonstrate several important points. Firstly, our algorithms are reliable, in that they provide a reliable starting point from which the probability that the nonlinear methods reach the optimal solution is very high. This is achieved through two rounds of globally convergent optimization, typically Linear Least Squares. Secondly, our algorithms are fast. They find the solution orders of magnitude faster than a nonlinear method with multiple starting points, and with a higher probability of success.

We believe that the batch algorithms derived in this paper will have an important practical impact. One possible extension is the incorporation of constraints on the factors such as non-negativity.

Chapitre 14

ALGORITHMS FOR BATCH MATRIX FACTORIZATION WITH APPLICATION TO STRUCTURE-FROM-MOTION

Cet article [154] a été publié comme l'indique la référence bibliographique.

Tardif J.P., Bartoli A., Trudeau M., Guilbert N., Roy S., Algorithms for Batch Matrix Factorization with Application to Structure-from-Motion, dans IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Mineapolis, États-Unis, 2007.

Abstract

Matrix factorization is a key component for solving several computer vision problems. It is particularly challenging in the presence of missing or erroneous data, which often arise in Structure-from-Motion. We propose batch algorithms for matrix factorization. They are based on *closure* and *basis* constraints, that are used either on the cameras or the structure, leading to four possible algorithms. The constraints are robustly computed from complete measurement sub-matrices with *e.g.* random data sampling. The cameras and 3D structure are then recovered through Linear Least Squares. Prior information about the scene such as identical camera positions or orientations, smooth camera trajectory, known 3D points and coplanarity of some 3D points can be directly incorporated. We demonstrate our algorithms on challenging image sequences with tracking error and more than 95% missing data.

14.1 Introduction

Matrix factorization is an essential tool for solving several computer vision problems including Structure-from-Motion [141, 159], plane-based pose estimation [143], non rigid 3D reconstruction [24] and motion segmentation [167]. When no data are missing or corrupted by outliers, an efficient algorithm based on Singular Value Decomposition (SVD) can be used. However, missing and erroneous data are certainly unavoidable in many real-life situations, and they make factorization much more difficult. Furthermore, the SVD-based algorithm makes it difficult to enforce constraints specific to the formulation or provided by prior information about the problem.

We propose algorithms for batch matrix factorization with special attention given to the Structure-from-Motion (SfM) problem. *Closure* or *basis constraints* on one of the two factors are computed from complete measurement sub-matrices. For example, *Camera Closure Constraints* (CCC) can be computed from their left kernel [162]. We investigate three variations: *Camera Basis Constraints* (CBC), *Structure Closure Constraints* (SCC) and *Structure Basis Constraints* (SBC), that are respectively left basis, right kernel and right basis of the measurement matrix. Our experiments and comparison with other state-of-the-art batch factorization algorithm show that basis constraints usually gives better results than closures, for both affine and perspective camera model. We also show that structure constraints can be used first to compute the 3D structure instead of the cameras, which allows to enforce directly constraints such as known 3D points or known planar surface. In the case of affine SBC, an approximation to the reprojection error is minimized.

Organization. The next section reviews previous work on factorization and Structure-from-Motion and points out the main differences with ours. In §14.3, we give our notation and formally introduce the factorization problem with his specialization to the affine Structure-from-Motion problem. Camera Closure Constraints (CCC) are first reviewed in §14.3.1, then follows our contribution in §14.4. The details are provided

for the affine camera model and we discuss how the theory applies to the perspective model in §14.5. Robustness is discussed in §14.6, experiments are described and analyzed in §14.7, followed by conclusion in §14.8.

14.2 Previous Work

Structure-from-Motion can be formulated most generally as a bilinear (modulo homogeneous scale) inverse problem. The seminal work on affine factorization [159] however showed that it could be relaxed to a bilinear problem for the affine camera model. For perspective cameras, it can be formulated similarly when the homogeneous feature point coordinates are rescaled by their projective depth, which must be estimated *a priori* [89, 141].

The algorithms for matrix factorization despite missing data can be divided into three main categories: iterative, batch and hierarchical. In the first one, factorization is performed by minimizing directly the factorization error either by non-linear methods [28] or alternation [23, 88, 125]. However, the convergence to the global minimum is not guaranteed because they can get stuck in a local minimum. Although good performances have been reported when initializing the algorithms with a random solution [28, 74, 125], an initialization as close as possible to the global minimum is recommended. Hierarchical approaches proceed by factorizing overlapping sub-blocks of the measurement matrix [37, 101]. The solutions are then merged in a hierarchical manner, and care must be taken choosing the merging scheme. This allows to deal with very large factorization problems.

Batch algorithms provide a solution for initializing iterative algorithms with a low computational cost. They usually minimize an approximation to the reprojection error to simplify the optimization and avoid local minima [57, 89, 162]. Under zero noise, they find the global minimum. The approximation is done by computing the two factors through two linear steps. Constraints on one of the two factors are

computed to span the whole solution space. Once the first factor is estimated, the second one can be easily computed. Non-linear and batch algorithms are usually considered complementary solutions. In [131], essential matrices are used between pairwise views to estimate the motion of all the cameras without the structure. This is similar to our solution in philosophy although a very different approach is taken.

It has been shown that the reconstruction problem is considerably simplified when observing a reference plane in all the images of the sequence [73, 118]. The structure constraints we propose handle this situation naturally. The solution must still proceed in two steps, but has the benefit that the objects **do not** need to be visible in all of the images. It has been noted in [90] that very few reconstruction algorithms can use two image tracks, which occur when using a sparse set of images with wide-baseline. Our solution is not subject to this limitation. Finally, robustness is enforced one constraint at a time, rather than globally [1, 4, 74], thereby allowing the use of RANSAC-type algorithms.

14.3 Notation and Preliminaries

Notation. Matrices are in *sans-serif*, *e.g.* M, and joint matrices in calligraphic characters, *e.g.* $\mathcal{M}$. Vectors are always in bold, *e.g.* $\mathbf{v}$. The matrix operator $\odot$ is the Hadamard element-wise product. Finally, $\mathbb{P}$ represents the projective space.

Problem statement. Factorization of a matrix M with missing data is formulated as the problem of finding a weighted approximation of M with the closest rank r matrix (AB) such that:

$$\min_{\mathsf{A},\mathsf{B}} \|\mathsf{W}_{(n\times m)} \odot \left(\mathsf{M}_{(n\times m)} - \mathsf{A}_{(n\times r)}\mathsf{B}_{(r\times m)}\right)\|, \tag{14.1}$$

where M is called the *measurement matrix* composed of points $\mathbf{m}_p^j$, and W is a weighting matrix with zeros for missing elements in M. In some problems, constraints on

elements of A and B must be enforced, *e.g.* affine SfM. In SfM, B is called the Joint Structure Matrix (JSM) and represents the 3D points $\mathbf{q}^j \in \mathbb{P}^3$, $\mathsf{A} = (\mathcal{P}\ \mathbf{t})$ is the Joint Projection Matrix (JPM) and consists of the stacked camera projection matrices.

Affine SfM can be formulated as a rank-3 or a rank-4 factorization of a joint matrix $\mathcal{M}$, depending on whether the input matrix has missing data or not (with the exception of [23], where predictions are made for the missing data). In the rank-3 case, the projection can be expressed with:

$$\mathbf{m}^j_{p(2\times 1)} = \mathsf{P}_{p(2\times 3)}\mathbf{q}^j_{(3\times 1)} + \mathbf{t}_{p(2\times 1)} \tag{14.2}$$

where an optimal choice for the *joint translation vector* $\mathbf{t}$ can be computed as the column means of $\mathcal{M}$. It can be eliminated from (14.2), giving the centered measurement matrix $(\mathcal{M} - \mathbf{t}\mathbf{1}^\mathsf{T})$. The factorization of this matrix computed using SVD is an optimal solution in terms of the reprojection error [159]. We have rank-4 when data are missing, so the joint translation vector cannot be computed *a priori*. Bilinear matrix factorization [125] provides a solution as long as the last row of the JSM is constrained to unity:

$$\mathcal{M}_{(2n\times m)} = \begin{pmatrix}\mathcal{P}_{(2n\times 3)} & \mathbf{t}\end{pmatrix}\begin{pmatrix}\mathcal{Q}_{(3\times m)} \\ \mathbf{1}^\mathsf{T}\end{pmatrix}. \tag{14.3}$$

14.3.1 Camera Closure Constraints (CCC)

We review *Closure Constraints* [57, 68, 162] for affine cameras and give a generic formulation for estimating matching tensors between many views.

Deriving the constraint. Let $\hat{\mathcal{M}}$ be a sub-block of $\mathcal{M}$ without missing data. Selecting a subset of views is done by multiplying to the left by some row-amputated block-diagonal matrix Π with (2×2) identity blocks. Selecting a subset of features is done similarly by multiplying to the right by Γ, an identity matrix amputated of

some of its columns, yielding:

$$\hat{\mathcal{M}}_{(2\hat{n}\times\hat{m})} \stackrel{\text{def}}{=} \Pi\mathcal{M}\Gamma. \tag{14.4}$$

In this case, the measurements can be expressed as:

$$\hat{\mathcal{M}} = \Pi\mathcal{M}\Gamma = \Pi\mathcal{P}\mathcal{Q}\Gamma + \Pi\mathbf{t}\mathbf{1}^{\mathsf{T}}\Gamma = \hat{\mathcal{P}}\hat{\mathcal{Q}} + \tilde{\mathbf{t}}\mathbf{1}^{\mathsf{T}}. \tag{14.5}$$

We define $\boldsymbol{\mu}_m$ as:

$$\boldsymbol{\mu}_m \stackrel{\text{def}}{=} \frac{1}{m}\mathbf{1}_{(m\times 1)}, \tag{14.6}$$

which computes the column means of an $(n \times m)$ matrix by multiplying to the right. We define the centered measurement matrix as:

$$\bar{\mathcal{M}} \stackrel{\text{def}}{=} \hat{\mathcal{M}} - \hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\mathbf{1}^{\mathsf{T}}_{(1\times\hat{m})}, \tag{14.7}$$

which does not equal $(\hat{\mathcal{P}}\hat{\mathcal{Q}})$ in general, as the row means of the sub-blocks are not necessarily those of the complete matrix. The SVD $\bar{\mathcal{M}} = \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}$ can be used to compute optimal rank-3 factors given by the leading 3 columns of U and 3 rows of $\Sigma\mathsf{V}^{\mathsf{T}}$. The first factor gives the partial Joint Projection Matrix $\hat{\mathcal{P}}$ while the remaining columns of U form a basis for the best approximation to the left kernel of $\bar{\mathcal{M}}$, which we call *centered matching tensor*, denoted $\bar{\mathcal{N}}$. We have:

$$\bar{\mathcal{N}}^{\mathsf{T}}\bar{\mathcal{M}} = \mathbf{0}, \tag{14.8}$$

and from (14.7):

$$\bar{\mathcal{N}}^{\mathsf{T}}\left(\hat{\mathcal{M}} \quad -\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\mathbf{1}^{\mathsf{T}}\right) = \mathbf{0}, \tag{14.9}$$

that rewrites as:

$$\underbrace{\begin{pmatrix} \bar{\mathcal{N}}^\mathsf{T} & -\bar{\mathcal{N}}^\mathsf{T}\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}} \end{pmatrix}}_{\mathcal{N}^\mathsf{T}} \begin{pmatrix} \hat{\mathcal{M}} \\ \mathbf{1}^\mathsf{T} \end{pmatrix} = \mathbf{0}, \tag{14.10}$$

where appears the non-centered matching tensor $\mathcal{N}$ we are seeking. Note that directly computing a tensor from $(\hat{\mathcal{M}}^\mathsf{T}\ \mathbf{1})^\mathsf{T}$ would not be optimal in terms of reprojection error. Tensor $\mathcal{N}$ corresponds to the classical affine matching tensors. The affine fundamental matrix has 4 degrees of freedom, and the non-centered left kernel obtained from a measurement matrix with four rows has 5 components up to scale. A matching tensor computed from a matrix with six rows is of size (3×6) and has orthonormal rows, which leaves 12 degrees of freedom like the affine trifocal tensor.

Estimating the Joint Projection Matrix. The unity constraint on the last row of the Joint Structure Matrix can be expressed with extra rows in $\mathcal{M}$ and $\mathcal{P}$:

$$\begin{pmatrix} \mathcal{M} \\ \mathbf{1}^\mathsf{T} \end{pmatrix} = \begin{pmatrix} \mathcal{P} & \mathbf{t} \\ \mathbf{0}_{(1\times 3)} & 1 \end{pmatrix} \begin{pmatrix} \mathcal{Q} \\ \mathbf{1}^\mathsf{T} \end{pmatrix}. \tag{14.11}$$

Let:

$$\mathsf{D}^\mathsf{T} \stackrel{\text{def}}{=} \bar{\mathcal{N}}^\mathsf{T}\Pi; \tag{14.12}$$

multiplying (14.11) by $\begin{pmatrix} \Pi & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix}$ to the left and Γ to the right and substituting in (14.10), we obtain:

$$\begin{pmatrix} \mathsf{D}^\mathsf{T} & -\bar{\mathcal{N}}^\mathsf{T}\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}} \end{pmatrix} \begin{pmatrix} \mathcal{P} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} = \mathbf{0}, \tag{14.13}$$

$$\boxed{\mathsf{D}^\mathsf{T} \begin{pmatrix} \mathcal{P} & \mathbf{t} \end{pmatrix} = \begin{pmatrix} \mathbf{0}_{(1\times 3)} & \bar{\mathcal{N}}^\mathsf{T}\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}} \end{pmatrix}.} \tag{14.14}$$

Stacking every such constraint computed from different sub-blocks of $\mathcal{M}$ in a single matrix equation, we get:

$$\begin{pmatrix} \mathsf{D}_1^\mathsf{T} \\ \vdots \\ \mathsf{D}_l^\mathsf{T} \end{pmatrix} \begin{pmatrix} \mathcal{P} & \mathbf{t} \end{pmatrix} = \begin{pmatrix} \mathbf{0} & \bar{\mathcal{N}}_1^\mathsf{T} \hat{\mathcal{M}}_1 \boldsymbol{\mu}_{\hat{m}} \\ \vdots & \vdots \\ \mathbf{0} & \bar{\mathcal{N}}_l^\mathsf{T} \hat{\mathcal{M}}_l \boldsymbol{\mu}_{\hat{m}} \end{pmatrix}. \tag{14.15}$$

The design matrix, denoted $\mathcal{D} \stackrel{\text{def}}{=} (\mathsf{D}_1, \ldots, \mathsf{D}_l)^\mathsf{T}$, is highly sparse. This is exploited when computing the Linear Least Square (LLS) solution. As explained in [57], choosing the projection matrix of some of the cameras fixes the gauge of the system and ensures a full-rank design matrix. The error minimized by this method is difficult to interpret because it is expressed in terms of matching tensors. Once the JPM is estimated, the structure can be computed by affine triangulation.

14.4 Batch Matrix Factorization for Affine SfM and Generalization of CCC

14.4.1 Summary of the Algorithms

The algorithms we propose follow the typical steps of batch algorithms: 1) Measurement sub-matrices without missing data are found. 2) Each of these matrices is used to compute a constraint, either on the partial Joint Projection Matrix or partial Joint Structure Matrix. 3) The constraints are combined to estimate one of the factors. 4) The second factor is estimated by camera resectioning or triangulation. 5) Finally, non-linear or alternation methods refine the solution.

14.4.2 Camera Basis Constraints (CBC)

We show how basis constraints can be used instead of matching tensors. The partial JPM $\mathcal{P}$ can be computed alone, *i.e.* without the joint translation vector. This is done

by aligning bases of the projection matrices of partial reconstructions performed on measurement sub-matrices. Once $\mathcal{P}$ is recovered, the joint translation vector and the structure can be computed together to minimize the reprojection error.

Consider a centered sub-matrix $\bar{\mathcal{M}}$ of $\mathcal{M}$ and compute its SVD $\bar{\mathcal{M}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$. The 3 leading columns of U, denoted $\bar{\mathsf{U}}$, form a basis of $\hat{\mathcal{P}}$, that is, there is a 3×3 invertible matrix Z (the aligning transformation) such that:

$$\hat{\mathcal{P}} = \bar{\mathsf{U}}\mathsf{Z}, \tag{14.16}$$

leading to the Camera Basis Constraint:

$$\boxed{\Pi\mathcal{P} = \bar{\mathsf{U}}\mathsf{Z}.} \tag{14.17}$$

Because of the remark following (14.7), this is not trivial. To demonstrate this, we note that multiplying an $(n \times m)$ matrix by:

$$\boldsymbol{\vartheta}_m \stackrel{\text{def}}{=} \mathrm{I} - \frac{1}{m}1_{(m\times m)} = \mathrm{I} - \boldsymbol{\mu}_m \mathbf{1}_{(1\times m)}, \tag{14.18}$$

to the right subtracts the row mean from each of its entries. Consequently:

$$\bar{\mathcal{M}} = \Pi\mathcal{M}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = \Pi\mathcal{P}\mathcal{Q}\Gamma\boldsymbol{\vartheta}_{\hat{m}} + \Pi\mathbf{t}\mathbf{1}^\mathsf{T}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = \Pi\mathcal{P}\mathcal{Q}\Gamma\boldsymbol{\vartheta}_{\hat{m}}, \tag{14.19}$$

since $\Pi\mathbf{t}\mathbf{1}^\mathsf{T}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = 0$. Hence, the columns of $\bar{\mathcal{M}}$ are linear combinations of those of $\Pi\mathcal{P}$, and since $\bar{\mathcal{M}}$ has rank 3, the same is true of $\bar{\mathsf{U}}$.

Solving for the Joint Projection Matrix

Computing many CBC's for different sub-blocks of the measurement matrix amounts to performing partial affine reconstructions. Solving for the Joint Projection Matrix is done by: 1) aligning basis constraints to recover the reduced JPM and 2) recovering

the joint translation vector and the structure.

Let l be the number of CBC's. Aligning bases together involves minimizing:

$$\sum_{k=1}^{l} \|\bar{\mathsf{U}}_k \mathsf{Z}_k - \Pi_k \mathcal{P}\|^2 = \sum_{k=1}^{l} \left\| \begin{pmatrix} \Pi_k & -\bar{\mathsf{U}}_k \end{pmatrix} \begin{pmatrix} \mathcal{P} \\ \mathsf{Z}_k \end{pmatrix} \right\|^2, \tag{14.20}$$

which is a simple LLS system, that can be rewritten as:

$$\left\| \underbrace{\begin{pmatrix} \Pi_1 & -\bar{\mathsf{U}}_1 & & \mathbf{0} \\ \vdots & & \ddots & \\ \Pi_l & \mathbf{0} & & -\bar{\mathsf{U}}_l \end{pmatrix}}_{\mathcal{D}} \begin{pmatrix} \mathcal{P} \\ \mathsf{Z}_1 \\ \vdots \\ \mathsf{Z}_l \end{pmatrix} \right\|^2. \tag{14.21}$$

Although the size of the design matrix $\mathcal{D}$ is quadratic in the number of bases, the equation system can be minimized efficiently. Indeed it is extremely sparse, and we do not need to compute the aligning matrices Z_k. The minimized error is the alignment of the optimal cameras of partial reconstructions. Although this error is algebraic, as when expressing the error in terms of matching tensor constraints, our experiments suggest it is more stable, *cf*.§14.7.

Once the partial JPM $\mathcal{P}$ is estimated, we propose two approaches for estimating the translations and the structure. The best solution comes from doing both at the same time, by using an LLS formulation. It minimizes the reprojection error. This is because the orientation and intrinsics of the camera are already estimated up to a (3×3) invertible transformation G, which has no effect on the minimized error:

$$\min_{\mathbf{t},\mathcal{Q}} \|\mathcal{M} - \mathcal{P}\mathcal{Q} - \mathbf{1}^\mathsf{T}\mathbf{t}\| = \min_{\mathbf{t},\mathcal{Q}} \|\mathcal{M} - \mathcal{P}\mathsf{G}\mathsf{G}^{-1}\mathcal{Q} - \mathbf{1}^\mathsf{T}\mathbf{t}\|. \tag{14.22}$$

However, for a long sequence, this equation system uses a lot of memory and is rather long to minimize. In this case, it is more efficient to estimate only the joint translation

vector and then perform individual triangulation for each feature track. To this end, we combine the computed basis $\bar{\mathsf{U}}$ with the translation $\bar{\mathbf{t}} = \hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}$ of the cameras of the partial reconstructions. Thus, the camera alignment can also be performed with:

$$\Pi\begin{pmatrix}\mathcal{P} & \mathbf{t}\end{pmatrix} = \begin{pmatrix}\hat{\mathcal{P}} & \tilde{\mathbf{t}}\end{pmatrix} = \begin{pmatrix}\bar{\mathsf{U}} & \bar{\mathbf{t}}\end{pmatrix}\begin{pmatrix}\mathsf{Z} & \mathbf{v} \\ \mathbf{0}_{(1\times 3)} & 1\end{pmatrix} \tag{14.23}$$

, where matrices $\hat{\mathsf{U}}$ and Z are those from (14.17). The joint translation vector can be estimated by minimizing:

$$\left\| \mathcal{D}\begin{pmatrix}\mathbf{t} \\ \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_l\end{pmatrix} - \begin{pmatrix}\bar{\mathbf{t}}_1 \\ \vdots \\ \bar{\mathbf{t}}_l\end{pmatrix} \right\|^2, \tag{14.24}$$

where $\mathcal{D}$ is the design matrix of (14.21).

In [89], bases for cameras were computed from $\hat{\mathcal{M}}$, not $\bar{\mathcal{M}}$ like we do. The partial reconstruction corresponding to these cameras is not optimal. Furthermore, a method for estimating $\tilde{\mathbf{t}}$ from the bases independently of the structure is not given. This is essential when dealing with very large sequences or data corrupted by outliers (*cf.*§14.6).

14.4.3 Structure Closure Constraints (SCC)

Structure Closure Constraint is the analogue of the CCC applied to the Joint Structure Matrix. From (14.3), a matching tensor should be in the column space of $\mathcal{Q}$ and $\mathbf{1}^\mathsf{T}$. It can be computed using SVD [60] by minimizing:

$$\boxed{\|\hat{\mathcal{M}}\mathcal{N}\|^2, \text{ subject to } \mathbf{1}^\mathsf{T}\mathcal{N} = 0.} \tag{14.25}$$

Note that unlike the CCC, the SCC is not computed from a centered input matrix. This is because the last row of the JSM must be $\mathbf{1}^\mathsf{T}$s. A more formal explanation is given in §14.4.4. By accumulating many constraints, we can form a design matrix $\mathcal{D}$ with:

$$\mathsf{D}_i^\mathsf{T} \stackrel{\text{def}}{=} \mathcal{N}_i^\mathsf{T} \Gamma^\mathsf{T}. \tag{14.26}$$

By construction $\mathcal{D}$ is rank deficient because each of its rows vanishes on $\mathbf{1}^\mathsf{T}$. Hence, the right eigenvector corresponding to the smallest singular value, equal to zero, is $\mathbf{1}/\|\mathbf{1}\|$. The estimate for $\mathcal{Q}^\mathsf{T}$ is given by the next three right eigenvectors of $\mathcal{D}$.

14.4.4 Structure Basis Constraints (SBC)

As with Camera Closures, using Structure Closures implies minimizing a purely algebraic function difficult to interpret. Structure Bases can be used to estimate partial reconstructions which are then aligned together in a single computation. From an SVD of $\bar{\mathcal{M}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$, the three leading columns of V estimate the structure, up to an affine transformation, in the partial reconstruction corresponding to $\hat{\mathcal{M}}$. However we prefer using $\bar{\mathsf{V}}$ as the three leading columns of $\mathsf{V}\Sigma^\mathsf{T}$, as explained below. It can be aligned with the structure through:

$$\hat{\mathcal{Q}}^\mathsf{T} = \mathsf{Z}_{(3\times 4)} \begin{pmatrix} \bar{\mathsf{V}} & \mathbf{1} \end{pmatrix}^\mathsf{T}, \tag{14.27}$$

leading to the Structure Basis Constraint:

$$\boxed{\Gamma^\mathsf{T} \mathcal{Q}^\mathsf{T} = \mathsf{Z}\bar{\mathsf{V}}^\mathsf{T}.} \tag{14.28}$$

Note that unlike a CBC, an SBC cannot be aligned with a (3×3) matrix. This is because the row space of $\bar{\mathsf{V}}$ is that of:

$$\Pi\mathcal{P}\mathcal{Q}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = \Pi\mathcal{P}\begin{pmatrix}\hat{\mathsf{Q}}^1\boldsymbol{\vartheta}_{\hat{m}}\\ \hat{\mathsf{Q}}^2\boldsymbol{\vartheta}_{\hat{m}}\\ \hat{\mathsf{Q}}^3\boldsymbol{\vartheta}_{\hat{m}}\\ \mathbf{1}^\mathsf{T}\boldsymbol{\vartheta}_{\hat{m}}\end{pmatrix} = \Pi\mathcal{P}\begin{pmatrix}\hat{\mathsf{Q}}^1 - \hat{\mathsf{Q}}^1\boldsymbol{\mu}_{\hat{m}}\mathbf{1}_{(1\times m)}\\ \hat{\mathsf{Q}}^2 - \hat{\mathsf{Q}}^2\boldsymbol{\mu}_{\hat{m}}\mathbf{1}_{(1\times m)}\\ \hat{\mathsf{Q}}^3 - \hat{\mathsf{Q}}^3\boldsymbol{\mu}_{\hat{m}}\mathbf{1}_{(1\times m)}\\ \mathbf{0}^\mathsf{T}\end{pmatrix}, \tag{14.29}$$

where the $\hat{\mathsf{Q}}^i$'s are the rows of $\hat{\mathsf{Q}}$, that is, the row space of $\bar{\mathsf{V}}$ is the one generated by the $\hat{\mathsf{Q}}^i$'s and $\mathbf{1}^\mathsf{T}$, but not necessarily only by the $\hat{\mathsf{Q}}^i$'s. Partial reconstructions can be aligned together by solving an equation system similar to (14.21).

Choosing the bases. Consider two partial reconstructions $\bar{\mathsf{V}}$ and $\bar{\mathsf{V}}'$. Aligning them together amounts to finding:

$$\min_{\mathsf{Z}} \|\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}}\|^2. \tag{14.30}$$

We show that when $\bar{\mathsf{V}}$ is chosen so that the corresponding projection matrix $\hat{\mathcal{P}}$ is orthonormal, the 3D error approximates the reprojection error [14]. Projecting the residual given by (14.30) into the images corresponding to the block, we obtain:

$$\|\hat{\mathcal{P}}\bar{\mathsf{V}}' + \mathbf{1}^\mathsf{T}\tilde{\mathbf{t}} - (\hat{\mathcal{P}}\mathsf{Z}\bar{\mathsf{V}} + \mathbf{1}^\mathsf{T}\tilde{\mathbf{t}})\|^2 = \|\hat{\mathcal{P}}\bar{\mathsf{V}}' - \hat{\mathcal{P}}\mathsf{Z}\bar{\mathsf{V}}\|^2 \tag{14.31}$$

$$= \|\hat{\mathcal{P}}(\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}})\|^2. \tag{14.32}$$

Thanks to the orthonormal property $\hat{\mathcal{P}}^\mathsf{T} = \hat{\mathcal{P}}^\dagger$, our error function simplifies to:

$$\mathrm{tr}\left((\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}})^\mathsf{T}\underbrace{\hat{\mathcal{P}}^\mathsf{T}\hat{\mathcal{P}}}_{\mathsf{I}_{(3\times 3)}}(\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}})\right) = \|\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}}\|^2. \tag{14.33}$$

The minimization is only exact if both $\bar{\mathsf{V}}$ and $\bar{\mathsf{V}}'$ have been estimated without error in their respective partial reconstruction. Hence, under noise, it only approximates

the reprojection error.

14.4.5 Enforcing Constraints

In many situations, prior knowledge about the configuration of the scene structure is available. Examples are two cameras with identical position and/or orientation, smooth camera path, known 3D points and planar structure. In our algorithms, a certain number of constraints can be enforced. This is done when estimating the first factor. As a consequence, one can only force constraints on either cameras or structure.

Most of our equation systems are homogeneous, like (14.21). In order to simplify the constrained optimization, a well known trick is to select a gauge, *i.e.* to give a value to some rows of X, and solve the resulting regression problem. A valid gauge fixes the degrees of freedom of the affine ambiguity, which makes the original design matrices rank deficient. Once this is done, each column $\mathbf{X}^i$ of the solution matrix X can be individually estimated by regression under linear constraints, which is an instance of convex quadratic programming [22]. Aligning two cameras together can be done by choosing:

$$\begin{pmatrix} \mathsf{P}_a & \mathbf{t}_a \end{pmatrix} = \begin{pmatrix} \mathsf{P}_b & \mathbf{t}_b \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \tag{14.34}$$

$$\begin{pmatrix} \mathsf{P}_c & \mathbf{t}_c \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ * & * & * & * \end{pmatrix} \tag{14.35}$$

Others can also be aligned through equality constraints $\mathsf{P}_e - \mathsf{P}_f = 0$, or variable elimination.

Enforcing known 3D points can be done similarly with structure constraints. The gauge is fixed with at least four non-coplanar points. Three planar surfaces can also be enforced by forcing groups of points to have their (X, Y, Z) coordinates to

either $(0,*,*)$, $(*,0,*)$ or $(*,*,0)$, where $*$ means the coordinate is not fixed[1]. For more than three planes, their absolute equation have to be known. Observe that the gauge is at least partially fixed by the points located on the planes. Consequently, caution must be taken to ensure that these planes are really orthogonal up to an affine transformation. Prior information from points and planar structures can also be incorporated as long as the constraints are compatible, *i.e.* the affine ambiguity is fixed by the same matrix.

14.5 The Perspective Camera Model

In projective SfM, the point coordinates are multiplied by their projective depth λ_p^j and the projection is performed by (3×4) matrices defined up to scale:

$$\lambda_p^j \begin{pmatrix} \mathbf{m}_p^j \\ 1 \end{pmatrix} = \begin{pmatrix} \mathsf{P}_{p(3\times3)} & \mathbf{t}_{p(3\times1)} \end{pmatrix} \mathbf{q}_{(4\times1)}^j \tag{14.36}$$

where $\mathbf{m}_p^j \in \mathbb{R}^2$ is an interest point tracked throughout the sequence and $\mathbf{q}^j \in \mathbb{R}^3$. We assume that the projective depths are estimated *a priori*. In our experiments, we used the algorithm presented in [89]. Rank-4 factorization is then performed without any restriction on $\mathbf{q}^j$, unlike for the affine case. Closures and rank-4 bases are purely algebraic and are computed from $\hat{\mathcal{M}}_{(3\tilde{n}\times\tilde{m})} = \Pi\mathcal{M}\Gamma$ instead of $\bar{\mathcal{M}}$. This is akin to what was presented in [89]. The main difference is the way system (14.21) is minimized. They used Matlab's EIGS method to estimate four orthonormal solution vectors. We used a solution based on fixing one of the bases $\bar{\mathsf{U}}_i$ to identity and solve the resulting sparse regression problem. Finally, we performed a QR factorization on the estimated JPM to orthonormalize its columns. It was suggested in [89] that gluing via points cannot be used with the perspective camera model. We did not encounter this limitation.

[1] Details will be provided in a technical report.

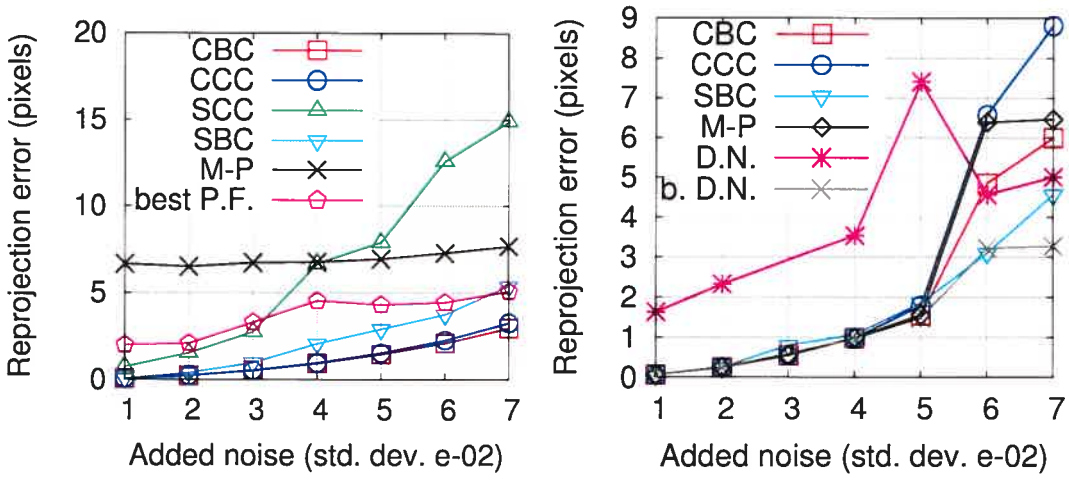


Figure 14.1. Comparison of the algorithms for the simulated sequence. M-P stands for Martinec-Pajdla [89], P.F. for Powerfactorization [125], D.N. for Damped Newton [28] and b. for the best solution out of the 15 trials. Left) Affine model. Right) Perspective model (SCC not shown).

14.6 Robustness

To deal with outliers, we use random sampling to compute each Closure/Basis Constraint. Once the camera path or 3D points have been computed, we rely again on random sampling to perform robust triangulation or resection. We do not reject points directly from the computed matching tensors, which can leave certain outliers behind. We avoid performing the optimization using a robust (non-convex) cost function, that would typically have a lot of local minima, or using alternation with re-weighting.

For CBC, we rely on (14.24) to compute the position of the cameras, because the estimation of the structure and the translation vectors in a single step (*i.e.* using (14.22)) would be computationally expensive in a random sampling strategy.

14.7 Experiments and Analysis

We compared our three algorithms to Guilbert *et al.* [57], Martinec-Pajdla [89], Hartley-Schaffalitzky [125] and Buchanan-Fitzgibbon [28] methods on simulated and real sequences. Powerfactorization and Damped Newton were used respectively for the affine and perspective camera model. They were limited to 1000 iterations, which was sufficient to attain convergence from a random solution in most cases. The tracks were sorted in order of appearance in the sequence and we assumed that the resulting measurement matrix was approximately band-diagonal as in figure 14.4. Heuristics were used to find complete sub-blocks, making sure that constraints are approximately equally distributed among the cameras or the 3D points, depending on the constraint type.

Simulation. The simulated sequence consisted of 50 cameras and around 900 3D points with a lot of occlusion, resulting in around 96% missing data. In figure 14.1, the algorithms are compared for both camera models at different levels of Gaussian noise. Our results strongly suggest that Camera Basis performs best, especially under affine projection. Under perspective projection, Closure and Structure Basis gave similar results with a slight advantage to SBC in the presence of very high noise. The advantage vanished when projective depths were exact (not shown here). Hence, SBC seems to be the most robust to erroneous projective depths. The CBC method performed slightly better than Martinec-Padjla's, a likely result of our balancing of the constraints. Structure Closures performed rather poorly. This is not surprising, at least for the affine case, since it is the only one of our algorithms whose constraint is not optimal. Powerfactorization and Damped Newton provided good performance in the best case, but on average, they did not converge to satisfying solutions.

Most of the computation time was spent finding complete sub-blocks from the measurement matrix. The time for solving the two factors was almost negligible. Hence, a good algorithm performs well even with a small number of constraints. We

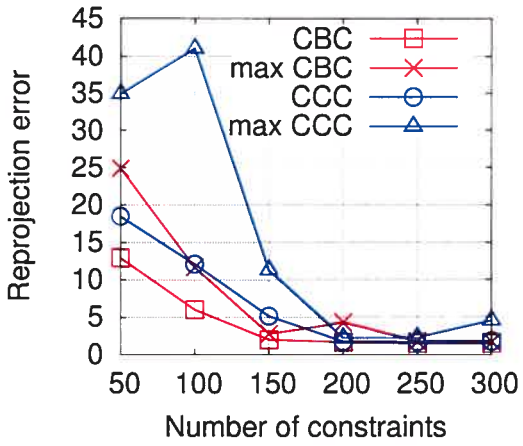


Figure 14.2. Comparison for the number of constraints between CCC and CBC, for our simulated sequence.

compared CCC and CBC on this criteria with the simulated sequence as well as with the *Teddy Bear* sequence (*cf*.figure 14.2 and 14.3(a)). The number of closures had to be nearly twice as large as that of bases to obtain a comparable average reprojection error. The maximal reprojection error also suggests more stability for bases.

Real sequences. We compared batch algorithms and Powerfactorization on five real sequences, four affine and one perspective, (*cf*.table 14.1). To take them out

Sequence (# Img.,# 3D pts, # 2D pts, miss. data)	Mean (max) reprojection error in pixels					
	CCC [57]	CBC	SCC	SBC [14]	P.F. [125]	M-P [89]
Dinosaur (36,2683,11832,96.9%)	0.56 (5.49)	**0.49 (4.62)**	0.65 (7.27)	0.66 (7.12)	1.75 (73.1)	0.56 (6.99)
Book (95, 254, 10253, 89%)	0.54 (6.86)	**0.50 (5.59)**	0.55 (5.25)	0.56 (5.66)	0.54 (5.96)	2.56 (41.1)
Building (194, 779, 17233, 97%)	**0.86 (14.4)**	0.95 (22.8)	1.24 (17.5)	1.21 (21.5)	(3.45) 256.8	1.28 (39.2)
Teddy Bear (196, 2480, 93589, 95%)	**0.65 (8.14)**	**0.65 (8.14)**	4.67 (174.5)	1.13 (35.3)	1.91 (38.8)	4.453 (96.97)
Desk (66, 2483, 26771, 95.9%)	0.99 (43.36)	**0.87 (19.5)**	3.93 (132.66)	1.44 (45.18)	—	**0.83 (24.4)**

Table 14.1. Comparison between batch methods and Powerfactorization for four real sequences. All were reconstructed using the affine camera model except for the *Desk* sequence. The best solutions are in bold.

of the comparison, we removed the outliers in each of the sequence using the robust CBC-based algorithm (see test below). For the *Dinosaur* sequence, the algorithm of Guilbert *et al.* and Martinec-Pajdla obtained a better mean reprojection error than what they reported in their paper, respectively 5.4 and 2.57 pixels. This is probably because we used more constraints (around 350). Computation time, in Matlab, for the camera estimation (not including sub-block search) were 0.01 and 0.29 seconds for CCC and CBC (for which five different gauges were tested) and 0.34 seconds for Martinec-Pajdla. On the larger *Teddy bear* sequence, computation time were respectively 0.49, 0.73, 0.91 seconds (not including the translations since it was much longer minimizing (14.22) than (14.24)). Structure Constraint based algorithms were slower because their equation systems are larger. Iterative algorithms achieved convergence after a few hundred iterations, resulting in minutes, if not hours, of computation. When initialized using a batch method, only a few iterations were necessary.

The *Desk* sequence (*cf*.figure 14.4, 14.5 and 14.6(c)) was reconstructed using the perspective algorithms and the one based on affine CBC. In the former case, outliers were removed before factorization using fundamental matrices. All batch algorithms but the one based on SCC provided satisfying results. The reconstruction shown in figure 14.6(c) is the result of refining the solution with Mahamud *et al.* method followed by self-calibration. Projective and Euclidean bundle adjustment improved the reconstruction only slightly. We also achieved reconstruction by initializing a Euclidean bundle adjustment with the robust algorithm based on affine CBC. After convergence, the recovered focal length was similar to the one recovered using projective reconstruction.

Outliers issue. The performance of CCC and CBC for handling outliers was also tested similarly to [130]. Up to 7 % outliers where added to the *Dinosaur* sequence. This gives up to $7n$ % unusable n-view constraints (we used up to 4). The constraints

were computed from robustly selected sub-blocks. Thus, as the number of outliers increased, the number of points used to compute the constraints decreased. We tested: 1) the percentage of recovered valid outliers, 2) the percentage of removed points that were in fact inliers (false positive) and 3) the average reprojection error of the reconstruction using the original data set. Our results are shown in figure 14.3(b) and 14.3(c).

14.8 Conclusion

We presented algorithms for efficient batch matrix factorization. Constraints on measurement sub-matrices are combined to estimate one of the two factors. We extended Trigg's Camera Closure Constraints to Structure Closure Constraints and proposed Camera and Structure Basis. Experimental results showed that Basis Constraints fared better than state-of-the-art methods on most of our tests, with simulated and real sequences and both for affine and perspective camera models. Future work will focus on the measurement sub-matrix search and selection mechanism, and on experiments for enforcing *a priori* on the structure.

Acknowledgments. To A. Buchanan and D. Martinec for providing their source code, to A. Fitzgibbon and A. Zisserman for the Dinosaur sequence and to K. McHenry and G. Petit for the Teddy bear sequence.

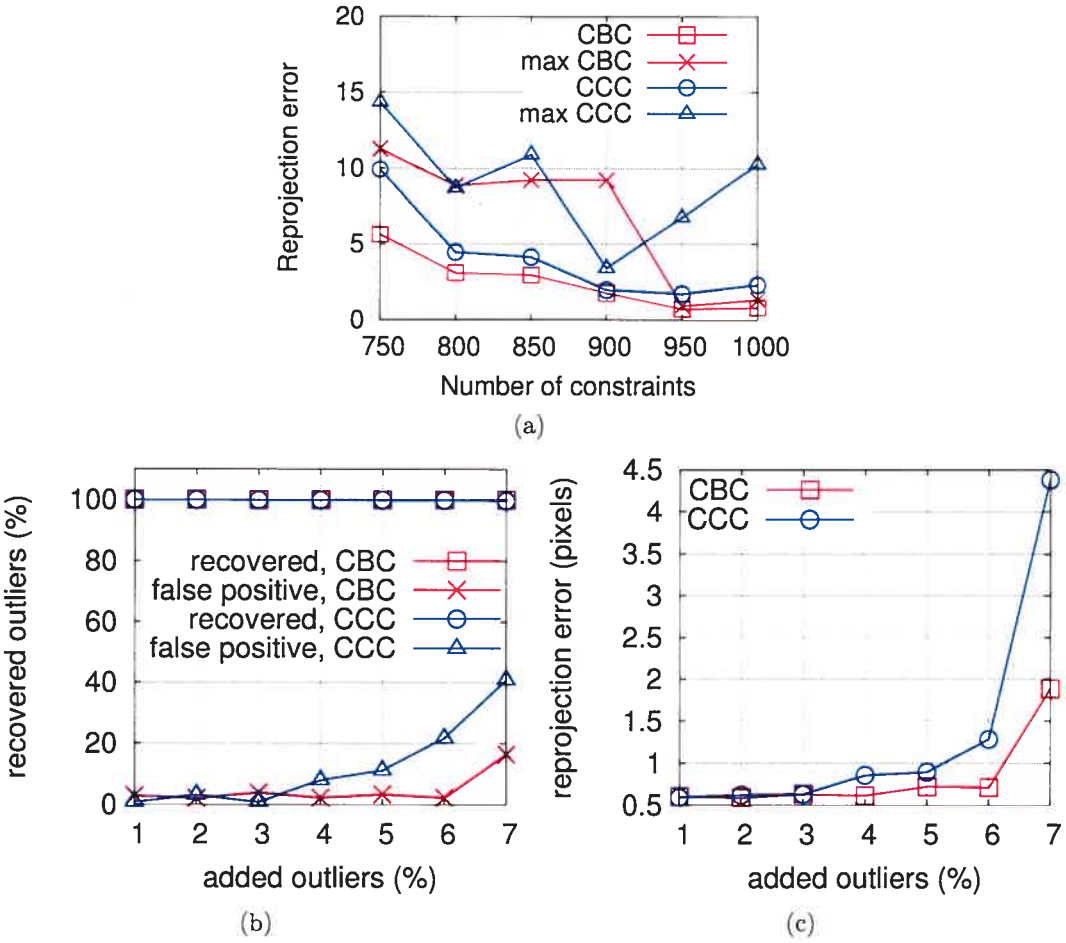


Figure 14.3. Comparison between CCC and CBC with real data. a) number of constraints for the *Teddy Bear* sequence. For added outliers in the *Dinosaur* sequence: b) percentage of outliers recovered and percentage of false positive, c) average reprojection error of the original data.

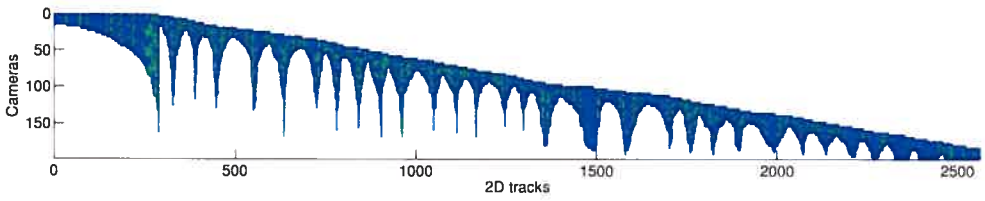


Figure 14.4. Tracks (blue/dark) and detected outliers (green/light) from the *Desk* sequence.

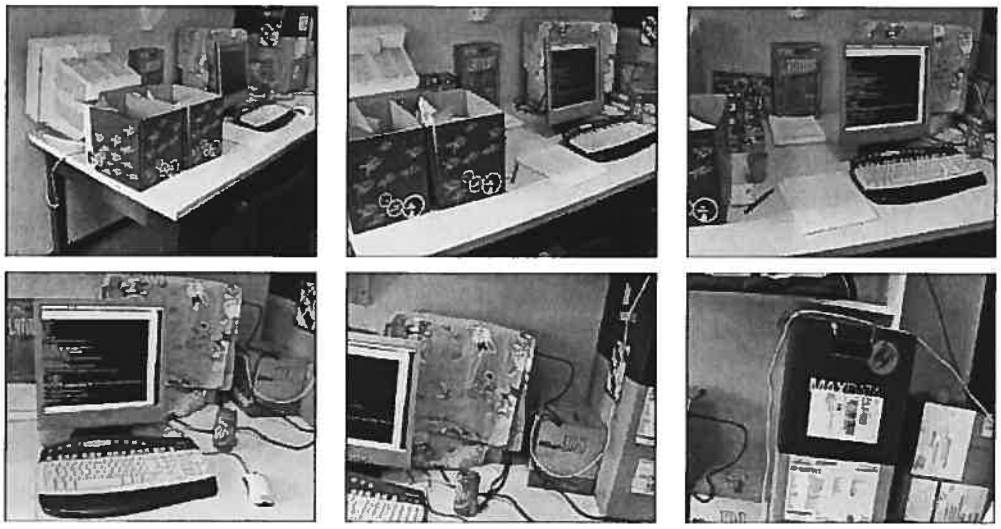


Figure 14.5. Five out of 66 images of the $Desk$ sequence. Reconstruction shown in 14.6(c).

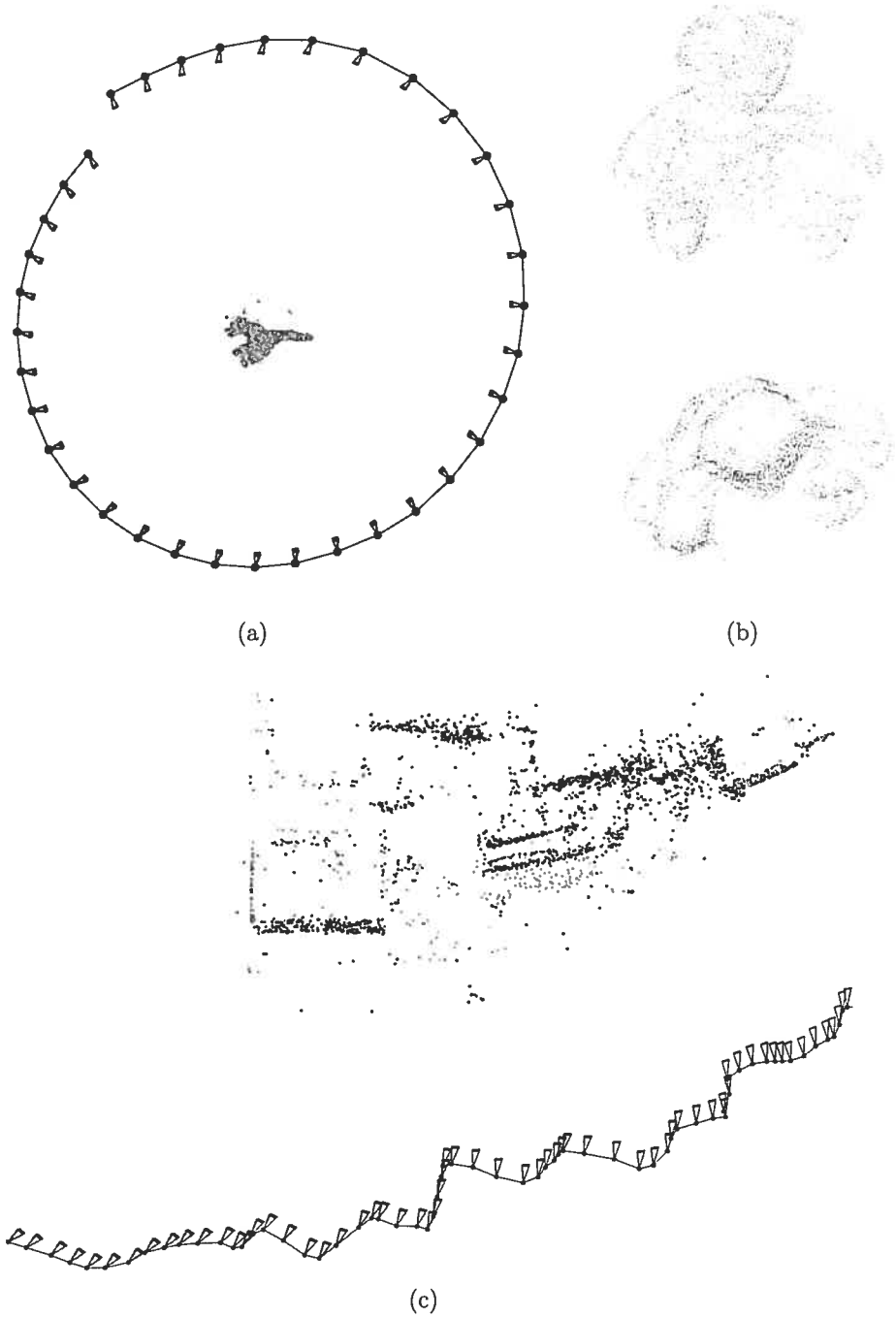


Figure 14.6. Reconstruction results. a) **Top view of the** *Dinosaur* **sequence without using a closing constraint.** b) **Side and top view of the** *Teddy Bear* **model.** c) **Top view of the** *Desk* **sequence. All but the** *Desk* **sequence were obtained solely with an algorithm based on CBC.**

Chapitre 15

CONCLUSION ET PERSPECTIVES

Nous résumons tout d'abord nos contributions. Puis, nous décrivons certains problèmes ouverts reliés à nos travaux ainsi que des perspectives de recherches que nous croyons prometteuses.

15.1 Résumé

▷ **Calibrage et auto-calibrage de caméra omnidirectionnelle**

Les améliorations technologiques des dernières années ont permis l'augmentation de l'angle de vue des caméras, permettant de couvrir la totalité de l'espace environnant avec très peu d'images. Cependant, les différentes variantes de ces caméras nécessitent généralement l'emploi de modèles de caméra spécifiques à chaque design. Notre première contribution est un modèle de caméra permettant de représenter efficacement une très grande partie des caméras. Le reste de nos contributions sont des algorithmes de calibrage et d'auto-calibrage pour différentes variantes du modèle : projection centrale ou non-centrale, et distorsions paramétrique ou non-paramétrique. Par ailleurs, nos expériences suggèrent que l'emploi d'une fonction de distorsion non-paramétrique procure des résultats plus précis et, de façon surprenante, produit parfois des algorithmes plus stables. Cependant, l'utilisation de fonctions de distorsion et de déplacement du centre optique, toutes deux discrètes, peut donner des résultats assez instables.

▷ **Reconstruction 3D active**

Les projecteurs multimédia, coûteux et encombrants encore récemment, sont de plus en plus abordables et de petite taille. Leur utilisation dépasse déjà la

projection classique ; il existe aujourd'hui plusieurs prototypes de système de reconstruction 3D et de système multi-projecteurs. Mais pour ne pas rester confinées aux laboratoires de recherche, les méthodes de reconstruction doivent pouvoir fonctionner dans des environnements pour lesquels le contrôle est limité : éclairage variable ou inconnu, contraste très bas dans les images, projecteurs situés très loin de l'écran. Dans ce genre de circonstances, le problème de mise en correspondance est plus difficile même pour des méthodes actives utilisant la lumière structurée. Notre contribution à cet égard est une formulation probabiliste du problème de décodage des motifs, permettant de combiner la cohérence spatiale des codes à la binarisation traditionnelle des images de motifs. Ceci réduit le nombre d'erreurs de décodage par rapport à un décodage classique en plus de permettre un lissage intelligent des codes qui préserve les discontinuités.

▷ **Algorithmes de factorisation de matrice avec données manquantes**
La très grande majorité des méthodes de factorisation de matrice avec données manquantes procède de manière itérative. Lorsque le pourcentage de données manquantes est très élevé (plus de 80%), elles sont lentes et donnent des résultats qui sont souvent insatisfaisants. Ceci vient surtout du fait que la minimisation est démarrée à partir de solutions choisies de façon aléatoire, ce qui rend nécessaire de répéter le processus plusieurs fois afin d'espérer trouver un minimum global. Conséquemment, la taille des problèmes pouvant être résolus est grandement limitée. Nos contributions portent sur une méthode de factorisation *batch* combinant des contraintes de fermeture généralisées et de nouvelles contraintes de base. L'utilisation de notre méthode pour initialiser un algorithme itératif permet d'obtenir des taux de convergence au minimum global supérieur à 50 % dans le cas de problèmes difficiles, et supérieur à 90% dans le cas de problèmes plus faciles. Par ailleurs, nous avons fait ces observations

en utilisant un des algorithmes itératifs les plus simples, soit PowerFactorization. De plus, notre méthode *batch* est très rapide, ne nécessitant que quelques secondes pour les petits problèmes ou quelques minutes pour les plus grands. Ainsi, il n'y pratiquement aucune pénalité en terme de temps de calcul à l'utiliser comme méthode d'initialisation. Nos contraintes et notre algorithme peuvent aussi être formulés pour le problème de reconstruction 3D affine qui présente la particularité de nécessiter l'ajout de contraintes sur le deuxième des deux facteurs. Dans cette situation, nos résultats expérimentaux montrent la supériorité des contraintes de base vis-à-vis des contraintes de fermeture. Celle-ci se situe surtout au niveau du nombre de contraintes nécessaires à l'obtention d'une reconstruction de bonne qualité (selon le critère d'erreur de reprojection). Ceci est critique surtout lorsque l'algorithme est utilisé dans la version robuste où chaque contrainte est calculée de manière robuste par échantillonage aléatoire (RANSAC).

15.2 Perspectives et problèmes ouverts

La vision par ordinateur est aujourd'hui à un stade permettant le développement de systèmes complets de reconstruction 3D passives de larges environnements et ce, de façon complètement automatique. Il s'agit tout de même d'une avancée que peu d'organisations ou groupes de recherches a pu réaliser [3, 80, 133]. Ces systèmes sont encore imparfaits, ils requièrent de très longs calculs ou l'intégration de données externes comme des coordonnées GPS, mais leurs résultats sont malgré tout très impressionnant.

Un autre défi de taille est celui d'intégrer les derniers développements de la reconstruction 3D à ceux provenant de domaine de l'apprentissage en vision par ordinateur. Il est logique de penser qu'inévitablement ces méthodes permettront de mieux guider le processus de reconstruction 3D.

Nous décrivons ici quelques perspectives et problèmes de façon plus détaillé par rapport aux trois grands thèmes abordés dans la thèse.

▷ **Calibrage et auto-calibrage de caméra omnidirectionnelle**

Le problème de calibrage de caméra est étudié depuis les années 60. Il est presque surprenant que des avancées de nature théorique et non seulement technologiques soient encore possibles aujourd'hui. Il faut rappeler que la plupart des travaux récents sur ce thème portent sur les systèmes omnidirectionnels qui existent depuis environ un dizaine d'années. Avec la multiplication des types de caméra, nous croyons que l'avènement des modèles de caméras génériques est une percée importante. Naturellement, ces modèles confèrent beaucoup de flexibilité, n'exigeant que très peu d'hypothèses sur le système d'acquisition.

L'auto-calibrage de caméra, datant d'une dizaine d'années, est un concept beaucoup plus récent que le calibrage. Il n'est évidemment appliqué que depuis peu aux systèmes omnidirectionnels. D'ailleurs, il n'existe toujours pas de solution au problème d'auto-calibrage de modèle générique. Toutes celles qui ont été proposées jusqu'à maintenant s'appliquent à des situations spécifiques (rotation pure, translation pure, modèle radial symétrique ou radial 1D, projection centrale). Il n'est pas évident que les modèles complètement génériques soient beaucoup plus utiles que ceux qui ne le sont que partiellement, comme les modèles radial symétrique et radial 1D. Cependant, les dernières innovations technologiques comme les *caméras computationnelles*[1] nous laisse croire que oui [100]. Par ailleurs, nous croyons qu'il est important d'explorer les limites de ce genre de modèle afin de découvrir le meilleur compromis à faire entre modèles génériques et paramétriques.

Depuis quelques années, plusieurs chercheurs explorent les possibilités offertes par de nouvelles méthodes d'optimisation [22]. La solution d'approximation

[1] *Computational cameras*

convexe que nous proposons pour le problème d'auto-calibrage de caméra radiale symétrique suit d'ailleurs cette tendance. Comme pour l'estimation de matrice fondamentale ou d'homographies basée sur une erreur algébrique, l'idée est d'approcher le mieux possible le problème non-convexe original par un problème convexe. Nous croyons que cette avenue de recherche est très prometteuse.

Une autre avenue de recherche intéressante qui a peu été explorée est l'élaboration d'outils d'évaluation de la qualité du calibrage d'une caméra et un interface permettant de communiquer les résultats à un utilisateur. Idéalement, des recommandations devraient lui être faites pour améliorer la qualité des résultats, comme par exemple, de prendre une nouvelle image du plan de calibrage dans une position donnée.

▷ **Reconstruction 3D active**

Il est surprenant de constater que les différentes disciplines de mise en correspondance active et passive sont développés de façon plutôt indépendante. Du côté actif, la recherche est surtout effectuée au niveau du design de motif, alors que du côté passif, l'intérêt est dirigé vers les difficiles problèmes d'optimisation que l'on rencontre. Nous croyons que chaque discipline aurait intérêt à emprunter des idées à l'autre. La solution de mise en correspondance active que nous proposons, quoique très simple, suit en quelque sorte cette tendance. En effet, le décodage des motifs est modélisé de façon probabiliste comme c'est le cas pour la mise en correspondance passive. Nous croyons aussi que l'intégration de solutions logicielles sophistiquées aux systèmes actifs sera facilitée par l'augmentation de la puissance et la miniaturisation des ordinateurs.

Une avenue de recherche qui nous semble intéressante consisterait à étendre notre approche probabiliste à des motifs à cohérence spatiale comme les codes de De Bruijn. Dans ces encodages les erreurs sur les étiquettes des pixels ont des conséquences encore plus néfastes que dans le cas de code de Gray. Le

système multi-projecteurs LightTWIST bénéficierait grandement de ce genre d'améliorations. En effet, un nombre total de 40 motifs doit être utilisés pour chaque projecteur : 10 motifs de code Gray et leur inverse pour chaque coordonnées X et Y). L'acquisition de ces images est longue et leur emmagasinage est coûteux en espace mémoire. Finalement, nous croyons que la cohérence temporelle pourrait, elle aussi, être modélisée de façon probabiliste pour la reconstruction 3D de scènes dynamiques.

▷ **Algorithmes de factorisation de matrice avec données manquantes**

Il reste encore plusieurs problèmes non-solutionnés de factorisation de matrice avec données manquantes. Notre algorithme *batch* est efficace parce que la factorisation de sous-blocs est possible à l'aide de la SVD, la matrice de poids W étant binaire. Dans le cas d'une matrice poids plus générale, le calcul de chaque contrainte n'est plus analytique. Nous aimerions analyser cette situation avec plus d'attention.

La contrainte de non-négativité des facteurs a aussi reçu une certaine attention dans le domaine de l'apprentissage par ordinateur [63]. Dans le cas de données manquantes, le problème est beaucoup plus difficile que dans le cas non-contraint, même avec une matrice binaire de poids. La difficulté provient essentiellement du fait que des contraintes sont imposées sur les deux facteurs. De plus, c'est en bonne partie grâce à l'ambiguïté de factorisation que les méthodes *batch* peuvent se concentrer sur l'estimation d'un seul des deux facteurs. Dans le cas non-négatif, l'ambiguïté est réduite par les contraintes.

Finalement, le problème de reconstruction passive n'est pas encore résolu même si plusieurs systèmes fonctionnels ont été présentés. Nous croyons que leur amélioration doit se faire du côté de la robustesse, mais surtout au niveau de leur temps de calcul. Par exemple, il faudrait trouver des façons de combiner plusieurs centaines d'images pour obtenir le plus rapidement possible un estimé de

la trajectoire d'une caméra.

Les méthodes *batch* de factorisation proposent des solutions intéressantes à ce problème. Cependant, elles sont imparfaites, puisque limitées à l'emploi du modèle de caméra affine ou, dans le cas perspectif, nécessitent l'estimation *a priori* des profondeurs projectives.

BIBLIOGRAPHIE

[1] H. Aanaes, R. Fisker, K. Astrom, J. M. Carstensen, Robust Factorization, Pattern Analysis and Machine Intelligence (PAMI), vol. 24, no. 9, pp. 1215-1225, Sept., 2002.

[2] I. Adler, N. Karmarkar, M. G.C. Resende, G. Veiga, An Implementation of Karmarkar's Algorithm for Linear Programming, Mathematical Programming, Vol 44, p. 297-335, 1989.

[3] A. Akbarzadeh, J.-M. Frahm, P. Mordohai, B. Clipp, C. Engels, D. Gallup, P. Merrell, M. Phelps, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, H. Towles, D. Nister, M. Pollefeys, Towards Urban 3D Reconstruction From Video, International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT), 2006.

[4] P. Anandan , M. Irani, Factorization with Uncertainty, International Journal of Computer Vision (IJCV), 49(2-3): 101-116, September 2002.

[5] W. E. Arnoldi, The principle of minimized iterations in the solution of the matrix eigenvalue problem, Quart. J. Applied Mathematics, 9:17-29, 1951.

[6] S. Baker, S. K. Nayar, A Theory of Single-Viewpoint Catadioptric Image Formation, International Journal of Computer Vision (IJCV), 35(2), 1–22, 1999.

[7] H. Bakstein , T. Pajdla, Panoramic mosaicing with a 180 field of view lens, Workshop on omnidirectional vision, camera networks, and non-classical cameras, pp. 60-67, 2002.

[8] J. P. Barreto , H. Araujo, Issues on the Geometry of Central Catadioptric Imaging, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 422-427, 2001.

[9] J.P. Barreto, K. Daniilidis, Unifying image plane lifting for central catadioptric and dioptric cameras, Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras, 151-162, 2004.

[10] Joao P. Barreto , Kostas Daniilidis, Fundamental Matrix for Cameras with Radial Distortion, IEEE International Conference on Computer Vision (ICCV), pp. 625-632, 2005.

[11] Joao P. Barreto , Helder Araujo, Geometric Properties of Central Catadioptric Line Images and Their Application in Calibration, Pattern Analysis and Machine Intelligence (PAMI), 27(8), pp. 1237–1333. August 2005.

[12] J.P. Barreto, K. Daniilidis, Epipolar geometry of central projection systems using veronese maps, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1258-1265, 2006.

[13] Joao P. Barreto, A Unifying Geometric Representation for Central Projection Systems, Computer Vision and Image Understanding, 103(3), pp. 207–217. September, 2006.

[14] A. Bartoli, H. Martinsson, F. Gaspard, J.-M. Lavest, On Aligning Sets of Points Reconstructed from Uncalibrated Affine Cameras, Scandinavian Conference on Image Analysis (SCIA), 2005.

[15] A. Basu, S. Licardie, Alternative models for fish-eye lenses, Pattern Recognition Letters 16: 433-441, (1995).

[16] J. Batllea, J. Salvia, Recent Progress in Structured Light in order to Solve the Correspondence Problem in Stereovision, IEEE International Conference on Robotics and Automation (ICRA), pp. 130-136, 1997.

[17] H. Bay, T. Tuytelaars, L. V. Gool, SURF: Speeded Up Robust Features, European Conference on Computer Vision (ECCV)pp 404-417, 2006.

[18] W. Boehm, H. Prautzsch, *Geometric Concepts for Geometric Design*, A.K. Peters, 1994.

[19] P. T. Boggs, R. H. Byrd, J. R. Donaldson, R. B. Schnabel, ODRPACK – Software for Weighted Orthogonal Distance Regression, ACM Transactions on Mathematical Software, Vol. 15, No. 4, pp 348-364, 1989.

[20] Bookstein, Fitting conic sections to scattered data, Computer Graphics and Image Processing, pp. 5671, 1979.

[21] M. Born, E. Wolf, Principles of Optics, Pergamon Press, 1965.

[22] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.

[23] S. Brandt, Closed-form solutions for affine reconstruction under missing data, European Conference on Computer Vision (ECCV), pp. 109-114, 2002.

[24] C. Bregler, A. Hertzmann, H. Biermann, Recovering non-rigid 3D shape from image streams, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 690-696, 2000.

[25] D.C. Brown, Decentering Distortion of Lenses Photometric Engineering, pp. 444-462, Vol. 32, No. 3, 1966.

[26] D.C. Brown, Close-Range Camera Calibration, Photogrammetric Engineering, 37(8), 855-866, 1971.

[27] A. M. Buchanan, Investigation into Matrix Factorization when Elements are Unknown, Technical report, University of Oxford, 2004.

[28] A. M. Buchanan, A. W. Fitzgibbon, Damped Newton Algorithms for Matrix Factorization with Missing Data, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 316-322, 2005.

[29] G. Champleboux, S. Lavallée, P. Sautot, P. Cinquin, Accurate Calibration of Cameras and Range Imaging Sensors: the NPBS Method, IEEE International Conference on Robotics and Automation (ICRA), pp. 1552-1557, 1992.

[30] N. L. Chang, Efficient Dense Correspondences using Temporally Encoded Light Patterns, Procams 2003.

[31] S. Christy, R. Horaud. Euclidean shape and motion from multiple perspective views by affine iteration, Pattern Analysis and Machine Intelligence (PAMI), 18(11):10981104, 1996.

[32] D. Claus, A.W. Fitzgibbon, Rational Function Model for Fish-eye Lens Distortion IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2005.

[33] D. Claus, A.W. Fitzgibbon, A Plumbline Constraint for the Rational Function Lens Distortion Model British Machine Vision Conference (BMVC)2005.

[34] A. Conrady Decentering lens systems, Monthly notices of the Royal Astronomical Society, Vol. 79, pp. 384-390, 1919.

[35] A. Dempster, N. Laird, D. Rubin, Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, Series B, 39(1):138, 1977

[36] Frédéric Devernay, Olivier Faugeras, Straight lines have to be straight: Automatic calibration and removal of distortion from scenes of structured environments, Machine Vision and Applications, MVA(13), No. 1, 2001, pp. 14-24. 2001.

[37] A.W. Fitzgibbon, A. Zisserman, Automatic Camera Recovery for Closed or Open Image Sequences, European Conference on Computer Vision (ECCV), pp. 311-326, 1998

[38] A. W. Fitzgibbon, R. B. Fisher, A buyers guide to conic fitting, British Machine Vision Conference (BMVC), pp. 513–522, 1995.

[39] A. W. Fitzgibbon, Simultaneous linear estimation of multiple view geometry and lens distortion, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 125-132, 2001.

[40] D. Claus, A.W. Fitzgibbon, Rational Function Model for Fish-eye Lens Distortion IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 213-219, 2005.

[41] M.M. Fleck, Perspective Projection: The Wrong Imaging Model, TR 95–01, University of Iowa, 1995.

[42] D. Forsyth, J. Ponce, Computer Vision: A Modern Approach, Prentice Hall, Upper Saddle River, NJ, 2002.

[43] W. Gander, G.H. Golub, R. Strebel, Fitting of Circles and Ellipses, *BIT*, 34, 556-577, 1994.

[44] C. Geyer, K. Daniilidis, Catadioptric Camera Calibration, IEEE International Conference on Computer Vision (ICCV), pp. 398-404, 1999.

[45] C. Geyer, K. Daniilidis, A Unifying Theory for Central Panoramic Systems and Practical Applications, European Conference on Computer Vision (ECCV), pp 445-461, 2000.

[46] C. Geyer, K. Daniilidis, Catadioptric Projective Geometry, International Journal of Computer Vision (IJCV), 45(3), 2001.

[47] C. Geyer, K. Daniilidis, Structure and Motion from Uncalibrated Catadioptric Views, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 279-286, 2001.

[48] C. Geyer, K. Daniilidis, Properties of the Catadioptric Fundamental Matrix, European Conference on Computer Vision (ECCV), pp. 140-154, 2002.

[49] C. Geyer, K. Daniilidis, Mirrors in Motion: Epipolar geometry and motion estimation, IEEE International Conference on Computer Vision (ICCV), pp. 766-773, 2003.

[50] J. M. Gluckman, S. K. Nayar, Planar catadioptric stereo: geometry and calibration, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 22–28, 1999.

[51] Gene H. Golub and Charles F. Van Loan, Matrix computations (3rd ed.), Johns Hopkins University Press, 1996.

[52] M. Grant, S. Boyd, Y. Ye, Disciplined Convex Programming, Global Optimization: From Theory to Implementation, Kluwer, 2005.

[53] K.D. Gremban, C.E. Thorpe, T. Kanade, Geometric Camera Calibration using Systems of Linear Equations, IEEE International Conference on Robotics and Automation (ICRA), 1988.

[54] M.D. Grossberg, S.K. Nayar, A general imaging model and a method for finding its parameters, IEEE International Conference on Computer Vision (ICCV), 2001.

[55] M.D. Grossberg, S.K. Nayar, The Raxel Imaging Model and Ray-Based Calibration, International Journal of Computer Vision, 61(2), 119-137, 2005.

[56] E. Grossmann, E-J Lee, P. Hislop, D. Nistér, H. Stewénius, Are two rotational flows sufficient to calibrate a smooth non-parametric sensor?, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006.

[57] N. Guilbert, A. Bartoli, A. Heyden, Affine Approximation for Direct Batch Recovery of Euclidean Motion From Sparse Data, International Journal of Computer Vision, Vol. 69, No. 3, pp. 317-333, September 2006.

[58] P.Gurdjos et R.Payrissat, Plane-based Calibration of a Camera with Varying Focal Length: the Centre Line Constraint, British Machine Vision Conference (BMVC), 2001.

[59] P. Gurdjos, A. Crouzil et R. Payrissat, Another Way of Looking at Plane-Based Calibration: the Centre Circle Constraint, European Conference on Computer Vision (ECCV), pp. 252-266, 2002.

[60] R. Hartley, A. Zisserman Multiple View Geometry in Computer Vision Cambridge University Press 2000.

[61] R.I. Hartley, S.B. Kang, Parameter-free Radial Distortion Correction with Centre of Distortion Estimation, International Conference on Computer Vision, 1834-1841, 2005.

[62] H. Hayakawa, Photometric stereo under a light source with arbitrary motion, JOSA, 11(11):3079-89, 1994.

[63] M. Heiler Image Models for Segmentation and Recognition, Thèse de doctorat, University of Mannheim, 2006.

[64] E. Horn, N. Kiryati, Toward Optimal Structured Light Patterns, International Conference on 3D Digital Imaging and Modeling (3DIM), pp. 28-35, 1997.

[65] P. Huang, S. Zhang, High-resolution, Real-time 3D Shape Acquisition, 2004 Conference on Computer Vision and Pattern Recognition Workshop, p. 28, 2004.

[66] S. Inokuchi, K. Sato, F. Matsuda, Range imaging system for 3-D object recognition, IEEE Conference on Pattern Recognition (ICPR), pp. 806808, 1984.

[67] Intel Open Source Computer Vision Library, `http://www.intel.com/research/mrl/research/opencv/`.

[68] F. Kahl, A. Heyden, Affine Structure and Motion from Points, Lines and Conics, International Journal of Computer Vision (IJCV), 33(3):163-180, 1999.

[69] K. Kanatani, Statistical bias of conic fitting and renormalization, Pattern Analysis and Machine Intelligence (PAMI), 16(3):320326, 1994.

[70] K. Kanatani, Ellipse Fitting with Hyperaccuracy, European Conference on Computer Vision (ECCV), pp. 484-495, 2006.

[71] S. B. Kang, Radial Distortion Snakes, IEICE Trans. Inf. & Syst., vol. E84-D, No. 12, Dec. 2001, pp. 1603-1611.

[72] J. Kannala, S.S. Brandt, A Generic Camera Model and Calibration Method for Conventional, Wide-Angle, and Fish-Eye Lenses, Pattern Analysis and Machine Intelligence (PAMI), 28(8), 1335-1340, 2006.

[73] R. Kaucic, R. Hartley, N. Dano, Plane-based Projective Reconstruction, IEEE International Conference on Computer Vision (ICCV), pp. 420-427, 2001.

[74] Q. Ke, T. Kanade, Robust L-1 Norm Factorization in the Presence of Outliers and Missing Data by Alternative Convex Programming, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 739-746, 2005.

[75] T. P. Koninckx, I. Geys, T. Jaeggli, L. V. Gool, A Graph Cut based Adaptive Structured Light approach for real-time Range Acquisition, International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT), pp. 413-421, 2004.

[76] Z. Kukelova, T. Pajdla, A minimal solution to the autocalibration of radial distortion, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007.

[77] D. D. Lee, H. S. Seung, Algorithms for Non-negative Matrix Factorization, Conference on Neural Information Processing Systems (NIPS), pp. 556-562, 2001.

[78] R. B. Lehoucq, Analysis and Implementation of an Implicitly Restarted Iteration, PhD thesis, Rice University, Houston, Texas, May 1995.

[79] R. B. Lehoucq, D. C. Sorensen, and C. Yang, ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods, SIAM, 1998.

[80] B. Leibe, N. Cornelis, K. Cornelis, L. Van Gool Dynamic 3D Scene Analysis from a Moving Vehicle IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007.

[81] S.Z. Li, Markov Random Field Modeling in Computer Vision, Springer-Verlag 1995.

[82] S.-S. Lin, R. Bajcsy, True single view point cone mirror omni-directional catadioptric system, International Conference on Computer Vision, 102-107, 2001.

[83] David G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision (IJCV), Vol. 60, 2, pp. 91-110, 2004.

[84] B. D. Lucas, T. Kanade, An Iterative Image Registration Technique with an Application to Stereo Vision, International Joint Conference on Artificial Intelligence, pages 674-679, 1981.

[85] Y. Ma, S. Soatto, J. Kosecka, S. Sastry, An Invitation to 3D Vision, Springer, 2003.

[86] L. Ma, Y. Chen, K. L. Moore, Flexible Camera Calibration Using a New Analytical Radial Undistortion Formula with Application to Mobile Robot Localization, 2003 IEEE International symposium on intelligent control (ISIC). October 5-8, 2003, Westin Galleria Houston,Texas, USA

[87] L. Ma, Y. Chen, K.L. Moore, Rational radial distortion models with analytical undistortion formulae, Internationnal Journal of Information Acquisition (IJA). vol. 1,no. 2, pp 135–147, 2004.

[88] S. Mahamud, M. Hebert, Y. Omori, J. Ponce, Provably-Convergent Iterative Methods for Projective Structure from Motion, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1018-1025, 2001.

[89] D. Martinec, T. Pajdla, 3D Reconstruction by Fitting Low-rank Matrices with Missing Data, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 198-205, 2005.

[90] D. Martinec, T. Pajdla, 3D Reconstruction by Gluing Pair-wise Euclidean Reconstructions, or "How to Achieve a Good Reconstruction from Bad Images", International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT), pp. 25-32, 2006.

[91] D. Martinec, T. Pajdla, Robust Rotation and Translation Estimation in Multiview Reconstruction, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007.

[92] H.-G. Maas, Robust Automatic Surface Reconstruction with Structured Light, International Archives of Photogrammetry and Remote Sensing Vol. XXIX (1992).

[93] S. Maybank, O. Faugeras, A Theory of Self-Calibration of a Moving Camera, International Journal of Computer Vision (IJCV), 8(2):123-151, 1992.

[94] B. Micusik, T. Pajdla, Autocalibration & 3D Reconstruction with Non-central Catadioptric Cameras, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 58-65, 2004.

[95] K. Mikolajczyk, C. Schmid, Scale and affine invariant interest point detectorçs, International Journal of Computer Vision (IJCV), Vol. 60, 1, pp. 63-86, 2004.

[96] R. A. Morano, C. Ozturk, R. Conn, S. Dubin, S. Zietz, J. Nissanov, Structured light using pseudorandom codes, Pattern Analysis and Machine Intelligence 20 (3) (1998) 322-327.

[97] V. Nalwa, A true omnidirectional viewer, Technical report, Bell Laboratories, Holmdel, NJ, USA, 1996.

[98] Shree K. Nayar, Catadioptric Omnidirectional Camera, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.482-488, 1997.

[99] S. K. Nayar, Venkat Peri, Folded Catadioptric Cameras, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), p. 2217, 1999.

[100] S. K. Nayar, Computational Cameras: Redefining the Image, IEEE Computer Magazine, Special Issue on Computational Photography, pp. 62-70, Aug, 2006.

[101] D. Nistèr, Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors, European Conference on Computer Vision (ECCV), pp. 649-663, 2003.

[102] D. Nistér, H. Stewénius, E. Grossman, Non-Parametric Self-Calibration, IEEE International Conference on Computer Vision (ICCV), pp. 120-127, 2005.

[103] J. Oliensis, R. Hartley, Iterative extensions of the Sturm/Triggs algorithm: convergence and nonconvergence, European Conference on Computer Vision (ECCV), pp. 214-227 2006.

[104] P. Paatero, U. Tapper Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values, Environmetrics 5:111-126.

[105] J. Pagès, J. Salvi, C. Matabosch, Implementation of a robust coded structured light technique for dynamic 3D measurements, ICIP, pp. 1073-1076, 2003.

[106] J. Pagés, J Salvi, C. Collewet, J. Forest, Optimised De Bruijn patterns for one-shot shape acquisition Image and Vision Computing 23 (2005) 707720.

[107] T. Pajdla, T. Svoboda, V. Hlavac Epipolar Geometry of Central Panoramic Catadioptric Cameras Panoramic Vision: Sensors, Theory and Applications, pp. 85-114. Springer, 2001.

[108] P. M. Pardalos and S. A. Vavasis. Quadratic programming with one negative eigenvalue is NP-hard, Journal of Global Optimization, Volume 1, Number 1, 1991, pg.15-22.

[109] R. Pless, Using many cameras as one, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 587-593, 2003.

[110] L. Quan, Self-calibration of an affine camera from multiple views, International Journal of Computer Vision (IJCV), 19(1):93–105, 1996.

[111] R. Andrew, G. Gill, A. Majumder, H. Towles, H. Fuchs, PixelFlex2: A Comprehensive, Automatic, Casually-Aligned Multi-Projector Display, PROCAMS 2003.

[112] S. Ramalingam, P. Sturm, S. K. Lodha, Towards Complete Generic Camera Calibration, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, USA, June 2005.

[113] S. Ramalingam, P. Sturm, S.K. Lodha, Towards Generic Self-Calibration of Central Cameras, Workshop on omnidirectional vision, camera networks, and non-classical cameras, pp. 20-27 2005.

[114] S. Ramalingam, P. Sturm, E. Boyer, A Factorization Based Self-Calibration for Radially Symmetric Cameras, International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT), 2006.

[115] R. Raskar, M. S. Brown, R. Yang, W. Chen, G. Welch, H. Towles, B. Seales, H. Fuchs, Multi-Projector Displays Using Camera-Based Registration, IEEE Visualization '1999.

[116] Realviz Co., ImageModeler, www.realviz.com 3D scenes using photographs.

[117] C. Rocchini, P. Cignoni, C. Montani, P. Pingi, R. Scopigno, A low cost 3D scanner based on structured light, Computer Graphics Forum (Eurographics 2001 Conf. Issue), Blackwell v 20,3.

[118] C. Rother, S. Carlsson, Linear Multi View reconstruction and Camera Recovery using a Reference Plane, International Journal of Computer Vision (IJCV), 2002, 49(2/3):117-141.

[119] S. Roy, LightTWIST, `http://tot.sat.qc.ca/logiciels_lighttwist.html`.

[120] S, Rusinkiewicz, O. Hall-Holt, M. Levoy, Real-Time 3D Model Acquisition, ACM Transactions on Graphics, pp. 438-446, 2002.

[121] F. Ruskey. The Combinatorial Object Server, http://www.theory.csc.uvic.ca/, 2007.

[122] J. Salvi, J. Pagès, J. Batlle, Pattern codification strategies in structured light systems, Pattern Recognition, 37(4), 827-849, 2004.

[123] J. Salvi, J. Pagés, J. Batlle, Pattern codification strategies in structured light systems, Pattern Recognition 37 (4) (2004) 827849.

[124] D. Sampson, Fitting conic sections to very scattered data: An iterative re-inement of the Bookstein algorithm, Computer Graphics and Image Processing, 18:97108, 1982.

[125] R. Hartley, F. Schaffalitzky, PowerFactorization: an approach to affine reconstruction with missing and uncertain data, In Australia-Japan Advanced Workshop on Computer Vision, 2003.

[126] D. Scharstein, R. Szeliski, A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, International Journal of Computer Vision (IJCV), 47(1/2/3):7-42, April-June 2002.

[127] D. Scharstein, R. Szeliski, High-Accuracy Stereo Depth Maps Using Structured Light, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 195-205, 2003.

[128] S. Shah, J.K. Aggarwal, Intrinsic Parameter Calibration Procedure for A (High-Distortion) Fish-Eye Lens Camera with Distortion Model and Accuracy Estimation, Pattern Recognition, 29(11), 1775-1788, 1996.

[129] H.-Y. Shum, R. Szeliski, Systems and Experiment Paper: Construction of Panoramic Image Mosaics with Global and Local Alignment, International Journal of Computer Vision (IJCV), 36(2):101-130, 2000.

[130] K. Sim, R. Hartley, Removing Outliers Using The L-∞ Norm, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 485-494, 2006.

[131] K. Sim, R. Hartley, Recovering Camera Motion Using L-∞ Minimization, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1230-1237, 2006.

[132] C. Slama, editor. Manual of Photogrammetry, American Society of Photogrammetry, Falls Church, VA, 4th edition, 1980.

[133] Noah Snavely, Steven M. Seitz, Richard Szeliski, Photo Tourism: Exploring photo collections in 3D, ACM Transactions on Graphics, 25(3), August 2006.

[134] N. Srebro, T. Jaakkola, Weighted Low-Rank Approximations, International Conference on Machine Learning (ICML) 2003.

[135] N. Srebro, T. Jaakkola, Learning with Matrix Factorizations Ph.D. Thesis, Massachusetts Institute of Technology, 2004, 132 pages.

[136] D.E. Stevenson, M.M. Fleck, Nonparametric correction of distortion, TR 95-07, University of Iowa, 1995.

[137] J. Stolfi, Oriented projective geometry Academic Press Professional, Inc, 1991.

[138] R. Strand, E. Hayman, Correcting Radial Distortion by Circle Fitting, British Machine Vision Conference (BMVC), 2005

[139] A. Strzebonski, Cylindrical Algebraic Decomposition, From MathWorld – A Wolfram Web Resource, created by E.W. Weisstein, `http://mathworld.wolfram.com/CylindricalAlgebraicDecomposition.html`.

[140] J.F. Sturm. Using SeDuMi 1.02, A Matlab toolbox for optimization over symmetric cones, Optimization Methods and Software, 1999.

[141] P. Sturm, B. Triggs, A factorization based algorithm for multi-image projective structure and motion, European Conference on Computer Vision (ECCV), 1996.

[142] P. Sturm, S. Maybank, On Plane-Based Camera Calibration, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 432-437, 1999.

[143] P. Sturm, Algorithms for Plane-Based Pose Estimation, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 706-711, 2000.

[144] P. Sturm, Mixing Catadioptric and Perspective Cameras, Workshop on omnidirectional vision, camera networks, and non-classical cameras, 37- 44 2002.

[145] P. Sturm, S. Ramalingam, A generic concept for camera calibration, European Conference on Computer Vision (ECCV), 1-13, 2004.

[146] R. Swaminathan, M. Grossberg, S. Nayar, Caustics of catadioptric cameras, IEEE International Conference on Computer Vision (ICCV), pp. 2-9, 2001.

[147] R. Swaminathan, S.K. Nayar, Non-Metric Calibration of Wide-Angle Lenses and Polycameras, IEEE Journal on Pattern Analysis and Machine Intelligence (PAMI), Vol 22, No 10, October 2000.

[148] R. Swaminathan, M. D. Grossberg, S. K. Nayar, Non-Single Viewpoint Catadioptric Cameras: Geometry and Analysis International Journal of Computer Vision (IJCV)Vol.66, No.3, pp.211-229, Mar, 2006.

[149] S. Roy, J.-P. Tardif, LightTwist, a multi-projector system, Live performance, ArtFutura 2004.

[150] J.-P. Tardif, S. Roy, A MRF formulation for coded structured light, International Conference on 3D Digital Imaging and Modeling (3DIM), pp. 22-29, 2005.

[151] J.-P. Tardif, P. Sturm, Calibration of Cameras with Radially Symmetric Distortion, Workshop on omnidirectional vision, camera networks, and non-classical cameras, pp. 44-51, 2005.

[152] J.-P. Tardif, P. Sturm, S. Roy, Self-calibration of a general radially symmetric distortion model European Conference on Computer Vision (ECCV), pp. 186-199, 2006.

[153] J.-P. Tardif, P. Sturm, S. Roy, Plane based self-calibration of radial distortion, IEEE International Conference on Computer Vision (ICCV), 2007.

[154] J.-P. Tardif, A. Bartoli, M. Trudeau, N. Guilbert, S. Roy, Algorithms for Batch Matrix Factorization with Application to Structure-from-Motion, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007.

[155] J.-P. Tardif, S. Roy, M. Trudeau, Multi-projectors for arbitrary surfaces without explicit calibration nor reconstruction, International Conference on 3D Digital Imaging and Modeling (3DIM), pp. 22-29, 2003.

[156] S. Thirthala, M. Pollefeys, The Radial Trifocal Tensor: A tool for calibrating the radial distortion of wide-angle cameras, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 321-328, 2005.

[157] S. Thirthala, M. Pollefeys, Multi-View Geometry of 1D Radial Cameras and its Application to Omnidirectional Camera Calibration, IEEE International Conference on Computer Vision (ICCV), pp. 1539-1546, 2005.

[158] Tisseur, F., K. Meerbergen, The quadratic eigenvalue problem, SIAM Rev., Vol. 43, Number 2, pp. 235-286, 2001.

[159] C. Tomasi, T. Kanade, Shape and motion from image streams under orthography: a factorization method, International Journal of Computer Vision (IJCV), 9(2):137-154, November 1992.

[160] B. Triggs, Matching constraints and joint image IEEE International Conference on Computer Vision (ICCV), pp. 338-343, 1995.

[161] B. Triggs, Factorization Methods for Projective Structure and Motion, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 845-851 1996.

[162] B. Triggs, Linear projective reconstruction from matching tensors, Image and Vision Computing, 15(8):617625, 1997.

[163] B. Triggs, Autocalibration from Planar Scenes, European Conference on Computer Vision (ECCV), p 89, 1998.

[164] B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon, Bundle Adjustment, A Modern Synthesis, Vision Algorithms, pp. 298-372, 1999.

[165] T. Tsai, A versatile camera calibration technique for high- accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses, IEEE J Robotics Autom 3(4): 323344, 1997.

[166] R. F. Vassalloy, J. Santos-Victory, H. J. Schneebelz, A General Approach for Egomotion Estimation with Omnidirectional Images Workshop on omnidirectional vision, camera networks, and non-classical cameras, pp. 97–103, 2002.

[167] R. Vidal, R. Hartley, Motion Segmentation with Missing Data using PowerFactorization and GPCA, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 310-316, 2004.

[168] E. W. Weisstein, Gray Code, MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/GrayCode.html.

[169] R. Yang, D. Gotz, J. Hensley, H. Towles, M.S. Brown, PixelFlex: a reconfigurable multi-projector display system, IEEE Visualization 2001.

[170] X. Ying, Z. Hu, Catadioptric Camera Calibration Using Geometric Invariants, IEEE International Conference on Computer Vision (ICCV), pp. 1260-1271, 2003.

[171] X. Ying, Z. Hu, Catadioptric Line Features Detection using Hough Transform, IEEE Conference on Pattern Recognition (ICPR), vol. 3, pp. 231-234, 2004.

[172] X. Ying, Z. Hu, Distortion Correction Of Fisheye Lenses Using A Non-Parametric Imaging Model, Asian Conference on Computer Vision (ACCV), pp.527-532, 2004.

[173] X. Ying, Z. Hu Can We Consider Central Catadioptric Cameras and Fisheye Cameras within a Uni ed Imaging Model, European Conference on Computer Vision (ECCV), pp. 442-455, 2004.

[174] X. Ying, H. Zha, Linear Catadioptric Camera Calibration from Sphere Images, Workshop on omnidirectional vision, camera networks, and non-classical cameras, pp. 28-34, 2005.

[175] X. Ying, H. Zha, Using Sphere Images for Calibrating Fisheye Cameras under the Unified Imaging Model of the Central Catadioptric and Fisheye Cameras, IEEE Conference on Pattern Recognition (ICPR), 2006.

[176] X. Ying, H. Zha, A Novel Linear Approach to Camera Calibration from Sphere Images, IEEE Conference on Pattern Recognition (ICPR), pp. 535-538, 2006.

[177] X. Ying, Z. Hu, H. Zha, Fisheye Lenses Calibration Using Straight-line Spherical Perspective Projection Constraint, Asian Conference on Computer Vision (ACCV), pp. 724-733, India, January 13-16, 2006.

[178] A. Zaharescu, R. P. Horaud, R. Ronfard, L. Lefort, Multiple Camera Calibration using Robust Perspective Factorization, International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT), 2006.

[179] Z. Zhang, Parameter estimation techniques: A tutorial with application to conic fitting, TR 2676, INRIA, 1995.

[180] Z. Zhang, On the Epipolar Geometry Between Two Images With Lens Distortion, IEEE Conference on Pattern Recognition (ICPR), pp. 407–411, 1996.

[181] Z. Zhang, A Flexible New Technique for Camera Calibration, Pattern Analysis and Machine Intelligence (PAMI), 22(11), 1330–1334, 2000.

[182] L. Zhang, B. Curless, S. M. Seitz, Rapid shape acquisition using color structured light and multi-pass dynamic programming, International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT), Padova, Italy,pp. 24-36, 2002.