Calibration par programmation linéaire et reconstruction spatio-temporelle à partir de réseaux d'images

Jérôme Courchay

UNIVERSITÉ PARIS-EST

T H È S E

pour obtenir le titre de

Docteur en Informatique

Jérôme COURCHAY

Structure à partir du mouvement par programmation linéaire et Reconstruction spatio-temporelle à partir d'images

soutenue le 5 janvier 2011

Jury :

Rapporteurs :	Long QUAN	-	Hong Kong University of Science and Technology
	Adrien BARTOLI	-	Professeur à l'Université d'Auvergne
Examinateurs :	Jean PONCE	-	ENS Paris
	Peter STURM	-	INRIA Grenoble
	Arnak DALALYAN	-	LIGM - IMAGINE - CSTB Marne-la-Vallée
Directeur :	Renaud KERIVEN	-	LIGM - IMAGINE - CSTB Marne-la-Vallée

Dédicaces

Je dédicace cette thèse à ma famille, dont le soutien a toujours été constant et sans faille.

Remerciements

Je remercie mon directeur de thèse, Renaud Keriven, pour la constance et la qualité de son suivi durant mon doctorat. Son entrain et sa motivation communicative ont été un soutien essentiel. Je tiens également à remercier Arnak Dalalyan pour la contribution solide et consistante qu'il a apporté à la seconde partie de cette thèse. Enfin je suis honoré que les membres du jury, qui ont tous apporté beaucoup à la vision par ordinateur ainsi qu'à la géométrie des images multiples, se soient réunis pour évaluer ma thèse.

Introduction

Le problème de la stéréovision à partir de caméras multiples calibrées capturant une scène fixe est étudié depuis plusieurs décennies. Les résultats présentés dans le benchmark de stéréovision proposé par Strecha *et al.* [SvHVG+08], attestent de la qualité des reconstructions obtenues. En particulier, les travaux de [VKLP09], mènent à des résultats visuellement impressionnants.
Aussi, il devient intéressant de calibrer des scènes de plus en plus vastes, afin d'appliquer ces algorithmes de stéréovision de façon optimale. Trois objectifs essentiels apparaissent :

- La précision de la calibration doit être améliorée. En effet comme pointé dans [FP09], même les benchmarks de stéréovision fournissent parfois des caméras bruitées à la précision imparfaite. Un des moyens pour améliorer les résultats de stéréovision consiste à augmenter la précision de la calibration.
- Il est important de pouvoir prendre en compte les cycles dans le graphe des caméras de façon globale. En effet la plupart des méthodes actuelles sont séquentielles, et dérivent. Ainsi ces méthodes ne garantissent pas, pour une très grande boucle, de retrouver cette configuration cyclique. Comme on calibre des réseaux d'images de plus en plus grands, ce point devient crucial.
- Pour calibrer des réseaux d'images très grands, il convient d'avoir des algorithmes rapides.

Les méthodes de structure à partir du mouvement ("structure from motion" en anglais) que nous proposons dans la première partie, permettent de retrouver les positions dans l'espace, orientations dans l'espace et distances focales des caméras avec une précision très proche de l'état de l'art. D'autre part elles traitent les contraintes de cyclicité de manière globale par le biais d'optimisations sous contraintes linéaires. Ainsi ces nouvelles méthodes bénéficient de la rapidité de la programmation linéaire.

Enfin, étant donné le nombre importants de travaux de recherche effectués en stéréovision, il convient de s'intéresser à l'étape suivante, à savoir la reconstruction spatio-temporelle. La méthode de [VKLP09] faisant partie de l'état de l'art en stéréovision, il est intéressant de développer cette méthode et de l'étendre à la reconstruction spatio-temporelle, c'est-à-dire la reconstruction d'une scène dynamique capturée par un dôme de caméras. Nous verrons cette méthode dans la seconde partie de ce manuscrit.

Ma thèse a été financée par le projet ANR Flamenco, qui est un projet sur la modélisation de scènes dynamiques mené en partenariat avec le laboratoire LIGM - IMAGINE, l'Ecole Nationale des Ponts et Chaussées et l'INRIA.

Table des matières

Première partie

“structure from motion” d’un réseau de caméras par programmation linéaire

Les résultats de stéréovision permettant d'acquérir des modèles 3D très précis, un objectif est désormais de reconstruire des scènes de plus en plus vastes et avec le plus de précision possible. Pour cela, il est important de calibrer, suffisamment vite des graphes d'images contenant des grands cycles de caméras. Nous proposons ici des méthodes efficaces et rapides essentiellement basées sur la programmation linéaire. D'autre part nous prenons en compte les contraintes de cyclicité de manière générale par le biais d'optimisations sous contraintes linéaires.

Pour cela nous pré-calculons certains objets grâce aux concepts fondamentaux de la calibration, et ensuite nous cherchons à déterminer les inconnues restantes, par des formulations matricielles liées aux correspondances dans les images, et donc à des informations géométriques. Les objets pré-calculés sont également obtenus grâce à des correspondances entre les images et sont gardés fixes et rendus cohérents entre eux durant nos optimisations. Le fait de n'utiliser que des informations géométriques, grâce à une construction cohérente, est central dans notre travail, et nous permet d'obtenir des résultats très proches de l'état de l'art sans recourir systématiquement a des optimisations coûteuses telles que le bundle adjustment (présenté en section 1.2.5), et à la connaissance a priori de données internes des caméras (centre, distance focale...).

Dans un premier temps, nous présentons les outils couramment utilisés en "structure from motion", ainsi que les concepts fondamentaux de la géométrie à plusieurs images. Le lecteur averti pourra passer directement au chapitre suivant, où nous présentons nos nouveaux algorithmes de "structure from motion".

Chapitre 1

La problématique de la structure à partir du mouvement

Contents

FIG. 1.1 – *Nuage de points d'une scène $3D$ et caméras autour de cette scène.*

1.1 Introduction

L'objectif de la "structure from motion", étant donné une scène $3D$ et des photos de cette scène qui ont été prises depuis différents points de vues, est de retrouver un nuage de points de cette scène à partir de correspondances entre les images, et aussi de retrouver les paramètres de ces caméras comme indiqué dans la figure 1.1. Nous cherchons à retrouver les positions du centre des caméras, les distances focales, les orientations des caméras...

Dans ce chapitre, nous étudierons l'ensemble des outils et concepts fondamentaux de la calibration utiles à la compréhension de ce manuscrit et plus précisément de la méthode de calibration par programmation linéaire que nous proposons.

Dans un premier temps il est nécessaire de détecter des "points d'intérêt" stables (par exemple invariants par transformation affine) dans les différentes images. Ensuite, il faut les décrire à l'aide de "descripteurs" puis les appareiller entre différentes images par ressemblance de ces descripteurs. Ces étapes sont étudiées en Annexe A. Une fois que nous avons ces ensembles de correspondances, nous pouvons retrouver la géométrie pour des paires d'images 1.3.1 et parfois pour des triplets d'images 1.3.2. Une fois que la géométrie entre les différentes images est décrite, il est possible de retrouver les caméras avec les outils tels que le "Bundle adjustment" vus dans la section 1.2.

Les descriptions détaillées des descripteurs SIFT et des méthodes de résolution de systèmes linéaires sont proposées en Annexe A et B. Il s'agit de méthodes classiques et bien connues en vision par ordinateur, elles sont uniquement reproduites dans ce manuscrit dans un but pédagogique et afin que ce manuscrit soit presque entièrement auto-consistant.

1.2 Outils fondamentaux pour la structure à partir du mouvement

1.2.1 Géométrie d'une caméra

1.2.1.1 Caméra métrique

Une caméra est composée d'une lentille, et les rayons passant sur les bords de la lentille ne sont pas déviés de la même façon ce qui crée des distorsions sur les bords de l'images. Ces problématiques n'ont pas été traitées dans la méthode proposée afin de conserver des modèles linéaires. On peut représenter de façon simple une caméra d'après le modèle "pinhole camera" montré en figure 1.2.

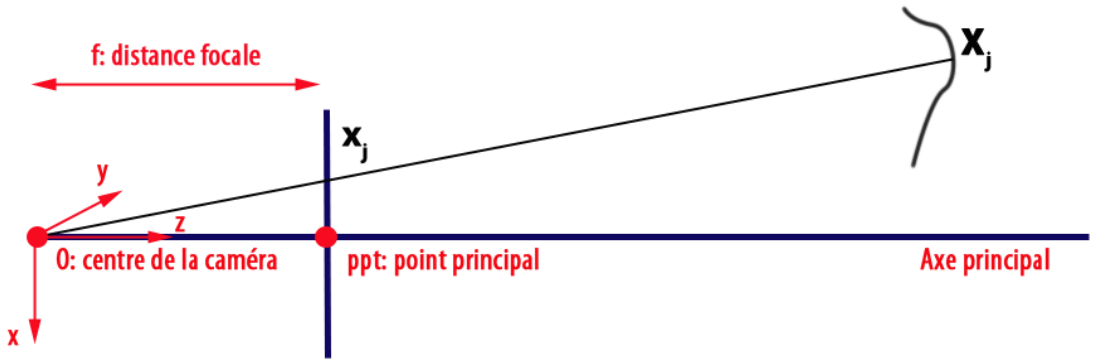


FIG. 1.2 – *Procédé de reprojection d'un point 3D $\mathbf{X}_j$ en un point image $\mathbf{x}_j$ par une caméra "pinhole" i de distance focale f_i.*

Si l'on note x_j, y_j les coordonnées euclidiennes de $\mathbf{x}_j$ dans le plan image de la camera i, et X_j, Y_j, Z_j les coordonnées euclidiennes du point réel $\mathbf{X}_j$ dans le repère R_i lié à la caméra i, on en déduit par une simple application du théorème de Thalès :

$$\begin{cases} x_j &= f_i \ \frac{X_j}{Z_j} \\ y_j &= f_i \ \frac{Y_j}{Z_j} \end{cases} \tag{1.1}$$

Sur la figure ci-dessus, les coordonnées $\mathbf{x}_j$ sont calculées à partir du centre de l'image, en général on les calcule à partir du coin supérieur gauche de l'image d'où la formule de reprojection :

$$\begin{cases} x_j &= f_i \ \frac{X_j}{Z_j} &+ u_i \\ y_j &= f_i \ \frac{Y_j}{Z_j} &+ v_i \end{cases} \tag{1.2}$$

Il est alors intéressant de transposer cette formule sous forme linéaire et matricielle en utilisant les coordonnées homogènes u_j,v_j,w_j empruntées à la géométrie projective. On a alors :

$$\begin{bmatrix} u_j \\ v_j \\ w_j \end{bmatrix} = \begin{bmatrix} f_i & 0 & u_i \\ 0 & f_i & v_i \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} \tag{1.3}$$

On retrouve les coordonnées euclidiennes x_j et y_j très facilement en divisant par la troisième coordonnée homogène, en effet :

$$\begin{cases} x_j &= \frac{u_j}{w_j} &= f_i \ \frac{X_j}{Z_j} &+ u_i \\ y_j &= \frac{v_j}{w_j} &= f_i \ \frac{Y_j}{Z_j} &+ v_i \end{cases} \tag{1.4}$$

Si l'on note X'_j, Y'_j, et Z'_j les coordonnées euclidiennes du point 3D $\mathbf{X}_j$, dans un repère quelconque R. Ce repère étant lié à R_i par une matrice de rotation $\mathtt{R}_i$ autour des trois axes et une translation $\mathbf{T}_i$ du centre du repère on a :

$$\begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} = \mathtt{R}_i \begin{bmatrix} X'_j \\ Y'_j \\ Z'_j \end{bmatrix} + \mathbf{T}_i \tag{1.5}$$

En utilisant les coordonnées homogènes X'_j, Y'_j, et Z'_j, T'_j du point 3D $\mathbf{X}_j$, dans le repère R on a l'égalité suivante à un facteur multiplicatif près :

$$\begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} \cong [\mathtt{R}_i \, \mathbf{T}_i] \begin{bmatrix} X'_j \\ Y'_j \\ Z'_j \\ T'_j \end{bmatrix} \tag{1.6}$$

On a donc l'équation de reprojection suivante du point 3D $\mathbf{X}_j$ dans un repère quelconque R vers les coordonnées pixelliques du point dans l'image, en coordonnées homogènes :

$$\begin{bmatrix} u_j \\ v_j \\ w_j \end{bmatrix} = \begin{bmatrix} f_i & 0 & u_i \\ 0 & f_i & v_i \\ 0 & 0 & 1 \end{bmatrix} [\mathtt{R}_i \, \mathbf{T}_i] \begin{bmatrix} X'_j \\ Y'_j \\ Z'_j \\ T'_j \end{bmatrix} \tag{1.7}$$

Finalement, la caméra peut être modélisée sous la forme d'une matrice $\mathtt{P}_i$ 3×4, se décomposant sous la forme $\mathtt{P}_i = \mathtt{K}_i \, [\mathtt{R}_i \, \mathbf{T}_i]$. $\mathtt{K}_i$ étant la matrice des paramètres intrinsèques : distance focale, centre de l'image... et $[\mathtt{R}_i \, \mathbf{T}_i]$ représentant les paramètres extrinsèques : position et orientation de la caméra dans l'espace.

1.2.1.2 Caméra projective et ambiguïté projective

Afin de conserver des calculs simples et linéaires on considèrera que les matrices des caméras s'écrivent simplement sous la forme de matrices quelconques $\mathtt{P}_i$ 3×4. On pourra donc reconstruire un ensemble de caméras $\{\mathtt{P}_i, \forall i = 1, ..., N\}$ ainsi qu'un ensemble de points $3D$ $\{\mathbf{X}_j, \forall j = 1, ..., M\}$ en coordonnées homogènes, et leurs reprojections dans les images $\{\mathbf{x}_{\mathbf{j}_i}, \forall j = 1, ..., M \, , \, \forall i = 1, ..., N\}$.

Si un point $\mathbf{X}_j$ se reprojette sur un point $\mathbf{x}_{\mathbf{j}_i}$ dans l'image i, alors on a :

$$\begin{bmatrix} u_{j_i} \\ v_{j_i} \\ w_{j_i} \end{bmatrix} = \mathtt{P}_i \begin{bmatrix} X_j \\ Y_j \\ Z_j \\ T_j \end{bmatrix} \tag{1.8}$$

Dans ce cas on note que pour une transformation projective ou en d'autres termes une homographie $\mathtt{H}$ 4×4 inversible, on a :

$$\begin{bmatrix} u_{j_i} \\ v_{j_i} \\ w_{j_i} \end{bmatrix} = \mathtt{P}_i\mathtt{H} \quad \mathtt{H}^{-1} \begin{bmatrix} X_j \\ Y_j \\ Z_j \\ T_j \end{bmatrix} \tag{1.9}$$

Aussi dans un autre espace projectif, on a un ensemble de caméras $\{\mathtt{P}_i\mathtt{H}, \forall i = 1, ..., N\}$ qui est une solution équivalente pour un ensemble de points $3D$ $\{\mathtt{H}^{-1}\mathbf{X}_j, \forall j = 1, ..., M\}$

en coordonnées homogènes, ainsi que leurs reprojections dans les images $\left\{\mathbf{x}_{\mathbf{j}_i}, \forall j = 1, ..., M\ , \forall i = 1, ..., N\right\}$.

C'est ce que l'on nomme l'ambiguïté projective. Aussi une fois que l'on aura retrouvé l'ensemble des caméras dans un espace projectif, on cherchera l'homographie telle que $\forall i = 1, ..., N\ \mathtt{HP}_i \approx \mathtt{K}_i\left[\mathtt{R}_i\ \mathbf{T}_i\right]$. Où $\mathtt{K}_i$ est une caméra métrique de la forme :

$$\mathtt{K}_i \quad = \quad \begin{bmatrix} f_i & 0 & u_i \\ 0 & f_i & v_i \\ 0 & 0 & 1 \end{bmatrix} \tag{1.10}$$

Cette méthode permet de retrouver à partir d'une reconstruction projective, la reconstruction métrique, c'est-à-dire la véritable configuration des caméras et des points $3D$ à une similarité près. Les méthodes de rectification métrique sont brièvement expliquées dans la section 1.2.2.

1.2.2 *Rectification métrique*

Il existe de nombreuses méthodes de rectification métrique, décrites dans le livre [HZ03]. L'une des plus connues est basée sur la "quadrique duale absolue" et nécessite une connaissance du centre de la caméra, qui peut presque toujours être choisi comme étant le centre pixellique de l'image. Une autre permet également d'obtenir une rectification métrique de manière linéaire et se base cette fois sur ce qui est appelé le "complexe quadratique absolu" et est développée par Ponce *et al.* [PMP+05]. L'avantage de cette dernière méthode est qu'elle permet d'obtenir une rectification métrique à partir de 10 images sans avoir aucune connaissance du centre de l'image.

Nous allons ici décrire très brièvement la méthode basée sur la "quadrique duale absolue".

Une matrice $\mathtt{P}_i$ 3×4 peut toujours s'écrire à l'aide d'une "décomposition QR" sous la forme :

$$\mathtt{P}_i \quad = \quad \begin{bmatrix} f_{ix} & s_i & u_i \\ 0 & f_{iy} & v_i \\ 0 & 0 & 1 \end{bmatrix} \left[\mathtt{R}\ \mathbf{T}\right] \tag{1.11}$$

s_i correspond au skew, f_{ix} et f_{iy} correspondent aux focales en x et en y, et $(u_i, v_i)^T$ correspond au centre de l'image. Nous avons vu dans la section 1.2.1.2, qu'il y a une ambiguïté projective. Connaissant les caméras, nous allons chercher l'homographie qui permet d'aller vers un espace métrique dans lequel les caméras sont métriques :

$$\mathtt{P}'_i = \quad \mathtt{P}_i\,\mathtt{H} \quad = \quad \begin{bmatrix} f_i & 0 & u_i \\ 0 & f_i & v_i \\ 0 & 0 & 1 \end{bmatrix} \left[\mathtt{R}\ \mathbf{T}\right] \tag{1.12}$$

Si l'on se place au centre pixellique de l'image en multipliant par une matrice de translation, la décomposition de la matrice une fois rectifiée doit avoir la forme :

$$\mathtt{P}'_i = \mathtt{P}_i\,\mathtt{H} = \begin{bmatrix} f_i & 0 & 0 \\ 0 & f_i & 0 \\ 0 & 0 & 1 \end{bmatrix} [\mathtt{R}\,\mathbf{T}] \tag{1.13}$$

Posons :

$$\tilde{\mathtt{I}} = \left[\begin{array}{c|c} \mathtt{I} & \mathbf{0} \\ \hline \mathbf{0} & 0 \end{array}\right] \tag{1.14}$$

Nous obtenons alors l'équation :

$$\mathtt{P}'_i\,\tilde{\mathtt{I}}\,{\mathtt{P}'_i}^T = \mathtt{P}_i\,\mathtt{H} = \begin{bmatrix} {f_i}^2 & 0 & 0 \\ 0 & {f_i}^2 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \mathtt{P}_i\left(\mathtt{H}\,\tilde{\mathtt{I}}\,\mathtt{H}^T\right){\mathtt{P}_i}^T \tag{1.15}$$

Si nous posons maintenant $\mathtt{Q} = \mathtt{H}\,\tilde{\mathtt{I}}\,\mathtt{H}^T$, nous avons alors pour chaque caméra i les équations :

$$\begin{bmatrix} {f_i}^2 & 0 & 0 \\ 0 & {f_i}^2 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \mathtt{P}_i\,\mathtt{Q}\,{\mathtt{P}_i}^T. \tag{1.16}$$

On obtient ainsi pour chaque image quatre équations. En effet les deux coordonnées du centre doivent être nulles dans la matrice gauche, le skew doit être nul et les deux coefficients liés à la distance focale doivent être égaux. $\mathtt{Q}$ est symétrique et a 16 coefficients, cette matrice est donc paramétrisable par 10 coefficients. 3 images et 12 équations suffisent donc à effectuer une rectification métrique.

Connaissant $\mathtt{Q}$, il est facile de trouver l'upgrade métrique $\mathtt{H}$ en effectuant dans un premier temps une "décomposition de cholesky" de $\mathtt{Q}$. Cette méthode de rectification métrique due à M. Pollefeys a l'avantage d'être rapide et linéaire.

Une étude plus approfondie de cette méthode et des autres méthodes de rectification métrique est effectuée dans le livre [HZ03].

1.2.3 Algorithme RANdom SAmple Consensus

Partons d'un exemple simple. Estimer une droite peut se faire avec un ensemble minimal de 2 points. Le problème illustré en figure 1.3, est le suivant : étant donné un ensemble de points $2D$, trouver la droite maximisant le nombre de points valides, en considérant qu'un point est valide s'il n'est pas plus éloigné de la droite qu'un seuil t. Le problème est double, on souhaite trouver la droite correspondant le mieux possible aux points et on veut également classifier les points comme "inliers" en adéquation avec un modèle, ou "outliers".

Nous allons ici décrire une méthode bien connue qui est l'algorithme RANdom SAmple Consensus (RANSAC) de Fischler et Bolles [BF81]. L'idée est très simple : 2 points sont sélectionnés aléatoirement. Ces points définissent une droite. Le support de cette droite est le nombre de points, parmi l'ensemble complet initial, qui sont à une distance de la droite inférieure à un

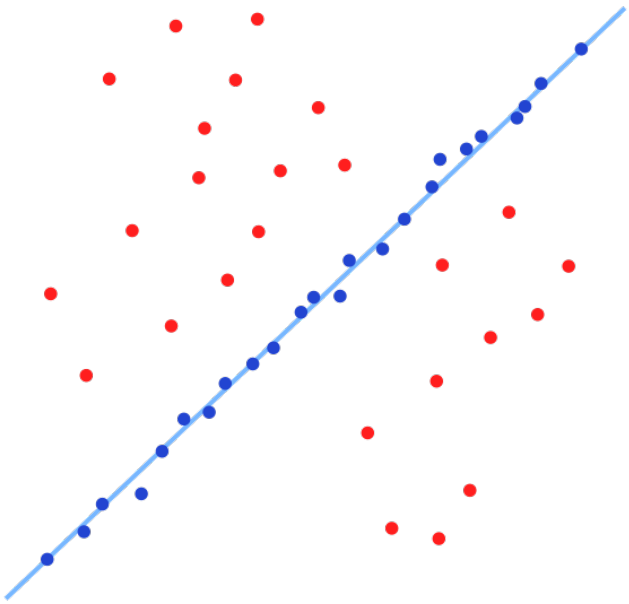


FIG. 1.3 – *Deux points suffisent à créer un modèle, ici une droite. Les points bleus, proches de la droite, sont classifiés comme "inliers" et les points en rouge, éloignés de la droite, sont classifiés comme "outliers".*

seuil t. Cette sélection aléatoire est répétée un grand nombre de fois, et à la fin la droite ayant le plus grand support est retenue comme modèle. Les points à une distance inférieure à un seuil t de la droite sont nommés "inliers" et constituent l'ensemble de plus grand consensus. Les autres points sont nommées "outliers". L'étape finale du RANSAC est une ré-estimation du modèle à partir de l'ensemble des inliers.

Dans le cas général le modèle ne sera pas estimé à partir de 2 points mais à partir d'un échantillon de s points. Nous souhaitons déterminer le nombre d'itérations nécessaires N. Nous voulons que N permette d'obtenir une probabilité p, que au moins un des échantillons aléatoires de s points ne contienne aucun outlier. On choisit généralement $p = 0.99$. $1 - p$ est la probabilité que chacun des N échantillons contienne au moins un outlier. Notons w la probabilité qu'un point soit un inlier qui n'est en fait que la proportion d'inliers par rapport à l'ensemble des points. Donc w^s est la probabilité qu'un échantillon n'ait pas d'outlier. $1 - w^s$ est la probabilité qu'un échantillon ait au moins un outlier. La probabilité que chacun des N échantillons contienne au moins un outlier est donc $(1 - w^s)^N = 1 - p$. Le nombre d'itérations nécessaires sera donc :

$$N = \frac{\log(1-p)}{\log(1-w^s)} \tag{1.17}$$

En pratique on fixera un seuil maximal pour le nombre d'itération, afin d'éviter un algorithme beaucoup trop long dans le cas où il y a très peu d'inliers. Le RANSAC est brièvement résumé dans l'algorithme 1.2.3.

Algorithm 1 RANSAC

Require: $\mathbf{S}$ un ensemble de points de cardinal C

Initialise $SampleCount = 0$, $I_{max} = 0$, $\mathbf{S_i} = \emptyset$, N_{max}, p, t
while $N > SampleCount$ et $N_{max} > SampleCount$ **do**
  Choisis un échantillon aléatoire de s points
  Avec les s points calcule le modèle M
  Pour chaque point de $\mathbf{S}$ calcule la distance au modèle d
  Détermine l'ensemble des inliers $\mathbf{S_{itemp}}$, tels que $d < t$
  Compte le nombre d'inliers I, $w = \frac{I}{C}$.
  if $I > I_{max}$ **then**
    Mets à jour $N = \frac{\log(1-p)}{\log(1-w^s)}$
    Mets à jour $I_{max} = I$
    Mets à jour $\mathbf{S_i} = \mathbf{S_{itemp}}$
  end if
  Incrémente $SampleCount = SampleCount + 1$
end while
Calcule le modèle M à partir de l'ensemble des inliers $\mathbf{S_i}$
return le modèle M et l'ensemble des inliers $\mathbf{S_i}$

1.2.4 Triangulation et erreur de reprojection

A partir d'une correspondance à n images $(\mathbf{p}_1, \cdots, \mathbf{p}_n)$ et n caméras $(\mathtt{P}_1, \cdots, \mathtt{P}_n)$ avec $n \geq 2$, on peut reconstruire le point tridimensionnel $\mathbf{X}$. Si l'on considère que les correspondances sont parfaites et non bruitées, alors le point $\mathbf{X}$ doit se reprojeter exactement en $\mathbf{p}_1, \cdots, \mathbf{p}_n$ au travers des caméras $\mathtt{P}_1, \cdots, \mathtt{P}_2$. On a donc les équations linéaires en coordonnées homogènes $\mathbf{p}_i \cong \mathtt{P}_1\mathbf{X}$ pour i allant de 1 à n, avec $\mathbf{X} = (X; Y; Z; 1)$. Autrement dit nous avons les $3n$ équations $\lambda_i \mathbf{p}_i = \mathtt{P}_1\mathbf{X}$. Ces équations sont linéaires par rapport aux n inconnues λ_i et aux 3 inconnues X, Y et Z. On peut donc mettre ces équations sous la forme matricielle :

$$\mathtt{A} \begin{bmatrix} X \\ Y \\ Z \\ \lambda_1 \\ \cdots \\ \lambda_n \end{bmatrix} = \mathbf{b} \tag{1.18}$$

Avec $\mathtt{A}$ de taille $3n \times n+3$. En pratique les données sont bruitées et imparfaites et il n'existe pas de solution exacte à ce système. A partir de $n = 2$ on a un système d'équations linéaires surdéterminé qui peut être résolu aux moindres carrés par SVD comme expliqué en Annexe B.1. Nous obtenons ainsi une première approximation du point tridimensionnel $\mathbf{X}$ par une méthode dite de triangulation linéaire.

Cette équation linéaire n'a pas véritablement de sens géométrique, aussi on souhaite calculer plus précisément le point tridimensionnel en minimisant une erreur ayant un sens géométrique.

On peut raffiner le point $3D$ reconstruit par triangulation linéaire, en minimisant par exemple la somme des erreurs géométriques aux carrés. Nous définissons ci-dessous l'erreur géométrique.

Définition 1 *Erreur Géométrique*
Etant donné un point $\mathbf{p}$ *dans une image* i *dont la caméra est* $\mathtt{P}_i$*, et un point tridimensionnel* $\mathbf{X}$ *en coordonnées homogènes, l'erreur géométrique correspond à l'erreur de reprojection, c'est-à-dire la distance entre le point* $\mathbf{p}$ *et la projection de* $\mathbf{X}$ *dans l'image* i*.*

La reprojection $\widehat{\mathbf{p}}$ de $\mathbf{X}$ par la caméra $\mathtt{P}_i$ est en coordonnées homogènes $\widehat{\mathbf{p}} = \mathtt{P}_i\mathbf{X}$. Comme expliqué en section 1.2.1, on retrouve les coordonnées euclidiennes de $\widehat{\mathbf{p}}$ en divisant par la troisième coordonnée homogène. Si l'on note ${\mathtt{P}_i}^k$ la $k^{\text{ième}}$ ligne de la caméra projective $\mathtt{P}_i$, le point reprojeté a pour coordonnées euclidiennes :

$$\begin{cases} \widehat{x} &= \frac{{\mathtt{P}_i}^1\mathbf{X}}{{\mathtt{P}_i}^3\mathbf{X}} \\ \\ \widehat{y} &= \frac{{\mathtt{P}_i}^2\mathbf{X}}{{\mathtt{P}_i}^3\mathbf{X}} \end{cases} \tag{1.19}$$

Concernant le problème d'une correspondance dans n images $(\mathbf{p}_1, \cdots, \mathbf{p}_n)$ et n caméras $(\mathtt{P}_1, \cdots, \mathtt{P}_n)$, on peut raffiner le point reconstruit $\mathbf{X}$ en minimisant l'erreur suivante par rapport à $\mathbf{X}$:

$$\sum_i^n \left[\left(x - \frac{{\mathtt{P}_i}^1\mathbf{X}}{{\mathtt{P}_i}^3\mathbf{X}} \right)^2 + \left(y - \frac{{\mathtt{P}_i}^2\mathbf{X}}{{\mathtt{P}_i}^3\mathbf{X}} \right)^2 \right] \tag{1.20}$$

Il s'agit d'un problème non linéaire par rapport aux 3 inconnues $\mathbf{X}$. La quatrième coordonnée de $\mathbf{X}$ est généralement arbitrairement fixée à 1, sauf dans le cas particulier d'un point à l'infini où la quatrième coordonnée homogène est nulle. On peut par exemple minimiser cette erreur en utilisant la méthode itérative de Levenberg-Marquardt brièvement décrite en Annexe B.2.3.

1.2.5 Ajustement des faisceaux

1.2.5.1 "Bundle adjustment" projectif

Considérons un ensemble de caméras $\{\mathtt{P}_i, \forall i = 1, ..., N\}$, ainsi qu'un ensemble de points $3D$ $\{\mathbf{X}_j, \forall j = 1, ..., M\}$ en coordonnées homogènes, les correspondances dans les images dont ces points $3D$ sont issus $\left\{ \mathbf{p_{j_i}} = \left(x_{j_i}, y_{j_i}, 1\right)^T, \forall j = 1, ..., M \, , \, \forall i = 1, ..., N \right\}$ et $\left\{ \epsilon_{j_i}, \forall j = 1, ..., M \, , \, \forall i = 1, ..., N \right\}$ avec $\epsilon_{j_i} = 1$ si $\mathbf{X}_j$ se reprojette dans l'image i et $\epsilon_{j_i} = 0$ sinon.

Dans ce cas on peut minimiser l'erreur géométrique, correspondant à la somme des erreurs de reprojection :

$$\mathbf{S} = \sum_{i=1}^n \sum_{j=1}^N \epsilon_{j_i} \left[\left(x_{j_i} - \frac{{\mathtt{P}_i}^1\mathbf{X}_j}{{\mathtt{P}_i}^3\mathbf{X}_j} \right)^2 + \left(y_{j_i} - \frac{{\mathtt{P}_i}^2\mathbf{X}_j}{{\mathtt{P}_i}^3\mathbf{X}_j} \right)^2 \right] \tag{1.21}$$

On appelle l'ajustement des faisceaux (en anglais "bundle adjustment") le fait de minimiser la somme des erreurs de reprojection définie ci-dessus, à la fois par rapport aux points $3D$ comme décrit en section 1.2.4, mais également par rapport aux caméras $\mathtt{P}_i$. Ceci se fait généralement avec la méthode itérative de Levenberg-Marquardt brièvement décrite en Annexe B.2.3. Pour un grand nombre de caméras il s'agit d'un problème à grande dimension, de plus le jacobien $\mathtt{J}$ correspondant a une forte sparsité, on utilisera donc une implémentation sparse de Levenberg-Marquardt disponible sur le web `http://www.ics.forth.gr/~lourakis/sba/` et basée sur un article de M.Lourakis *et al.* [LA09].

Généralement l'ajustement des faisceaux est basé sur une méthode de recherche de Levenberg-Marquardt, qui allie rapidité et garanties de convergence. Dans [MBG09] une méthode de recherche alternative est proposée. Elle consiste a calculer le pas de chaque itération de façon optimale en se basant sur une distance algébrique. Cet algorithme offre les mêmes garanties de convergence que la méthode itérative de Levenberg-Marquardt et converge souvent plus rapidement dans le cas qui nous intéresse, à savoir le calcul de la "Structure à partir du mouvement".

Définition 2 *"Mean Square" Erreur*
L'erreur pixellique "Mean Square" ou erreur MS correspond à la somme des erreurs de reprojection divisée par la somme des reprojections. Celle-ci s'écrit comme suit :

$$\mathbf{MS} = \sum_{i=1}^{n} \sum_{j=1}^{N} \frac{\epsilon_{j_i}}{\sum_{i=1}^{n} \sum_{j=1}^{N} \epsilon_{j_i}} \left[\left(x_{j_i} - \frac{{P_i}^1 \mathbf{X}_j}{{P_i}^3 \mathbf{X}_j} \right)^2 + \left(y_{j_i} - \frac{{P_i}^2 \mathbf{X}_j}{{P_i}^3 \mathbf{X}_j} \right)^2 \right] \tag{1.22}$$

Définition 3 *"Root Mean Square" Erreur*
L'erreur pixellique "Root Mean Square" ou erreur RMS correspond à la racine de l'erreur "Mean Square". Celle-ci s'écrit comme suit :

$$\mathbf{RMS} = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{N} \frac{\epsilon_{j_i}}{\sum_{i=1}^{n} \sum_{j=1}^{N} \epsilon_{j_i}} \left[\left(x_{j_i} - \frac{{P_i}^1 \mathbf{X}_j}{{P_i}^3 \mathbf{X}_j} \right)^2 + \left(y_{j_i} - \frac{{P_i}^2 \mathbf{X}_j}{{P_i}^3 \mathbf{X}_j} \right)^2 \right]} \tag{1.23}$$

Lorsque l'on minimise l'erreur $\mathbf{S}$ *par ajustement des faisceaux par rapport aux caméras et aux points* $3D$*, on minimise également l'erreur pixellique RMS. L'erreur RMS est une bonne mesure de qualité de la reconstruction, lorsque* $\mathbf{RMS} < 1$*, on dira que l'on a une erreur de reprojection sous-pixellique.*

1.2.5.2 "Bundle adjustment" métrique

Si l'on part de caméras métriques ou quasiment métriques, définies en section 1.2.1.1, il est possible d'affiner la qualité de la reconstruction, avec les points reconstruits. Par exemple on

peut décomposer les matrices, après rectification métrique linéaire (décrit en section 1.2.2), sous la forme quasiment métrique $\mathtt{P}_i = \mathtt{K}_i \left[\mathtt{R}_i \, \mathbf{T}_i\right]$. Puis forcer $\mathtt{P}_i = \mathtt{K'}_i \left[\mathtt{R}_i \, \mathbf{T}_i\right]$, $\mathtt{K'}_i$ étant la matrice la plus proche de $\mathtt{K}_i$ vérifiant les conditions skew nul et distances focales égales, c'est-à-dire s'écrivant simplement :

$$\mathtt{K}'_i \;=\; \begin{bmatrix} f_i & 0 & \widehat{u_i} \\ 0 & f_i & \widehat{v_i} \\ 0 & 0 & 1 \end{bmatrix}. \tag{1.24}$$

Puis on effectue une minimisation aux moindres carrés de la distance de reprojection, en utilisant comme paramètres les points reconstruits ainsi que la décomposition des matrices : $\mathtt{K'}_i$ se réduisant au paramètre unique f_i ainsi qu'aux coordonnées du centre. Et pour $\mathtt{R}_i$ on peut utiliser une paramétrisation par quaternions. La méthode utilisée pour le programme n'est pas celle-là, mais consiste à partir directement des matrices $\mathtt{K}_i$ ayant la forme générale

$$\mathtt{K}_i \;=\; \begin{bmatrix} f_{i_x} & s_i & u_i \\ 0 & f_{i_y} & v_i \\ 0 & 0 & 1 \end{bmatrix}, \tag{1.25}$$

et à minimiser la distance de reprojection ainsi qu'une fonction de coût liée aux différences quadratiques des distances focales suivant les axes $\left(f_{i_x} - f_{i_y}\right)^2$, au carré du skew ${s_i}^2$ ainsi qu'à la différence quadratique entre le centre obtenu et le véritable centre pixellique de l'image $\left(u_i - \widehat{u_i}\right)^2 + \left(v_i - \widehat{v_i}\right)^2$. La paramétrisation et la fonction d'erreur ainsi définies, on utilise la méthode itérative de Levenberg-Marquardt brièvement décrite en Annexe B.2.3 pour minimiser la fonctionnelle.

1.3 Concepts fondamentaux de géométrie à plusieurs images

1.3.1 Géométrie à deux images

1.3.1.1 La matrice fondamentale

Sur la figure 1.4 on remarque que $\vec{OP}$, $\vec{OO'}$ et $\vec{O'P}$ sont coplanaires, il s'ensuit que :

$$\vec{O'P} \cdot \left(\vec{OO'} \times \vec{OP}\right) = 0 \tag{1.26}$$

On note $\mathbf{t}$ la translation entre les repères des deux caméras correspondant à $\vec{OO'}$, $\mathtt{R}$ la rotation entre les deux caméras, $\mathbf{P}$ les coordonnées non homogènes du point P dans le repère de la première caméra, et $\mathbf{P}'$ les coordonnées non homogènes du point P dans le repère de la seconde caméra. Alors l'équation précédente s'écrit simplement,

$$\mathbf{P}' \cdot \left[\mathbf{t} \times (\mathtt{R}\mathbf{P})\right] = {\mathbf{P}'}^T \left[[\mathbf{t}]_\times \, \mathtt{R}\right] \mathbf{P} = 0 \tag{1.27}$$

Comme montré dans la section 1.2.1 consacrée à la modélisation d'une caméra, les coordonnées homogènes $\mathbf{p}$ et $\mathbf{p}'$ des points dans les deux images s'écrivent en fonction des paramètres intrinsèques $\mathtt{K}_1$ et $\mathtt{K}_2$ des deux caméras :

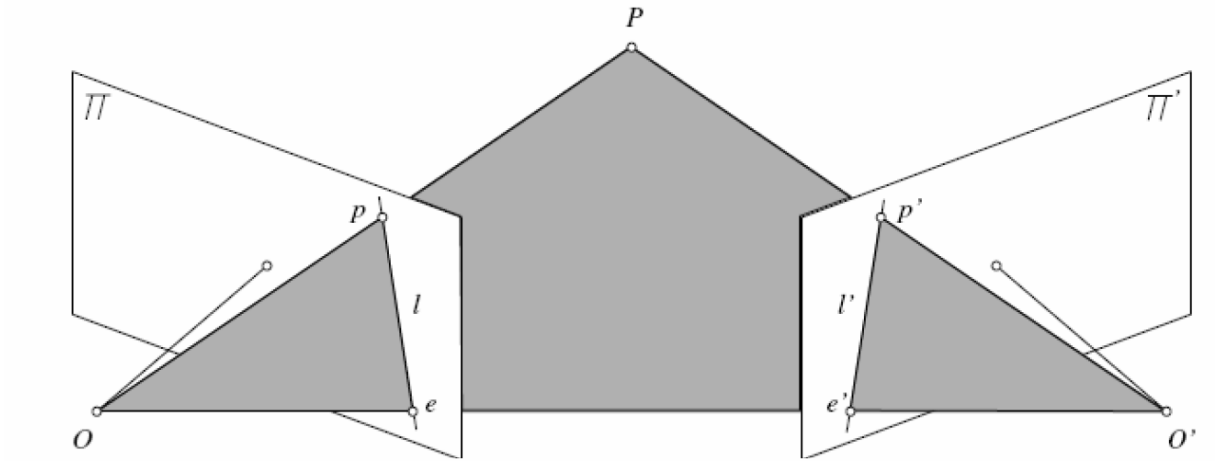


FIG. 1.4 – *Géométrie épipolaire.*

$$\begin{cases} \mathbf{p} &= \mathtt{K}_1 \quad \mathbf{P} \\ \mathbf{p}' &= \mathtt{K}_2 \quad \mathbf{P}' \end{cases} \text{ ou, } \begin{cases} \mathbf{P} &= {\mathtt{K}_1}^{-1} \quad \mathbf{p} \\ \mathbf{P}' &= {\mathtt{K}_2}^{-1} \quad \mathbf{p}' \end{cases} \tag{1.28}$$

On en déduit donc l'équation :

$$\mathbf{p}'^T {\mathtt{K}_2}^{-T} [\mathbf{t}]_\times \mathtt{R} {\mathtt{K}_1}^{-1} \mathbf{p} = 0 \tag{1.29}$$

La matrice du milieu ne dépendant pas du point considéré, pour toute correspondance $(\mathbf{p}, \mathbf{p}')$ entre deux images on a :

$$\mathbf{p}'^T \mathtt{F} \mathbf{p} = 0 \quad \text{avec,} \quad \mathtt{F} = {\mathtt{K}_2}^{-T} [\mathbf{t}]_\times \mathtt{R} {\mathtt{K}_1}^{-1} \tag{1.30}$$

On note que $\mathtt{F}\ (\mathtt{K}_1 \mathtt{R}^{-1} \mathbf{t}) = {\mathtt{K}_2}^{-T} [\mathbf{t}]_\times \mathtt{R} {\mathtt{K}_1}^{-1} (\mathtt{K}_1 \mathtt{R}^{-1} \mathbf{t}) = {\mathtt{K}_2}^{-T} [\mathbf{t}]_\times \mathbf{t} = \mathbf{0}$, soit $\mathtt{F}\mathbf{e} = \mathbf{0}$. $\mathbf{e}$ est l'épipole de la première image, c'est-à-dire la reprojection dans la première image du centre de la seconde caméra. De même on a l'équation $\mathbf{e}'\mathtt{F} = \mathbf{0}$. Les épipoles $\mathbf{e}$ et $\mathbf{e}'$ sont représentées en figure 1.4.

Cette matrice $\mathtt{F}$, appelée " matrice fondamentale " est singulière et peut être calculée à partir d'un nombre minimal de correspondances entre deux images.

1.3.1.2 Equivalences entre caméras projectives et matrice fondamentale

L'épipole $\mathbf{e}$ correspond au vecteur nul de la matrice fondamentale $\mathtt{F}$, de même l'épipole $\mathbf{e}'$ correspond au vecteur nul gauche de $\mathtt{F}$. A partir de $\mathtt{F}$ et des épipoles, il est possible d'estimer un couple de matrices projectives pour chaque caméra qui soit en adéquation avec la géométrie épipolaire. Il est toujours possible de fixer l'espace projectif arbitrairement de telle manière que la matrice de la première caméra soit $\mathtt{P}_1 = [\mathtt{I} \mid \mathbf{0}]$.

Proposition 1 *La matrice $\mathtt{F}$ est la matrice fondamentale associée aux caméras $\mathtt{P}_1$ et $\mathtt{P}_2$ si et seulement si $\mathtt{P}_2^T \mathtt{F} \mathtt{P}_1$ est antisymétrique.*

Preuve
$P_2^T F P_1$ *est antisymétrique* $\Leftrightarrow \mathbf{X}^T P_2^T F P_1 \mathbf{X} = 0 \; \forall \mathbf{X}$. *En posant* $\mathbf{p}' = P_2\mathbf{X}$ *et* $\mathbf{p} = P_1\mathbf{X}$ *on vérifie la relation de la matrice fondamentale :* $\mathbf{p}'^T F \mathbf{p} = 0$.

Il est alors facile de vérifier que $\mathtt{P}_2^T \mathtt{F} \mathtt{P}_1$ est antisymétrique en posant $\mathtt{P}_1 = [\mathtt{I} \,|\, \mathbf{0}]$ et $\mathtt{P}_2 = [[\mathbf{e}']_\times \mathtt{F} \,|\, \mathbf{e}']$. Ainsi ces deux caméras projectives satisfont la géométrie épipolaire décrite par $\mathtt{F}$.

Proposition 2 *La matrice fondamentale associée à deux caméras* $P_1 = [I \,|\, \mathbf{0}]$ *et* $P_2 = [M \,|\, \mathbf{m}]$ *est* $F = [\mathbf{m}]_\times M$.

Preuve
En effet deux caméras sont associées à F *si et seulement si* ${P_2}^T F P_1$ *est antisymétrique d'après la proposition 1. Or en posant* $F = [\mathbf{m}]_\times M$, *on a :*

$$ {P_2}^T F P_1 = \left[\begin{array}{c|c} M^T [\mathbf{m}]_\times M & \mathbf{0} \\ \hline \mathbf{0} & \end{array} \right] \tag{1.31} $$

Or $\forall \mathbf{X}$, $(\mathbf{X}^T M^T)[\mathbf{m}]_\times (M\mathbf{X}) = \mathbf{Y}^T [\mathbf{m}]_\times \mathbf{Y} = 0$. *La matrice ci-dessus est donc antisymétrique ce qui termine la preuve.*

Proposition 3 *Deux paires de caméras* $(P_1\,,\,P_2)$ *et* $(P'_1\,,\,P'_2)$ *sont associées à la même matrice fondamentale si et seulement si* $\exists H$ *tel que* $P_1 \cong P'_1 H$ *et* $P_2 \cong P'_2 H$.

Preuve
Supposons que la matrice fondamentale associée à $(P_1\,,\,P_2)$ *soit* F. *Dans ce cas* ${P_2}^T F P_1$ *est anti-symétrique. Supposons qu'il existe une homographie* H *telle que* $P_1 = P'_1 H$ *et* $P_2 = P'_2 H$, *alors on a évidemment* ${H^{-T}\, {P_2}^T F P_1\, H^{-1}}$ *est anti-symétrique. En d'autres termes* ${P'_2}^T F P'_1$ *est anti-symétrique et donc la matrice fondamentale associée aux caméras* $(P'_1\,,\,P'_2)$ *est également* F.

Nous avons démontré le "si", nous allons maintenant montrer le "seulement si". Supposons que $(P_1\,,\,P_2)$ *et* $(P'_1\,,\,P'_2)$ *sont associées à la même matrice fondamentale, c'est-à-dire associées à* F_1 *et* F_2 *telles que* $F_1 \cong F_2$, *nous allons montrer qu'il existe une homographie reliant ces paires de caméras.*
$(P_1\,,\,P_2)$ *est associée à* F_1. P_1 *est de rang* 3 *et a un centre* $\mathbf{C}$ *tel que* $P_1\mathbf{C} = \mathbf{0}$. *En posant* $H_1 = [{P_1}^+ \,|\, \mathbf{C}]$, *on a* $(P_1H_1\,,\,P_2H_1) = ([I \,|\, \mathbf{0}]\;,\;[M \,|\, \mathbf{m}])$ *est une paire de caméras associées à la matrice fondamentale* F_1.

$(P'_1\,,\,P'_2)$ *est associée à* F_2. P'_1 *a un centre* $\mathbf{C}'$. *En posant* $H_2 = \left[{P'_1}^+ \,|\, \mathbf{C}'\right]$, *on a* $(P'_1H_2\,,\,P'_2H_2) = ([I \,|\, \mathbf{0}]\;,\;[M' \,|\, \mathbf{m}'])$ *est une paire de caméras associée à la matrice fondamentale* F_2. *Et* $F_1 \cong F_2$.

Nous n'avons plus qu'à montrer qu'il existe une homographie H *telle que* $[I \,|\, \mathbf{0}] \cong [I \,|\, \mathbf{0}]\, H$ *et* $[M \,|\, \mathbf{m}] \cong [M' \,|\, \mathbf{m}']\, H$.

D'après la proposition 2, nous savons que $F_1 \cong F_2$ *s'écrit* $[\mathbf{m}]_\times M \cong [\mathbf{m}']_\times M'$. *On en déduit que nécessairement* $\mathbf{m} \cong \mathbf{m}'$. *Nous avons donc :*

$$\begin{cases} \mathbf{m}' & \cong & \mathbf{m} \\ [\mathbf{m}]_\times M & \cong & [\mathbf{m}]_\times M' \end{cases} \tag{1.32}$$

Avec des calculs vectoriels simples on en arrive à la conclusion que :

$$\begin{cases} \mathbf{m}' & = & v_4 \mathbf{m} \\ M & = & \lambda M' + \mathbf{m}\,\mathbf{v}^T \end{cases} \tag{1.33}$$

Pour terminer cette preuve, il existe donc une homographie H *reliant les paires de caméras, s'écrivant :*

$$H = \left[\begin{array}{c|c} \lambda I & \mathbf{0} \\ \hline \mathbf{v}_{1:3} & v_4 \end{array}\right] \tag{1.34}$$

Ce résultat est également démontré dans le livre [FLP01]. En d'autres termes les caméras associées à une matrice fondamentale sont uniques à une transformation projective près.
L'ensemble des caméras associées à une matrice fondamentale $\mathtt{F}$ est donc $\mathtt{P}_1 = [\mathtt{I} \,|\, \mathbf{0}]\, \mathtt{H}$ et $\mathtt{P}_2 = [[\mathbf{e}']_\times \mathtt{F} \,|\, \mathbf{e}']\, \mathtt{H}$. Les homographies telles que $\mathtt{P}_1 = [\mathtt{I} \,|\, \mathbf{0}]$ sont de la forme :

$$\mathtt{H} = \left[\begin{array}{c|c} \mathtt{I} & \mathbf{0} \\ \hline \mathbf{v}_{1:3} & v_4 \end{array}\right] \tag{1.35}$$

Les caméras dans la seconde images sont donc de la forme :

$$\mathtt{P}_2 = [[\mathbf{e}']_\times \mathtt{F} \,|\, \mathbf{0}] + \mathbf{e}'\mathbf{v}^T \tag{1.36}$$

Proposition 4 *L'ensemble des caméras* P_1 *et* P_2 *associées à la matrice fondamentale entre la première et la seconde image, et telles que* $P_1 = [I \,|\, \mathbf{0}]$, *sont de la forme :*

$$P_1 = [I \,|\, \mathbf{0}] \;\; et \;\; P_2 = [[\mathbf{e}']_\times F \,|\, \mathbf{0}] + \mathbf{e}'\mathbf{v}^T. \tag{1.37}$$

A contrario il est possible de calculer la matrice fondamentale à partir de 2 caméras projectives comme vu en proposition 2.

D'après les propositions 3 et 2, l'ensemble des caméras projectives associées à la matrice fondamentale $\mathtt{F} = [\mathbf{m}]_\times \mathtt{M}$, sont de la forme $\mathtt{P}_1 = [\mathtt{I} \,|\, \mathbf{0}]\, \mathtt{H}$ et $\mathtt{P}_2 = [\mathtt{M} \,|\, \mathbf{m}]\, \mathtt{H}$.

Les homographies telles que $\mathtt{P}_1 = [\mathtt{I} \,|\, \mathbf{0}]$ sont de la forme :

$$\mathtt{H} = \left[\begin{array}{c|c} \mathtt{I} & \mathbf{0} \\ \hline \mathbf{v}_{1:3} & v_4 \end{array}\right] \tag{1.38}$$

Nous en déduisons la proposition suivante,

Proposition 5 *L'ensemble des caméras associées à une matrice fondamentale* $\mathtt{F} = [\mathbf{m}]_\times M$, *telles que* $P_1 = [I \,|\, \mathbf{0}]$ *sont de la forme*

$$P_1 = [I \,|\, \mathbf{0}] \quad et \quad P_2 = [M|\, \mathbf{0}] + \mathbf{m}\mathbf{v}^T \tag{1.39}$$

1.3.1.3 Calcul de la matrice fondamentale

Algorithme des 8 points Pour une correspondance $\left(\mathbf{p_j}, \mathbf{p'_j}\right)$ L'équation $\mathbf{p'_j}^T \mathtt{F} \mathbf{p_j} = 0$, peut être mise sous la forme d'une équation vectorielle ${\mathbf{A}_j}^T \mathbf{F} = 0$, où $\mathbf{F}$ est un vecteur de taille 9 correspondent aux coefficients de la matrices fondamentale en ligne. Ainsi avec 8 correspondances on a une équation matricielle de la forme $\mathtt{A}\mathbf{F} = \mathbf{0}$, ou $\mathtt{A}$ est une matrice 8×9. Dans le cas général, lorsque la matrice $\mathtt{A}$ n'est pas dégénérée, en effectuant une décomposition en valeurs singulières ou valeurs propres on obtient le vecteur propre solution de l'équation $\mathtt{A}\mathbf{F} = \mathbf{0}$. La décomposition en valeurs singulières est expliquée en Annexe B.1.

Avec $n > 8$ correspondances on obtient l'équation $\mathtt{A}\mathbf{F} = \mathbf{0}$, ou $\mathtt{A}$ est une matrice $n \times 9$. Dans le cas général, on effectue un décomposition en valeurs singulières SVD, et on prend comme meilleure solution approchante le vecteur singulier correspondant à la plus petite valeur singulière.

D'une manière générale pour $n \geq 8$ correspondances, la matrice $\mathtt{F}$ obtenue n'est pas singulière, on effectue donc un post-processing consistant à prendre la matrice fondamentale singulière la plus proche possible de $\mathtt{F}$. Pour cela, on effectue une décomposition en valeurs singulières de $\mathtt{F}$ sous la forme :

$$\mathtt{F} = \mathtt{U} \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{bmatrix} \mathtt{V}^T \tag{1.40}$$

s_1, s_2 et s_3 étant les valeurs singulières positives de la plus grande à la plus petite. La matrice fondamentale retenue $\mathtt{F}'$ est celle obtenue en remplaçant la plus petite valeur singulière par 0 :

$$\mathtt{F}' = \mathtt{U} \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathtt{V}^T \tag{1.41}$$

Algorithme des 7 points Il est également possible de calculer la matrice fondamentale à partir de 7 correspondances. Avec 7 correspondances on obtient comme précédemment l'équation $\mathtt{A}\mathbf{F} = \mathbf{0}$, ou $\mathtt{A}$ est une matrice 7×9. Dans le cas général, en effectuant une décomposition en valeurs singulières ou valeurs propres on obtient deux vecteurs propres $\mathbf{F}_1$ et $\mathbf{F}_2$ solutions de l'équation $\mathtt{A}\mathbf{F} = \mathbf{0}$. La matrice fondamentale étant définie à un scalaire près, elle peut s'écrire $\mathtt{F} = \alpha \mathtt{F}_1 + (1 - \alpha)\, \mathtt{F}_2$. Avec la contrainte que la matrice $\mathtt{F}$ a un déterminant nul on obtient une équation polynomiale de degré 3 en α. On obtient donc de 1 à 3 solutions possibles pour la matrice fondamentale.

Algorithm 2 Algorithme *Gold Standard*

Require: $\mathbf{S}$ un ensemble de correspondances de cardinal C, entre deux images 1 et 2

Initialise $SampleCount = 0$, $I_{max} = 0$, $\mathbf{S_i} = \emptyset$, N_{max}, p, t
while $N > SampleCount$ et $N_{max} > SampleCount$ **do**
  Choisis un échantillon aléatoire de 7 correspondances
  Avec les 7 points calcule la matrice fondamentale $\mathtt{F}$ (Algorithme des 7 points)
  Avec $\mathtt{F}$ calcule les caméras $\mathtt{P}_1 = [\mathtt{I} \mid \mathbf{0}]$ et $\mathtt{P}_2 = [[\mathbf{e}']_\times \mathtt{F} \mid \mathbf{e}']$ comme décrit dans la section 1.3.1.2
  Pour chaque correspondance de $\mathbf{S}$ $\mathbf{p_j} \leftrightarrow \mathbf{p'_j}$ reconstruit le point $3D$ $\widehat{\mathbf{X}_\mathbf{j}}$ par triangulation 1.2.4
  Calcule les reprojections au travers des caméras 1 et 2, $\widehat{\mathbf{p}}_\mathbf{j} = \mathtt{P}_1\widehat{\mathbf{X}_\mathbf{j}}$ et $\widehat{\mathbf{p}}'_\mathbf{j} = \mathtt{P}_2\widehat{\mathbf{X}_\mathbf{j}}$
  Calcule l'erreur géométrique $d = d\left(\mathbf{p_j}, \widehat{\mathbf{p}}_\mathbf{j}\right)^2 + d\left(\mathbf{p'}_\mathbf{j}, \widehat{\mathbf{p}}'_\mathbf{j}\right)^2$
  Détermine l'ensemble des inliers $\mathbf{S_{itemp}}$, tels que $d < t$
  Compte le nombre d'inliers I, $w = \frac{I}{C}$.
  if $I > I_{max}$ **then**
    Mets à jour $N = \frac{\log(1-p)}{\log(1-w^s)}$
    Mets à jour $I_{max} = I$
    Mets à jour $\mathbf{S_i} = \mathbf{S_{itemp}}$
  end if
  Incrémente $SampleCount = SampleCount + 1$
end while
Effectue un ajustement des faisceaux décrit en 1.2.5, sur l'ensemble des inliers $\mathbf{S_i}$. $\mathtt{P}_1$ reste fixe, on minimise suivant $\mathtt{P}_2$ et les $\widehat{\mathbf{X}_\mathbf{j}}$, $12 + 3I$ variables
return $\mathtt{P}_2 = [\mathtt{M} \mid \mathbf{t}]$ $\mathtt{F} = [\mathbf{t}]_\times \mathtt{M}$ voir proposition 2

On pourra donc utiliser l'algorithme de RANSAC expliqué en section 1.2.3, afin de calculer la matrice fondamentale à partir d'échantillons de 7 correspondances ou plus.

Algorithme *Gold Standard* Cet algorithme consiste, à partir d'un ensemble $\mathbf{S}$ de correspondances, à calculer une première estimation de la matrice par RANSAC, décrite dans la section 1.2.3, puis à minimiser l'erreur de reprojection suivant les points $3D$ reconstruits et suivant les paramètres des caméras par la méthode itérative de Levenberg-Marquardt. Dans la section 1.2.4 l'erreur de reprojection est définie et une méthode de reconstruction des points $3D$ par triangulation est proposée. La minimisation de l'erreur de reprojection suivant les caméras et les points reconstruits est brièvement décrite dans la section 1.2.5. La méthode *Gold Standard* est résumée dans l'algorithme 1.3.1.3.

1.3.2 Géométrie à trois images

1.3.2.1 Le tenseur trifocal

Tout d'abord nous nous intéressons à l'équation du plan dans l'espace $3D$. L'équation du plan est $aX + bY + cZ + d = 0$. Considérons en géométrie le point $3D$, $\mathbf{X} = (X, Y, Z, 1)^T$ et le plan $\mathbf{\Pi} = (a, b, c, d)$. Un point $\mathbf{X}$ appartient au plan $\mathbf{\Pi}$ si et seulement si $\mathbf{\Pi}^T\,\mathbf{X} = 0$.

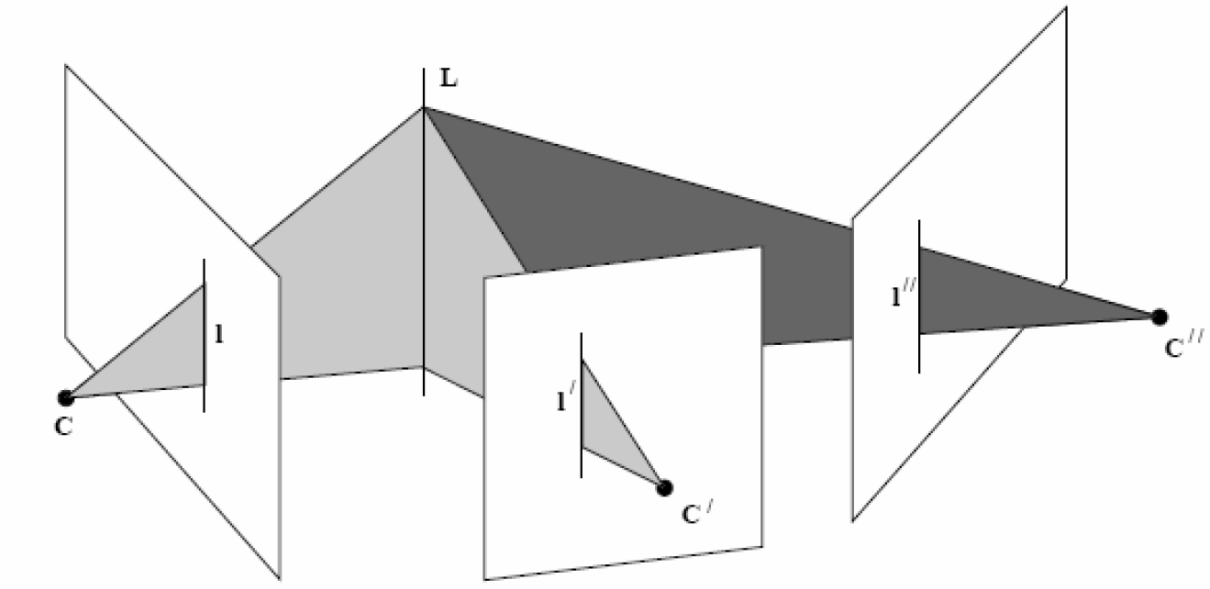


FIG. 1.5 – *Géométrie trifocale.*

L'équation d'une droite dans le plan est $eu+fv+g=0$. Considérons en géométrie projective le point $2D$ $\mathbf{p}=(u,v,1)$ et la droite $\mathbf{l}=(e,f,g)^T$. Un point $\mathbf{p}$ appartient à la droite $\mathbf{l}$ si et seulement si $\mathbf{l}^T\mathbf{p}=0$.

Maintenant, un point $\mathbf{X}$ appartient au plan pré-image de la droite $\mathbf{l}$ au travers de la caméra $\mathtt{P}$ si et seulement si il se reprojette sur la droite $\mathbf{l}$. Autrement dit, si et seulement si $\mathtt{P}X$ appartient à $\mathbf{l}$, soit $\mathbf{X}^T\mathtt{P}^T\mathbf{l}=0$. Le plan pré-image de la droite $\mathbf{l}$ est donc $\mathbf{\Pi}=\mathtt{P}^T\mathbf{l}$.

Deux plans s'intersectent toujours en une droite, en revanche comme on le voit en figure 1.5, si trois droites sont en correspondance, alors les 3 plans pré-image doivent s'intersecter en une droite dans l'espace $3D$, ce qui constitue un cas dégénéré. Si l'on considère trois caméras $\mathtt{P}_1$, $\mathtt{P}_2$ et $\mathtt{P}_3$, et trois droites en correspondance $\mathbf{l}$, $\mathbf{l}'$ et $\mathbf{l}''$, alors la matrice des plans pré-image à un rang égal à 2 :

$$\mathtt{M}=[\mathbf{\Pi}\,,\,\mathbf{\Pi}'\,,\,\mathbf{\Pi}'']=[\mathtt{P}_1^T\mathbf{l}\,,\,\mathtt{P}_2^T\mathbf{l}'\,,\,\mathtt{P}_3^T\mathbf{l}''] \tag{1.42}$$

Les 3 caméras étant définies à une transformation projective près on peut toujours poser $\mathtt{P}_1=[\mathtt{I}\,|\,\mathbf{0}]$, $\mathtt{P}_2=[\mathtt{A}\,|\,\mathbf{a}_4]$ et $\mathtt{P}_3=[\mathtt{B}\,|\,\mathbf{b}_4]$. On a donc :

$$\mathtt{M}=[\mathbf{m_1}\,,\,\mathbf{m_2}\,,\,\mathbf{m_3}]=\begin{bmatrix}\mathbf{l} & \mathtt{A}^T\mathbf{l}' & \mathtt{B}^T\mathbf{l}''\\ 0 & \mathbf{a}_4^T\mathbf{l}' & \mathbf{b}_4^T\mathbf{l}''\end{bmatrix} \tag{1.43}$$

Le fait que $\mathtt{M}$ ait un rang égal à 2, s'écrit $\mathbf{m}_1=\alpha\mathbf{m}_2+\beta\mathbf{m}_3$. dans la mesure où le coefficient en bas à gauche de $\mathtt{M}$ vaut 0, on a nécessairement, $\alpha=k\mathbf{l}''^T\mathbf{b}_4$ et $\beta=-k\mathbf{l}'^T\mathbf{a}_4$, pour un scalaire k. On a donc l'équation suivante à un scalaire près :

$$\mathbf{l}=(\mathbf{l}''^T\mathbf{b}_4)\mathtt{A}^T\mathbf{l}'-(\mathbf{l}'^T\mathbf{a}_4)\mathtt{B}^T\mathbf{l}'' \tag{1.44}$$

Si l'on note $\mathbf{a}_i$ la $i^{\text{ème}}$ colonne de $\mathtt{A}$ et $\mathbf{b}_i$ la $i^{\text{ème}}$ colonne de $\mathtt{B}$, la $i^{\text{ème}}$ coordonnée de la ligne $\mathbf{l}$, peut s'écrire :

$$l_i = \mathbf{l}'^T(\mathbf{a}_i\mathbf{b}_4^T)\mathbf{l}'' - \mathbf{l}'^T(\mathbf{a}_4\mathbf{b}_i^T)\mathbf{l}'' \tag{1.45}$$

Notons,

$$\mathtt{T}_i^{123} = \mathbf{a}_i\mathbf{b}_4^T - \mathbf{a}_4\mathbf{b}_i^T. \tag{1.46}$$

On a alors la relation droite-droite-droite suivante :

$$\mathbf{l}^T = \mathbf{l}'^T\left[\mathtt{T}_1^{123}, \mathtt{T}_2^{123}, \mathtt{T}_3^{123}\right]\mathbf{l}'' \tag{1.47}$$

L'ensemble des matrices tensorielles $\mathtt{T}_i^{123}$ constituent le tenseur trifocal entre les images 1, 2 et 3, il est noté $\mathcal{T}^{123}$.

Proposition 6 *Etant donné trois caméras il existe un tenseur trifocal noté $\mathcal{T}^{123}$ composé de trois matrices 3×3 T_i^{123} telles que l'on a la relation droite-droite-droite suivante :*

$$\mathbf{l}^T = \mathbf{l}'^T\left[T_1^{123}, T_2^{123}, T_3^{123}\right]\mathbf{l}''. \tag{1.48}$$

1.3.2.2 Epipoles associées au tenseur trifocal

Nous avons vu au paragraphe précédent que nous avons la relation droite-droite-droite suivante :

$$\mathbf{l}^T = \mathbf{l}'^T\left[\mathtt{T}_1^{123}, \mathtt{T}_2^{123}, \mathtt{T}_3^{123}\right]\mathbf{l}'' \tag{1.49}$$

Supposons que nous avons une relation point-droite-droite, étant donné deux droites $\mathbf{l}'$ et $\mathbf{l}''$, nous connaissons la droite $\mathbf{l}$ et nous savons également que le point $\mathbf{p}$ passe nécessairement par cette droite. soit $\mathbf{l}^T\mathbf{p} = 0$, ce qui s'écrit $\mathbf{l}'^T\left[\mathtt{T}_1^{123}, \mathtt{T}_2^{123}, \mathtt{T}_3^{123}\right]\mathbf{l}''\,\mathbf{p} = 0$. Soit :

$$\mathbf{l}'^T\left(\sum p_i\,\mathtt{T}_i^{123}\right)\mathbf{l}'' = 0 \tag{1.50}$$

Proposition 7 *Etant donné trois caméras et leur tenseur trifocal associé $\mathcal{T}^{123}$, nous avons la relation point-droite-droite suivante, entre un point $\mathbf{p}$ et deux droites $\mathbf{l}$ et $\mathbf{l}'$ correspondantes dans les images 2 et 3 :*

$$\mathbf{l}'^T\left(\sum p_i\,T_i^{123}\right)\mathbf{l}'' = 0. \tag{1.51}$$

Supposons maintenant que $\mathbf{l}'$ soit la droite épipolaire correspondant au point $\mathbf{p}$ dans la première image, *i.e.* $\mathbf{l}' = \mathtt{F}\mathbf{p}$ ($\mathtt{F}$ étant la matrice fondamentale entre la première et la seconde image). Ce cas de figure est montré dans la figure 1.4. On remarque que le rayon correspondant à la pré-image du point $\mathbf{p}$ est entièrement inclus dans le plan pré-image $\mathbf{\Pi}'$ de la droite $\mathbf{l}'$. Dans ce cas, quelle que soit la droite $\mathbf{l}''$, son plan pré-image $\mathbf{\Pi}''$ intersectera $\mathbf{\Pi}'$ et cette intersection sera une droite $\mathbf{L}$ incluse dans le plan $\mathbf{\Pi}'$. Cette droite $\mathbf{L}$ et le rayon pré-image du point $\mathbf{p}$ étant entièrement inclus dans le même plan, ils auront nécessairement une intersection non vide.

Autrement dit, si $\mathbf{l}'$ est la droite épipolaire dans la deuxième image correspondant au point $\mathbf{p}$ dans la première image, alors pour toute droite $\mathbf{l}''$ dans la troisième image, le rayon pré-image de $\mathbf{p}$, le plan pré-image de $\mathbf{l}'$ et le plan pré-image de $\mathbf{l}''$ s'intersecteront en un point. Nous avons donc une relation point-droite-droite.

si $\mathbf{l}'$ est la droite épipolaire dans la deuxième image correspondant au point $\mathbf{p}$ dans la première image, alors $\forall \mathbf{l}''$, $\mathbf{l}'^T \left(\sum p_i \, \mathtt{T}_i^{123}\right) \mathbf{l}'' = 0$. Dans ce cas on a donc simplement $\mathbf{l}'^T \left(\sum p_i \, \mathtt{T}_i^{123}\right) = \mathbf{0}^T$. Nous en déduisons la proposition suivante :

Proposition 8 *Si $\mathbf{p}$ est un point et $\mathbf{l}'$ et $\mathbf{l}''$ sont les droites épipolaires dans la deuxième et la troisième image, alors*

$$\mathbf{l}'^T \left(\sum p_i \, T_i^{123}\right) = \mathbf{0}^T \text{ et } \left(\sum p_i \, T_i^{123}\right) \mathbf{l}' = \mathbf{0}. \tag{1.52}$$

Preuve
La première relation a été montrée précédemment et la seconde est obtenue par le raisonnement transposé qui dit que si $\mathbf{l}''$ est la droite épipolaire dans la troisième image correspondant au point $\mathbf{p}$ dans la première image, i.e. $\mathbf{l}'' = F'\mathbf{p}$ *(F' étant la matrice fondamentale entre la première et la troisième image), alors $\forall \mathbf{l}'$, $\mathbf{l}'^T \left(\sum p_i \, T_i^{123}\right) \mathbf{l}'' = 0$.*

Remarquons que en prenant $\mathbf{p}_1 = (1, 0, 0)$, $\mathbf{p}_2 = (0, 1, 0)$ et $\mathbf{p}_3 = (0, 0, 1)$ nous obtenons trois droites épipolaires dans la second image, $\mathbf{l}'_1$, $\mathbf{l}'_2$, et $\mathbf{l}'_3$, qui sont respectivement le vecteur nul gauche de $\mathtt{T}_1^{123}$, de $\mathtt{T}_2^{123}$ et de $\mathtt{T}_3^{123}$, $\mathbf{l}'_1$. Dans la mesure où ces trois droites épipolaires passent par l'épipole on a $\mathbf{e}'^T \mathbf{l}'_1 = 0$, $\mathbf{e}'^T \mathbf{l}'_2 = 0$ et $\mathbf{e}'^T \mathbf{l}'_3 = 0$.

Proposition 9 *Si l'on note $\mathbf{u}_i$ et $\mathbf{v}_i$ les vecteurs nuls gauche et droit de T_i^{123}, c'est-à-dire $\mathbf{u}_i^T T_i^{123} = \mathbf{0}^T$, $T_i^{123} \mathbf{v}_i = \mathbf{0}$. Alors les épipoles $\mathbf{e}'$ et $\mathbf{e}''$ correspondant au centre de la première caméra dans la seconde et la troisième image sont les vecteurs nuls des matrices suivantes :*

$$\mathbf{e}'^T \left[\mathbf{u}_1 \, , \, \mathbf{u}_2 \, , \, \mathbf{u}_3\right] = \mathbf{0} \text{ et } \mathbf{e}''^T \left[\mathbf{v}_1 \, , \, \mathbf{v}_2 \, , \, \mathbf{v}_3\right] = \mathbf{0}. \tag{1.53}$$

Preuve
La première relation a été montrée précédemment et la seconde est obtenue par le raisonnement transposé.

1.3.2.3 Matrices fondamentales associées au tenseur trifocal

Si nous avons une correspondance point-point-droite entre $\mathbf{p}$, $\mathbf{p}'$ et $\mathbf{l}''$, alors pour toute droite $\mathbf{l}'$ passant par $\mathbf{p}'$ nous avons une correspondance point-droite-droite, et nous en déduisons d'après la proposition 7 que $\forall \mathbf{l}'$ passant par $\mathbf{p}'$ on a $\mathbf{l}'^T \left(\sum p_i \, \mathtt{T}_i^{123}\right) \mathbf{l}'' = 0$. Si l'on note

$\mathbf{u} = (\sum p_i \, \mathtt{T}_i^{123}) \, \mathbf{l}''$ nous avons donc $\forall \mathbf{l}'$ passant par $\mathbf{p}'$ $\mathbf{l}'^T \mathbf{u} = 0$. Autrement dit $\mathbf{p} = \mathbf{u}$ à un scalaire près. Nous en déduisons la proposition suivante :

Proposition 10 *Etant donné une correspondance point-point-droite, entre un point* $\mathbf{p}$*, une correspondance* $\mathbf{p}'$ *dans la seconde image et une droite* $\mathbf{l}''$ *dans la troisième image nous avons la relation suivante à un scalaire près :*

$$\mathbf{p}' = \left(\sum p_i \, T_i^{123} \right) \mathbf{l}''. \tag{1.54}$$

Par le raisonnement transposé nous obtenons également la proposition suivante :

Proposition 11 *Etant donné une correspondance point-droite-point, entre un point* $\mathbf{p}$*, une correspondance* $\mathbf{p}''$ *dans la troisième image et une droite* $\mathbf{l}'$ *dans la seconde image nous avons la relation suivante à un scalaire près :*

$$\mathbf{p}'' = \left(\sum p_i \, T_i^{123^T} \right) \mathbf{l}'. \tag{1.55}$$

D'après la proposition 10, nous avons la relation $\mathbf{p}' = ([\mathtt{T}_1^{123}, \mathtt{T}_2^{123}, \mathtt{T}_3^{123}]\mathbf{l}'') \, \mathbf{p}$. La droite épipolaire correspondant à $\mathbf{p}$ est donc $\mathbf{l}' = [\mathbf{e}']_\times ([\mathtt{T}_1^{123}, \mathtt{T}_2^{123}, \mathtt{T}_3^{123}]\mathbf{l}'') \, \mathbf{p}$. Ceci étant vrai pour toute droite $\mathbf{l}''$, il nous suffit de choisir $\mathbf{l}''$ en évitant le cas dégénéré ou $\mathbf{l}''$ est dans l'espace nul des tenseurs, un bon choix est donc $\mathbf{l}'' = \mathbf{e}''$ qui est orthogonal à cet espace nul. On en déduit que la droite épipolaire s'écrit $\mathbf{l}' = [\mathbf{e}']_\times ([\mathtt{T}_1^{123}, \mathtt{T}_2^{123}, \mathtt{T}_3^{123}]\mathbf{e}'') \, \mathbf{p}$. Or nous savons que, étant donné la matrice fondamentale $\mathtt{F}_{21}$ entre la première et la seconde image, cette droite épipolaire est également $\mathbf{l}' = \mathtt{F}_{21} \, \mathbf{p}$. Nous en déduisons que la matrice fondamentale entre la première et la seconde image est :

$$\mathtt{F}_{21} = [\mathbf{e}']_\times [\mathtt{T}_1^{123}, \mathtt{T}_2^{123}, \mathtt{T}_3^{123}]\mathbf{e}'' \tag{1.56}$$

Proposition 12 *Etant donné un tenseur trifocal entre trois images, les matrices fondamentales* F_{21} *et* F_{31} *entre la première et la seconde image, et entre la première et la troisième image sont :*

$$F_{21} = [\mathbf{e}']_\times [T_1^{123}, T_2^{123}, T_3^{123}]\mathbf{e}'' \;\; et \;\; F_{31} = [\mathbf{e}'']_\times [T_1^{123^T}, T_2^{123^T}, T_3^{123^T}]\mathbf{e}'. \tag{1.57}$$

Preuve
La première relation a été montrée précédemment et la seconde est obtenue par le raisonnement transposé.

1.3.2.4 Caméras associées au tenseur trifocal

D'après la proposition 12, la matrice fondamentale entre la première et la seconde image s'écrit :

$$\mathtt{F}_{21} = [\mathbf{e}']_\times [\mathtt{T}_1^{123}\ ,\ \mathtt{T}_2^{123}\ ,\ \mathtt{T}_3^{123}]\mathbf{e}'' \tag{1.58}$$

On déduit de la section 1.3.1.2 et plus précisément en proposition 2, que l'on peut fixer l'espace projectif tel que $\mathtt{P}_1 = [\mathtt{I}\,|\,\mathbf{0}]$ et $\mathtt{P}_2 = [[\mathtt{T}_1^{123}\ ,\ \mathtt{T}_2^{123}\ ,\ \mathtt{T}_3^{123}]\mathbf{e}''\,|\,\mathbf{e}']$.

La matrice fondamentale entre la première et la troisième image est

$$\mathtt{F}_{31} = [\mathbf{e}'']_\times [\mathtt{T}_1^{123^T}\ ,\ \mathtt{T}_2^{123^T}\ ,\ \mathtt{T}_3^{123^T}]\mathbf{e}' \tag{1.59}$$

Sachant que l'on a fixé la première caméra $\mathtt{P}_1 = [\mathtt{I}\,|\,\mathbf{0}]$ et d'après la proposition 5 la troisième caméra est nécessairement de la forme :

$$\mathtt{P}_3 = \left[[\mathtt{T}_1^{123^T}\ ,\ \mathtt{T}_2^{123^T}\ ,\ \mathtt{T}_3^{123^T}]\mathbf{e}'\,|\,\mathbf{0}\right] + \mathbf{e}''\,\mathbf{v}^T \tag{1.60}$$

en fixant $v_4 = 1$, ce qui revient à fixer le facteur d'échelle on peut écrire :

$$\mathtt{P}_3 = \left[[\mathtt{T}_1^{123^T}\ ,\ \mathtt{T}_2^{123^T}\ ,\ \mathtt{T}_3^{123^T}]\mathbf{e}'\,|\,\mathbf{e}''\right] + \mathbf{e}''\,(\mathbf{v}_{1:3}, 0) \tag{1.61}$$

Plus généralement on écrira $\mathtt{P}_3 = [\mathtt{B}\,|\,\mathbf{e}'']$. Comme écrit dans la section 1.3.2.1, les matrices tensorielles s'écrivent d'après l'équation 1.46, $\mathtt{T}_i^{123} = \mathbf{a}_i\mathbf{b}_4^T - \mathbf{a}_4\mathbf{b}_i^T$, c'est-à-dire ici :

$$\mathtt{T}_i^{123} = (\mathtt{T}_i^{123}\mathbf{e}'')\mathbf{e}''^T\ -\ \mathbf{e}'\mathbf{b}_i^T \tag{1.62}$$

En multipliant à gauche par $\mathbf{e}'^T$, en supposant que nous avons des épipoles de normes unitaires, et en prenant l'équation transposée on obtient nécessairement $\mathbf{b}_i = \left(\mathbf{e}''\,\mathbf{e}''^T\ -\mathtt{I}\right)\mathtt{T}_i^{123^T}\mathbf{e}'$. Ainsi nous avons donc l'unicité des caméras associées dans un espace projectif donné. Mais pas forcément l'existence.

Définition 4 *Un tenseur est dit "géométriquement valide" s'il est compatible avec* 3 *caméras* $P_1 = [I\,|\,\mathbf{0}]$, P_2 *et* P_3.

Autrement dit un tenseur est dit géométriquement valide, si il peut-être calculé avec 3 caméras $\mathtt{P}_1 = [\mathtt{I}\,|\,\mathbf{0}]$, $\mathtt{P}_2$ et $\mathtt{P}_3$ suivant l'équation 1.46.

Proposition 13 *Si un tenseur trifocal est géométriquement valide alors il existe un espace projectif tel que les caméras s'écrivent :*

$$\begin{aligned} P_1 &= [I\,|\,\mathbf{0}],\ P_2 = [\,[T_1^{123}\ ,\ T_2^{123}\ ,\ T_3^{123}]\ \mathbf{e}''\,|\,\mathbf{e}'] \\ P_3 &= \left[\left(\mathbf{e}''\,\mathbf{e}''^T\ -\ I\right)\left[T_1^{123^T}\ ,\ T_2^{123^T}\ ,\ T_3^{123^T}\right]\ \mathbf{e}'\,|\,\mathbf{e}''\right] \end{aligned} \tag{1.63}$$

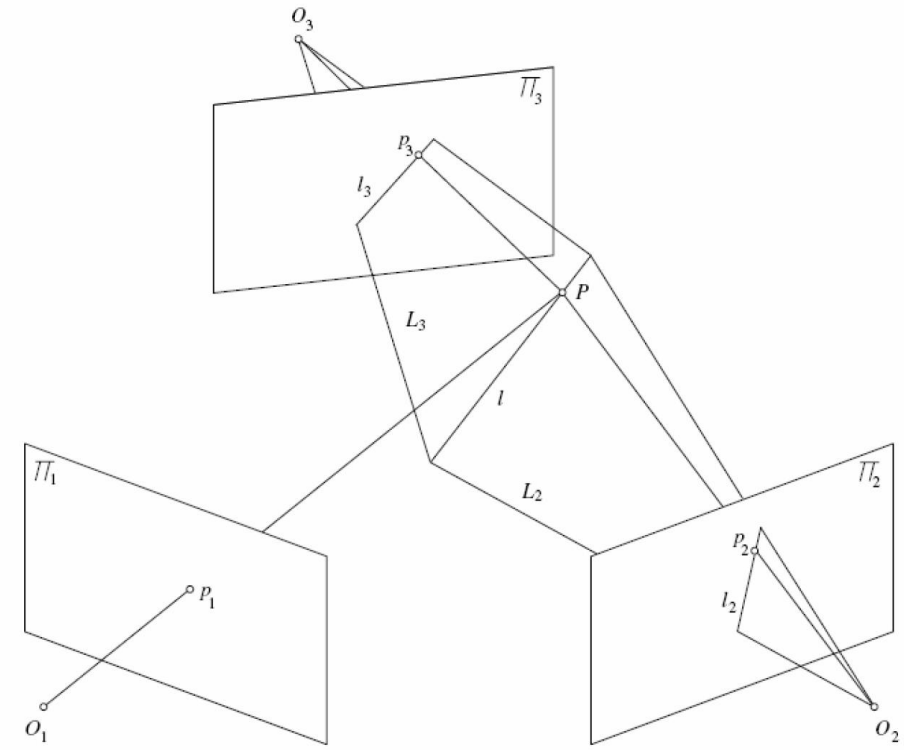


FIG. 1.6 – *Relation point-droite-droite*

1.3.2.5 Calcul du tenseur trifocal

Algorithme des 7 points Le tenseur trifocal comporte 27 entrées et est défini à un scalaire près. 26 équations sont donc suffisantes pour le calculer. Il est possible de le calculer à partir de correspondances entre les points. Supposons que l'on a les correspondances $\mathbf{p}_1$, $\mathbf{p}_2$ et $\mathbf{p}_3$ sur les trois images, alors dans ce cas en prenant des droites quelconques $\mathbf{l}_2$ et $\mathbf{l}_3$ passant par les points $\mathbf{p}_2$ et $\mathbf{p}_3$, elles correspondent nécessairement à une droite réelle. En effet la pré-image de $\mathbf{l}_2$ et de $\mathbf{l}_3$ forment 2 plans qui s'intersectent nécessairement en une droite. Après on sait que cette droite d'intersection se reprojette sur une droite $\mathbf{l}_1$ passant par le point $\mathbf{p}_1$. Ceci se voit sur la figure 1.6.

Pour ces droites nous avons une équation qui est ${\mathbf{l_2}}^T \left(\sum p_{1i}\, \mathtt{T}_i^{123}\right) \mathbf{l}_3 = 0$. Or il est possible d'obtenir deux droites indépendantes passant par $\mathbf{p}_2$ et également deux passant par $\mathbf{p}_3$:

$$\mathbf{p}_i = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \quad \mathbf{l}'_i = \begin{bmatrix} 1 \\ 0 \\ -u_i \end{bmatrix} \quad \mathbf{l}''_i = \begin{bmatrix} 0 \\ 1 \\ -v_i \end{bmatrix} \tag{1.64}$$

En utilisant les quatre couples possibles de droites, on a 4 équations pour chaque correspondance trifocale. Ainsi à partir de 7 correspondances trifocales on a 28 équations qui permettent d'évaluer le tenseur qui n'a que 27 coefficients. Il suffit de réarranger ces équations sous la forme $\mathtt{A}\mathbf{T} = 0$ ($\mathbf{T}$ étant le tenseur en ligne). La solution minimale est obtenue en effectuant une SVD comme expliqué en section B.1.

Une fois que l'on a trouvé les 27 entrées du tenseurs, on remarque que ce tenseur n'est a

priori pas géométriquement valide. Pour qu'il soit valide il faut qu'il puisse être calculé à partir de 3 caméras, et donc qu'il puisse s'écrire $\mathtt{T}_i^{123} = \mathbf{a}_i\mathbf{b}_4^T - \mathbf{a}_4\mathbf{b}_i^T$, d'après l'équation 1.46. Si l'on note que les caméras peuvent s'écrire en fonction des épipoles comme indiqué dans la proposition 13, alors le tenseur doit pouvoir s'écrire

$$\mathtt{T}_i^{123} = \mathbf{a}_i\mathbf{e}''^T - \mathbf{e}'\mathbf{b}_i^T. \tag{1.65}$$

Dans un premier temps nous calculons les épipoles suivant la méthode proposée dans la section 1.3.2.2. Et nous remarquons, d'après l'équation 1.65, que le tenseur en ligne doit pouvoir s'écrire $\mathbf{T} = \mathtt{E}\widehat{\mathbf{x}}$, pour être valide. $\mathtt{E}$ dépendant uniquement des épipoles. Finalement nous appliquons un post-processing pour calculer un tenseur valide solution du problème :

$$\begin{array}{c} \min \ \|\mathtt{A}\mathbf{T}\| \\ \text{sous contraintes } \|\mathbf{T}\| = 1 \text{ et } \mathbf{T} = \mathtt{E}\widehat{\mathbf{x}} \end{array} \tag{1.66}$$

Une résolution de ce problème sous contraintes est proposée dans la section B.1. Finalement nous récapitulons cet algorithme 1.3.2.5, avec une méthode itérative afin d'améliorer la qualité de la solution.

Algorithme des 6 points Il existe une méthode minimale pour calculer un triplet de caméras à partir de 6 points. Comme démontré par Quan [Qua95], il permet d'obtenir au plus 3 solutions valides. Cet algorithme est à la base du principe de dualité disant qu'il est possible d'intervertir le rôle des points et des centres des caméras. Ce principe de dualité est connu depuis les travaux de Carlsson [Car95] et Weinshall *et al.* [MWS95].

Algorithme *Gold Standard* pour calculer le tenseur trifocal Etant donné $n \geq 7$ correspondances, entre trois images 1, 2 et 3, on peut estimer le tenseur trifocal en choisissant itérativement 7 correspondances triples aléatoirement. Pour cela on utilise un RANSAC similaire à celui étudié dans la section sur la géométrie épipolaire 1.3.1.3. Le calcul du tenseur à partir de l'échantillon aléatoire de 7 points est décrit dans l'algorithme 1.3.2.5. Ensuite, d'après la proposition 13, on peut en déduire les caméras associées au tenseur. Une fois ces matrices de caméras connues, on peut définir le score de l'échantillon comme étant le nombre de correspondances triples dont l'erreur de reprojection est sous-pixellique (voir section 1.2.4 pour plus de détails sur la triangulation et la définition de l'erreur de reprojection). A la fin on retient les correspondances "inliers" dont l'erreur de reprojection était sous-pixellique pour le meilleur échantillon. Et on calcule à nouveau les matrices tensorielles à partir de tous les inliers, toujours suivant l'algorithme 1.3.1.3. Finalement on minimise l'erreur de reprojection suivant les points $3D$ reconstruits pour tous les inliers et suivant les paramètres des caméras par la méthode itérative de Levenberg-Marquardt comme décrit dans la section 1.2.5. Dans la section 1.2.4 l'erreur de reprojection est définie et une méthode de reconstruction des points $3D$ par triangulation est proposée.

Algorithm 3 Algorithme des 7 points pour calculer le tenseur trifocal

Require: $iter = 0$ et $\mathbf{S}$ un ensemble de $n \geq 7$ correspondances, entre trois images 1, 2 et 3

1: Pour chaque correspondance $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$, trouver 4 correspondances point-droite-droite indépendantes, voir équation 1.64.
2: Pour chaque correspondance $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$, en déduire les 4 équations point-droite-droite correspondantes ${\mathbf{l_2}}^T \left(\sum p_{1i}\, \mathtt{T}_i^{123}\right) \mathbf{l}_3 = 0$.
3: $\mathbf{T}$ étant le tenseur en ligne, réarranger les équations sous la forme $\mathtt{A}\mathbf{T} = \mathbf{0}$, $\mathtt{A}$ matrice $4n \times 27$.
4: Résoudre et trouver $\mathbf{T}$ en utilisant la méthode vue en Annexe B.1.2.
5: Trouver les épipoles $\mathbf{e}'$ et $\mathbf{e}''$, comme étant les perpendiculaires aux espaces nuls gauche et droit des 3 matrices tensorielles $\mathtt{T}_i^{123}$.
6: **while** $iter < Niter$ **do**
7: Trouver la matrice $\mathtt{E}$ dépendant de $\mathbf{e}'$ et $\mathbf{e}''$, tel que $\mathbf{T} = \mathtt{E}\mathbf{f}(\mathbf{a}, \mathbf{b})$, suivant les équations $\mathtt{T}_i^{123} = \mathbf{a}_i \mathbf{e}''^T - \mathbf{e}' \mathbf{b}_i^T$.
8: Trouver le tenseur $\mathbf{T}$ solution de min $\|\mathtt{A}\mathbf{T}\|$, sous contraintes $\|\mathbf{T}\| = 1$ et $\mathbf{T} = \mathtt{E}\widehat{\mathbf{x}}$, vu en B.1.4.
9: Paramétriser les épipoles pour avoir une norme unitaire, par exemple $\mathbf{e} = \frac{1}{\sqrt{{e_x}^2 + {e_y}^2}} (e_x, e_y, 1)^T$.
10: Chercher suivant les épipoles le zéro de la fonctionnelle $(\mathbf{e}, \mathbf{e}') \rightarrow \mathtt{AE}\mathbf{f}(\mathbf{a}, \mathbf{b})$ avec l'algorithme de Levenberg-Marquardt vu en Annexe B.2.3.
11: Incrémenter : $iter = iter + 1$.
12: **end while**
13: **return** Les 3 matrices tensorielles raffinées, $\mathtt{T}_i^{123}$.

Chapitre 2

Une nouvelle approche de “structure from motion” à partir d’un graphe de triplets

Contents

2.1 Etat de l’art sur la structure à partir du mouvement

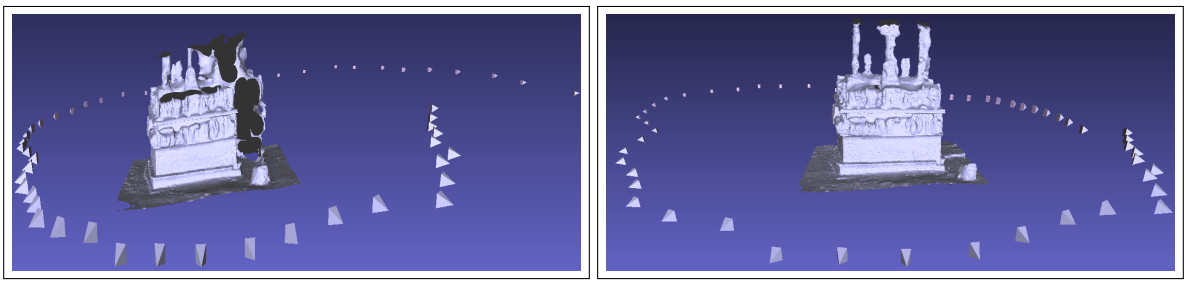


FIG. 2.1 – *Reconstruction stéréo multi-vues [VKLP09] en utilisant des caméras calibrées sans (gauche) et avec (droite) les contraintes de boucle. Lorsque les contraintes de boucles ne sont pas utilisées, l’accumulation d’erreurs mène à une reconstruction de très mauvaise qualité.*

La calibration de caméras à partir d’une scène tridimensionnelle a toujours été un problème central en vision par ordinateur. Le succès de livres tels que [HZ03, FLP01] atteste de cet intérêt. Au cours de ces dernières années, de nombreuses méthodes de calibration ont été proposées. La plupart de ces méthodes supposent que les paramètres internes sont connus ou partiellement connus [SSS06, MP07, SSS08, FP09, BKP09, ASS+09, HTKP09, MLD+09] ou traitent une séquence ordonnée de caméras [FZ98, AS01, PVGV+04, SPM04]. Cependant, dans bien des cas, les paramètres internes des caméras ne sont pas connus, ou le sont mais de manière très imprécise. L’absence d’ordre dans un ensemble de caméras est également très courant, lorsque l’on traite, par exemple, des images à partir du web.

Dans ce manuscrit, nous cherchons à calibrer un réseau de caméras non ordonnées, l’objectif principal étant d’obtenir une reconstruction projective de caméras suffisamment précise. Généralement, ceci est effectué par factorisation de la matrice des mesures [TK92, ST96], qui peut avoir des données manquantes [Jac97, MP05, TBT+07] en raison d’occlusions. Dans [TBT+07] la méthode de factorisation proposée permet de traiter des séquences avec 95% de données manquantes. Cette méthode se révèle très efficace comparativement aux autres méthodes similaires, et permet d’introduire facilement des contraintes sur les caméras. Ces contraintes peuvent être que les caméras sont identiques, varient de façon lisse le long de la séquence, ou encore que la configuration de certains points $3D$ est connue. La méthode proposée dans ce manuscrit est très différente et est basée sur la notion de graphe de tenseurs trifocaux plutôt que sur la factorisation. Les expériences faites sur des jeux de données réelles montrent que notre approche est très compétitive et qu’elle fournit une très bonne initialisation pour l’algorithme de “Bundle adjustment” (BA) [TMHF99], vu en section 1.2.5. L’utilisation d’une fusion de triplets pour obtenir une séquence de caméras a déjà été proposé par [FZ98] et [QL02]. Dans [FZ98] les triplets sont fusionnés en estimant les meilleures transformations projectives possibles minimisant une erreur géométrique. Cette erreur étant non linéaire la minimisation est susceptible de rester bloquée dans des minima locaux. D’autre par les tenseurs étant incohérents deux à deux, ce problème n’a pas de solution exacte. Une méthode pour forcer les contraintes de cyclicité, par un ajustement des faisceaux global sur toute la séquence, est également proposée dans ce papier. Dans [QL02] une méthode très similaire de fusion de triplets est également proposé.

Cette fois encore les contraintes de cyclicité sont imposées par un ajustement des faisceaux, mais de façon quasi-dense et non pas avec un ensemble parcimonieux de correspondances. On obtient grâce à cette mise en correspondances quasi-dense une bonne initialisation pour des algorithmes de stéréovision. Il est à noter que cet ajustement des faisceaux est également susceptible de rester bloqué dans des minima locaux et il n'y a aucune garantie d'atteindre les contraintes de cyclicité. Dans ce manuscrit nous proposons une fusion de triplets rendus parfaitement cohérents, c'est-à-dire que les transformations projectives entre deux tenseurs sont obtenues de façon exacte et sont connues sous forme annalytique. Cette fusion de triplets bien posée permet ultérieurement de formuler les contraintes de cyclicité comme étant la solution d'un problème polynomial. En abordant ces contraintes de cyclicité de façon différente nous remarquons que nous pouvons résoudre ces contraintes par la simple résolution de problèmes d'algèbre linéaire.

Même dans le cas de caméras calibrées, la plupart des méthodes citées plus haut sont basées sur un graphe de caméras (dans lequel les arêtes sont les géométries épipolaires 1.3.1) qui est rendu acyclique en retirant certaines arêtes. En revanche, certaines études récentes basées sur la modélisation de villes à partir de voitures ou de vues aériennes, mettent en avant l'intérêt d'utiliser les contraintes de boucles. Le fait de considérer les boucles dans un graphe permet de réduire la dérive des caméras due à l'accumulation d'erreurs lorsque l'on calcule les caméras de façon séquentielle (cf. Fig. 2.1). [KZIS08] fusionne des reconstructions partielles, [SFP10] contraint des rotations cohérentes avec les boucles et un mouvement plan. Adaptées à leurs données d'entrées spécifiques, ces méthodes reposent sur une régularisation de trajectoire ou une mise en correspondance dense [CCVG06, TPD08]. [THP09] est une exception notable, où les contraintes de boucles sont ajoutées à l'algorithme de calibration ("structure from motion") sparse, en tenant compte cependant, d'une séquence d'image ordonnées et de paramètres internes connus.

Les deux méthodes proposées dans ce manuscrit sont constituées des différentes étapes indiquées ci-dessous :

1. Notre point de départ est un ensemble de géométries épipolaires. Celles-ci sont calculées à l'aide d'une version du RANSAC 1.2.3 correspondant à l'état de l'art, le PROSAC [CM05], puis elles sont raffinées à l'aide d'une méthode basée sur le maximum de vraisemblance expliquée dans [PVGV+04]. Par l'intermédiaire du RANSAC, nous avons filtré les correspondances "outliers" et nous pouvons supposer que nous avons conservé des correspondances épipolaires, point-point, correctes. Enfin nous fusionnons ces correspondances épipolaires en correspondances à trois vues. En effet si nous avons une correspondance épipolaire entre i et j et une autre entre j et k, si les coordonnées sont identiques sur l'image j, alors nous les fusionnons en une correspondance point-point-point.

2. Nous regroupons les vues en triplets. Trois vues (i, j, k) sont considérées comme formant

un triplet valide si (a) les géométries épipolaires entre i et j mais également entre j et k ont été calculées avec succès lors de l’étape précédente, et (b) il y a au moins quatre correspondances à trois vues, point-point-point, dans ces images. Afin de réduire le nombre d’arêtes épipolaires dans le graphe des caméras, certaines des géométries épipolaires estimées, sont ignorées, de telles sorte que pour un triplet donné, seulement deux matrices fondamentales sont supposées connues et fixées. L’avantage de cette stratégie est que nous n’avons pas besoin de forcer la cohérence entre les matrices fondamentales. A première vue on pourrait considérer cela comme une perte d’information. Ce n’est pas le cas car cette information est retrouvée en calculant par la suite les tenseurs trifocaux.

3. Nous définissons un graphe ayant pour noeuds les triplets de caméras valides. Nous avons donc deux matrices fondamentales valides pour chaque noeud. Deux noeuds sont connectés par une arête si ils ont en commun une matrice fondamentale. Nous montrons que pour chaque noeud il existe un vecteur de taille 4 γ, tel que les entrées des caméras associées au triplet sont fonctions affines de ce vecteur. Nous proposons une méthode qui permet de calculer ces 4 inconnues à partir de quatre correspondances point-point-point par programmation linéaire. Aussi nous calculons donc ces 4 inconnues par RANSAC, en éliminant ainsi les correspondances à trois vues “outliers”.. Ainsi nous obtenons un ensemble de correspondances à trois vues que nous considérons comme particulièrement fiable. D’autre part, les homographies permettant de relier deux noeuds adjacents ν et ν' dans le graphe des triplets sont également fonctions affines de 4 des 8 paramètres inconnus correspondant aux deux vecteurs de taille 4 liés à ν et ν'.

4. Si le graphe des triplets est acyclique, l’estimation des γ faite à l’étape précédente, permet de retrouver directement l’ensemble des caméras avec existence et unicité. Dans le cas où, le graphe des triplets a un ou plusieurs cycles, chaque boucle mène à un ensemble de contraintes polynômiales par rapport aux inconnues γ. En partant des valeurs préalablement calculées pour les γ, nous linéarisons les contraintes de cyclicité par un procédé séquentiel. Pour linéariser les contraintes non linéaires, nous faisons simplement appel à la méthode de Gauss-Newton décrite en Annexe B.2.1. Ainsi nous résolvons de manière itérative des systèmes parcimonieux sous contraintes linéaires. En pratique, cette méthode converge rapidement, mais les contraintes de cyclicité ne sont que partiellement atteintes.

5. Dans le cas où les contraintes de cyclicité ne sont pas parfaitement atteintes, nous procédons à un répartition de l’erreur résiduelle de cyclicité par enregistrement d’homographies et estimation des matrices de caméras par programmation linéaire sous contraintes de norme. Ceci est effectué via “décomposition en valeurs singulières” expliquée en Annexe B.1. Nous obtenons ainsi l’ensemble des caméras dans un même espace projectif.

6. En remplacement des étapes 4 et 5, nous proposons également une seconde approche.

Cette méthode est basée sur une minimisation alternée, qui pour toutes nos expériences, sans exception, nous a permis de converger exactement vers les contraintes de cyclicité. Cette méthode utilise le fait que, si l'on réduit une boucle, à une boucle faite de 4 tenseurs trifocaux choisis alternativement, alors les contraintes de cyclicité peuvent s'écrire comme un problème linéaire. Ceci est expliqué dans la section 2.5. Cette méthode permet d'obtenir de très bons résultats et est intéressante pour appréhender les contraintes de cyclicité d'une manière différente, en les envisageant d'avantage comme étant des contraintes de cohérence.

7. Dans un premier temps nous raffinons ces caméras projectives en appliquant quelques "Bundle Adjustment" Projectifs successifs en retirant petit à petit les plus mauvais points. Nous retrouvons finalement l'espace métrique en utilisant une implémentation de [PMP+05], puis quelques "Bundle Adjustment" Euclidiens successifs en retirant petit à petit les plus mauvais points, afin de raffiner les caméras. Les "Bundle Adjustment" projectifs et euclidiens sont expliqués en section 1.2.5.

Ainsi, nous proposons deux méthodes de calibration permettant de retrouver la géométrie des caméras de façon précise, sans recourir à un procédé incrémental et tout en utilisant les informations de boucles afin d'éviter la dérive des caméras. Un des avantages de notre méthode est qu'elle n'utilise qu'un petit nombre de paramètres, dans la mesure où nous n'avons que quatre inconnues par triplet de caméras. Notre reconstruction est finalement raffinée à l'aide de quelques ajustements des faisceaux ("Bundle Adjustment"). Le fait de tenir compte des boucles et d'éviter les accumulations d'erreurs nous permet de diminuer nos chances de rester bloqués dans des minima locaux.

2.2 Objectif : un graphe de triplets cohérents

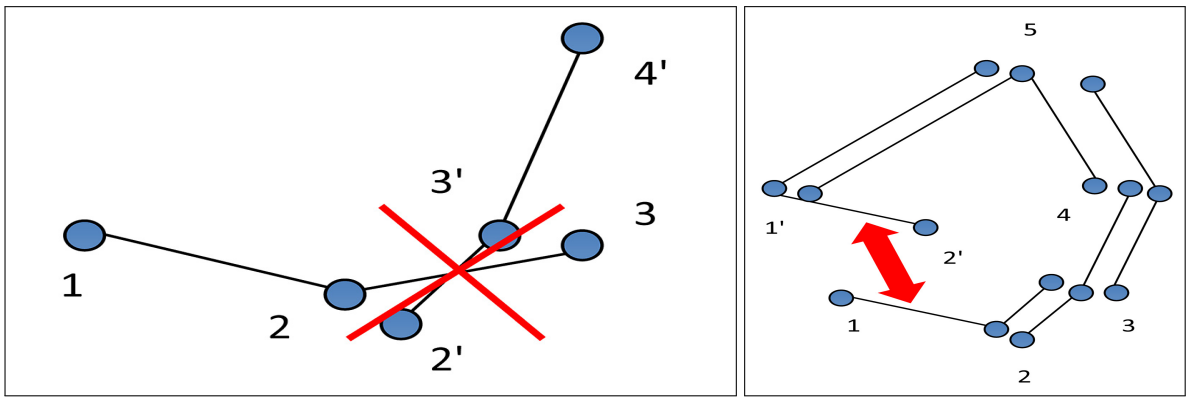


FIG. 2.2 – *Si l'on considère un graphe de triplets, 2 triplets ne sont pas cohérents entre eux (gauche), en revanche si on rend 2 triplets cohérents avec la même géométrie épipolaire alors les triplets sont cohérents deux à deux (droite). Comme on le voit sur la figure de droite, il faut en plus ajouter des contraintes de boucle.*

Pour calibrer des caméras, la méthodologie proposée consiste à utiliser un graphe de triplets

de caméras. En supposant que l’on a un ensemble de triplets cohérents (comme indiqué dans la partie droite de la figure 2.2) on peut fusionner l’ensemble des triplets de caméras dans un même espace.

Le principal problème est que ces triplets sont en pratique incohérents deux à deux comme on le voit sur la partie gauche de la figure 2.2. Ceci est dû au fait que l’on est toujours en présence de données bruitées et non idéales. Pour résoudre ce problème nous allons calculer dans un premier temps les géométries épipolaires (vu en section 1.3.1) pour les arêtes de notre graphe de triplets. Dans un second temps nous rendrons tous les triplets de caméras, c’est-à-dire les tenseurs trifocaux 1.3.2 qui sont les sommets du graphes, compatibles avec les mêmes géométries épipolaires. Ainsi deux tenseurs voisins dans le graphe seront compatibles avec la même géométrie épipolaire et donc seront cohérents entre eux.

D’autre part même si les tenseurs sont cohérents deux à deux il reste une contrainte de boucle, que l’on visualise en figure 2.2. Si l’on aborde le problème d’un point de vue simpliste, c’est comme si l’on construisait un ensemble avec des légos qui doivent être cohérents deux à deux, et à la fin le premier bout de la construction doit rejoindre l’autre bout de cette construction, ce qui constitue les contraintes de boucles.

Dans la section 2.3.1, nous verrons comment forcer un tenseur trifocal à être compatible avec une ou deux géométries épipolaires. Ensuite nous verrons que cette formulation permet de retrouver un ensemble de caméras de manière unique pour un arbre de triplets 2.3.2.1. Enfin nous aborderons la contrainte de cyclicité en section 2.3.2.2. Notre première approche de la contrainte de cyclicité mène à un problème polynomial complexe, nous proposerons donc une première résolution en section 2.4 faisant appel à une méthode itérative où on linéarise la contrainte de cyclicité suivant une approche de type Gauss-Newton B.2.1. Finalement nous verrons dans la section 2.5 que l’on peut transformer cette contrainte de cyclicité en un problème linéaire.

2.3 Concepts fondamentaux utilisés

2.3.1 Formulation réduite du tenseur trifocal

2.3.1.1 Réflexion sur les degrés de liberté

Une caméra projective est une matrice 3×4 définie à un facteur d’échelle près. Elle a donc 11 degrés de liberté. Maintenant, un ensemble de N caméras est défini à une transformation projective près comme on l’a vu dans la section 1.2.1.2. Une transformation projective, ou homographie, est une matrice 4×4 à un facteur d’échelle près qui a donc 15 degrés de liberté. Aussi un ensemble de N caméras a donc $11\,N\,-\,15$ degrés de liberté.

Ainsi, on retrouve le fait que la géométrie épipolaire, ou matrice fondamentale, a $11 \times 2\,-\,15\,=\,7$ degrés de liberté. Ce qui correspond bien à une matrice $\mathtt{F}$ 3×3, définie à un facteur d’échelle près et de déterminant nul.

De même le tenseur trifocal a $11 \times 3\,-\,15\,=\,18$ degrés de liberté. Aussi, étant donné 2 matrices fondamentales connues, soit $7 \times 2\,=\,14$ données, le tenseur trifocal compatible

avec ces géométries épipolaires n'a plus que $18 - 14 = 4$ degrés de liberté. Etant donné deux matrices fondamentales, nous allons maintenant expliciter ces 4 inconnues du tenseur trifocal de façon simple et linéaire.

2.3.1.2 Tenseur trifocal connaissant deux matrices fondamentales

Proposition 14 *Pour trois images i, j et k, étant donné deux matrices fondamentales $\mathtt{F}^{ij}$ et $\mathtt{F}^{ik}$, il existe un vecteur de taille 4 $\boldsymbol{\gamma} = [\gamma'_0, \ldots, \gamma_3]$ tel que le Tenseur Trifocal $\mathcal{T}^{ijk}$ s'écrit :*

$$T_t^{ijk} = A_t^{ij} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \gamma'_0 \\ 0 & 1 & \gamma_t \end{bmatrix} \left(A_t^{ik}\right)^\top \tag{2.1}$$

pout tout $t = 1, 2, 3$, où

$$A_t^{is} = \left[\left(F_{t,1:3}^{is}\right)^\top \, , \, \left(F_{t,1:3}^{is}\right)^\top \times \mathbf{e}^{is} \, , \, \mathbf{e}^{is} \right], \; s = j, k.$$

De plus, ce Tenseur Trifocal est géométriquement valide, c'est-à-dire, il existe 3 caméras $\mathtt{P}^i$, $\mathtt{P}^j$ et $\mathtt{P}^k$ compatibles avec les matrices fondamentales $\mathtt{F}^{ij}$ et $\mathtt{F}^{ik}$ et ayant $\mathcal{T}^{ijk}$ comme Tenseur Trifocal associé.

La preuve de ce résultat est en Annexe C. Il est notable que le résultat de la proposition 14 reste vrai dans tous les cas, même si les centres des trois caméras sont alignés. Il est important de présenter la forme des caméras paramétrisées par $\boldsymbol{\gamma}$ qui sont compatibles avec les matrices fondamentales $\mathtt{F}^{ij}$ et $\mathtt{F}^{ik}$ ainsi que le Tenseur Trifocal défini par l'équation 2.1. En fait, le triplet de caméras, qui est unique à une transformation projective près, est

$$\begin{aligned} \mathtt{P}^i &= [\mathtt{I}_{3\times 3} \,|\, \mathbf{0}_{3\times 1}], \qquad \mathtt{P}^k = [[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} \,|\, \mathbf{e}^{ik}], \\ \mathtt{P}^j &= \mathrm{kron}\left([\boldsymbol{\gamma}^0{}_{1:3}, 1]; \mathbf{e}^{ij}\right) - \gamma'^0{}_0 [[\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} \,|\, \mathbf{0}_{3\times 1}], \end{aligned} \tag{2.2}$$

où $\mathrm{kron}(\cdot, \cdot)$ représente le produit de Kronecker de deux matrices. Ces caméras sont obtenues directement à partir du tenseur trifocal, en utilisant les résultats présentés en section 1.3.2.4. Dans le cas de données idéales, la proposition 14 offre un moyen minimal de calculer les quatre inconnues restantes à partir de quatre correspondances point-point-point. A première vue on pourrait penser qu'une correspondance point-point-point, donnant lieu à 4 équations (voir section 1.3.2.5), est suffisante pour retrouver les 4 inconnues. Cependant, dans la mesure où deux géométries épipolaires sont connues, parmi les 4 équations, seule une apporte une information nouvelle non redondante avec les informations épipolaires connues. Ainsi nous avons besoin d'au moins 4 correspondances point-point-point pour calculer le tenseur trifocal compatible avec les deux matrices fondamentales données. Dans le cas de données bruitées, il est intéressant d'utiliser l'ensemble des 4 équations associées à une correspondance point-point-point. Le système est alors surdéterminé et on calcule la solution du système aux moindres carrés : $\min \|\mathtt{M}\boldsymbol{\gamma} - \mathbf{m}\|_2$, comme indiqué en Annexe B.1.

2.3.1.3 Transformation projective entre deux tenseurs

Le second ingrédient de notre approche est la paramétrisation de l’homographie reliant deux triplets de caméras ayant une matrice fondamentale en commun. Soient i, j, k et ℓ quatre images telles que (a) pour les vues i et k nous avons calculé avec succès la matrice fondamentale $\mathtt{F}^{ik}$ et (b) pour chacun des triplets (i, j, k) et (k, i, ℓ) l’estimation de deux matrices fondamentales est disponible. Alors, les triplets (i, j, k) et (k, i, ℓ) ont en commun la même matrice fondamentale $\mathtt{F}^{ik}$. En utilisant les équations (C.30), on obtient deux reconstructions projectives des caméras des images i et j basées sur deux vecteurs de taille 4, $\boldsymbol{\gamma}$ et $\underline{\boldsymbol{\gamma}}$. Notons les reconstructions des triplets (i, j, k) et respectivement (k, i, ℓ), $\mathtt{P}^i_{\boldsymbol{\gamma}}$ et $\mathtt{P}^k_{\boldsymbol{\gamma}}$ et respectivement $\mathtt{P}^i_{\underline{\boldsymbol{\gamma}}}$ et $\mathtt{P}^k_{\underline{\boldsymbol{\gamma}}}$. Si les centres des caméras i et k ne sont pas confondus, alors il existe une unique homographie $\mathtt{H}_{\boldsymbol{\gamma},\underline{\boldsymbol{\gamma}}}$ telle que

$$\mathtt{P}^i_{\boldsymbol{\gamma}}\underline{\mathtt{H}}_{\boldsymbol{\gamma},\underline{\boldsymbol{\gamma}}} \cong \mathtt{P}^i_{\underline{\boldsymbol{\gamma}}}, \quad \mathtt{P}^k_{\boldsymbol{\gamma}}\underline{\mathtt{H}}_{\boldsymbol{\gamma},\underline{\boldsymbol{\gamma}}} \cong \mathtt{P}^k_{\underline{\boldsymbol{\gamma}}}, \tag{2.3}$$

où $\cong$ représente l’égalité à un facteur multiplicatif près. En considérant les matrices des caméras et les inconnues $\boldsymbol{\gamma}$ connues, on peut résoudre (2.3) c’est-à-dire $\underline{\mathtt{H}}_{\boldsymbol{\gamma},\underline{\boldsymbol{\gamma}}}$ et son inverse $\mathtt{H}_{\boldsymbol{\gamma},\underline{\boldsymbol{\gamma}}}$. Le lecteur retrouve facilement que [1]

$$\underline{\mathtt{H}}_{\boldsymbol{\gamma},\underline{\boldsymbol{\gamma}}} = \left[\begin{array}{c|c} \mathrm{kron}(\underline{\boldsymbol{\gamma}}_{1:3}, \mathbf{e}^{ki}) - \underline{\gamma}'_0[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} & \mathbf{e}^{ki} \\ \hline -\frac{1}{2}\,\mathrm{tr}([\mathbf{e}^{ik}]_\times \mathtt{F}^{ki}[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik})(\mathbf{e}^{ik})^\top & 0 \end{array}\right], \tag{2.4}$$

et

$$\mathtt{H}_{\boldsymbol{\gamma},\underline{\boldsymbol{\gamma}}} = \begin{bmatrix} [\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} & \mathbf{e}^{ik} \\ \underline{\gamma}'_0 \lambda \mathbf{e}^{ki\top} - \underline{\boldsymbol{\gamma}}^T_{1:3}[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} & -\underline{\boldsymbol{\gamma}}_{1:3} \cdot \mathbf{e}^{ik} \end{bmatrix}. \tag{2.5}$$

Pour conclure cette section, le principal message à retenir de ces formules est que l’homographie $\mathtt{H}_{\boldsymbol{\gamma},\underline{\boldsymbol{\gamma}}}$, son inverse, ainsi que les caméras obtenues en (C.30) sont linéaires en $(\boldsymbol{\gamma}, \underline{\boldsymbol{\gamma}})$.

2.3.2 Graphe de triplets de caméras

Cette section contient l’essence même de notre contribution qui est basée sur la représentation sous forme de graphe des triplets de caméras. Cela rejoint les travaux de [SSS08], où le graphe des paires de caméras est utilisé. L’avantage qu’il y a à utiliser des triplets plutôt que des paires est qu’il n’est plus nécessaire de faire la distinction entre les chemins faisables et infaisables.

Le point de départ de notre algorithme est un ensemble de géométries épipolaires qui nous permet de définir un graphe $\mathcal{G}_{\text{cam}}$ tel que (a) $\mathcal{G}_{\text{cam}}$ a N noeuds correspondant aux N caméras et (b) deux noeuds de $\mathcal{G}_{\text{cam}}$ sont connectés par une arête si une estimation correcte de la géométrie épipolaire correspondante est disponible. Alors, un triplet de noeuds i, j, k de $\mathcal{G}_{\text{cam}}$ est dit valide si

- Il y a suffisamment de correspondances trifocales entre les vues i, j et k,
- Au moins deux parmi les trois paires de noeuds sont adjacentes dans $\mathcal{G}_{\text{cam}}$.

[1] Voir Annexe C pour le détail des calculs

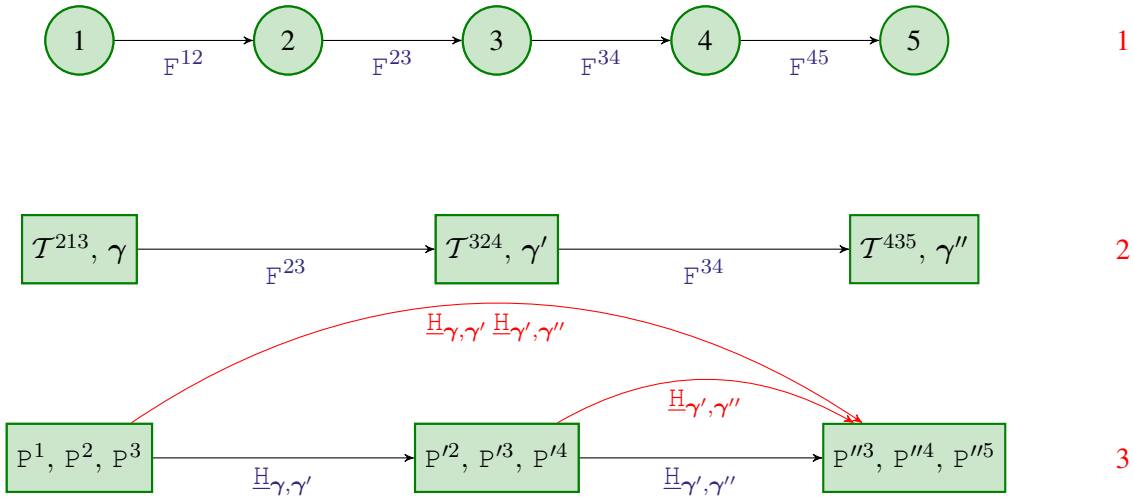


FIG. 2.3 – *En étape (1) nous construisons l'arbre des caméras et calculons les matrices fondamentales. En étape (2) nous construisons l'arbre des triplets de caméras et calculons les inconnus γ à l'aide d'un RANSAC et d'un algorithme de type "Gold Standard". En étape (3) nous retrouvons les caméras dans le même espace projectif, ici l'espace projectif de $\mathcal{T}^{435}$.*

Si pour un triplet valide les trois géométries épipolaires sont disponibles, nous supprimons la géométrie épipolaire la moins "digne de confiance" et définissons le graphe $\mathcal{G}_{\text{triplet}} = (\mathcal{V}_{\text{triplet}}, \mathcal{E}_{\text{triplet}})$ ayant pour noeuds les triplets de caméras valides. Dans ce graphe on applique une arête entre deux triplets, si ces triplets ont une matrice fondamentale en commun. D'après la proposition 14, la calibration globale du réseau de caméras est équivalente à l'estimation des vecteurs $\boldsymbol{\gamma}$ pour chaque triplet de caméras. Ainsi, à chaque noeud v du graphe de triplets nous associons un vecteur $\boldsymbol{\gamma}^v \in \mathbb{R}^4$. Le vecteur concaténé $\Gamma = (\boldsymbol{\gamma}^v : v \in \mathcal{V}_{\text{triplet}})$ est le paramètre auquel on s'intéresse dans le cadre de ce manuscrit.

2.3.2.1 *Existence et Unicité pour un arbre de triplets de caméras*

Si par chance il s'avère que le graphe de triplets est acyclique, alors le problème de l'estimation de Γ se réduit simplement à estimer $N_V = \text{Card}(\mathcal{V}_{\text{triplet}})$ vecteurs $\boldsymbol{\gamma}^v$ indépendants. Cette tâche est effectuée en utilisant des correspondances point-point-point ainsi que les résultats de la section 1.3.2.5. Plus précisément on utilisera un RANSAC à 4 points imbriqué dans un algorithme "Gold Standard" à partir des correspondances trifocales, dans l'esprit de la section 1.3.2.5. Cet algorithme faisant appel à un RANSAC, on pourra donc filtrer naturellement les correspondances trifocales pour ne conserver que les meilleures. Comme nous l'avons noté dans la section 2.3.1.2, quelques correspondances point-point-point suffisent pour obtenir une estimation de $\boldsymbol{\gamma}^v$.

Une fois que ces ensembles de Tenseurs Trifocaux et d'inconnues $\boldsymbol{\gamma}^v$ ont été calculés, il y a existence et unicité de l'ensemble des caméras associées. Comme on peut le voir en figure 2.3, on peut transporter les triplets de caméras dans le même espace projectif en composant les homographies calculées en équations 2.4. Si l'on considère les degrés de liberté de N caméras, ils correspondent à N matrices 3×4 définies à un facteur multiplicatif près, et à une transfor-

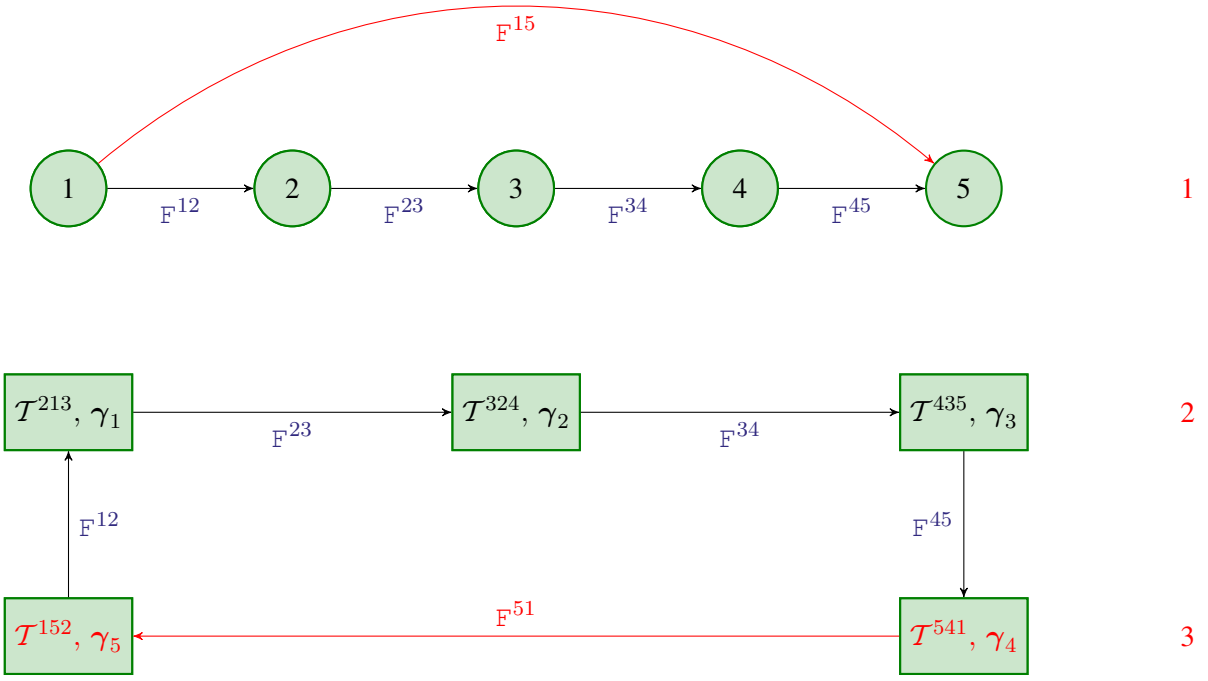


FIG. 2.4 – *En étape (1) on construit le cycle des caméras, et nous avons une matrice fondamentale additionnelle par rapport au cas d’un arbre vu en figure 2.3, c’est-à-dire* 7 *contraintes. En étape (2) nous construisons le cycle des triplets de caméras et il y a deux tenseurs réduits additionnels, c’est-à-dire* 8 *contraintes. Enfin, nous obtenons donc* 15 *contraintes qui peuvent s’écrire* $\prod_{i=1}^{N} \underline{H}_{\gamma_i,\gamma_{i+1}} \cong I$.

mation projective près. Une transformation projective ou homographie est une matrice 4×4 définie à un facteur multiplicatif près. N caméras ont donc $11N - 15$ degrés de liberté. Notre paramétrisation d’un arbre de N caméras est composée de $N-1$ matrices fondamentales, et de $N-2$ tenseurs ayant 4 inconnues $\boldsymbol{\gamma}$. Autrement dit nous avons $7(N-1) + 4(N-2)$ paramètres, c’est-à-dire $11N - 15$ paramètres. Ainsi le fait que notre paramétrisation est minimale est cohérent avec le fait qu’il y a existence et unicité d’un ensemble de caméras à une homographie près.

2.3.2.2 Contrainte de cyclicité pour un cycle de triplets de caméras

Cependant, les graphes acycliques sont une exception plutôt qu’une règle. Même si le graphe des caméras est acyclique, le graphe des triplets résultant peut contenir des boucles. S’il y a une boucle, nous avons alors 15 contraintes de cyclicité, c’est-à-dire une matrice fondamentale additionnelle (7 contraintes) et deux tenseurs réduits additionnels ($4 \times 2 = 8$ contraintes), de 4 inconnues chacun, comme montré en figure 2.4. Toujours sur la figure 2.4, nous pouvons écrire la contrainte comme : si nous transportons le triplet de caméras du tenseur trifocal $\mathcal{T}^{213}$ dans l’espace projectif du même tenseur trifocal $\mathcal{T}^{213}$ en parcourant la boucle complète, nous devons retrouver le même triplet de caméras à un facteur multiplicatif près. Ceci peut s’écrire,

$$\prod_{i=1}^{N} \underline{\mathtt{H}}_{\gamma_i,\gamma_{i+1}} \cong \mathtt{I}. \tag{2.6}$$

L'équation (2.6) définit un ensemble de 15 contraintes polynomiales par rapport au vecteur Γ des inconnues. Si le graphe des triplets contient N_{loop} boucles, alors nous obtenons $15N_{\text{loop}}$ contraintes. Pour donner davantage de détails, notons que chaque contrainte de boucle (2.6) peut s'écrire $f_j(\Gamma) = 0$, $j = 1, \ldots, 15$, pour une fonction polynomiale f_j. En regroupant ces contraintes pour les N_{loop} boucle, nous obtenons

$$f_j(\Gamma) = 0, \quad j = 1, \ldots, 15N_{\text{loop}}. \tag{2.7}$$

D'autre part d'après la formulation du tenseur trifocal (2.1), et d'après les résultats vus en section 1.3.2.5, les correspondances point-point-point peuvent s'écrire comme un système d'équations linéaires et non homogènes en Γ

$$\mathtt{M}\Gamma = \mathbf{m}, \tag{2.8}$$

où $\mathtt{M}$ est une matrice $4N_{\text{3-corr}} \times 4N$ et $\mathbf{m}$ est un vecteur de taille $4N_{\text{3-corr}}$, $N_{\text{3-corr}}$ étant le nombre de correspondances à trois vues. La matrice $\mathtt{M}$ et le vecteur $\mathbf{m}$ sont calculés en utilisant les matrices fondamentales connues. Dans la mesure où ces matrices sont estimées à partir de correspondances bruitées, le système (2.8) ne doit pas nécessairement être parfaitement vérifié. Il est donc naturel d'estimer le vecteur Γ des inconnues en résolvant le problème

$$\begin{aligned} &\min \|\mathtt{M}\Gamma - \mathbf{m}\|_q^q \\ \text{subject to} \quad &f_j(\Gamma) = 0, \ \forall j = 1, \ldots, 15N_{\text{loop}}, \end{aligned} \tag{2.9}$$

pour $q \geq 1$. Malheureusement avec cette formulation du problème de cyclicité, il n'existe pas de q tel que ce problème soit convexe, il est donc très complexe à résoudre. Pour gérer cette difficulté, nous proposons deux stratégies, la première basée sur une linéarisation locale et la seconde faisant appel à une minimisation alternée.

L'avantage de cette approche est que si une solution du problème d'optimisation est trouvée, alors elle est forcément cohérente avec les boucles. Ceci signifie que l'ensemble des caméras sera obtenu de manière unique à un facteur multiplicatif près et à une transformation projective près, ce qui correspond à l'ambiguïté projective vue en section 1.2.1.2.

2.4 Une première solution : programmation linéaire séquentielle

2.4.1 Linéarisation de la contrainte de cyclicité

Plutôt que de résoudre le problème d'optimisation obtenu en combinant les contraintes de boucle polynomiales, nous proposons de le remplacer par un problème linéaire. Pour linéariser les contraintes nous nous basons sur une méthode de type Gauss-Newton vue en Annexe B.2.1.

Nous commençons avec une estimation initiale de Γ, *e.g.*, en résolvant le problème sans contraintes (convexe) pour un $q \geq 1$. Dans notre implémentation nous calculons séparément les 4 inconnues de chaque triplet avec une méthode basée sur un RANSAC pour garantir la robustesse par rapport à d'éventuelles fausses correspondances trifocales. Nous utilisons aussi $q = 2$ dans notre programme. Alors, étant donné une première estimation Γ_0, nous définissons

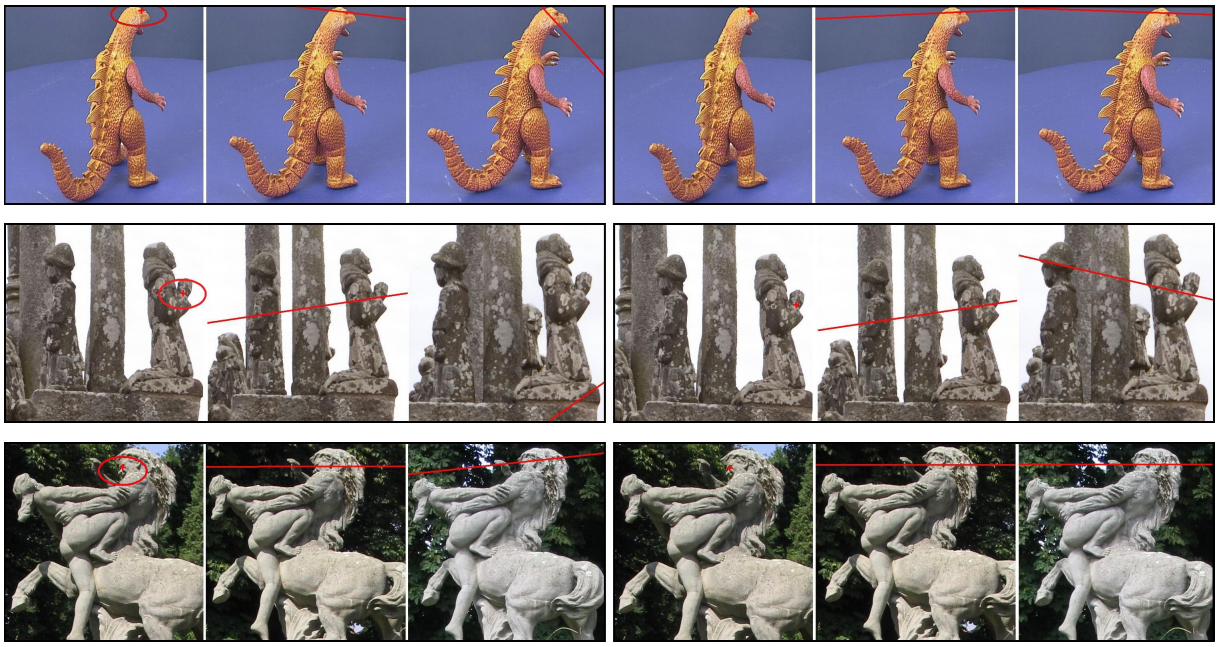


FIG. 2.5 – *Pour chaque jeu de donnée acyclique composé de N images numérotées de 0 à $N-1$, nous montrons la géométrie épipolaire obtenue. Les droites épipolaires, pour les images $N-1$ et 0 correspondent à un point cliqué dans l’image $N-2$, sont dessinées. Pour chaque ligne, les trois images sur la gauche correspondent à une calibration sans contrainte de cyclicité, et les trois images sur la droite à une calibration avec des contraintes de boucle. Naturellement, les droites épipolaires dans l’image 0 sont bien plus précises avec l’optimisation sous contraintes.*

la séquence Γ_k par la relation récursive suivante : Γ_{k+1} est la solution du problème linéaire

$$\begin{aligned} &\min \|\mathrm{M}\Gamma - \mathbf{m}\|_1 \\ \text{subject to} \quad &|f_j(\Gamma_k) + \nabla f_j(\Gamma_k)(\Gamma - \Gamma_k)| \leq \epsilon, \end{aligned} \tag{2.10}$$

où ϵ est un petit paramètre (nous utilisons $\epsilon = 10^{-6}$). Il existe de nombreux logiciels—tels que GLPK, SeDuMi, SDP3—pour résoudre le problème (2.10) avec des temps de réponse très attractifs même pour des milliers de variables et de contraintes. De plus, les expériences empiriques nous ont montré que la séquence Γ_k converge très rapidement. Ainsi nous obtenons une solution satisfaisante après une dizaine d’itérations.

Le jeu de données *Dinosaur* est composé de 36 images. Le jeu de données *Detenice Fountain* est composé de 34 images. Le jeu de données *Calvary* est composé de 52 images.

Comme on peut le voir sur la figure 2.5, pour les trois jeux de données, sans optimisation sous contraintes, les droites épipolaires dans la dernière image, correspondant à un point cliqué dans l’avant dernière image, sont très précises. ceci est naturel puisque notre méthode calcule les caméras en adéquation parfaite avec les matrices fondamentales calculées précédemment. En revanche on note que les droites épipolaires dans la première image, correspondant à un point cliqué dans l’avant dernière image, sont peu précises lorsque les contraintes de cyclicité ne sont pas utilisées. Ceci est particulièrement vrai pour le *Dinosaur*, et même encore plus pour le *Calvary*. Lorsque l’on ajoute l’optimisation sous contraintes, on voit sur la partie droite de la

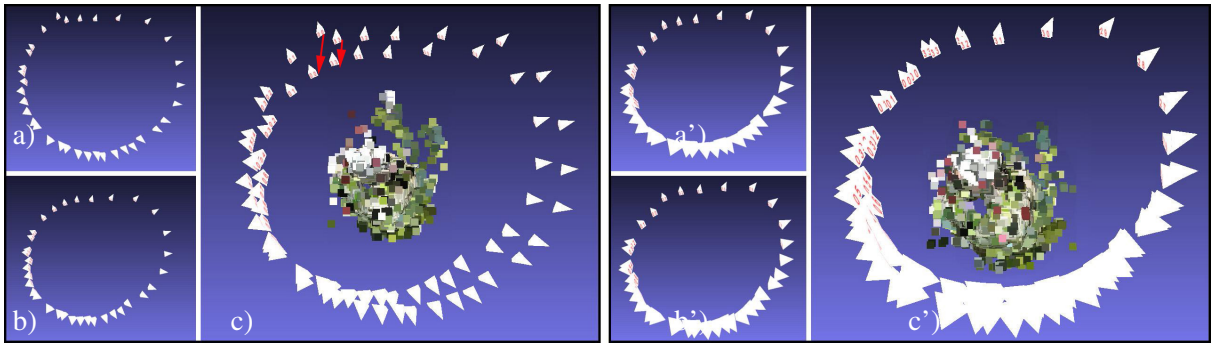


FIG. 2.6 – *Pour le Detenice Fountain, nous comparons notre méthode sans les contraintes de boucle sur la gauche, et avec une optimisation sous contraintes sur la droite. a) et a') montrent les caméras résultantes numérotées avant d'appliquer un BA ("bundle adjustment"). b) et b') montrent les caméras obtenues après BA. c) et c') montrent les caméras avant et après BA afin de comparer le mouvement des caméras pendant le BA. c) et c') montrent les nuages de points $3D$ au centre. Comme nous pouvons le voir, sans les contraintes de cyclicité il y a un grand mouvement pendant le BA, alors qu'avec l'optimisation sous contraintes le mouvement effectué au cours du BA est petit, ceci montre le potentiel de notre approche comme alternative au BA.*

figure 2.5, que les droites épipolaires fermant la boucle deviennent bien plus précises.

Puisqu'il n'y a pas de vérité terrain disponible pour les jeux de données *Detenice Fountain* et *Calvary*, nous utilisons le résultat du BA pour évaluer l'impact des contraintes de cyclicité sur la qualité d'estimation des caméras. Sur la figure 2.6, nous remarquons que les positions des caméras estimées sans imposer les contraintes de cyclicité sont très différentes de celles obtenues après BA. Plus particulièrement, la caméra 33 est devant la caméra 00, alors que suivant l'estimation fournie par le BA elle est légèrement derrière la caméra 00. En revanche, si l'optimisation sous contraintes est utilisée, les positions relatives des caméras 33 et 00 sont presque les mêmes avant et après BA.

La figure 2.7, montre un example de séquence longue dans laquelle l'optimisation sans contrainte produit une forte dérive (la caméra 00 est loin derrière la caméra 51). Avec l'optimisation sous contraintes, la caméra 51 est proche de la caméra 00. Comme on le voit sur les imagettes en haut, la première et la dernière photo sont capturées depuis le même point de vue ou presque. Cette configuration est confirmée par le BA. Afin d'avoir une mesure quantitative de l'amélioration obtenue en forçant les contraintes de cyclicité, nous effectuons un BA avec des estimateurs sans contraintes et avec des estimateurs contraints pour initialiser le BA. La l'erreur de reprojection pixellique "root mean square" sur l'ensemble de la séquence est 0.87 après le BA initialisé avec des estimateurs contraints, alors qu'elle est de 1.61 après le BA initialisé avec des estimateurs non contraints.

2.4.2 Prendre en compte l'hétéroscédasticité

L'énergie que nous minimisons dans (2.10), est purement algébrique, notre objectif est de lui donner une signification d'un point de vue statistique. En supposant que les équations (2.8)

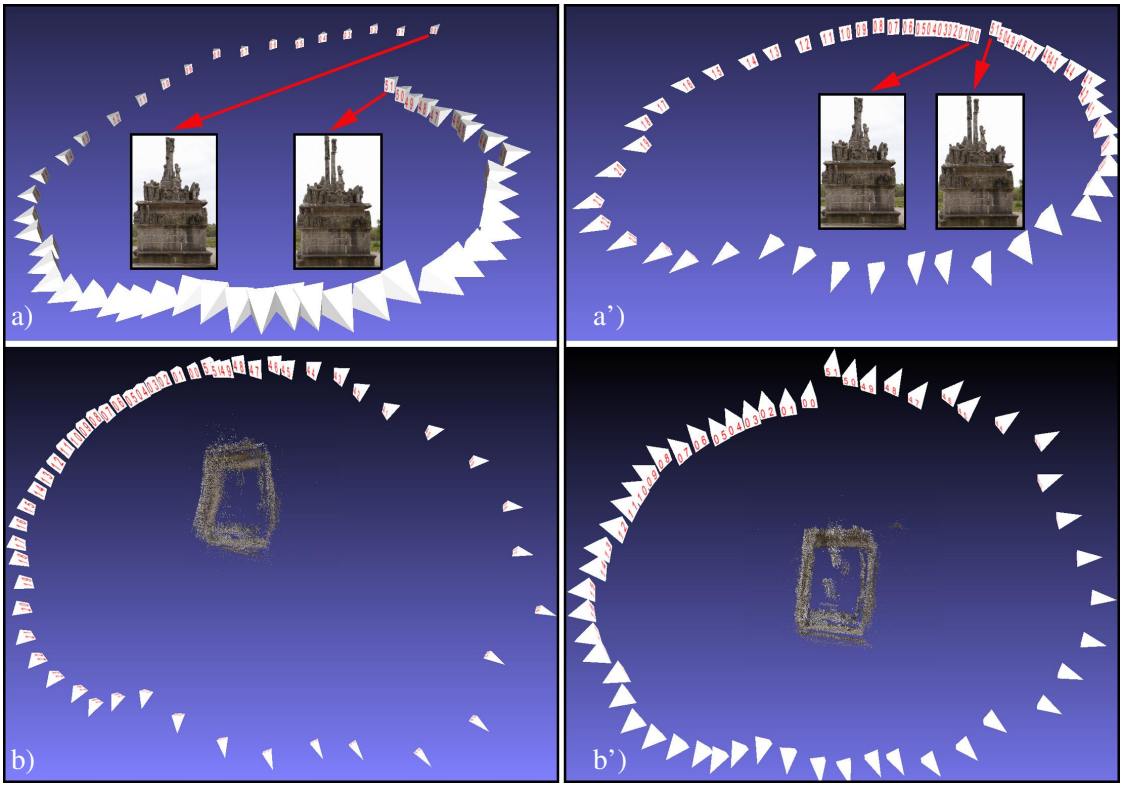


FIG. 2.7 – *Jeu de données Calvary : dans a) et a’) nous montrons les caméras obtenues avant BA sans contrainte de cyclicité sur la gauche, et avec optimisation sous contraintes sur la droite. Dans b) et b’) nous montrons les caméras après BA sans contrainte de boucle (gauche), et avec optimisation sous contraintes (droite). Comme on le voit sur les imagettes en haut, la première et la dernière photo sont capturées depuis le même point de vue ou presque. Cette configuration n’est pas retrouvée sans les contraintes de boucle, en revanche elle est correctement retrouvée avec l’optimisation sous contraintes. Après BA dans b’) la configuration circulaire est bien récupérée, alors que dans b), sans les contraintes de cyclicité, la configuration des caméras n’est pas circulaire mais s’étire sur la droite.*

sont satisfaites à un bruit aléatoire près : $\mathtt{M}\Gamma = \mathbf{m} + \boldsymbol{\xi}$, où le vecteur aléatoire $\boldsymbol{\xi}$ a des coordonnées indépendantes autour d'une distribution centrée de Laplace avec une échelle constante. Alors l'énergie dans (2.10) est proportionnelle à l'opposé de la vraisemblance logarithmique. Le caractère constant du facteur d'échelle signifie que les erreurs sont homoscédastiques ce qui constitue un hypothèse forte. Nous remarquons que toutes les correspondances trifocales, point-point-point, liées à un même triplet ont à peu près le même facteur d'échelle, tandis que les échelles varient très fortement d'un triplet à l'autre (voir Fig. 2.8). Afin de prendre en compte cette hétéroscédasticité du bruit, nous utilisons l'estimation initiale de Γ pour estimer un paramètre d'échelle σ_v par noeud $v \in \mathcal{V}_{\text{triplet}}$. Ceci est effectué en calculant la déviation standard des erreurs résiduelles calculées. En utilisant $\{\sigma_v\}$, l'énergie du problème (2.10) est remplacée par $\sum_v \|\mathtt{M}_v\Gamma - \mathbf{m}_v\|_1/\sigma_v$. Ici, $\mathtt{M}_v$ est la partie matricielle de $\mathtt{M}$ contenant les lignes obtenues à partir de correspondances à trois vues pour le triplet v. Le vecteur $\mathbf{m}_v$ est obtenu de façon similaire à partir de $\mathbf{m}$.

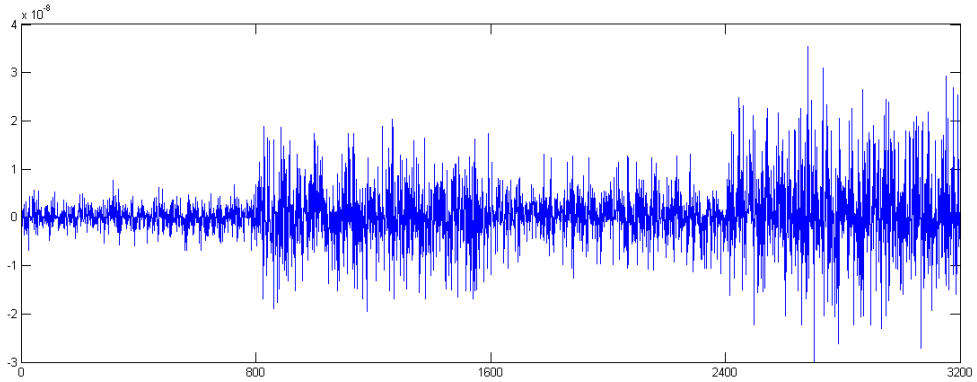


FIG. 2.8 – *Cette figure illustre l'hétéroscédasticité du bruit dans le système (2.8) lorsque la matrice* $\mathtt{M}$ *et le vecteur* $\mathbf{m}$ *sont calculés en utilisant les géométries épipolaires préalablement calculées. Les erreurs résiduelles sont ici affichées pour 4 noeuds du graphe de triplet, avec 800 mesures disponibles pour chaque noeud.*

2.4.3 Répartition de l'erreur résiduelle de cyclicité par homographies

Etant donné un graphe de Tenseurs Trifocaux, $\mathcal{G}_{\text{triplet}}$, chaque noeud de ce graphe sera noté $v_1, v_2, \ldots, v_n$. A l'étape précédente nous avons déterminé les paramètres $\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_n$, tels que $\boldsymbol{\gamma}_i$ caractérise le tenseur trifocal représenté dans le graphe par v_i. Une stratégie naïve pour estimer les matrices de caméras est de fixer les 3 caméras d'un triplet v et de retrouver les autres caméras dans l'espace projectif du tenseur v en appliquant successivement des homographies $\mathtt{H}_{\gamma,\gamma'}$ aux matrices des caméras reconstruites à partir des équations (C.30). Cependant, dans la plupart des cas, le vecteur Γ calculé par programmation linéaire séquentielle, comme décrit dans la section précédente, ne satisfait les contraintes de cyclicité qu'à une erreur résiduelle près. Ainsi, la méthode naïve proposée précédemment a le désavantage d'accumuler les erreurs et de dévier légèrement pour un grand nombre d'homographies appliquées. Afin d'éviter cela et de répartir

l’erreur résiduelle de cyclicité uniformément parmi les caméras, nous proposons une méthode basée sur l’enregistrement d’homographies par SVD. Le point d’entrée pour cette méthode est un vecteur Γ pour lequel les contraintes de boucle sont satisfaites à une petite erreur près.

2.4.3.1 Le cas d’une boucle unique

Nous supposons dans un premier temps que $\mathcal{G}_{\text{triplet}}$ se réduit à une boucle, c’est-à-dire chaque noeud v_i a exactement deux voisins v_{i-1} et v_{i+1} avec la convention standard et $v_{n+i} = v_i$ pour tout i. Pour chaque noeud v_i représentant trois images, nous avons déjà calculé une version des matrices de caméras $\mathtt{P}^{1,\gamma_i}, \mathtt{P}^{2,\gamma_i}, \mathtt{P}^{3,\gamma_i}$. De plus, pour deux noeuds voisins v_i et v_{i+1} nous avons calculé une homographie $\mathtt{H}^{i,i+1}$ telle que

$$\mathtt{P}^{j,\gamma_{i+1}} \cong \mathtt{P}^{j+1,\gamma_i}\mathtt{H}^{i,i+1}, \quad j \in \{1,2\}. \tag{2.11}$$

En se basant sur ces homographies relatives $\{\mathtt{H}^{i,i+1}\}$ nous souhaitons retrouver les homographies absolues $\mathtt{H}^{v_i}$ qui permettent de représenter toutes les matrices de caméras $\mathtt{P}^{j,\gamma_i}$ dans un espace projectif commun. En d’autres termes, dans le cas idéal où il n’y a pas d’erreur résiduelle de cyclicité, les matrices $\mathtt{H}^{v_i}$ devraient satisfaire

$$\mathtt{P}^{j,\gamma_i}\,\mathtt{H}^{v_i} \cong \mathtt{P}^{j+i-1,*}, \qquad j \in \{1,2,3\}. \tag{2.12}$$

Évidemment, l’ensemble $\{\mathtt{H}^{v_i}\}$ ne peut être déterminé qu’à une ambiguïté projective près.

Proposition 15 *Si pour $i = 1, \ldots, n$, les caméras $\mathtt{P}^{i+1,*}$ et $\mathtt{P}^{i+2,*}$ ont des centres distincts, alors $\mathtt{H}^{v_i} \cong \mathtt{H}^{i,i+1}\mathtt{H}^{v_{i+1}}$. De plus, si les centres de chaque paire de caméras consécutives sont distincts, alors on peut trouver un espace projectif tel que*

i) $\mathtt{H}^{v_i} = \mathtt{H}^{i,i+1}\mathtt{H}^{v_{i+1}}, \qquad \forall i = 1, \ldots, n-1,$

ii) $\alpha\,\mathtt{H}^{v_n} = \mathtt{H}^{n,1}\mathtt{H}^{v_1}$, *où α peut être déterminé par* $\alpha = \frac{1}{4}\text{Trace}(\prod_{i=1}^{n}\mathtt{H}^{i,i+1})$,

iii) Soit $\bar{\mathtt{H}}$ une matrice $(4n)\times 4$ résultant de la concaténation verticale des matrices $\mathtt{H}^{v_i}$. Les quatre colonnes de $\bar{\mathtt{H}}$ sont orthonormées.

La preuve de ce résultat est proposée en Annexe C. Ce résultat, nous permet de définir l’algorithme suivant pour estimer les matrices projectives absolues $\{\mathtt{H}^{v_i}\}$. Etant donné les homographies relatives $\{\mathtt{H}^{i,i+1}\}$, nous calculons dans un premier temps α suivant la formule de ii) et nous minimisons alors la fonction de coût

$$\sum_{i=1}^{n-1}\frac{\|\mathtt{H}^{v_i} - \mathtt{H}^{i,i+1}\mathtt{H}^{v_{i+1}}\|_2^2}{\max(\sigma_{v_i}^2, \sigma_{v_{i+1}}^2)} + \frac{\|\alpha\,\mathtt{H}^{v_n} - \mathtt{H}^{n,1}\mathtt{H}^{v_1}\|_2^2}{\max(\sigma_{v_1}^2, \sigma_{v_n}^2)} \tag{2.13}$$

Les $\{\mathtt{H}^{v_i}\}$ étant sous contraintes d’orthonormalité des colonnes de $\bar{\mathtt{H}}$. Ici, $\|\cdot\|_2$ est la norme de Frobenius. La solution exacte de ce problème d’optimisation non convexe peut être obtenue en utilisant une décomposition en valeurs singulières d’une matrice de taille $4n \times 4n$ construite à partir de α et $\{\mathtt{H}^{i,i+1}\}$. Dans la mesure où ce problème est standard (basé sur le théorème Courant-Fisher minimax [GVL96, Thm. 8.1.2]), nous ne donnons pas plus de détails ici. Nous proposons une démonstration de ce résultat optimisation linéaire sous contraintes d’orthonormalité en annexe B.1.4.

1^{er} frame $2^{ième}$ frame dernier frame

FIG. 2.9 – *Cette figure illustre l'amélioration faite à chaque étape de notre algorithme. Si les caméras sont reconstruites sans imposer de contraintes de boucle, les droites épipolaires entre le premier et le dernier frame sont très imprécises (1ère ligne). Elles deviennent bien plus précises une fois que l'optimisation linéaire séquentielle a été effectuée (2nde ligne). Enfin, le résultat est visuellement quasiment parfait une fois que l'enregistrement des homographies a été effectué (3ième ligne). La figure peut être mieux visualisée en agrandissant le pdf.*

2.4.3.2 Le cas de plusieurs boucles

Supposons maintenant que nous avons identifié plusieurs boucles dans le graphe des tenseurs trifocaux. Soit N_{loop} le nombre de ces boucles. Nous appliquons pour chaque boucle la méthode de la section précédente et obtenons une homographie pour chaque noeud de la boucle. Dans le cas général, un noeud de $\mathcal{G}_{\text{triplet}}$ peut appartenir à plusieurs boucles, dans ce cas nous aurons plusieurs homographies pour ce noeud. Il est nécessaire de forcer la cohérence de ces homographies. Dans ce but, nous définissons le graphe $\mathcal{G}_{\text{loop}}$ ayant N_{loop} noeuds, chaque noeud représentant une boucle. Deux noeuds de $\mathcal{G}_{\text{loop}}$ sont liés par une arête, si les cycles correspondants ont des intersections non vides. Nous supposerons que le graphe $\mathcal{G}_{\text{loop}}$ est connecté, sinon il est impossible de calibrer différentes composantes connectées dans un même espace projectif.

L’étape suivante consiste à déterminer un arbre $\mathcal{T}_{\text{loop}}$ à partir de $\mathcal{G}_{\text{loop}}$ de profondeur minimale. Dans la mesure où le nombre de cycles est supposé relativement petit, cette étape ne prend que peu de temps. Soit $(\mathcal{L}, \mathcal{L}')$ une paire de noeuds adjacents de $\mathcal{T}_{\text{loop}}$. Par un argument analogue à celui de la proposition 15, on peut montrer qu’il existe une homographie 4×4 $\mathtt{H}^{\mathcal{L},\mathcal{L}'}$ telle que $\mathtt{H}^{v,\mathcal{L}} \cong \mathtt{H}^{v,\mathcal{L}'}\mathtt{H}^{\mathcal{L}',\mathcal{L}}$ à une erreur de cyclicité près, pour chaque triplet de caméras $v \in \mathcal{L} \cap \mathcal{L}'$. Ici, $\mathtt{H}^{v,\mathcal{L}}$ (resp. $\mathtt{H}^{v,\mathcal{L}'}$) étant l’homographie assignée (*voir* section précédente) au triplet v faisant parti de la boucle $\mathcal{L}$ (resp. $\mathcal{L}'$). L’homographie $\mathtt{H}^{\mathcal{L}',\mathcal{L}}$ peut être estimée en minimisant la fonction

$$\sum_{v \in \mathcal{L} \cap \mathcal{L}'} \|\alpha_v \mathtt{H}^{v,\mathcal{L}} - \mathtt{H}^{v,\mathcal{L}'}\mathtt{H}^{\mathcal{L}',\mathcal{L}}\|_2^2 / \sigma_v^2 \tag{2.14}$$

où la matrice $\mathtt{H}^{\mathcal{L}',\mathcal{L}}$ et les paramètres $\{\alpha_v\}$ sont sous contrainte

$$\|\mathtt{H}^{\mathcal{L}',\mathcal{L}}\|_2^2 + \sum_{v \in \mathcal{L} \cap \mathcal{L}'} \alpha_v^2 = 1. \tag{2.15}$$

Une fois de plus, cette minimisation peut être effectuée en calculant les vecteurs correspondant à la plus petite valeur singulière d’une matrice adéquate. Enfin, pour forcer la cohérence des homographies absolues calculées à partir de plusieurs boucles, nous procédons comme suit.

Nous ne modifions pas les homographies calculées pour la boucle $\mathcal{L}_0$ constituant la racine de l’arbre de profondeur minimale $\mathcal{T}_{\text{loop}}$. Pour toutes les autres boucles $\mathcal{L}$, soit $\mathcal{L}_0 \rightarrow \mathcal{L}_1 \rightarrow \ldots \rightarrow \mathcal{L}_k \rightarrow \mathcal{L}$ le chemin unique reliant $\mathcal{L}$ à la racine. Alors, chaque homographie absolue $\mathtt{H}^{v,\mathcal{L}}$, $v \in \mathcal{L}$, calculée pour la boucle $\mathcal{L}$ en utilisant la méthode de la section précédente est remplacée par $\mathtt{H}^{v,\mathcal{L}}\mathtt{H}^{\mathcal{L},\mathcal{L}_k} \cdots \mathtt{H}^{\mathcal{L}_1,\mathcal{L}_0}$. Après cette modification, les images par $\mathtt{H}^{v,\mathcal{L}}$ des matrices de caméras $\mathtt{P}^{j,\gamma_v}$ $(j = 1, 2, 3)$ seront presque toutes dans le même espace projectif. Il est donc possible de retrouver les matrices des caméras définitives $\mathtt{P}^i$ par un calcul simple présenté dans la section suivante.

2.4.3.3 Estimation des matrices des caméras

Une fois que l’ensemble des homographies absolues a été trouvé, nous passons à l’estimations des matrices des caméras $\{\mathtt{P}^{j,*}\}$. Grâce aux calculs effectués dans les étapes précédentes, chaque matrice de projection $\mathtt{P}^{j,*}$ peut être estimée indépendamment des autres. Pour faciliter les notations et puisqu’il n’y a pas de perte de généralité, concentrons nous sur le calcul de $\mathtt{P}^{1,*}$.

Nous commençons par déterminer les noeuds de $\mathcal{G}_{\text{triplet}}$ qui contiennent la première vue. Soit $\mathcal{V}_1$ représentant l'ensemble de ces noeuds. A chaque noeud $v \in \mathcal{V}_1$ correspond une estimation de $\mathtt{P}^{1,*}$, notée $\mathtt{P}^{1,\gamma_v}$. De plus, nous avons un ensemble d'homographies estimées $\mathtt{H}^{v,\mathcal{L}}$ qui satisfont, à une erreur résiduelle de cyclicité près, la relation $\mathtt{P}^{1,v}\mathtt{H}^{v,\mathcal{L}} \cong \mathtt{P}^{1,*}$. Ceci est équivalent à

$$\alpha_{v,\mathcal{L}}\mathtt{P}^{1,v}\mathtt{H}^{v,\mathcal{L}} = \mathtt{P}^{1,*}, \qquad \forall v \in \mathcal{V}_1,\ \forall \mathcal{L} \supset \{v\} \tag{2.16}$$

pour $\alpha_{v,\mathcal{L}} \in \mathbb{R}$. Dans l'équation (2.16), les inconnues sont les réels $\alpha_{v,\mathcal{L}}$ et la matrice $\mathtt{P}^{1,*}$. Puisque cette matrice doit être de rang 3, elle a une norme de Frobenius non nulle. Ainsi, nous calculons $\mathtt{P}^1$ grâce à $\mathtt{P}^{1,*}$ comme étant la solution de

$$\arg\min_{\mathtt{P}} \min_{\|\mathtt{P}\|_2^2+\|\boldsymbol{\alpha}\|_2^2=1} \sum_{\mathcal{L}} \sum_{v\in\mathcal{L}\cap\mathcal{V}_1} \frac{\left\|\alpha_{v,\mathcal{L}}\mathtt{P}^{1,v}\mathtt{H}^{v,\mathcal{L}} - \mathtt{P}\right\|_2^2}{\sigma_v^2}, \tag{2.17}$$

où $\boldsymbol{\alpha}$ représente le vecteur qui a pour coordonnées les réels $\alpha_{v,\mathcal{L}}$. Une fois de plus, le problème (2.17) peut être résolu de manière explicite en calculant la SVD d'une matrice adéquate.

2.5 Une seconde solution : minimisation linéaire alternée

2.5.1 Concepts fondamentaux utilisés

Les preuves des propositions de cette section sont proposées en Annexe C.

2.5.1.1 Réduction du cycle de N triplets à K triplets

Proposition 16 *Si l'on considère un ensemble de Tenseurs Trifocaux connectés et cohérents deux à deux dans un arbre, avec la paramétrisation proposée dans la proposition 14 :*

$$\left(\left[\mathcal{T}^{opn}, \boldsymbol{\gamma}^{-K}\right] \ \ldots\ \left[\mathcal{T}^{ijk}, \boldsymbol{\gamma}^{0}\right] \ \ldots\ \left[\mathcal{T}^{lmh}, \boldsymbol{\gamma}^{N}\right]\right), \tag{2.18}$$

Cet arbre peut être réduit à un unique tenseur trifocal $\left[\mathcal{T}^{iol}, f\left(\boldsymbol{\gamma}^0\right)\right]$ où f est une fonction linéaire et où les matrices fondamentales reconstruites F^{li} et F^{oi} ne dépendent pas de $\boldsymbol{\gamma}^{-K}$, $\boldsymbol{\gamma}^0$ et $\boldsymbol{\gamma}^N$. De plus, $f\left(\boldsymbol{\gamma}^0\right)$ peut s'écrire :

$$f\left(\boldsymbol{\gamma}^0\right) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ a'_1 & -1 & 0 & 0 \\ a'_2 & 0 & -1 & 0 \\ a'_3 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \gamma'^0{}_0 \\ \boldsymbol{\gamma}^0{}_1 \\ \boldsymbol{\gamma}^0{}_2 \\ \boldsymbol{\gamma}^0{}_3 \end{bmatrix} + \begin{bmatrix} 0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}. \tag{2.19}$$

De la proposition précédente, il s'en suit qu'un cycle de N triplets de caméras peut être divisé en K arbres connectés. Finalement, le cycle peut être réduit à un cycle de K Tenseurs Trifocaux, en considérant fixes les autres $N-K$ Tenseurs Trifocaux. Plus particulièrement nous réduirons le cycle à un cycle de $K = 4$ Tenseurs Trifocaux duaux.

2.5.1.2 Linéarisation de la contrainte de cyclicité pour 4 triplets

La proposition suivante établit que, avec notre paramétrisation des tenseurs trifocaux, la troisième matrice fondamentale peut être calculée simplement et linéairement en fonction des inconnues. Ce résultat sera utile pour la preuve de la proposition 18.

Proposition 17 *Si l’on considère le Tenseur Trifocal $\mathcal{T}^{ijk}$, avec notre paramétrisation vue en proposition 14, la matrice fondamentale F^{jk} est linéaire en $\boldsymbol{\gamma}$ et peut s’écrire simplement,*

$$\begin{aligned} F^{jk} \quad = \quad & \left(\boldsymbol{\gamma}_{1:3} \cdot \mathbf{e}^{ki}\right) [\mathbf{e}^{ij}]_\times G^{ji} G^{ik} + \\ & \left[G^{ji}\mathbf{e}^{ki}\right]_\times \left[\left(\mathbf{e}^{ij}{\boldsymbol{\gamma}_{1:3}}^T - {\gamma'}_0 G^{ji}\right) G^{ik} + \beta \mathbf{e}^{ij}{\mathbf{e}^{ik}}^T\right]. \end{aligned} \tag{2.20}$$

En notant $G^{ji} = [\mathbf{e}^{ij}]_\times F^{ji}$.

Pour chaque boucle il y a 15 contraintes de cyclicité. Ces contraintes de cyclicité peuvent être formulées comme “le produit des homographies est égal à la matrice identité à un facteur multiplicatif près”. Maintenant nous allons formuler les contraintes de cyclicité d’une façon différente, et les considérer plus précisément comme des contraintes de cohérence dans le cas d’une boucle composée de 4 Tenseurs Trifocaux. Dans ce cas nous allons voir que les contraintes de cyclicité ne sont rien d’autre qu’un problème linéaire.

Proposition 18 *Nous considérons la boucle composée de 4 Tenseurs Trifocaux, $\left[\mathcal{T}^{ijk}, \boldsymbol{\gamma}^0\right]$, $\left[\mathcal{T}^{kil}, \boldsymbol{\gamma}^1\right]$, $\left[\mathcal{T}^{lkj}, \boldsymbol{\gamma}^2\right]$ et $\left[\mathcal{T}^{jli}, \boldsymbol{\gamma}^3\right]$. En considérant le vecteur concaténé $\Gamma = \left(\boldsymbol{\gamma}^0, \boldsymbol{\gamma}^1, \boldsymbol{\gamma}^2, \boldsymbol{\gamma}^3\right)^T$, les contraintes de cyclicité peuvent s’écrire simplement, $\mathtt{C}\Gamma = \mathbf{d}$, où $\mathtt{C}$ est une matrice de taille 15×16 et de rang 15.*

Généralement nous avons un cycle de N triplets, et suivant la proposition 16, nous pouvons travailler sur un cycle réduit de 4 Tenseurs Trifocaux duaux. Ainsi, en considérant les autres tenseurs fixes, les contraintes de cyclicités s’écrivent comme un problème linéaire. On peut revenir, à partir des tenseurs duaux, au système initial de Γ, en utilisant le changement de variables de l’équation (2.19). Un fois que ce changement de variables a été effectué, en connaissant le vecteur nul de $\mathtt{C}$, la solution générale de ces contraintes de cyclicité peut s’écrire simplement $\Gamma = \Gamma_{sol} + \alpha\Gamma_{null}$. En fait nous avons $4 \times 4 = 16$ inconnues Γ et 15 contraintes, et donc il reste exactement un degré de liberté. Nous souhaitons calculer l’optimum global du système suivant :

$$\begin{aligned} &\min \|\mathtt{M}\Gamma - \mathbf{m}\|_2 \\ &\text{subject to } \mathtt{C}\Gamma = \mathbf{d}, \end{aligned} \tag{2.21}$$

Où de manière équivalente $\min \sum_p \left(\mathtt{M}_{p,:}\Gamma_{sol} + \alpha \mathtt{M}_{p,:}\Gamma_{null} - m_p\right)^2$, qui est simplement, une fois cette expression dérivée, une équation du premier degré en α. Comme on le voit, en utilisant 4 Tenseurs Trifocaux duaux, l’optimisation sous contraintes peut être effectuée très rapidement sans même recourir à un logiciel d’optimisation.

Algorithm 4 Seconde approche : minimisation linéaire alternée

Require: 1 cycle, N Tenseurs, $\Gamma_{initial}$ concaténation des $\gamma^i_{initial}$, $\mathtt{M}$, $\mathbf{m}$, et $1 < K < 20$.

$bool\ Ended = false$.

$\Gamma = \Gamma_{initial}$, $\gamma^i = \gamma^i_{initial}$.

while $Ended == false$ **do**

Calculer le vecteur des Erreurs $\mathbf{E}$, $\mathbf{E}[i] = \frac{\|\mathtt{M}_i\gamma^i - \mathbf{m}_i\|}{\|\mathtt{M}_i\gamma^i_{initial} - \mathbf{m}_i\|}$.

Trier $\mathbf{E}$ et choisir les 4 Tenseurs Trifocaux $i_1 < i_2 < i_3 < i_4$ avec les plus petites erreurs $\mathbf{E}[i]$.

Calculer pour les 4 Tenseurs duaux les matrices de changement de variable $\mathtt{A}_\mathbf{S}$ et $\mathbf{b}_\mathbf{S}$, suivant la proposition 2.19.

Calculer $\mathtt{C}$, et $\mathbf{d}$ tels que les contraintes de cyclicité pour ces 4 tenseurs duaux est $\mathtt{C}\Gamma_n = \mathbf{d}$, suivant la proposition 18.

Revenir du système dual vers le système initial. $\mathtt{C}' = \mathtt{C}\mathtt{A}_\mathbf{S}$ et $\mathbf{d}' = \mathbf{d} - \mathtt{C}\mathbf{b}_\mathbf{S}$.

Par SVD calculer Γ_{sol}, solution de $\mathtt{C}'\Gamma_s = \mathbf{d}'$, et Γ_{null}, le vecteur nul de $\mathtt{C}'$.

Calculer α solution de $\min \|\mathtt{M}\Gamma_{sol} + \alpha\mathtt{M}\Gamma_{null} - \mathbf{m}\|_2 \Leftrightarrow$ équation du premier degré.

La solution optimale est $\Gamma_s = \Gamma_{sol} + \alpha\Gamma_{null}$.

$\beta = \frac{\|\Gamma\|}{\|\Gamma - \Gamma_s\|\, K\, N}$.

if $\beta < 1$ **then**

Déplacer légèrement Γ vers Γ_s : $\Gamma = \Gamma - \beta(\Gamma - \Gamma_s)$

else

Forcer les contraintes de cyclicité : $\Gamma = \Gamma_s$

$Ended = true$

end if

end while

return Γ.

2.5.2 *Algorithme de minimisation alternée*

L'idée principale de notre algorithme est de choisir alternativement 4 Tenseurs Trifocaux parmi les N et de minimiser pour les Tenseurs trifocaux choisis la fonction de coût $\|\mathtt{M}\Gamma - \mathbf{m}\|_2$ sujette aux contraintes de cyclicité linéaires. De cette façon nous obtenons une fonction de coût décroissante. En effet, pour la première itération nous choisissons 4 Tenseurs et minimisons la fonction de coût sujette aux contraintes de cyclicité linéaires, et à la seconde itération nous minimisons également la fonction de coût avec d'autres tenseurs sujette aux contraintes de cyclicité linéaires. Mais dans la mesure où après la première itération les contraintes de cyclicité étaient déjà remplies parfaitement, nous sommes certains de minimiser la fonction de coût durant la seconde optimisation. Ainsi nous obtenons un algorithme convergent permettant de satisfaire exactement les contraintes de cyclicité. Cependant les 4 Tenseurs Trifocaux choisis lors de la première itération ont une forte erreur pour la fonction de coût après la première optimisation sous contraintes cycliques. En pratique, nous remarquons que cette erreur n'est pratiquement pas répartie avec les autres tenseurs durant les optimisations alternées suivantes.

Si la méthode précédente offre des garanties théoriques très intéressantes, nous proposons un algorithme permettant de répartir régulièrement les erreurs parmi les Tenseurs Trifocaux.

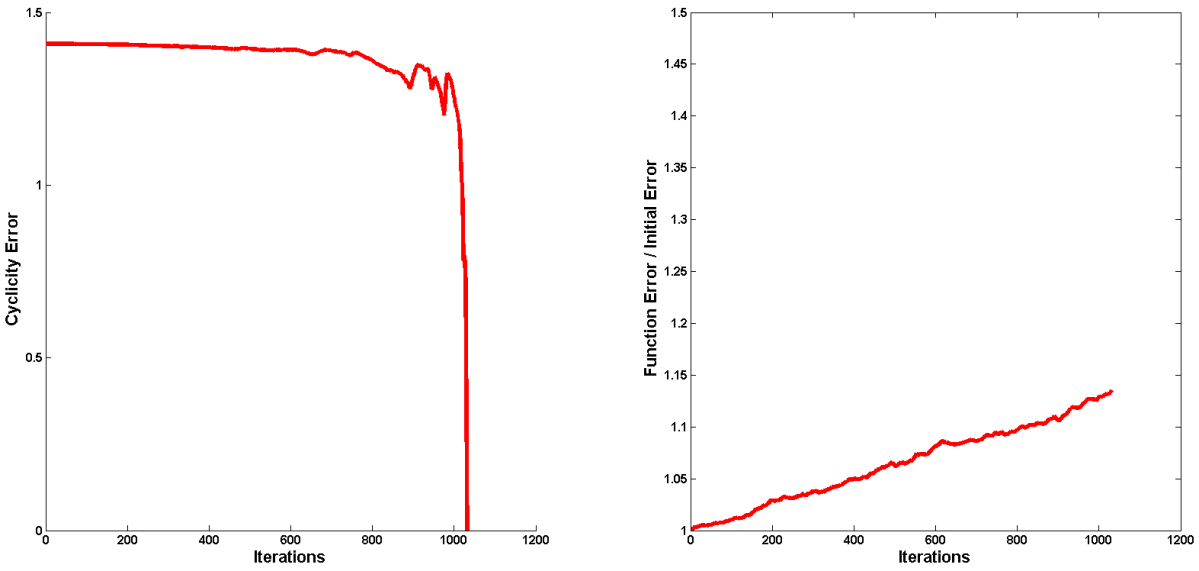


FIG. 2.10 – *Sur la gauche nous remarquons que pour le jeu de données dinosaur, l’erreur de cyclicité croît ou décroît alternativement, mais elle finit par converger parfaitement vers les contraintes de cyclicité après environ* 1000 *itérations. Sur la droite nous voyons que l’erreur de la fonction de coût* $\|\mathbb{M}\Gamma - \mathbf{m}\|_2$ *est seulement multipliée par moins de* 1.15 *après convergence. Cette figure peut être mieux visualisée en agrandissant la visualisation du pdf.*

Comme ci-dessus, pour les 4 Tenseurs Trifocaux duaux choisis nous calculons la meilleure solution Γ_s satisfaisant exactement les contraintes de cyclicité et minimisant la fonction de coût. Etant donné le vecteur Γ actuel, nous mettons à jour Γ de la façon suivante : $\Gamma = \Gamma - \alpha(\Gamma - \Gamma_s)$. Avec

$$\alpha = \frac{\|\Gamma\|}{\|\Gamma - \Gamma_s\| \, K \, N}. \tag{2.22}$$

N étant le nombre d’images et K en pratique choisi entre 1 et 20, suivant la précision ou la rapidité de l’algorithme désirée. Plus K est grand, plus l’algorithme converge doucement mais avec une très bonne répartition des erreurs parmi les tenseurs. Si pour une itération $\alpha \geq 1$ cela signifie que nous avons quasiment atteint les contraintes de cyclicité, et nous les forçons totalement : $\Gamma = \Gamma_s$. Pour choisir les Tenseurs Trifocaux duaux au début de chaque itération nous calculons pour tous les tenseurs les erreurs :

$$\mathbf{E}_\gamma = \frac{\|\mathbb{M}\gamma - \mathbf{m}\|}{\|\mathbb{M}\gamma_{initial} - \mathbf{m}\|}. \tag{2.23}$$

Et nous choisissons les 4 tenseurs ayant les plus petites erreurs, de cette façon nous garantissons une répartition régulière des erreurs tout au long de notre méthode itérative.

2.5.2.1 *Résultats et comparaison avec la première approche*

Pour la séquence du *dinosaur* composée de 36 images $1 < i < 36$, comme on peut le voir sur la figure 2.10 notre seconde approche permet de converger exactement vers les contraintes

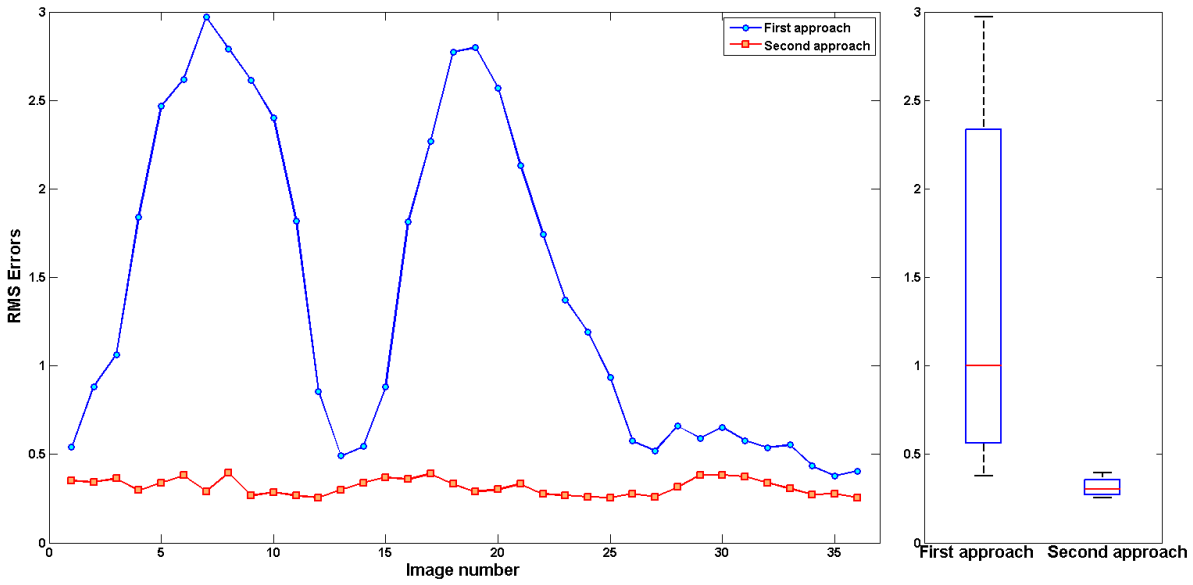


FIG. 2.11 – *Pour le jeu de données dinosaur, pour chacun des* 36 *triplets de caméras, nous comparons les erreurs de reprojection entre la première et la seconde approche. Sur la gauche on observe que la seconde approche donne lieu à des RMSE pixelliques (pixel Root Mean Square Errors) bien plus petites qu'avec la première approche. De plus, la variance des RMSE est considérablement plus petite avec la seconde approche par minimisation linéaire alternée. Cette figure peut être mieux visualisée en agrandissant la visualisation du pdf.*

de cyclicité après environ 1000 itérations (nous avons choisi $K = 15$ pour nos expériences, ce qui donne lieu à une convergence lente mais à une bonne répartition de l'erreur). Dans cette figure, l'erreur de la contrainte de cyclicité est non linéaire et est tout simplement la norme de Frobenius de la différence entre la matrice identité et le produit des homographies normalisées. Sur la partie droite de la figure 2.10, on remarque que l'erreur de la fonction de coût $\|\mathtt{M}\Gamma - \mathbf{m}\|_2$ n'augmente pratiquement pas pendant la minimisation alternée. En effet, après convergence celle-ci a été multipliée par moins de 1.15.

Sur la figure 2.11, sur l'axe horizontal nous montrons les triplets de caméras i, avec $1 < i < 36$. Le triplet de caméras i étant constitué des images i, $i+1$ et $i+2$. Le triplet 36 est constitué des images 36, 1 et 2, ainsi pour les triplets 35 et 36 notre erreur pixellique de reprojection représentera également l'erreur de cyclicité qui s'annule avec notre seconde approche. Etant donné des correspondances point-point-point nous calculons les points $3D$ et alors les erreurs de reprojection. Finalement, sur l'axe vertical nous montrons pour chaque triplet l'erreur Root Mean Square (RMSE), et comparons la première approche et la seconde approche. Avant même qu'aucun Ajustement des faisceaux (Bundle adjustment) ne soit effectué, nous voyons que l'erreur moyenne de reprojection est d'environ 2 pixels pour la première approche alors qu'elle est inférieure à 0.5 pixel avec la seconde approche. De plus, la variance de la RMSE est considérablement plus petite avec la seconde approche par minimisation alternée.

Comme montré dans la table 2.1, pour toutes nos expériences notre seconde approche par

Jeux de données	Méthode	# Itérations Jusqu’à convergence	RMSE (pixel)
Dinosaur	1ère Approche	-	1.01
	2nde Approche	1027	0.31
Temple	1ère Approche	-	0.10
	2nde Approche	340	0.09
Detenice	1ère Approche	-	2.28
	2nde Approche	1393	1.27
Calvary	1ère Approche	-	1.02
	2nde Approche	1286	1.76

TAB. 2.1 – *Pour les différents jeux de données nous comparons, la RMSE pixellique (pixel Root Mean Square Error), entre la première et la seconde approche. Nous voyons que la RMSE est souvent plus petite pour la seconde approche avec minimisation linéaire alternée. De plus, pour toutes nos expériences, la seconde approche a permis de converger parfaitement vers les contraintes de cyclicité.*

minimisation linéaire alternée converge systématiquement et exactement vers les contraintes de cyclicité. De plus lorsque nous la comparons à la première approche, la seconde approche donne lieu à une plus petite RMSE, sauf pour le jeu de données *Calvary*. Puisque notre dernière approche, a convergé exactement vers les contraintes de cyclicité pour tous les jeux de données testés, il serait très intéressant de savoir pourquoi elle converge et s’il existe des garanties théoriques pour cela. Probablement, si cette hypothèse était vérifiée, pour effectuer cette démonstration, il faudrait étudier une autre erreur de cyclicité que celle proposée pour la figure 2.10. En effet, dans la figure 2.10, nous remarquons que la méthode converge vers les contraintes de cyclicité mais avec une erreur de cyclicité tour à tour croissante puis décroissante. En fait l’erreur de cyclicité utilisée est fortement non linéaire. On pourrait également supposer que, la raison pour laquelle cette méthode converge est qu’il existe d’autres linéarités, et la contrainte de cyclicité pourrait être linéaire pour un cycle composé de plus que 4 Tenseurs Trifocaux.

2.6 Résultats Expérimentaux

2.6.1 Implémentation

Afin d’implémenter les méthodes ci-dessus, nous extrayons et mettons en correspondance des points SIFT dans toutes les images, suivant la procédure expliquée en Annexe A qui est en adéquation avec [Low04a]. Enfin, nous calculons les géométries épipolaires par DEGENSAC [CWM05]. Le DEGENSAC est une méthode proche du RANSAC 1.2.3 qui prend en compte les cas dégénérés, comme le cas où 4 correspondances parmi les 7 sont coplanaires et reliées par une homographie 2D 3×3. Il est possible d’accélérer les calculs pour cette partie en utilisant des versions récentes du RANSAC, telles que [CM05, SLK09]. Ces géométries épipolaires nous permettent d’obtenir des correspondances à deux images fiables en retirant les mauvaises

correspondances non compatibles avec les matrices fondamentales calculées. Nous fusionnons ensuite ces correspondances à deux images en correspondances à trois vues. Il est alors possible de faire des RANSAC à 4 points pour estimer d'une part les tenseurs trifocaux réduits, c'est-à-dire les inconnues γ, et d'autre part pour filtrer les mauvaises correspondances trifocales. Nous avons ainsi les données suffisantes pour nos deux méthodes présentées dans ce manuscrit, c'est-à-dire une première estimation de γ et les correspondances à trois vues ainsi que les géométries épipolaires qui nous permettent d'obtenir les matrices $\mathtt{M}$ et m. Une fois nos deux méthodes effectuées nous avons pour sortie notre ensemble de caméras dans un espace projectif. Nous créons des "track" s'étalant sur plus de trois images en fusionnant nos correspondances précédentes. Enfin, nous effectuons alors quelques bundle adjustment projectifs (Annexe 1.2.5) afin de raffiner ces caméras, puis nous récupérons les caméras dans un espace métrique suivant la méthode J. Ponce *et al.* [PMP+05].

Jeux des données	**#frames**	**résolution**	**# reprojections**		**RMSE (pixel)**	
			Notre	Bundler	Notre	Bundler
Dinosaur	36	576 × 720	45,250	37,860	0.27	0.25
Temple	45	480 × 500	26,535	23,761	0.08	0.11
Fountain P11	11	2048 × 3072	57,547	23,648	0.16	0.13
Herz-Jesu R23	23	2048 × 3072	129,803	–	0.41	–
Detenice	34	1536 × 2048	30,200	–	0.15	–
Calvary	52	2624 × 3972	54,798	–	0.51	–

TAB. 2.2 – *Caractéristiques des jeux de données utilisés pour notre validation expérimentale. De gauche à droite : nombre de frames pour chaque séquence, la résolution des images (toutes les images dans un même jeu de données ont la même résolution), le nombre de reprojections dans les images 2D utilisées pour le BA final pour notre méthode et pour bundler [SSS08], La Root mean square error (RMSE) pixellique pour chaque méthode. Les nombres indiqués dans les 4 dernières colonnes sont là pour montrer que les deux méthodes (la nôtre ou bundler) fonctionnent bien, et non pour les comparer.*

2.6.2 Jeux de données

Nous avons testé notre méthodologie sur six jeux de données : la séquence du *dinosaur* (36 frames), la séquence du *temple* (45 frames),la séquence du *fountain P11* (11 frames), la séquence du *Herz-Jesu R23* (23 frames), la séquence du *Detenice fountain* (34 frames) et la séquence du *calvary* (52 frames). Pour les trois premiers jeux de données, la vérité terrain des matrices des caméras est disponible sur Internet.

2.6.3 Mesures de qualité

Dans la mesure où la principale contribution de la partie de ce manuscrit consacrée à la calibration concerne la reconstruction projective, il semble naturel d'évaluer la qualité de notre

FIG. 2.12 – *Trois frames de chaque jeu de données utilisés pour tester notre procédure de calibration. De gauche à droite : dinosaur, temple, fountain P11, Herz-Jesu R23 [SvHVG+08], Calvary, Detenice fountain.*

approche en utilisant la distance suivante :

$$d_{proj}(\{\mathtt{P}^j\},\{\mathtt{P}^{j,*}\}) = \inf_{\boldsymbol{\alpha},\mathtt{H}} \sum_{j=1}^{n} \|\alpha_j \mathtt{P}^j \mathtt{H} - \mathtt{P}^{j,*}\|_2^2, \tag{2.24}$$

où $\mathtt{P}^j$ et $\mathtt{P}^{j,*}$ sont respectivement les caméras reconstruites et réelles, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ est un vecteur de nombre réels et $\mathtt{H}$ est une homographie 3D. Naturellement, cette mesure ne peut être utilisée que pour des jeux de données ou la vérité terrain est disponible. Notons également que le calcul du minimum de (2.24) est un problème d’optimisation non convexe. Nous le résolvons en calculant dans un premier temps la solution de norme unitaire du problème aux moindres carrés suivant $\min_{\boldsymbol{\alpha},\mathtt{H}} \sum_{j=1}^{n} \|\mathtt{P}^j\mathtt{H} - \alpha_j^{-1}\mathtt{P}^{j,*}\|_2^2$, et alors nous utilisons cette solution comme point de départ pour une minimisation alternée. Pour les exemples considérés ici, l’algorithme converge très rapidement et, puisque les résultats sont très bons, nous pensons que le minimum local que nous trouvons est en fait un minimum global, ou bien n’en est pas très éloigné.

2.6.4 Résultats

Pour les séquences du dinosaur, du temple et de fountain P11, dans la mesure où la vérité terrain existe, nous comparons nos résultats avec ceux obtenus avec bundler [SSS08], qui est un logiciel de calibration représentant l’état de l’art. La vérité terrain a été normalisée de sorte que la norme de Frobenius de toutes les caméras soit égale à 1. Pour chaque reconstruction (la notre et celle de bundler), nous avons calculé les nombres α_j et une homographie $\mathtt{H}$ en minimisant (2.24). Ceci nous permet de définir l’erreur par caméra comme $\|\alpha_j\mathtt{P}^j\mathtt{H} - \mathtt{P}^{j,*}\|_2^2$ pour la $j$ième caméra. Comme on peut le voir sur la Fig. 2.13, non seulement ces erreurs sont petites, mais en plus nos résultats sont très comparables à ceux obtenus avec bundler en dépit du fait que notre méthode n’utilise pas de Bundle Adjustment (BA décrit en section 1.2.5) intermédiaires

et ne suppose ni que le point principal est au centre de l'image, ni que le skew est nul. D'autre part bundler utilise la connaissance de la distance focale terrain, information que nous avons ici fournie à bundler, mais que nous n'avons pas utilisée pour notre méthode. On note également que l'erreur est bien distribuée sur l'ensemble des caméras dans la mesure où chacune des méthodes prend en compte le fait que la séquence est fermée (forme une boucle). De plus, les résultats montrés pour fountain P11 sont obtenus sans BA final, ce qui montre que notre méthode fournit une bonne initialisation pour l'optimisation non linéaire.

Pour les jeux de données où il n'y a pas de vérité terrain disponible, nous avons choisi d'utiliser la reconstruction par stéréovision multi-vues comme mesure d'évaluation. La reconstruction utilisée est basée sur la méthode [VKLP09][2]. Les résultats sont montrés en Fig. 2.1 (droite) pour la séquence du calvary et en Fig. 2.14 pour les séquences Herz-Jesu R23 et Detenice fountain. Afin de comparer nos résultats avec d'autres approches, rappelons que (comme écrit dans [SvHVG+08]) pour le Herz-Jesu R23 le logiciel d'ARC3D n'a réussi à calibrer que 4 caméras parmi les 23, alors que la méthode proposée dans [MP07] a calibré toutes les caméras avec une erreur assez grande pour les caméras 6-11. Bien que nous ne comparons pas quantitativement notre reconstruction à celle obtenue par [MP07], la précision de la reconstruction de la scène tridimensionnelle nous laisse penser que les caméras estimées sont très proches des véritables caméras.

2.7 Pour aller plus loin...

2.7.1 Une formulation relaxée du tenseur trifocal

Nous pouvons reformuler le tenseur trifocal défini en proposition 14, de manière plus générale :

$$\mathrm{T}_t^{ijk} = \mathrm{A}_t^{ij} \begin{bmatrix} 0 & 0 & a_t \\ 0 & 0 & b_t \\ d_t & e_t & c_t \end{bmatrix} (\mathrm{A}_t^{ik})^\top,$$

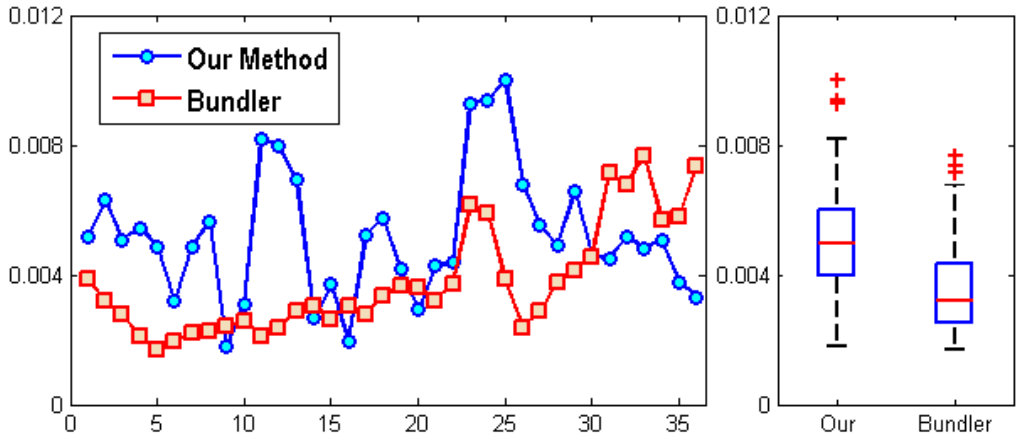
Avec cette formulation, comme établi dans [HZ03, Eq. 15.1] et montré en section 1.3.2, les épipoles correspondant à ce tenseur relaxé restent inchangées. En fait cette formulation est la plus générale permettant de ne pas changer les épipoles et donc de conserver des propriétés de linéarité. De plus les matrices fondamentales s'écrivent alors :

$$\mathrm{F}^{ji}{}_{1:3,t} = a_t [\mathbf{e}^{ij}]_\times \mathrm{F}^{ji}{}_{1:3,t} + b_t \mathrm{F}^{ji}{}_{1:3,t} \tag{2.25}$$

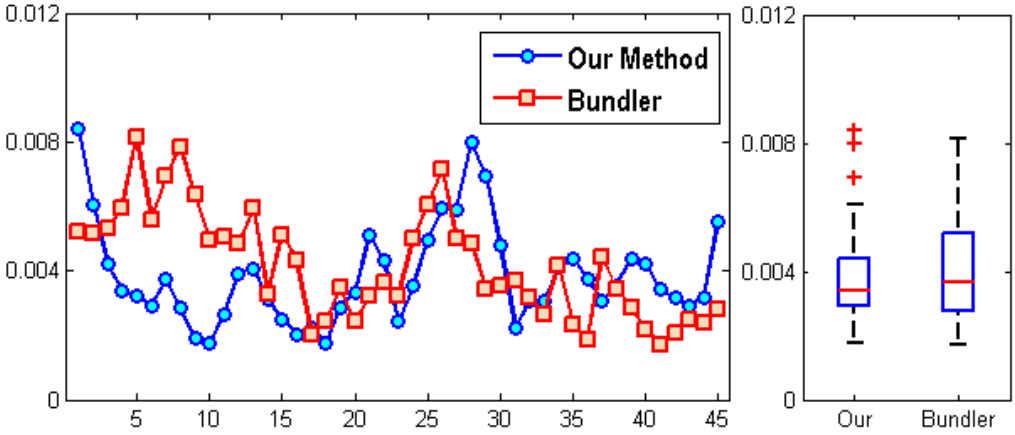
$$\mathrm{F}^{ki}{}_{1:3,t} = d_t [\mathbf{e}^{ik}]_\times \mathrm{F}^{ki}{}_{1:3,t} + e_t \mathrm{F}^{ki}{}_{1:3,t} \tag{2.26}$$

En effet comme établi dans [HZ03] et en section 1.3.2, les matrices fondamentales liées au tenseur $[\mathcal{T}^{ijk}]$ sont :

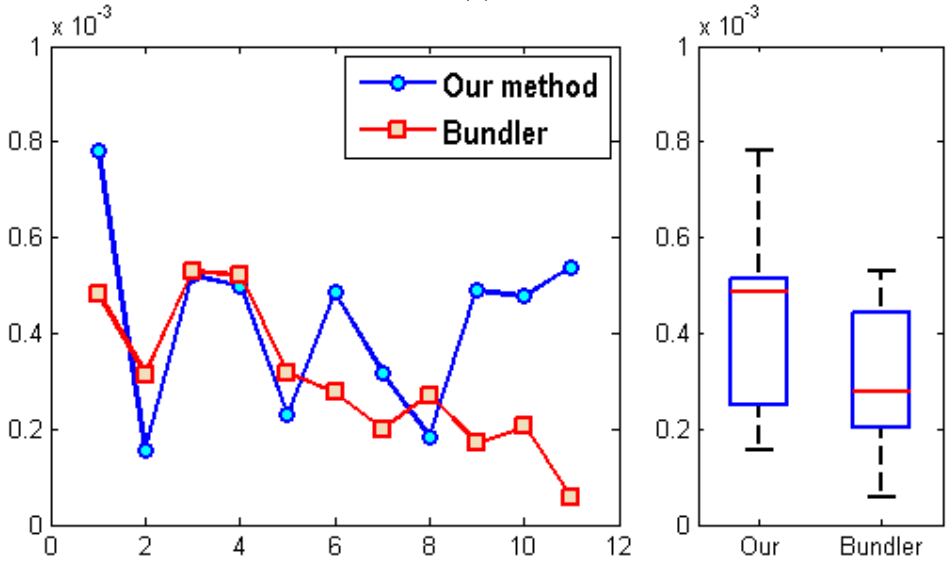
[2] Les résultats montrés en Fig. 2.1 sont obtenus sans raffinement de maillage final.



(a)



(b)



(c)

FIG. 2.13 – *Cette figure montre les erreurs en estimant les matrices des caméras pour notre méthode et pour bundler [SSS08]. L’erreur par caméra et leurs boxplots pour la séquence du dinosaur (a), du temple (b) et pour fountain P11 (c). Nous remarquons que notre méthode permet d’obtenir le même degré de précision que bundler, en dépit du fait que nous ne supposons pas les paramètres internes connus, alors que bundler récupère la distance focale terrain, suppose que le skew est nul et que le point principal est au centre.*

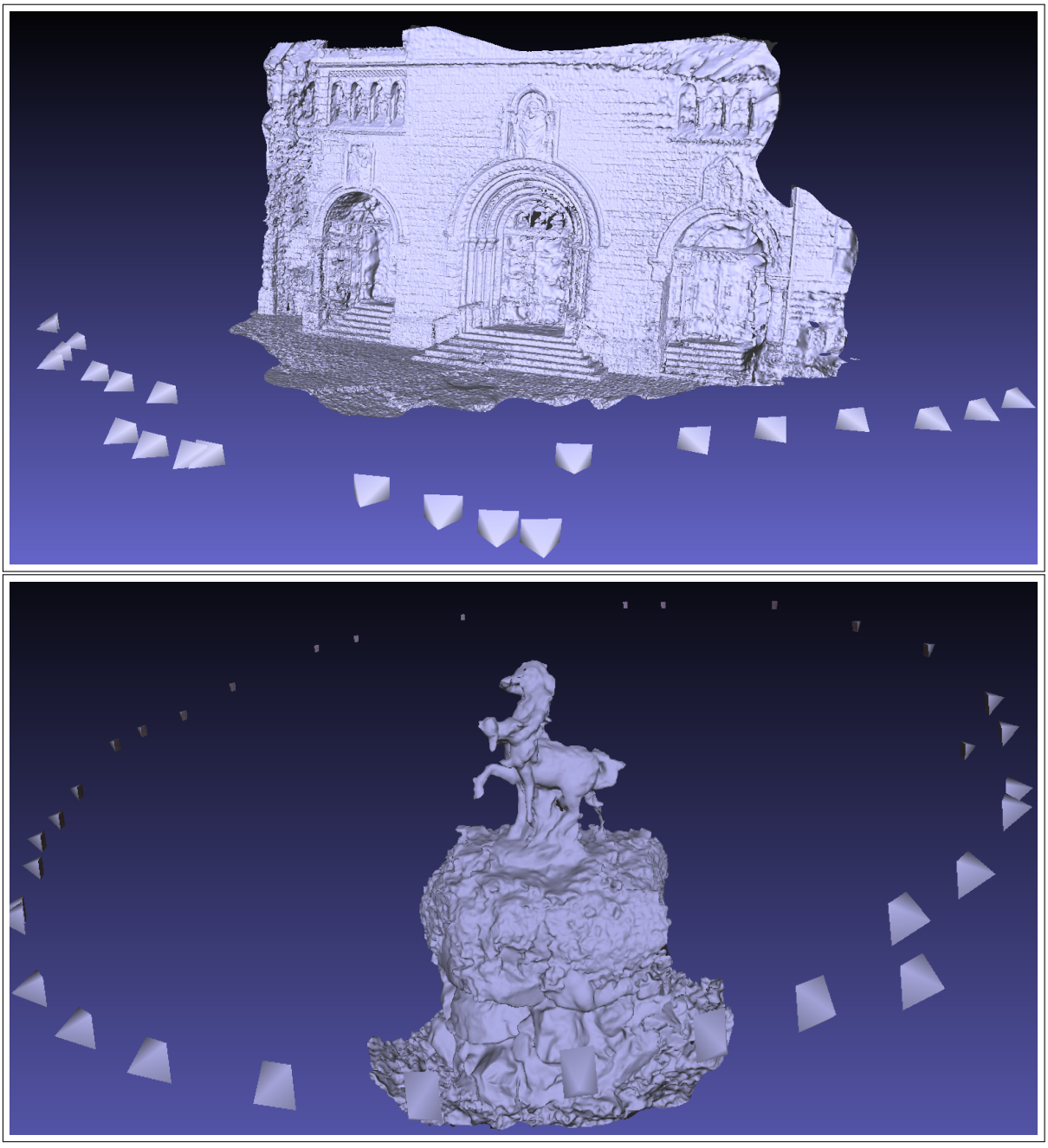


FIG. 2.14 – *Stéréovision Multi-vues en utilisant les matrices des caméras estimées avec notre méthode pour les jeux de données Herz-Jesu R23 et Detenice fountain. Pour ces jeux de données la vérité terrain n'est pas disponible mais la qualité de la reconstruction obtenue démontre la précision et la fiabilité des caméras estimées.*

$$\mathtt{F}^{ji} = [\mathbf{e}^{ij}]_\times [\mathtt{T}_1^{ijk}, \mathtt{T}_2^{ijk}, \mathtt{T}_3^{ijk}]\mathbf{e}^{ik} \tag{2.27}$$

$$\mathtt{F}^{ki} = [\mathbf{e}^{ik}]_\times [\mathtt{T}_1^{ijk^T}, \mathtt{T}_2^{ijk^T}, \mathtt{T}_3^{ijk^T}]\mathbf{e}^{ij} \tag{2.28}$$

En formulation homogène les 15 inconnues $\zeta = [a_t, b_t, c_t, d_t, e_t]$ peuvent être calculées linéairement à partir de correspondances point-point-point sous la forme $\mathtt{A}\zeta = \mathbf{0}$. Dans la formulation non homogène en posant par exemple $b_1 = 1$, nous calculons linéairement les 14 inconnues.

Remarquons que nous avons fixé dans la formulation relaxée deux épipoles 2×2 données. Le tenseur trifocal ayant 18 degrés de liberté, nous retrouvons les $18 - 4$ inconnues de notre tenseur relaxé.

2.7.2 Raffiner les matrices fondamentales

Grâce à la formulation vue en proposition 14, comme les matrices fondamentales sont fixes, deux tenseurs trifocaux consécutifs partagent la même matrice fondamentale et donc peuvent être reliés parfaitement par une transformation projective. Afin de conserver cette propriété essentielle nous pouvons forcer la contrainte pour que deux tenseurs consécutifs $\mathcal{T}^{ijk}$ et $\mathcal{T}^{kil}$ aient la même matrice fondamentale, c’est-à-dire $\mathtt{F}^{ki} = \mathtt{F}^{ik^T}$:

$$[\mathbf{e}^{ik}]_\times [\mathtt{T}_1^{ijk^T}, \mathtt{T}_2^{ijk^T}, \mathtt{T}_3^{ijk^T}]\mathbf{e}^{ij} = \left[[\mathbf{e}^{ki}]_\times [\mathtt{T}_1^{kil}, \mathtt{T}_2^{kil}, \mathtt{T}_3^{kil}]\mathbf{e}^{kl}\right]^T$$

Pour ne pas perdre de linéarité, nous pouvons transformer l’égalité à un facteur multiplicatif près en une égalité stricte, dans le mesure où nous travaillons avec des coordonnées homogènes. Néanmoins, la formulation homogène peut favoriser des solutions de vecteurs nuls pour des longues séquences. Aussi nous procédons autrement, et remarquons que pour la formulation non homogène initiale Γ, nous avions la relation suivante :

$\gamma'^{ijk}{}_0 \mathtt{F}^{ij}\left([\mathtt{T}^{ijk}]\right) = \mathtt{F}^{ij}\left([\mathtt{T}^{kil}]\right)$ et $\gamma^{ijk}{}_0 = 1$. Aussi nous pouvons, par exemple, utiliser les valeurs initiales de $\gamma'^{ijk}{}_0$ pour trouver le bon facteur d’échelle dans les égalisations entre matrices fondamentales avec notre formulation relaxée. Nous pouvons également fixer $e_1 = 1$ pour les tenseurs relaxés. De cette façon nous pouvons calculer les inconnues relaxées en résolvant le système d’optimisation convexe sous contraintes :

$$\begin{aligned} &\min \|\mathtt{A}Z\|_2 \\ &\text{subject to } \mathtt{C}\zeta = \mathbf{d}, \end{aligned} \tag{2.29}$$

Nous remarquons que la solution initiale Γ, permet de trouver une solution des inconnues ζ concaténées sous la forme Z, qui vérifie les contraintes, c’est-à-dire les contraintes d’égalité des matrices fondamentales entre tenseurs consécutifs. Nous utilisons donc cette solution initiale comme première estimation pour un logiciel d’optimisation convexe tel que “Matlab Optimization Toolbox” avec la fonction “lsqlin”. Cette optimisation avec estimation initiale et avec des matrices parcimonieuses est particulièrement rapide.

Ainsi nous pouvons conserver nos propriétés tout en faisant varier les matrices fondamentales, et en ne fixant que les épipoles. Cette méthode peut être intéressante car nos matrices

fondamentales initiales ne sont pas parfaites et sont bruitées, de plus nous utilisons ainsi les synergies entre tenseurs le long de la séquence pour raffiner les matrices fondamentales.

2.7.3 Vers une formulation symétrique du tenseur trifocal

Les formulations du tenseur trifocal vues en section 1.3.2, ou même en proposition 14, ne sont pas symétriques. Dans toute cette partie pour le tenseur $\mathcal{T}^{ijk}$, les matrices $\mathtt{F}^{ij}$ et $\mathtt{F}^{ik}$ jouent un rôle symétrique mais différent du rôle de la matrice fondamentale $\mathtt{F}^{jk}$. Du fait de cette dissymétrie, nous sommes obligés de rejeter une matrice fondamentale dans les triplets pour lesquels nous avons pu calculer les 3. L'algorithme pour choisir les matrices fondamentales à rejeter dans un graphe n'est pas forcément simple, et plusieurs solutions sont possibles. Aussi une formulation symétrique nous permettrait d'éluder cette question d'une part, et d'autre part d'utiliser les informations riches provenant de matrices fondamentales supplémentaires. Enfin comme nous l'avons vu dans cette partie, avec notre seconde approche qui permet en pratique d'atteindre systématiquement et parfaitement les contraintes de cyclicité, la question se pose de savoir si ces contraintes de cyclicité peuvent se linéariser pour plus qu'une boucle composée de 4 tenseurs. Une formulation symétrique permettrait peut-être également de simplifier davantage les contraintes de cyclicité.

Nous pourrions, par exemple, dire que le triplet i, j, k est décrit non par un tenseur trifocal, mais par 3 tenseurs trifocaux relaxés : $\mathcal{T}^{ijk}$, $\mathcal{T}^{jki}$ et $\mathcal{T}^{kij}$. Comme on peut le vérifier facilement ces 3 tenseurs permettraient de fixer les épipoles $\mathbf{e}^{ij}$, $\mathbf{e}^{ji}$, $\mathbf{e}^{ki}$, $\mathbf{e}^{ik}$, $\mathbf{e}^{kj}$ et $\mathbf{e}^{jk}$, soit 6×2 données. Nous aurions ainsi un tenseur trifocal ayant $18 - 12 = 6$ degrés de liberté.

Notre ensemble de 3 tenseurs relaxés ayant $14 \times 3 = 42$ paramètres, afin de le rendre cohérent nous devrions avoir un ensemble de $42 - 6 = 36$ contraintes internes, relatant que les matrices fondamentales et les épipoles sont cohérentes entre les tenseurs.

Une autre possibilité pour obtenir une formulation symétrique du tenseur trifocal, serait d'utiliser les résultats de [Pon10], où le tenseur trifocal est vu comme : 3 lignes dans l'espace s'intersectent en un point. Cette contrainte est symétrique, et se traduit par la nullité des mineurs d'une matrice. Cette méthode fait appel aux coordonnées de Plücker et aux matrices de projections inverses mappant un point dans l'image, vers une droite dans l'espace 3D (le rayon optique) écrite dans les coordonnées de Plücker.

Deuxième partie

Reconstruction spatio-temporelle

Dans ce manuscrit, nous décrivons une nouvelle méthode pour estimer simultanément la forme 3D et le mouvement 3D d'une scène dynamique à partir de vidéos calibrées filmant la scène depuis divers points de vue. Nous utilisons une approche variationnelle dans l'esprit de travaux précédents de reconstruction tridimensionnelle par stéréovision. Nous représentons une scène dynamique sous forme d'un maillage animé c'est-à-dire un maillage polygonal ayant une connectivité fixe au cours du temps, où les sommets ayant des positions variables dans le temps permettent d'échantillonner les trajectoires des points matériels. Cette représentation par maillage animé a pour intérêt d'assurer un encodage consistant à la fois de la forme et du mouvement. Notre méthode permet de retrouver la forme tridimensionnelle ainsi que le mouvement tridimensionnel de façon dense et précise en optimisant les positions dans le temps des sommets du maillage animé. Cette optimisation consiste à minimiser une énergie fonctionnelle qui incorpore la photo-consistance entre les images et également entre les frames temporels, la régularité de la surface spatio-temporelle et du champ de vitesse. Un des points clés de ce travail est un score de photo-consistance entièrement basé sur les images, et même intégré dans le repère des images et non celui du maillage, qui peut être calculé de manière efficace sur carte graphique et qui permet de traiter la distorsion projective ainsi que les occlusions. Nous démontrons l'efficacité de notre méthode sur plusieurs jeux de données complexes de scènes dynamiques réelles.

Chapitre 3

Etat de l'art de la reconstruction spatio-temporelle

Contents

Au cours de ces dernières années, plusieurs méthodes de génération automatique de modèles spatio-temporels à partir de scènes dynamiques réelles capturées par plusieurs vidéos, ont été proposées [APSK07, AdAT+05, ATR+08, dAST+08, dATSS07, FP08, GM04, NA02, PKF07, SH07, TAL+07, VZBH08, VBK05, VBSK00, VBMP08]. En particulier, les plus récentes se sont révélées efficaces pour la capture du mouvement de corps humains sans nécessiter de recours aux méthodes basées sur les marqueurs. Ces méthodes donnent souvent des résultats visuels impressionnants. Cependant, si l'on s'intéresse plus en détail aux méthodes citées ci-dessus, on se rend compte que très peu d'entre elles permettent d'obtenir une *estimation dense, précise et couplée de la forme 3D et du mouvement 3D*.

3.1 Forme 3D précise.

De nombreuses techniques récentes produisent encore une géométrie approximative : les méthodes basées sur la déformation libre de modèles humains [AdAT+05, TAL+07, VBMP08], sur les silhouettes [APSK07, ATR+08, VBMP08], ou sur la déformation Laplacienne d'un scan laser 3D d'une pose initiale [dAST+08, dATSS07]. Ces méthodes sont incapables de retrouver des détails géométriques fins tels que les expressions faciales ou les plis des vêtements.

3.2 Estimation précise du mouvement 3D

L'estimation précise du mouvement 3D est cruciale dans certaines applications telles que le transfert de mouvement, et l'interpolation temporelle. Aussi, une estimation précise du mouvement suppose de forcer une cohérence temporelle au cours de l'estimation couplée de la forme et du mouvement. Cependant, pour la plupart des techniques performantes de capture de mouvement, le flux tridimensionnel de la scène (3D scene flow) [VB05], c'est-à-dire l'estimation dense du champ de vitesse de la scène, n'est pas calculé de façon précise. Souvent, il est interpolé à partir de correspondances 3D parcimonieuses [ATR+08, dAST+08, VZBH08]. Certaines méthodes ne s'intéressent même pas du tout à l'estimation du mouvement tridimensionnel : [GM04] utilise une représentation 4D par level set qui, en plus de son coût de calcul élevé et de son utilisation importante de mémoire, n'encode pas les correspondances 3D dans le temps. [SH07, VBMP08] produit des maillages animés mais, en dépit des apparences, les correspondances 3D sont en fait purement artefactuelles.

3.3 Estimation couplée de la forme 3D et du mouvement 3D

L'estimation couplée de la forme 3D et du mouvement 3D permet d'exploiter leurs redondances, et a longtemps été reconnue [ZK00] comme un moyen souhaité pour améliorer leurs performances. Cependant, la plupart des méthodes de capture de mouvement non basées sur des marqueurs ne parviennent pas à intégrer simultanément les contraintes spatiales et temporelles. Dans [ATR+08, VZBH08, VBK05], la forme est calculée indépendamment à chaque étape temporelle, avant l'estimation du mouvement. Dans [PKF07], la forme et le mouvement sont cal-

culés de façon séquentielle, et non simultanément. Dans [dATSS07], un maillage initial est propagé en cohérence avec le flux de scène 3D, sous contraintes de silhouettes, mais sans recours aux avantages de la stéréovision ; il en résulte que cette méthode dévie dans le temps. Cet inconvénient est atténué dans [dAST+08] en substituant des correspondances 3D parcimonieuses au flux de scène 3D dense, mais ni la forme ni le mouvement ne sont suffisamment précis pour permettre une véritable cohérence spatio-temporelle. Dans [APSK07, GM04], un certain niveau de cohérence spatio-temporelle est obtenu par des représentations 4D, mais dans la mesure où ces représentations n'encodent pas les correspondances dans le temps, elles n'exploitent pas les mises en correspondances et similarités entre différents frames. Dans [VBSK00], la forme et le mouvement sont calculés simultanément en utilisant un algorithme de modelage (plane-sweep carving) dans un espace 6D, mais cette approche a un coût en mémoire et en temps de calcul très important, est limitée à deux frames, et ne peut forcer la régularité de la forme et du champ de vitesse.

Ainsi, à notre connaissance, seulement deux méthodes [FP08, NA02] parviennent à obtenir cette estimation, tant désirée, dense, précise et couplée de la forme 3D et du mouvement 3D. Dans [NA02], la forme et le mouvement sont représentés par le biais de coefficients de subdivision de surface variables dans le temps. Ces coefficients sont calculés en optimisant simultanément une photo-consistance à la fois entre les images et entre les frames temporels. Cependant, la non linéarité de la représentation multi-résolution choisie rend cette optimisation compliquée. De plus, l'estimation nécessaire du mouvement initial utilise les dérivées spatio-temporelles des images en entrée, ce qui rend cette méthode applicable principalement à des scènes totalement lambertiennes avec un mouvement lent et une illumination constante.

[FP08, PSDB+10] sont à notre connaissance les principaux travaux à ce jour permettant de traiter des scènes dynamiques, réelles et complexes. Malgré l'efficacité de la méthode présentée dans [FP08], nous pensons que la méthodologie par expansion utilisée, ne permet pas de prendre en compte la visibilité complète dépendant de "patchs" occluants n'ayant pas encore été calculés.

Le travail de [PSDB+10] est constitué d'un pipeline, dans un premier temps les maillages sont construits indépendamment, puis les maillages sont débruités selon une méthode développée dans des travaux précédents [BBH08]. Enfin une stratégie par expansion est utilisée pour mettre les maillages en correspondance. Pour celà, le fait que les distances intrinsèques doivent se conserver sous une hypothèse de mouvements peu élastiques, est utilisé. Ainsi le débruitage des maillages effectué à l'étape précédente prend toute son importance dans la mesure où un bruit important dans les maillages ne permet plus d'utiliser une hypothèse de conservation des distances intrinsèques dans le temps. Ce pipeline permet d'obtenir des résultats visuellement impressionnants tout en gérant les changements de topologies et les déplacements importants entre les frames temporels. Néanmoins, il n'y a pas une énergie basée sur les images qui soit minimisée de manière globale dans cette méthode. Aussi, ce travail pourrait être utilisé comme initialisation pouvant être raffinée avec notre procédure de minimisation d'énergie entièrement basée sur les images. Dans le section 4.5, nous proposons un pipeline qui pourrait permettre de gérer les grands déplacements, il est clair que le travail de [PSDB+10] pourrait également être

utilisé pour les premières étapes de ce pipeline.

Dans ce manuscrit, nous proposons une méthode permettant de retrouver simultanément et précisément la forme 3D ainsi que le champ de vitesse 3D d'une scène dynamique capturée par des caméras depuis différents points de vue. En premier lieu, nous utilisons une **approche variationnelle** dans l'esprit de travaux précédents en stéréovision et estimation de scene flow [PKF07, ZK00, HD07, PKFH, WRV+08, ZK01]. Telles qu'elles ont été conçues, aucune de ces méthodes ne couvre nos besoins : La plupart sont limitées à une carte de profondeur de la scène variant dans le temps [ZK00, HD07, PKFH, WRV+08, ZK01], tandis que les autres n'imposent pas les contraintes de cohérence spatio-temporelle [PKF07, PKFH].

En second lieu, nous adoptons un **maillage animé** pour représenter une scène dynamique par le biais d'un maillage polygonal ayant une connectivité fixe dans le temps, et dont les sommets ont des positions variables dans le temps échantillonnant la trajectoire des points matériels. Cette représentation par maillage animé a pour intérêt d'assurer un encodage consistant à la fois de la forme et du mouvement. Notre méthode permet de retrouver la forme tridimensionnelle ainsi que le mouvement tridimensionnel. Elle est largement utilisée en vision par ordinateur, et tout particulièrement en animation par ordinateur. Elle est également populaire pour les méthodes performantes de capture de mouvement à partir de vidéos [ATR+08, dAST+08, dATSS07, FP08, SH07, TAL+07, VBMP08] ou à partir de nuages de points mobiles dans le temps [SWG08, WJH+07] (Ces nuages de points étant obtenus à partir de vidéos ou d'un scanner laser 3D).

Notre méthode permet de retrouver la forme tridimensionnelle ainsi que le mouvement tridimensionnel de façon dense et précise en optimisant les positions dans le temps des sommets du maillage animé. Cette optimisation consiste à minimiser une énergie fonctionnelle qui incorpore la photo-consistance entre les images et également entre les frames temporels, la régularité de la surface spatio-temporelle et du champ de vitesse. Un des points clés de ce travail est un score de photo-consistance (basé sur la corrélation) entièrement basé sur les images dans l'esprit de [PKF07], et même intégré dans le repère des images (et non celui du maillage), qui peut être calculé de manière efficace sur carte graphique et qui permet de traiter la distorsion projective ainsi que les occlusions. Un intérêt d'effectuer l'intégrale de l'énergie dans la grille pixellique des images, est que cette grille reste fixe et que l'on contrôle parfaitement la taille de la fenêtre de corrélation dans les images. En effet lorsque les méthodes sons basées sur une corrélation effectuée sur des points échantillonnés sur le maillage polygonal, la taille de la fenêtre de corrélation reprojetée dans les images peut être de l'ordre de un pixel si les polygones sont petits, et alors la corrélation n'aura plus de sens et vaudra toujours environ un. Dans le cas de zones où les polygones sont grands, la taille de la fenêtre de corrélation reprojetée dans les images est très grande alors les points échantillonnés sur le maillage seront trop éloignés dans l'espace et là encore la mesure de corrélation perdra du sens. Il est toujours possible de pallier ce problème en faisant varier la taille de la zone utilisée autour d'un sommet, mais cette taille variera d'une itération à l'autre, et cette formulation n'est donc pas très bien posée.

Chapitre 4

Approche proposée

Contents

4.1 Discrétiser puis optimiser

La majorité des méthodes variationnelles dans ce domaine [PKF07, ZK00, HD07, PKFH, ZK01] et plus généralement en vision par ordinateur, sont du type *optimiser puis discrétiser* : On considère une fonctionnelle d'énergie dépendant d'une représentation spatio-temporelle continue et de dimension infinie, le gradient de cette énergie est calculé de façon analytique, puis le flux obtenu est discrétisé.

Au contraire, nous utilisons une approche de type *discrétiser puis optimiser* : nous définissons une fonction d'énergie dépendant d'une représentation spatio-temporelle discrète et de dimension finie, puis nous utilisons un outil classique d'optimisation non convexe, la descente de gradient. Les avantages de cette approche ont depuis longtemps été démontrés en gestion et opérations sur les maillages, mais ont rarement été démontrés en vision par ordinateur [DPG+08, SU05, VKLP09]. Ainsi, le choix d'une représentation spatio-temporelle discrète et bien choisie, est central dans notre travail.

4.2 Représentation par maillage animé

Dans notre contexte, les maillages animés et polygonaux présentent de nombreux avantages. Comparés à des maillages déconnectés les uns des autres dans le temps, ils sont plus compacts, plus faciles à stocker et à manipuler. Ils donnent un accès direct à la fois à la forme de la scène à un instant donné, et aux trajectoires des points de la scène. La forme 3D et le mouvement 3D sont mutuellement consistants par construction.

Plus particulièrement nous adoptons un maillage triangulé ayant une connectivité fixe dans le temps, cette représentation est très pratique pour le stockage et les calculs sur carte graphique en utilisant les technologies OpenGl et Cg.

Sa connectivité fixe peut être considérée comme étant une limitation, comme pointé dans [VZBH08]. Nous estimons que ce n'est pas le cas, dans la mesure où le corps humain a une topologie sphérique constante. Il est douteux de considérer une personne posant ses mains sur ses hanches comme un objet de genus 2 tel qu'un tore. Cette personne devrait davantage être considérée comme un objet ayant la topologie d'une sphère avec des régions de contact temporaires.

De plus, notons que notre méthode n'est pas limitée à une topologie sphérique : si la topologie du maillage animé est constante dans le temps, nous pouvons la modifier durant la procédure d'optimisation en utilisant une version spatio-temporelle des modèles déformables de Delaunay [PB07].

4.3 Reconstruction par minimisation d'une énergie

4.3.1 Formulation de l'énergie

Pour la suite, nous considérons une scène dynamique, mise en images par N vidéos séquences calibrées et synchronisées, composées de T frames. La scène dynamique est représen-

tée par un maillage triangulé animé composé de K sommets. Nous notons :

- $I_{i,t} : \Omega_i \subset \mathbb{R}^2 \rightarrow \mathbb{R}^d$, $i \in \{1..N\}$, $t \in \{1..T\}$ les images d'entrée. En pratique $d = 1$ pour des images en noir et blanc et $d = 3$ pour des images en couleur.
- $\mathbf{X} = \{\mathbf{x}_{k,t}, k \in \{1..K\}, t \in \{1..T\}\}$ les positions 3D des sommets du maillage animé aux différents instants temporels,
- $\mathbf{X}_t = \{\mathbf{x}_{k,t}, k \in \{1..K\}\}$ le $t^{\text{ième}}$ maillage animé à un instant donné t, ou une coupe temporelle du maillage.

Pour la suite, par un léger abus de notation, nous utiliserons tour à tour $\mathbf{X}$ ou $\mathbf{X}_t$ pour représenter le maillage animé ou les positions de ses sommets.

L'énergie à minimiser par rapport à $\mathbf{X}$ est composée d'un terme d'attache aux données, d'un terme de régularisation de la surface des coupes temporelles du maillage animé et d'un terme de régularisation du champ de vitesse :

$$E(\mathbf{X}) = E_D(\mathbf{X}) + \lambda_S E_S(\mathbf{X}) + \lambda_V E_V(\mathbf{X}) \ . \tag{4.1}$$

Nous minimisons l'énergie ci-dessus en utilisant une descente de gradient standard par rapport aux positions spatio-temporelles $\mathbf{X}$. Afin d'éviter de tomber dans des minimums locaux, nous adoptons un schéma chronologique et une approche multi-résolution. Dans un premier temps, nous optimisons les trois premiers frames ensemble à une version basse résolution du maillage ainsi que des images. Puis nous initialisons la coupe temporelle suivante du maillage animé en extrapolant ses positions 3D à partir des vitesses et des accélérations des sommets des maillages aux instants précédents. Nous ajoutons de façon itérative des coupes temporelles du maillage, et optimisons la séquence en utilisant une fenêtre temporelle glissante de quelques frames, jusqu'à ce que nous ayons reconstruit la séquence temporelle complète à basse résolution. Nous appliquons la procédure décrite ci-dessus de manière itérative en augmentant la résolution des images et du maillage, jusqu'à ce que nous atteignions la précision souhaitée. A la plus basse résolution nous créons et ajoutons des coupes temporelles, mais pour les résolutions suivantes nous ne faisons que raffiner le maillage animé en utilisant la fenêtre temporelle glissante de quelques frames. Afin d'augmenter ou de diminuer la résolution du maillage, nous calculons une longueur caractéristique du maillage devant correspondre à quelques pixels en moyenne dans les images reprojetées (cette distance pixellique dans les images variant suivant la résolution des images utilisées). Si une arête est plus grande que cette distance caractéristique à au moins un instant nous ajoutons un point au milieu de l'arête. Si une arête est plus petite que cette longueur caractéristique à tous les instants, nous supprimons un des points de cette arête. Enfin à partir de ce nouveau nuage de points nous calculons la triangulation de Delaunay, puis la surface de topologie correcte suivant une version spatio-temporelle de [PB07] expliquée en Annexe D.

4.3.1.1 Lissage de la surface

E_S favorise la régularité de la surface des coupes temporelles du maillage animé. Nous utilisons l'aire totale du maillage animé. La minimisation de ce terme par descente de gradient

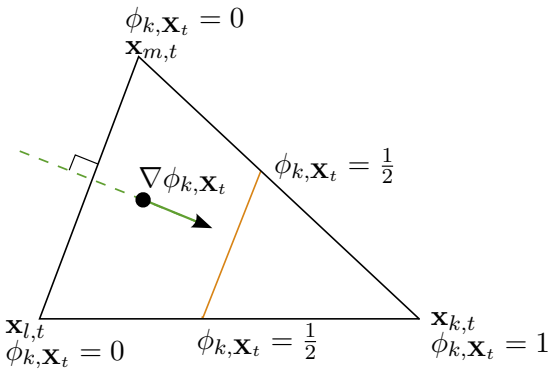


FIG. 4.1 – *Représentation par éléments finis sur une facette (k, l, m) du maillage animé.*

mène à une version discrète du mouvement par courbure moyenne, que nous implémentons suivant la méthode décrite dans [KCVS98].

4.3.1.2 Lissage de la vitesse

E_V pénalise les variations rapides du champ de vitesse le long du maillage animé. Il s'agit de l'intégrale des normes au carré L^2 des gradients intrinsèques, le long de la surface, des vitesses de l'ensemble du maillage animé. La description détaillée de ce terme est effectuée en section 4.3.2.1.

4.3.1.3 Attache au données

E_D encourage la consistance photo-métrique entre les images et entre les frames temporels. Elle est définie comme la somme suivant les paires de caméras (i, j) et les paires d'instants temporels (t, u) d'une mesure de dissimilarité entre les images $I_{i,t}$ et la reprojection de l'image $I_{j,u}$ par le biais du maillage animé.

La définition formelle de E_D et de sont gradient analytique nécessite quelques notations supplémentaires. La projection effectuée par la caméra i est notée $\Pi_i : \mathbb{R}^3 \rightarrow \mathbb{R}^2$. Notre méthode prend en compte la visibilité des points de la surface. Nous notons $\mathbf{X}_{i,t}$ comme étant la partie visible dans l'image i de la coupe temporelle $\mathbf{X}_t$. La back-projection d'un point de la caméra i sur le maillage animé à l'instant t est notée $\Pi^{-1}_{i,\mathbf{X}_t} : \Pi_i(\mathbf{X}_t) \rightarrow \mathbf{X}_{i,t}$.

Nous définissons également une fonction de transport 3D $T_{\mathbf{X}_t \rightharpoonup \mathbf{X}_u}$ qui transporte les points de $\mathbf{X}_t$ vers des points de $\mathbf{X}_u$. Ceci peut s'écrire de façon formelle en utilisant la représentation linéaire par éléments finis décrite dans la figure 4.1. Pour chaque sommet k du maillage animé à un instant donné t, nous définissons une fonction de base $\phi_{k,\mathbf{X}_t}$ telle que (i) $\phi_{k,\mathbf{X}_t}(\mathbf{x}_{k,t}) = 1$ (ii) $\forall l \neq k$, $\phi_{k,\mathbf{X}_t}(\mathbf{x}_{l,t}) = 0$ (iii) $\phi_{k,\mathbf{X}_t}$ varie linéairement à l'intérieur des facettes triangulaires adjacentes au $k^{\text{ième}}$ sommet, s'annule à l'extérieur de cette couronne.

$$T_{\mathbf{X}_t \rightharpoonup \mathbf{X}_u} = \sum_k \mathbf{x}_{k,u}\, \phi_{k,\mathbf{X}_t} \ . \tag{4.2}$$

De façon plus simple nous pouvons dire que la back-projection Y_t du pixel p_i se situe à l'intérieur de la facette f et a pour coordonnées barycentriques $\phi_{k,\mathbf{X}_t}(Y_t)$, $\phi_{l,\mathbf{X}_t}(Y_t)$, $\phi_{m,\mathbf{X}_t}(Y_t)$

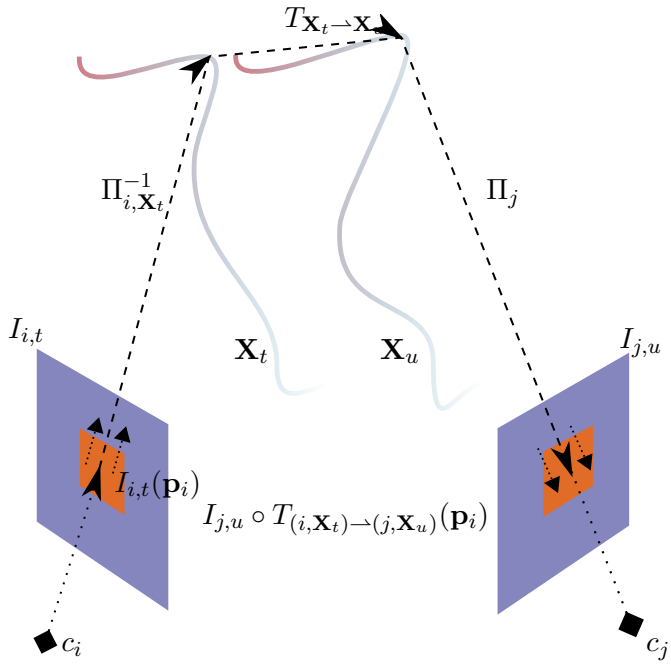


FIG. 4.2 – *Reprojection de l'image j au temps u dans l'image i au temps t par le biais du maillage animé.*

à l'instant t, k, l, m étant les sommets de la facette f. Aussi la position de cette particule à l'instant u est $Y_u = \sum_{l \in f} \mathbf{x}_{l,u}\, \phi_{l,\mathbf{X}_t}(Y_t)$, soit $Y_u = \sum_{k \in \mathbf{X}} \mathbf{x}_{k,u}\, \phi_{k,\mathbf{X}_t}(Y_t)$ dans la mesure où comme nous l'avons défini plus haut $\phi_{k,\mathbf{X}_t}(Y_t)$ s'annule si le sommet k n'appartient pas à la facette f.

Finalement, nous pouvons écrire la fonction de transport d'image $T_{(i,\mathbf{X}_t) \rightarrow (j,\mathbf{X}_u)}$ qui transforme les positions dans l'image $I_{i,t}$ vers les positions dans l'image $I_{j,u}$ par le biais de la back-projection, du maillage animé et de la reprojection :

$$T_{(i,\mathbf{X}_t) \rightarrow (j,\mathbf{X}_u)} = \Pi_j \circ T_{\mathbf{X}_t \rightarrow \mathbf{X}_u} \circ \Pi^{-1}_{i,\mathbf{X}_t} \,. \tag{4.3}$$

Avec ces notations en main la reprojection de l'image j au temps u dans l'image i au temps t par le biais du maillage animé s'écrit $I_{j,u} \circ T_{(i,\mathbf{X}_t) \rightarrow (j,\mathbf{X}_u)}$. Ceci est illustré en figure 4.2.

Le terme d'attache aux données est la somme suivant les paires de caméras (i, j) et les paires orientées de frames temporels (t, u) d'une mesure de dissimilarité M entre l'image $I_{i,t}$ et la reprojection définie ci-dessus de $I_{j,u}$ par le biais du maillage animé. Cette dissimilarité n'est calculée que dans les régions du plan image de la caméra i où les deux images sont définies, c'est-à-dire après avoir retiré les régions occluses. Cette région visible correspond à l'ensemble des pixels p_i dans le cadre de l'image i qui se back-projette sur la surface au temps t en une particule P, et dont cette particule une fois transportée au temps u est visible dans l'image $I_{j,u}$ (ni occluse, ni en dehors du cadre de l'image). Cette région sera notée $\Omega_{i,j,t,u}$. Cette région visible est calculée avant chaque étape d'optimisation sur carte graphique. Pour plus de concision, nous omettons ce terme de visibilité pour la suite des calculs et intégrons sur l'image entière Ω_i par abus de notation, c'est-à-dire sur tous les pixels de l'image. Nous avons l'équation

suivante :

$$E_D(\mathbf{X}) = \sum_{i,j} \sum_{t,u} \int_{\Omega_i} M\left[I_{i,t}\ ,\ I_{j,u} \circ T_{(i,\mathbf{X}_t)\rightarrow(j,\mathbf{X}_u)}\right]\ . \tag{4.4}$$

4.3.2 Calcul du gradient

4.3.2.1 Gradient de l'énergie de lissage de la vitesse

Le champ de vitesse est explicitement encodé par le maillage animé $\mathbf{X}$. Plus particulièrement, il s'agit d'une fonction continue et linéaire par morceaux de $\mathbf{X}_t \rightarrow \mathbb{R}^3$ définie par

$$\mathbf{v}_{\mathbf{X},t}(\mathbf{x}) = T_{\mathbf{X}_t \rightarrow \mathbf{X}_{t+1}}(\mathbf{x}) - \mathbf{x}\ , \tag{4.5}$$

ou de façon équivalente par

$$\mathbf{v}_{\mathbf{X},t} = \sum_k (\mathbf{x}_{k,t+1} - \mathbf{x}_{k,t})\, \phi_{k,\mathbf{X}_t}\ . \tag{4.6}$$

L'énergie de régularisation du champ de vitesse s'écrit :

$$E_V(\mathbf{X}) = \sum_t \int_{\mathbf{X}_t} \|\nabla \mathbf{v}_{\mathbf{X},t}(\mathbf{x})\|^2 d\mathbf{x}\ . \tag{4.7}$$

Afin de simplifier cette expression, remarquons que $\nabla \phi_{k,\mathbf{X}_t}$ est constant à l'intérieur de chaque facette triangulaire f de $\mathbf{X}_t$ et vaut $\frac{\mathbf{h}_{k,f}}{\|\mathbf{h}_{k,f}\|^2}$, où $\mathbf{h}_{k,f}$ est la hauteur du triangle passant par le sommet k. Ceci peut être très facilement vérifié et est illustré en figure 4.1. A_f étant l'aire de la facette f, l'énergie de régularisation devient

$$E_V(\mathbf{X}) = \sum_t \sum_{f \in \mathbf{X}_t} A_f \left\| \sum_{k \in f} \frac{\mathbf{h}_{k,f}}{\|\mathbf{h}_{k,f}\|^2} (\mathbf{x}_{k,t+1} - \mathbf{x}_{k,t}) \right\|^2\ . \tag{4.8}$$

En développant ce terme et en remarquant que la longueur de $\mathbf{h}_{k,f}$ est liée à l'aire, puis en négligeant les variations de l'aire par rapport aux déplacements des sommets, les dérivées partielles $\frac{\partial E_V}{\partial \mathbf{x}_{k,t}}$ de cette énergie peuvent facilement être obtenues. Nous négligeons certains termes, car minimiser l'aire serait redondant avec l'énergie de lissage de la surface. Nous ne développons pas le calcul car il est très facile, mais nous développons dans la section suivante le calcul du gradient de l'énergie d'attache aux données, qui est plus complexe et présente toutes les techniques nécessitées pour le gradient de l'énergie de régularisation du champ de vitesse.

4.3.2.2 Gradient de l'énergie d'attache aux données : calcul formel

Nous calculons maintenant les dérivées partielles de cette énergie suivant les variations de la position d'un sommet $\mathbf{x}_{k,t}$ à un instant donné dans le maillage animé. Tout d'abord, remarquons que les seules paires orientées d'instants concernées par cette variation sont (u,t) and (t,u), $u \in \{1..T\}$. De plus, lorsque le maillage animé bouge, les images reprojetées changent. Ainsi les dérivées partielles de l'énergie E_D impliquent les dérivées de la mesure de dissimilarité M par rapport au second argument (la première image restant fixe). Nous noterons cette dérivée $\partial_2 M$.

En utilisant la dérivation de fonctions composées, et après quelques manipulations d'indexes, nous avons :

$$\frac{\partial E_D}{\partial \mathbf{x}_{k,t}} = \sum_{i,j} \sum_{u} \sum_{k \in \mathbf{X}} \int_{\Omega_i} \partial_2 M \left[I_{i,t} \, , \, I_{j,u} \circ T_{(i,\mathbf{x}_t) \rightharpoonup (j,\mathbf{x}_u)} \right] DI_{j,u} \, \frac{\partial T_{(i,\mathbf{x}_t) \rightharpoonup (j,\mathbf{x}_u)}}{\partial \mathbf{x}_{k,t}} (\mathbf{p}_i) \, d\mathbf{p}_i$$
$$+ \int_{\Omega_j} \partial_2 M \left[I_{j,u} \, , \, I_{i,t} \circ T_{(j,\mathbf{x}_u) \rightharpoonup (i,\mathbf{x}_t)} \right] DI_{i,t} \, \frac{\partial T_{(j,\mathbf{x}_u) \rightharpoonup (i,\mathbf{x}_t)}}{\partial \mathbf{x}_{k,t}} (\mathbf{p}_j) \, d\mathbf{p}_j \ , \quad (4.9)$$

où $DI_{.,.}$ représente les matrices jacobiennes des images d'entrées, c'est-à-dire les gradients des images par rapport à l'horizontale et à la verticale, et $\mathbf{p}_i$ représente un pixel du domaine de l'image Ω_i. Pour plus de concision, nous avons omis les points où certaines fonctions sont évaluées.

En observant plus attentivement les dérivées $\frac{\partial T_{...}}{\partial \mathbf{x}_{k,t}}$, nous pouvons noter plusieurs points. En premier lieu, elles sont purement géométriques, c'est-à-dire indépendantes des données dans les images. D'autre part, le pixel $\mathbf{p}_i$ se back-projetant sur la facette f, ces quantités s'annulent si k n'est pas un sommet de f. Ainsi, étant donné un sommet k l'intégration n'est effectuée que pour les pixels se back-projetant sur la couronne de k, c'est-à-dire, sur les facettes adjacentes au sommet k. Enfin, ces quantités impliquent les normales aux facettes triangulaires se projetant sur le pixel p_i, ainsi que les coordonnées barycentriques de la back-projection du pixel p_i sur cette facette. Les expressions analytiques complètes peuvent être obtenues en effectuant des raisonnements géométriques non triviaux. Nous proposons ci-dessous les calculs formels détaillés, ainsi qu'une résolution intuitive. Les calculs formels consistent essentiellement à calculer les variations des cordonnées barycentriques des pixels back-projetés, pour des petites variations du maillage spatio-temporel animé.

La back-projection sur $\mathbf{X}$ du pixel $\mathbf{p}_i$ dans l'image i s'écrit :

$$\Pi^{-1}_{i,\mathbf{X}_t} (p_i) = \sum_k \mathbf{x}_{k,t} \, \phi_{k,\mathbf{X}_t}$$

Après une petite perturbation $\delta\mathbf{X}$ de la surface spatio-temporelle, le point back-projeté change dans la direction du rayon optique $\mathbf{d}_i$, ainsi nous obtenons :

$$\Pi^{-1}_{i,\mathbf{X}_t+\delta X_t} (p_i) = \sum_k \left(\mathbf{x}_{k,t} + \delta \mathbf{x}_{\mathbf{k},\mathbf{t}} \right) \, \phi_{k,\mathbf{X}_t+\delta X_t}$$

Et α étant inconnu :

$$\sum_k \mathbf{x}_{k,t} \, \phi_{k,\mathbf{X}_t} + \alpha \mathbf{d}_i = \sum_k \left(\mathbf{x}_{k,t} + \delta \mathbf{x}_{\mathbf{k},\mathbf{t}} \right) \, \phi_{k,\mathbf{X}_t+\delta X_t}$$

En d'autres termes,

$$\alpha \mathbf{d}_i = \sum_k \mathbf{x}_{k,t} \, \left(\phi_{k,\mathbf{X}_t+\delta X_t} - \phi_{k,\mathbf{X}_t} \right) + \sum_k \delta \mathbf{x}_{\mathbf{k},\mathbf{t}} \, \phi_{k,\mathbf{X}_t+\delta X_t}$$

Dans la mesure où $\sum_k \mathbf{x}_{k,t} \, \left(\phi_{k,\mathbf{X}_t+\delta X_t} - \phi_{k,\mathbf{X}_t} \right)$ est un vecteur le long de la surface, alors son produit scalaire avec la normale à la facette est nul, ainsi :

$$\alpha = \sum_k \phi_{k,\mathbf{X}_t+\delta X_t} \frac{\mathbf{N}\cdot\delta\mathbf{x_{k,t}}}{\mathbf{N}\cdot\mathbf{d}_i}$$

En notant $\mathbf{K}_{k,t} = \delta\mathbf{x_{k,t}} - \frac{\mathbf{N}\cdot\delta\mathbf{x_{k,t}}}{\mathbf{N}\cdot\mathbf{d}_i}\mathbf{d}_i$ nous avons les équations suivantes :

$$\sum_k (\mathbf{x}_{k,t} + \mathbf{K}_{k,t})\ \phi_{k,\mathbf{X}_t+\delta X_t} = \sum_k \mathbf{x}_{k,t}\,\phi_{k,\mathbf{X}_t}$$

Comme la back-projection est à l'intérieur de la facette f nous avons $\sum_{k\in f}\phi_{k,\mathbf{X}_t} = 1$, on peut donc remarquer que :

$$\phi_{1,\mathbf{X}_t}(\mathbf{x}_{1,t}-\mathbf{x}_{3,t}) + \phi_{2,\mathbf{X}_t}(\mathbf{x}_{2,t}-\mathbf{x}_{3,t}) + \mathbf{x}_{3,t} = \sum_k \mathbf{x}_{k,t}\,\phi_{k,\mathbf{X}_t}$$

Et si nous considérons de petits déplacements, la back-projection après perturbations reste toujours à l'intérieur de la même facette f et avec les même considérations, à savoir que la somme des coefficients barycentriques vaut un, on peut écrire :

$$\sum_{k=1,2} \phi_{k,\mathbf{X}_t}(\mathbf{x}_{k,t}-\mathbf{x}_{3,t}) = \sum_{k=1,2} \phi_{k,\mathbf{X}_t+\delta X_t}\left[(\mathbf{x}_{k,t}-\mathbf{x}_{3,t}) + (\mathbf{K}_{k,t}-\mathbf{K}_{3,t})\right] \tag{4.10}$$

Notre objectif est de calculer les inconnues $\phi_{k,\mathbf{X}_t+\delta X_t}$ qui permettent de retrouver les positions des particules correspondant aux back-projections des pixels sur le maillage après une petite perturbation. Pour trouver $\phi_{2,\mathbf{X}_t+\delta X_t}$ nous n'avons qu'à calculer le produit vectoriel des parties gauche et droite de l'équation 4.10 avec le vecteur $(\mathbf{x}_{1,t}-\mathbf{x}_{3,t}) + (\mathbf{K}_{2,t}-\mathbf{K}_{3,t})$ pour obtenir finalement par un développement limité à l'ordre un $\phi_{2,\mathbf{X}_t+\delta X_t}$ et plus généralement $\phi_{k,\mathbf{X}_t+\delta X_t}$, avec des produits vectoriels et des symétries de raisonnement :

$$\phi_{k,\mathbf{X}_t+\delta X_t} = \phi_{k,\mathbf{X}_t} + \frac{\left[(\mathbf{x}_{k-1,t}-\mathbf{x}_{k+1,t})\times(\sum_v \phi_{v,\mathbf{X}_t}\mathbf{K}_{v,t})\right]\cdot\mathbf{N}}{\left[(\mathbf{x}_{k,t}-\mathbf{x}_{k+1,t})\times(\mathbf{x}_{k-1,t}-\mathbf{x}_{k+1,t})\right]\cdot\mathbf{N}}$$

Au temps u la variation de positions avant et après perturbations est :

$$\mathbf{\Delta X} = \sum_k (\mathbf{x}_{k,u} + \delta\mathbf{x_{k,u}})\ \phi_{k,\mathbf{X}_t+\delta X_t} - \sum_k \mathbf{x}_{k,u}\,\phi_{k,\mathbf{X}_t}$$

Aussi en développant la formule avec la valeur de $\mathbf{K}_{k,t}$ en fonction de $\delta\mathbf{x_{k,t}}$ et avec une reformulation simple étant donné que $\mathbf{a}\cdot(\mathbf{c}\times\mathbf{b}) = -\mathbf{b}\cdot(\mathbf{c}\times\mathbf{a})$ il est facile de vérifier que l'on obtient :

$$\begin{aligned}\mathbf{\Delta X} \quad &= \textstyle\sum_k \delta\mathbf{x_{k,u}}\,\phi_{k,\mathbf{X}_t} \\ &+ \textstyle\sum_k \left(\sum_v \delta\mathbf{x_{v,t}}\,\phi_{v,\mathbf{X}_t}\right)\cdot \frac{\mathbf{N}\times\left(\mathbf{x}_{k-1,t}-\mathbf{x}_{k+1,t}\right)}{\left[\left(\mathbf{x}_{k,t}-\mathbf{x}_{k+1,t}\right)\times\left(\mathbf{x}_{k-1,t}-\mathbf{x}_{k+1,t}\right)\right]\cdot\mathbf{N}}\,\mathbf{x}_{k,u} \\ &- \textstyle\sum_k \left(\sum_v \delta\mathbf{x_{v,t}}\,\phi_{v,\mathbf{X}_t}\right)\cdot \frac{\left[\left(\mathbf{x}_{k-1,t}-\mathbf{x}_{k+1,t}\right)\times\mathbf{d}_i\right]\cdot\mathbf{N}}{\left[\left(\mathbf{x}_{k,t}-\mathbf{x}_{k+1,t}\right)\times\left(\mathbf{x}_{k-1,t}-\mathbf{x}_{k+1,t}\right)\right]\cdot\mathbf{N}}\frac{\mathbf{N}}{\mathbf{N}\cdot\mathbf{d}_i}\,\mathbf{x}_{k,u}\end{aligned}$$

Toujours avec la formule du triple produit vectoriel $\mathbf{b}(\mathbf{a}\cdot\mathbf{c}) = \mathbf{c}(\mathbf{a}\cdot\mathbf{b}) + \mathbf{a}\times(\mathbf{b}\times\mathbf{c})$, et en notant

$$\mathbf{V_{k,t}} = \frac{\mathbf{N} \times (\mathbf{x}_{k-1,t} - \mathbf{x}_{k+1,t})}{[(\mathbf{x}_{k,t} - \mathbf{x}_{k+1,t}) \times (\mathbf{x}_{k-1,t} - \mathbf{x}_{k+1,t})] \cdot \mathbf{N}},$$

nous en déduisons :

$$\begin{aligned}\mathbf{\Delta X} &= \textstyle\sum_k \delta\mathbf{x_{k,u}}\, \phi_{k,\mathbf{X}_t} \\ &+ \left(\textstyle\sum_k [\mathbf{V_{k,t}}]_\times [\mathbf{x}_{k,u}]_\times + (\mathbf{V_{k,t}} \cdot \mathbf{x}_{k,u})\ Id\right) \left(\textstyle\sum_v \delta\mathbf{x_{v,t}}\, \phi_{v,\mathbf{X}_t}\right) \\ &- \left(\textstyle\sum_k \mathbf{V_{k,t}} \cdot \mathbf{d}_i\, \mathbf{x}_{k,u}\right) \left(\textstyle\sum_v \frac{\phi_{v,\mathbf{X}_t} \delta\mathbf{x_{v,t}} \cdot \mathbf{N}}{\mathbf{N} \cdot \mathbf{d}_i}\right)\end{aligned}$$

C'est-à-dire,

$$\begin{aligned}\mathbf{\Delta X} &= \textstyle\sum_k \delta\mathbf{x_{k,u}}\, \phi_{k,\mathbf{X}_t} \\ &+ \left(\textstyle\sum_k [\mathbf{V_{k,t}}]_\times [\mathbf{x}_{k,u}]_\times + (\mathbf{V_{k,t}} \cdot \mathbf{x}_{k,u})\ Id\right) \left(\textstyle\sum_v \delta\mathbf{x_{v,t}}\, \phi_{v,\mathbf{X}_t}\right) \\ &- \left(\textstyle\sum_k [\mathbf{V_{k,t}}]_\times [\mathbf{x}_{k,u}]_\times + (\mathbf{V_{k,t}} \cdot \mathbf{x}_{k,u})\ Id\right)\ \mathbf{d}_i \left(\textstyle\sum_v \frac{\phi_{v,\mathbf{X}_t} \delta\mathbf{x_{v,t}} \cdot \mathbf{N}}{\mathbf{N} \cdot \mathbf{d}_i}\right)\end{aligned}$$

En notant la matrice de déformation relative à la facette

$$D_f\,(t,u) = -\left(\sum_k [\mathbf{V_{k,t}}]_\times [\mathbf{x}_{k,u}]_\times + (\mathbf{V_{k,t}} \cdot \mathbf{x}_{k,u})\ Id\right)$$

Nous obtenons une formulation simple :

$$\mathbf{\Delta X} = \sum_v \delta\mathbf{x_{v,u}}\, \phi_{v,\mathbf{X}_t} + D_f\,(t,u) \sum_v \phi_{v,\mathbf{X}_t} \left(-\delta\mathbf{x_{v,t}} + \mathbf{d}_i \frac{\delta\mathbf{x_{v,t}} \cdot \mathbf{N}}{\mathbf{N} \cdot \mathbf{d}_i}\right)$$

La variation d'énergie correspondant à une petite perturbation de surface $\epsilon\delta$ est :

$$\frac{dE_D}{d\epsilon} = \sum_{i,j} \sum_{t,\mu} \sum_{\mathbf{p}_i \in \Omega_i} \frac{\partial M}{\partial 2}(t,\mu)\, DI_{j,\mu}\, D\Pi_j \mathbf{\Delta X}(\mathbf{p_i},\mu) \tag{4.11}$$

Le point restant fixe dans la première image nous ne considérons que $\frac{\partial M}{\partial 2}(t,\mu)$ la dérivée partielle par rapport au second terme de la mesure de dissimilarité. Le calcul de cette dérivée partielle sera décrit plus en détails dans la prochaine section. En réarrangeant ces sommes et termes de manière adéquate, et en utilisant à nouveau les informations de visibilité, nous obtenons :

$$\frac{dE_D}{d\epsilon} = \sum_t \sum_v \langle \nabla E_D(v,t), \delta\mathbf{x_{v,t}} \rangle$$

Avec les trois gradients suivants :

$$\nabla\mathbf{E_D}(\mathbf{v},\mathbf{t}) = \sum_{i,j} \sum_u \sum_{f \ni v} \delta\mathbf{E_1} + \delta\mathbf{E_2} + \delta\mathbf{E_3}$$

$$\begin{cases} \delta\mathbf{E_1} &= \sum_{\mathbf{p}_i \in \Omega_{f,t,u}} \frac{\phi_{v,\mathbf{X}_t}}{\mathbf{N} \cdot \mathbf{d}_i} \left[\frac{\partial M}{\partial 2}(t,u)\, DI_{j,u}\, D\Pi_j\, D_f(t,u)\, \mathbf{d}_i\right] \mathbf{N} \\ \delta\mathbf{E_2} &= -\sum_{\mathbf{p}_i \in \Omega_{f,t,u}} \phi_{v,\mathbf{X}_t} \left[\frac{\partial M}{\partial 2}(t,u)\, DI_{j,u}\, D\Pi_j\, D_f(t,u)\right]^T \\ \delta\mathbf{E_3} &= \sum_{\mathbf{p}_i \in \Omega_{f,u,t}} \phi_{v,\mathbf{X}_u} \left[\frac{\partial M}{\partial 2}(u,t)\, DI_{j,t}\, D\Pi_j\right]^T \end{cases}$$

Ces équations sont obtenues en notant $\Omega_{f,i,t}$ la région de la facette f reprojetée dans la caméra i au temps t. Et $\Omega_{f,t,u} = \Omega_{f,i,t} \cap \Omega_{i,j,t,u}$

4.3.2.3 Gradient de l'énergie d'attache au données : calcul intuitif

L'objectif de cette section, est, à partir de raisonnements intuitifs, de donner un sens à la matrice $D_f(t,u)$ vue dans la section précédente, et de montrer que cette matrice correspond bien à une matrice de déformation de la facette entre les instants t et u.

Si nous ajoutons une petite perturbation spatio-temporelle $\delta\mathbf{X}$, en notant $\mathbf{d_i}$ le rayon optique depuis le centre de la caméra vers le point back-projeté sur la surface, et $\mathbf{N}$ la normale à la surface le nouveau point back-projeté sera :

$$\Pi^{-1}_{i,(\mathbf{X}_t+\epsilon\delta X_t)^t}(x) = \Pi^{-1}_{i,\mathbf{X}_t}(x) + \frac{\mathbf{N}\cdot\delta}{\mathbf{N}\cdot\mathbf{d_i}}\mathbf{d_i}$$

Ce calcul a été introduit dans [PKF07], et est un simple calcul géométrique pouvant être facilement retrouvé en établissant une figure. Nous remarquons que dans cette formule les points dans l'image i restent fixes puisque nous nous déplaçons nécessairement le long du rayon optique. Ceci est nécessaire dans notre cas, dans la mesure où nous intégrons dans la grille des pixels de l'image i. Maintenant nous souhaitons calculer les positions des particules back-projetées, non seulement au temps t (comme effectué ci-dessus) mais également au temps u. Au temps t nous avons calculé une position mais il ne s'agit que de coordonnées, pas d'une particule pouvant être suivie dans le temps. Aussi nous souhaitons identifier à quelle particule, sur le maillage avant perturbation, cette position correspond. Alors, nous pourrons suivre cette particule dans le temps.

Nous observons sur la Figure 4.3 le procédé global pour retrouver et tracer la particule dans le temps. Notre back-projection du pixel x au temps t sur une surface légèrement perturbée, vient de la particule X qui avait au temps t et avant perturbation, les coordonnées suivantes :

$$X(t) = \Pi^{-1}_{i,\mathbf{X}_t}(x) + \frac{\mathbf{N}\cdot\delta}{\mathbf{N}\cdot\mathbf{d_i}}\mathbf{d_i} - \delta$$

Nous souhaitons écrire ces équations dans le maillage comme des coordonnées barycentriques par rapport aux sommets, le point back-projeté s'écrit :

$$\Pi^{-1}_{i,\mathbf{X}_t}(x) = \sum_k \mathbf{x}_{k,t}\,\phi_{k,\mathbf{X}_t}$$

La perturbation au niveau du point back-projeté au temps t, sur la facette f est :

$$\delta = \sum_{v\in f} \delta\mathbf{x}_{\mathbf{v},\mathbf{t}}\,\phi_{v,\mathbf{X}_t}$$

La particule X s'écrit dans le maillage triangulé discret :

$$X(t) = \Pi^{-1}_{i,\mathbf{X}_t}(x) + \sum_{v\in f}\phi_{v,\mathbf{X}_t}\left(\frac{\mathbf{N}\cdot\delta\mathbf{x}_{\mathbf{v},\mathbf{t}}}{\mathbf{N}\cdot\mathbf{d_i}}\mathbf{d_i} - \delta\mathbf{x}_{\mathbf{v},\mathbf{t}}\right)$$

Si l'on considère la variation de position le long de la surface entre, le point X_1 back-projeté sur la surface avant perturbation, et la particule X au temps t nous obtenons :

$$\begin{aligned}\mathbf{\Delta X_{Surf}}(x,t) &= X(t) - \Pi^{-1}_{i,\mathbf{X}_t}(x) \\ &= \sum_{v\in f}\left(\phi_{v,\mathbf{X}_t}\frac{\mathbf{N}\cdot\delta\mathbf{x_{v,t}}}{\mathbf{N}\cdot\mathbf{d_i}}\mathbf{d_i} - \phi_{v,\mathbf{X}_t}\delta\mathbf{x_{v,t}}\right)\end{aligned}$$

Notons $D_f(t,u)$ la matrice de déformation intrinsèque de la facette f entre les instants t et u. Cette matrice de déformation permet d'obtenir le vecteur au temps u si on la multiplie par le vecteur au temps t. Aussi, la nouvelle variation de position au temps u s'écrit :

$$\mathbf{\Delta X_{Surf}}(x,u) = \sum_{v\in f}\phi_{v,\mathbf{X}_t}D_f(t,u)\left(\frac{\mathbf{N}\cdot\delta\mathbf{x_{v,t}}}{\mathbf{N}\cdot\mathbf{d_i}}\mathbf{d_i} - \delta\mathbf{x_{v,t}}\right)$$

Il est maintenant facile, avec un développement limité à l'ordre un, de calculer la variation de position entre les coordonnées de la particule initiale back-projetée avant perturbation puis déplacée au temps u et la position de la particule X sur la surface perturbée puis déplacée au temps u :

$$\begin{aligned}\mathbf{\Delta X}(x,u) &= \mathbf{\Delta X_{Surf}}(x,u) + \sum_{v\in f}\phi_{v,\mathbf{X}_t}\delta\mathbf{x_{v,u}} \\ \mathbf{\Delta X}(x,u) &= \sum_{v\in f}\phi_{v,\mathbf{X}_t}\left[D_f(t,u)\left(\frac{\mathbf{N}\cdot\delta\mathbf{x_{v,t}}}{\mathbf{N}\cdot\mathbf{d_i}}\mathbf{d_i} - \delta\mathbf{x_{v,t}}\right) + \delta\mathbf{x_{v,u}}\right]\end{aligned}$$

Comme on peut le voir, ce raisonnement intuitif mène aux mêmes résultats que dans la section 4.3.2.2, sachant que dans la section précédente nous avons obtenu une formulation analytique pour la matrice de déformation $D_f(t,u)$.

4.3.2.4 Mesure de dissimilarité

Nous avons choisi d'utiliser l'opposé de la corrélation croisée normalisée NCC, qui est une des mesures les plus populaires pour calculer les similarités en stéréovision. Elle permet d'obtenir une forme de robustesse par rapport aux changements de luminosité, de calibration de couleur des caméras, et par rapport au fait que la scène n'est pas parfaitement lambertienne.

Concernant le gradient de l'énergie d'attache aux données utilisé dans les sections précédentes, l'équation détaillée est en fait un peu plus complexe dans la mesure où la variation de la corrélation dans la fenêtre Ω_{x_k} centrée en x_k dépendra du pixel x_k mais également de tous les autres pixels à l'intérieur de la fenêtre de corrélation :

$$\frac{dE_D}{d\epsilon} = \sum_{i,j}\sum_{t,\mu}\sum_{x\in\Omega_i}\sum_{x_k\in\Omega_x}\frac{\partial M_{\Omega_{x_k}}}{\partial 2}(x,t,\mu)\, DI_{j,\mu}\, D\Pi_j\mathbf{\Delta X}(\mathbf{x},\mu)$$

où $\frac{\partial M_{\Omega_{x_k}}}{\partial 2}(x,t,\mu)$ correspond au gradient au pixel x de la corrélation dans la fenêtre centrée en x_k ceci peut être calculé plus simplement en sommant et en posant :

$$\frac{\partial M}{\partial 2}(\mu,t) = \sum_{x_k\in\Omega_x}\frac{\partial M_{\Omega_{x_k}}}{\partial 2}(x,t,\mu)$$

Ce calcul montre que pour une fenêtre de corrélation $k\times k$ pixels, le calcul s'avère k^2 fois plus complexe. Pour la fenêtre utilisée de 9×9 pixels, le calcul devrait être environ 80 fois plus long. En pratique, afin de gagner en temps de calcul, il est possible de ne pas prendre en compte ce qui est écrit ci-dessus, et de ne calculer le gradient de corrélation qu'au centre de la

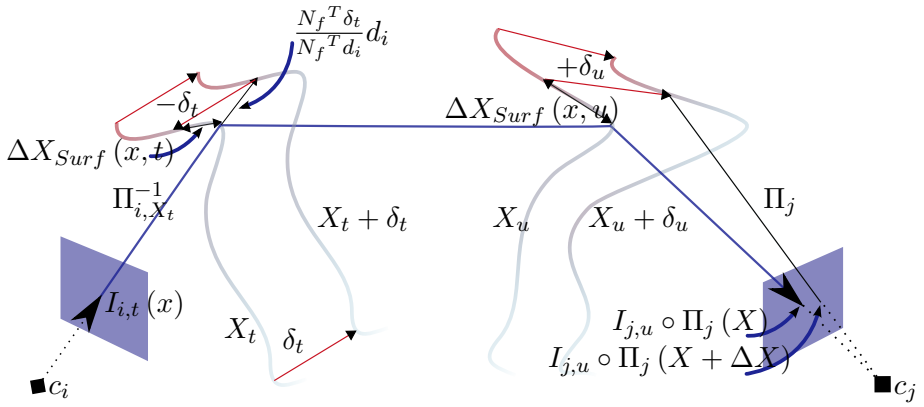


FIG. 4.3 – *Variation de l'intensité dans l'image j pour une perturbation spatio-temporelle de la surface δ*

fenêtre. En revanche le gradient de corrélation utilise des termes qui peuvent être pré-calculés sur carte graphique, de sorte qu'avec une implémentation adéquate et en pré-calculant des bons coefficients pour chaque pixel (variance...), la complexité du calcul reste du même ordre de grandeur et est au pire multipliée par 2. En pratique, nous avons noté que lorsque ce calcul complet du gradient était utilisé, alors la convergence de notre optimisation était plus rapide et produisait de meilleurs résultats.

4.4 Résultats

4.4.1 Implémentation

Le calcul des reprojections des images via le maillage animé, ainsi que le calcul du gradient par sommet de l'énergie d'attache aux données, sont les parties les plus coûteuses de l'algorithme. Ces calculs sont parallèles et ont été efficacement développés sur carte graphique en utilisant OpenGL et le langage Cg.

Pour toutes nos expériences, nous avons choisi l'opposé de la corrélation croisée normalisée comme mesure de dissimilarité M, afin de traiter efficacement les cas où les conditions de luminosité sont variables, et où des ombres peuvent se déplacer dans le temps. Nous utilisons une petite fenêtre de corrélation de taille 9×9 pixels, toujours pour rester local et ne pas être trop gênés par des ombres ou des conditions de luminosité non constantes sur l'objet. Le stockage du maillage animé et le calcul des termes de lissage sont effectués sous l'environnement C++ Computational Geometry Algorithms Library (CGAL)[1]. Ce maillage est également transféré sur la carte graphique, tout comme les images, afin de faire les calculs mentionnés plus haut à l'aide du langage Cg.

Afin de limiter les temps de calculs, les sommes dans l'équation 4.4 sont restreintes à des paires de caméras voisines et à des instants temporels voisins contenus dans une fenêtre temporelle glissante. La taille de cette fenêtre glissante est au départ de 2 frames, puis augmente petit

[1] `http ://www.cgal.org/`

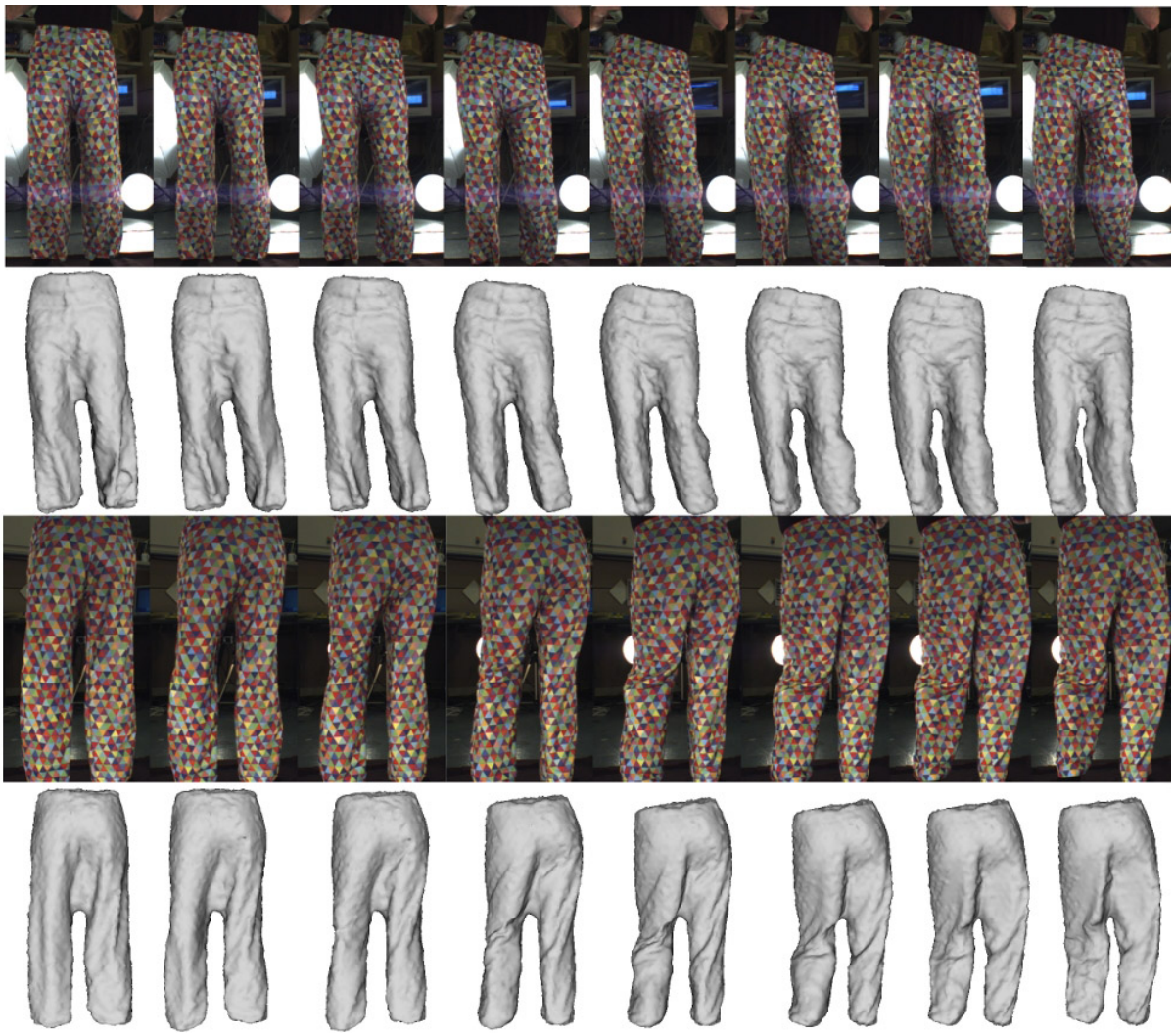


FIG. 4.4 – *Nos résultats pour le jeu de données "Pants". Voir le texte pour de plus amples détails.*

à petit jusqu'à atteindre 4 frames pour nos expériences. Cette réduction du nombre de calculs n'a pas dégradé la qualité des résultats obtenus.

La résolution du maillage est contrôlée à l'aide d'un seuil maximal et minimal de longueur des arêtes : une arête est coupée en deux si elle est plus grande qu'un seuil pour *au moins un* frame temporel ; une arête est "collapsée" si elle est plus petite qu'un seuil pour *tous* les instants.

Si nous souhaitons effectuer des changements de topologie de manière automatique, en pratique nous utilisons une méthode basée sur le fait que l'on ajoute ou retire des sommets à l'aide des seuils maximaux et minimaux, puis nous effectuons une triangulation de Delaunay de ces points, et enfin on retrouve la surface triangulée de topologie correcte par rapport à un instant temporel de référence en utilisant les informations sur la surface avant modifications. Ce procédé est expliqué plus en détail en Annexe D et est une application directe des modèles de Delaunay déformables [PB07]. L'utilisateur peut choisir un instant de référence qui reflète la véritable topologie de la scène : c'est-à-dire dans le cas d'une personne, un instant où les bras et les jambes sont suffisamment dégagés du reste du corps.

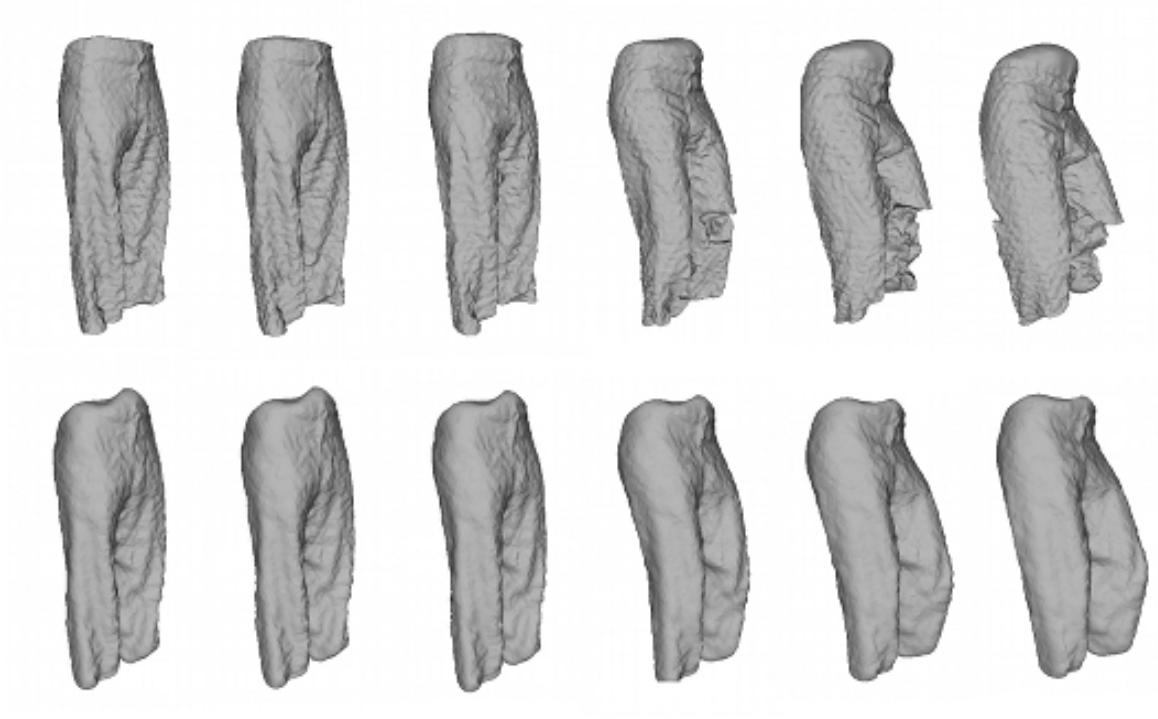


FIG. 4.5 – *Comparaison entre une méthode de stéréovision avec une approche instant par instant (en haut) et notre approche spatio-temporelle (en bas) pour le jeu de données "Pants". Voir le texte pour de plus amples explications.*

4.4.2 Résultats expérimentaux

Le jeu de données "Pants" est composé de 8 caméras. Il nous a été fourni par R. White, K. Crane and D.A. Forsyth [WCF07]. Nous avons appliqué notre algorithme avec succès sur les 60 premiers instants temporels de ce jeu de données. En raison de la haute résolution de ces images, quatre échelles multi-résolution ont été nécessaires pour obtenir la précision de reconstruction spatio-temporelle montrée en Figure 4.4.

La Figure 4.5 démontre la supériorité de notre approche spatio-temporelle par rapport à une reconstruction instant par instant utilisant une méthode de stéréovision de l'état de l'art [PKF07], sur le jeu de données "Pants". Concernant l'approche instant par instant, l'idée est d'utiliser le frame obtenu à l'instant un après raffinement, pour initialiser le second instant. Puis on raffine l'instant deux à l'aide d'une optimisation variationnelle basée sur une énergie similaire mais sans interaction entre les différents instants. Enfin nous ajoutons petit à petit des nouveaux frames de la même façon et de manière séquentielle. Les améliorations sont sur trois niveaux : (i) notre approche permet d'exploiter la vitesse et l'accélération pour avoir une bonne estimation initiale de l'instant suivant, et est donc moins sujette à être bloquée dans des minimas locaux (ii) grâce aux contraintes de cohérences spatio-temporelles utilisées, notre approche est moins sujette à échouer dans des régions où la photo-consistance est mauvaise (iii) Notre approche permet d'acquérir simultanément et de manière consistante la forme 3D et le champ de vitesse 3D.

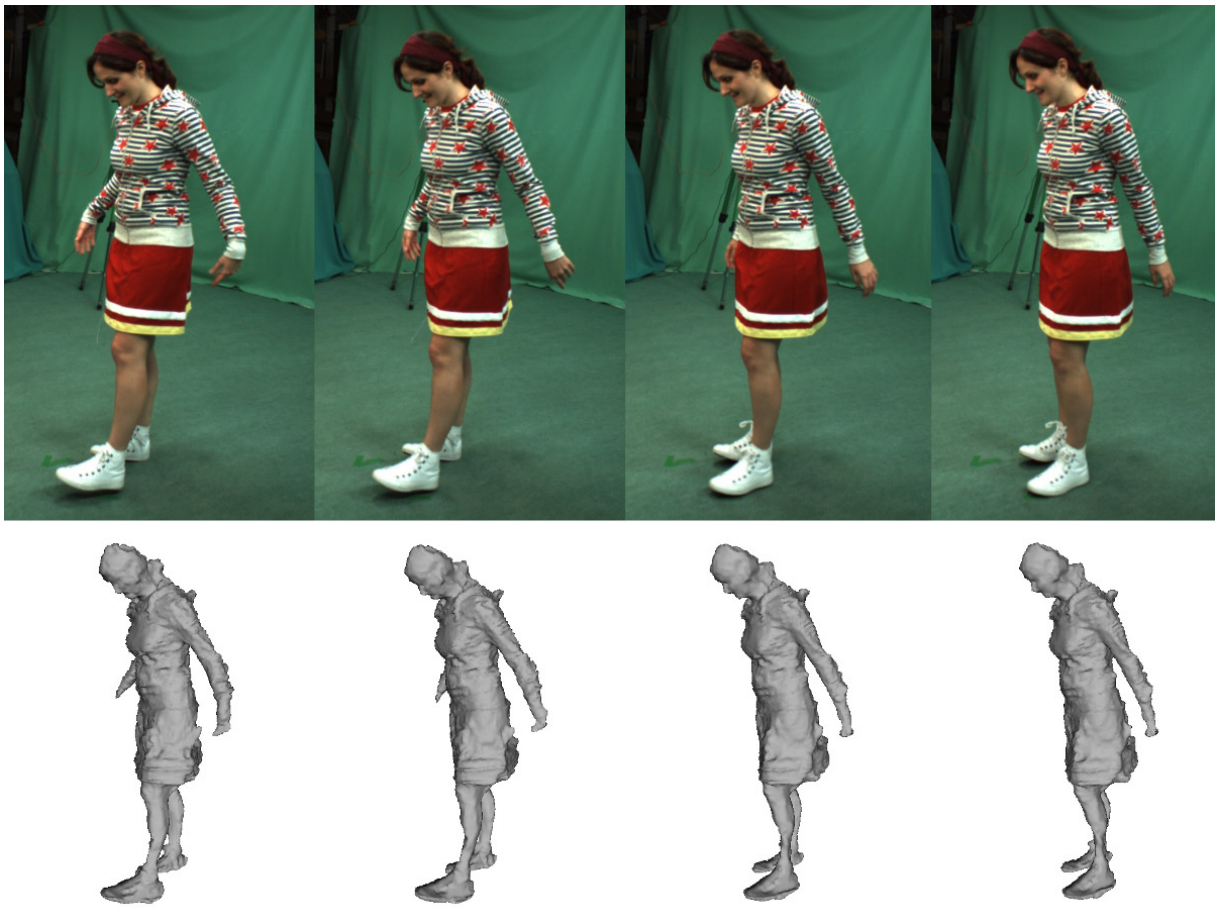


FIG. 4.6 – *Nos résultats pour le jeu de données du "Dancer". Voir le texte pour de plus amples explications.*

Le jeu de données "Dancer" nous a été fourni par l'entreprise 4Dviews [2]. Ce jeu de données a été acquis par 14 vidéos calibrées et synchronisées. Nous avons appliqué notre algorithme aux 10 premiers instants de ce jeu de données. Pour initialiser notre procédure séquentielle et multi-résolution, nous avons utilisé un algorithme standard de stéréovision pour effectuer la reconstruction du premier instant. La reconstruction spatio-temporelle obtenue après avoir appliqué trois niveaux de multi-résolution est présentée dans la Figure 4.6. Nous insistons sur le fait que nous n'avons utilisé aucune information de silhouette pour notre algorithme et qu'effectuer de la stéréovision sur un tel jeu de données est particulièrement délicat. En effet, ce jeu de données a été conçu pour des méthodes basées sur les silhouettes, et de nombreuses parties du sujet sont très peu texturées.

4.5 Pour aller plus loin...

Nous avons présenté une nouvelle méthode variationnelle permettant d'estimer de manière dense, précise et simultanée la forme tridimensionnelle ainsi que le champ de vitesse tridimensionnel à partir de séquences vidéos acquises depuis différents points de vue. Notre approche

[2] http ://4dviews.com

bénéficie des avantages de la représentation par un maillage triangulé animé, d'une énergie de photo-consistance calculée dans la grille des images, d'une optimisation géométrique discrète et des calculs rapides effectués sur carte graphique. Nous avons appliqué avec succès notre algorithme sur des jeux de données compétitifs, et obtenu des résultats rivalisant avec les méthodes constituant l'état de l'art.

En regardant de plus près les techniques mentionnées au début de cette partie consacrée à la reconstruction spatio-temporelle, nous avons vu que très peu de ces méthodes constituant l'état de l'art permettent d'obtenir l'estimation *couplée, dense et précise de la forme 3D et du mouvement 3D* souhaitée. Nous avons proposé une approche permettant d'atteindre ces objectifs avec notre cadre variationnel. Nous pouvons également noter que ces méthodes de l'état de l'art nécessitent une *bonne initialisation* et ne parviennent pas à fournir une reconstruction spatio-temporelle correcte en présence de *larges déplacements entre des instants consécutifs*.

Bonne initialisation La plupart des méthodes de reconstruction utilisant la photo-consistance sont basées sur des optimisations variationnelles [GM04, NA02]. Ces méthodes sont sujettes à rester bloquées dans des minimas locaux, à moins qu'on ne leur fournisse une estimation initiale proche de la véritable scène dynamique. L'absence d'une bonne initialisation disqualifie la plupart des méthodes de stéréovision les plus performantes, mais également les meilleures méthodes de reconstruction spatio-temporelle. Ce problème se pose d'autant plus dans le cas de scènes encombrées, lorsqu'une bonne approximation de la scène par les silhouettes visuelles (Visual Hull) ou par un scan laser ne peut être obtenue.

Larges déplacements entre des instants consécutifs De nombreuses méthodes de reconstruction spatio-temporelle, sont basées, tout comme la nôtre, sur une approche séquentielle. C'est-à-dire que le nouvel instant est généré à partir des instants précédents. Aussi il en résulte une accumulation d'erreurs, une dérive par rapport à la scène dynamique réelle pouvant augmenter au cours du temps. De plus, en cas de mouvements rapides même notre utilisation des informations de vitesse et d'accélération n'est pas suffisante pour suivre un mouvement très volatile. Dans notre cas, cette génération séquentielle n'est effectuée, comme nous l'avons mentionné plus haut, qu'à une basse résolution, ce qui permet de capturer d'assez larges déplacements. Néanmoins, pour de très grands déplacements même notre approche multi-résolution est insuffisante. D'autre part plus la résolution d'image utilisée est basse plus on perd des informations de formes (mains, parties saillantes...) qu'on ne pourra pas nécessairement retrouver par la suite en augmentant la résolution.

Dans cette section nous proposons une nouvelle méthode de reconstruction spatio-temporelle sous la forme d'un pipeline, qui pourrait permettre de retrouver la forme 3D et le mouvement 3D simultanément, et ce, même en présence de larges déplacements et dans le cas de scènes dynamiques encombrées. Dans un premier temps, un optimum global d'une énergie spatio-temporelle (4D) serait calculé, afin d'obtenir un maillage 4D. Un ensemble de maillages 3D pour chaque instant de la séquence temporelle serait alors retrouvé en extrayant des coupes temporelles dans le maillage 4D. Dans un second temps, des correspondances temporelles se-

raient alors calculées de manière dense sur le maillage entre les instants consécutifs, et nous pourrions en déduire un maillage spatio-temporel. Enfin, le maillage spatio-temporel ainsi obtenu serait raffiné suivant la méthode variationnelle présentée dans cette partie consacrée à la reconstruction spatio-temporelle à partir de vidéos.

La première contribution de notre pipeline serait de fournir une initialisation grâce à une optimisation globale qui permet de gérer les scènes larges et encombrées. De plus, les grands déplacements entre les frames sont bien gérés en trouvant des correspondances entre les différents maillages 3D, grâce à une hypothèse selon laquelle les distances géodésiques sont conservées. Finalement, une forme 3D précise ainsi qu'un mouvement 3D précis peuvent être obtenus grâce à notre méthode d'optimisation variationnelle basée sur une énergie spatio-temporelle intégrée dans la grille des images.

Dans les sections 4.5.1, et 4.5.2 nous décrivons en détail les deux premières étapes du pipeline proposé.

4.5.1 Maillage 4D à partir d'une optimisation globale

La première étape consiste à produire un maillage spatio-temporel prenant en compte la visibilité et suffisamment précis pour être raffiné par notre optimisation variationnelle sans tomber dans des minimas locaux trop éloignés de la véritable scène. Cette partie est une extension du travail de P. Labatut et al. [LPK07]. Notre méthode se compose de quatre étapes : (i) un nuage de points 3D quasi-dense est généré pour chaque frame, chaque point mémorisant les deux images ou plus à partir desquelles il a été triangulé. Un nuage de points 4D spatio-temporels est obtenu en ajoutant une quatrième coordonnée temporelle à chaque point 3D ; (ii) la triangulation de Delaunay du nuage de points 4D est calculée ; (iii) les pentatopes de Delaunay (simplexes de pleine dimension dans $\mathbb{R}^4$) sont étiquetés internes ou externes par rapport à l'objet spatio-temporel minimisant une énergie spatio-temporelle prenant en compte la visibilité de façon similaire à [LPK07], une surface 4D orientée est alors extraite comme étant l'ensemble des tétraèdres 4D situés à la frontière entre les pentatopes externes et les pentatopes internes. L'énergie prend en effet en compte la visibilité : Toutes les facettes 4D, intersectant le rayon optique entre les caméras et les points 3D, sont pénalisées. Cette énergie est minimisée de manière globale en calculant une coupe minimale *s-t* (source-puits) par Graph-cut ; (iv) la surface 3D à un instant donné est obtenue en intersectant cette hypersurface 4D avec un plan temporel. Cette méthode fait appel à des connaissances sur la triangulation de Delaunay et sur une minimisation d'énergie par Graph-cut expliquées en Annexe D. Les graph-cut sont brièvement introduits en Annexe B.3. Pour le lecteur intéressé nous signalons que cette méthode a été décrite en détails par E. Aganj dans [APK09].

4.5.2 Mise en correspondance de maillages

La méthode présentée ci-dessus permet de retrouver des maillages 3D à chaque instant, en utilisant une forme de cohérence temporelle. Néanmoins ces maillages sont indépendants et ne sont pas mis en correspondances. Pour pouvoir utiliser notre méthode variationnelle de raffi-

nement présentée dans cette partie de la thèse, nous devons avoir des correspondances dans le temps, et plus particulièrement un maillage spatio-temporel initial avec une connectivité fixe et des positions de sommets variables dans le temps. Nous proposons donc une approche variationnelle pour retrouver un ensemble de correspondances dans le temps entre les maillages.
Nous utilisons l'hypothèse de la conservation des distances géodésiques entre les différents points du maillage dans le temps. Cette hypothèse est raisonnable dans le contexte d'objets articulés, dans la mesure où la plupart des déformations élastiques sont locales et n'affectent pas fortement l'hypothèse d'isométrie d'un point de vue global. Dans ce but, Bronstein *et al.* [BBK06] ont proposé une généralisation multi-échelle pour la mise en correspondance isométrique. Leur méthode repose essentiellement sur la métrique de la surface, rendant cette approche purement géométrique. Dans [TK09], les auteurs proposent une extension des travaux de Bronstein *et al.* en ajoutant la photo-consistance dans le cas de maillages issus de stéréovision multi-vues. La plupart du temps de tels maillages sont bruités et ont de fortes déformations élastiques locales induisant des erreurs lorsque l'on utilise une approche purement géométrique comme dans [BBK06]. Néanmoins l'hypothèse d'isométrie est utilisée comme terme de régularisation pour l'approche variationnelle dans [TK09]. De plus, dans cet article, un ensemble de correspondances initiales est utilisé à l'aide des SIFT[Low04b] calculés dans les images et mis en correspondances entre les maillages. Afin d'améliorer la mise en correspondance des points SIFT détectés, chaque triangle est étiqueté par rapport à l'image par laquelle il est le mieux vu. Ainsi nous mettons en correspondances les points visibles dans une caméra au temps t avec les points visibles dans la même caméra ou une caméra très proche. Nous avons ainsi une initialisation plus robuste des correspondances et sans ambiguïté dans les cas de maillages symétriques grâce à notre information spatiale provenant des caméras étiquetées. Ces correspondances initiales sont ensuite raffinées en minimisant l'énergie suivante :

$$E_{tot}(\Theta) = \alpha E_{geom}(\Theta) + \beta E_{photo}(\Theta). \tag{4.12}$$

Le premier terme E_{geom} est la partie géométrique, reprise des travaux de Bronstein *et al.* [BBK06] tandis que le terme E_{photo} correspond à un terme de photo-consistance et d'attache aux données dans les images. α et β sont des nombres réels positifs contrôlant les poids relatifs de ces termes. Comme les points correspondants sont représentés localement en utilisant les coordonnées barycentriques Θ dans les triangles, l'énergie (4.12) est minimisée par rapport à Θ. Ainsi nous raffinons les positions des points pour que leurs reprojections dans les images minimisent une dissimilarité (l'opposé de la corrélation comme pour notre méthode variationnelle) entre les images, ainsi que des informations géométriques intrinsèques entre les maillages. Une stratégie par expansion est utilisée au cours des itérations successives de cette minimisation par descente de gradient. Une approche multi-résolution est également utilisée. La topologie de la séquence est supposée fixe et initialisée par le maillage à l'instant $t = 0$. A partir du temps $t = 0$, nous propageons les sommets dans le temps par mises en correspondance successives de paires d'instants. Dans la mesure où nous avons besoin de correspondances pour chaque sommet du maillage $\mathcal{M}^0$ dans tous les frames suivants, la méthode de [TK09] n'est pas directement applicable. Ainsi nous ne minimisons l'énergie que pour un petit sous ensemble de sommets jugés fiables qui constituent un maillage basse résolution. Une fois que la méthode

converge, les correspondances sont interpolées à une échelle plus fine du maillage et des images en utilisant les distances géodésiques comme dans le travail de [BBK06]. L'idée consiste donc à utiliser une approche multi-résolution en terme d'images mais également de maillage. Nous avons vu dans cette partie comment on peut changer la résolution du maillage en fonction de celle des images avec des longueurs caractéristiques maximales et minimales (Voir Annexe D pour de plus amples détails).

Les principales difficultés rencontrées pour notre pipeline pourraient être la non conservation des distances géodésiques dans le cas de changements de topologie d'une part et de maillages bruités d'autre part. Les travaux de [PSDB+10, BBH08] mènent une réflexion sur la gestion des changements de topologie, en considérant par exemple la conservation des distances intrinsèques d'un point de vue local et non global. Un débruitage approprié des maillages est également effectué pour que l'hypothèse de conservation locale des distances intrinsèques reste applicable. Nous pourrions nous inspirer de ces travaux pour rendre notre pipeline plus robuste.

Annexe

Annexe A

Descripteurs SIFT

Il existe de nombreux détecteurs de points d'intérêt tels que les SIFT ou les MSER. Ici nous nous intéresserons plus particulièrement aux descripteurs SIFT qui est la méthode de mise en correspondance que nous avons utilisée pour implémenter les méthodes présentées dans ce manuscrit. Nous faisons une brève étude des descripteurs SIFT, mais pour une étude plus exhaustive le lecteur peut se référer à l'article de D. Lowe qui est l'inventeur des descripteurs SIFT [Low04a].

A.1 Étude de détecteurs de points d'intérêts

A.1.1 Détecteur Laplacien

Si l'on note $\mathbf{u}(\sigma, \mathbf{x}) = \mathtt{G}_\sigma(\mathbf{u}_0(\mathbf{x}))$ la convolution de l'image $\mathbf{u}_0$ par une gaussienne de variance σ alors l'équation de la chaleur est satisfaite (l'équation n'étant pas ici paramétrée par $t = \sigma^2$) :

$$\frac{\partial \mathbf{u}}{\partial \sigma} = \sigma \Delta^2 \mathbf{u} \tag{A.1}$$

Cette équation étant invariante par transformation affine, il a été démontré par Lindeberg (1994) [Lin94], que pour avoir une invariance de région locale par changement d'échelle il faut étudier $\sigma^2 \Delta^2 \mathbf{u}$. Mycolajczyk [Mik02] a ensuite montré par des études comparatives détaillées que l'étude des extrema de $\sigma^2 \Delta^2 \mathbf{u}$ fournit les régions les plus stables par transformation affine.

A.1.2 Détecteur DOG

Afin d'obtenir des temps de traitements plus rapides on remarque qu'il est possible d'effectuer un lissage suivant l'horizontale avec une gaussienne, puis une convolution suivant la verticale avec une autre gaussienne (la fonction gaussienne étant séparable). D'autre part il est possible d'utiliser une échelle pyramidale afin d'approximer $\sigma^2 \Delta^2 \mathbf{u}$ par simple soustraction, comme indiqué en figure A.1.

On peut paramétrer le nombre d'octaves o, le nombre d'échelles par octave s (précision). A chaque octave on double σ.

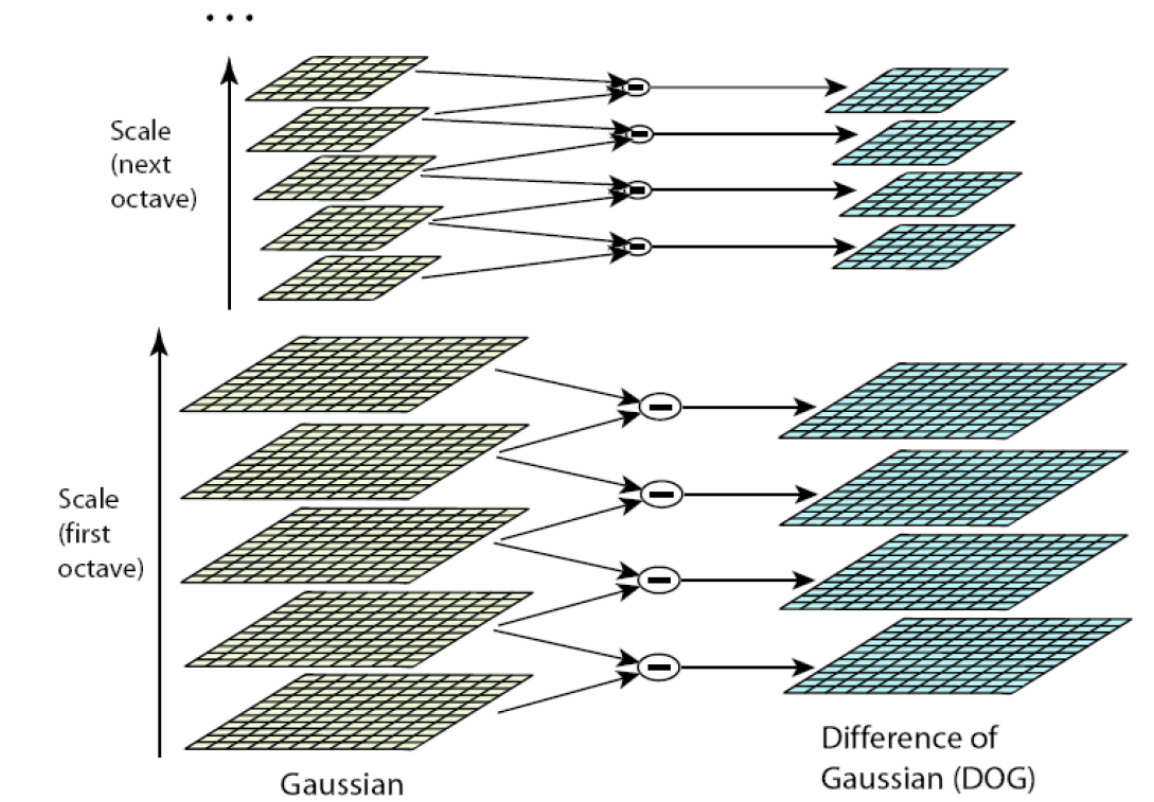


FIG. A.1 – *Calcul des différences de gaussiennes.*

L'opérateur de convolution par la gaussienne étant pyramidal il existe $\lambda(t,h)$ tel que $\mathbb{G}_{t+h}(\mathbf{u}) = \mathbb{G}_{\lambda}(\mathbb{G}_{t}(\mathbf{u}))$. Il suffit donc de lisser l'image de l'échelle précédente par une gaussienne de petite taille λ ce qui permet d'avoir un gain en temps de calcul.

Enfin, pour passer à l'octave suivant, c'est-à-dire lisser par une gaussienne de variance deux fois plus grande, il suffit de sous-échantillonner d'un facteur 2 l'image, puis de convoler par une gaussienne de variance σ.

Un rapide calcul montre que la soustraction, entre les images lissées à deux échelles successives, permet d'obtenir une bonne approximation de $\sigma^2\Delta^2\mathbf{u}$ à un facteur constant près, qui ne gêne pas pour la détection des extrema :

$$\begin{aligned} k &= 2^{\frac{1}{s}} \\ DOG &\approx (k-1)\,\sigma^2\Delta^2\mathbf{u} \end{aligned} \tag{A.2}$$

Pour détecter les extrema, il suffit de comparer un pixel à ses proches voisins à la même échelle et aux échelles consécutives comme indiqué en figure A.2.

A.2 Étude des descripteurs SIFT

A.2.1 Descripteur SIFT et histogramme de gradient

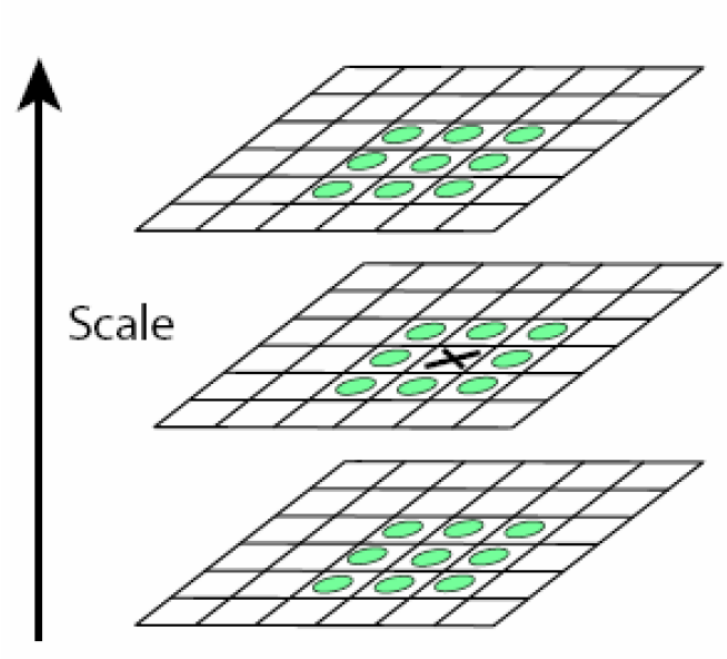


FIG. A.2 – *Détection des extréma par comparaison entre les proches voisins.*

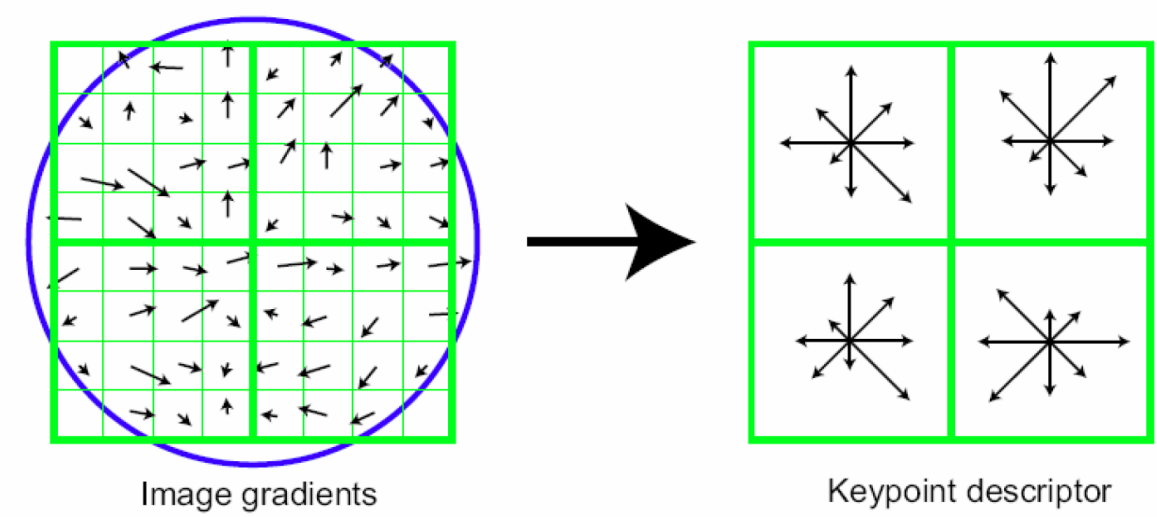


FIG. A.3 – *Décomposition de la zone d'intérêt en* $4 \times 4 \times 8$ *cases.*

A.2.1.1 Orientation

Afin de définir l'orientation, on effectue un histogramme sur 36 cases (pour les 360°) des orientations du gradient dans la région du point clé. A chaque pixel on assimile un poids à l'orientation correspondant à l'amplitude du gradient ainsi qu'un poids lié à une fenêtre circulaire gaussienne de taille 1.5σ. Ce qui permet d'attribuer un poids plus important aux forts gradients situés près de l'origine du point d'intérêt.

Pour effectuer une mise en correspondance plus robuste on utilise plusieurs orientations possibles pour le point d'intérêt. Si une des orientations est supérieure a 80% du pic d'orientation dans l'histogramme on ajoute un point d'intérêt avec une nouvelle orientation. En moyenne 15% des points détectés se voient assignées plusieurs orientations.

A.2.1.2 Description

Sur une région 16×16 pixels, on calcule tous les gradients. Alors on recalcule les vecteurs dans le repère défini par l'orientation trouvée pour le point, afin d'obtenir une invariance par rotation. Ensuite, on découpe cette région en 4×4 zones, comme indiqué sur la figure A.3. Finalement on calcule l'histogramme sur 8 cases (8 orientations possibles) en pondérant chaque vecteur par une fenêtre gaussienne de taille 0.5σ, toujours afin de donner un poids plus important au centre de chaque zone et pour éviter de fortes variations aux frontières si il y a des inexactitudes dans les échelles trouvées et des gradients importants aux frontières. On obtient un vecteur descripteur de taille $4 \times 4 \times 8 = 128$.

Afin de réduire les effets dus aux changements de luminosité, le vecteur est normalisé à 1. D'autre part les changements non linéaires (saturation, variation de la surface) peuvent affecter les amplitudes des gradients mais sont moins susceptibles de modifier les orientations. Ainsi pour limiter l'influence des très forts gradients, les éléments du vecteur sont seuillés à 0.2, puis le vecteur est normalisé à nouveau.

A.2.1.3 Mise en correspondance

Supposons que nous ayons des points d'intérêts et leurs descripteurs dans 2 images. Il suffit pour chaque point détecté dans la première image de calculer les 2 plus proches parmi les points détectés dans la seconde image. La distance peut être définie comme la distance euclidienne entre les vecteurs descripteurs. Nous obtenons alors 2 correspondances, la plus proche ayant une distance d_1 et la seconde ayant une distance d_2 entre les descripteurs. Nous considérons la meilleure correspondance comme étant retenue si le ratio $r = \frac{d_1}{d_2} < 0.8$.

Il s'agit du critère de David Lowe qui permet de ne garder que les correspondances distinctives, c'est-à-dire celles pour lesquelles la meilleure correspondance est "bien meilleure" que les autres.

A.2.2 Expériences

La figure A.4 présente les "Difference Of Gaussian" obtenues aux différentes échelles.

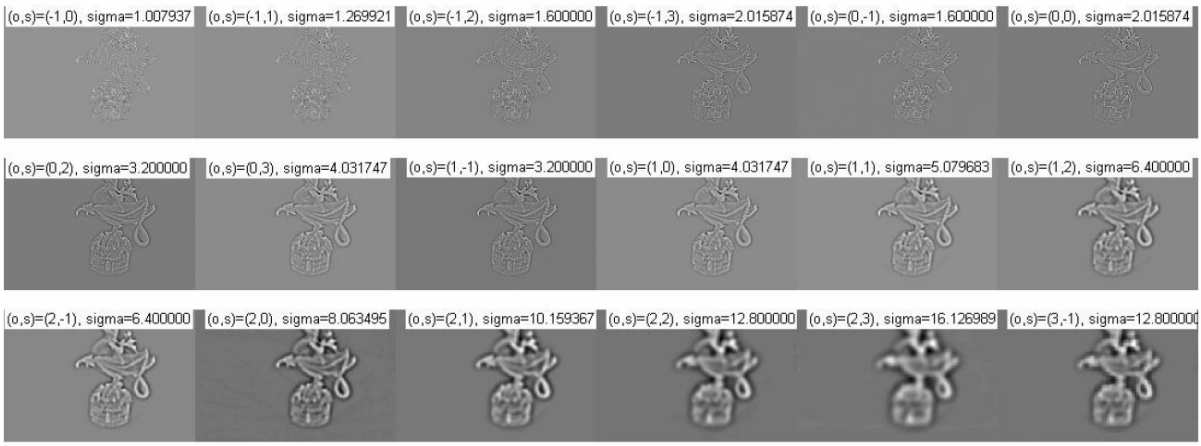


FIG. A.4 – *"Difference Of Gaussian" obtenues aux différentes échelles.*

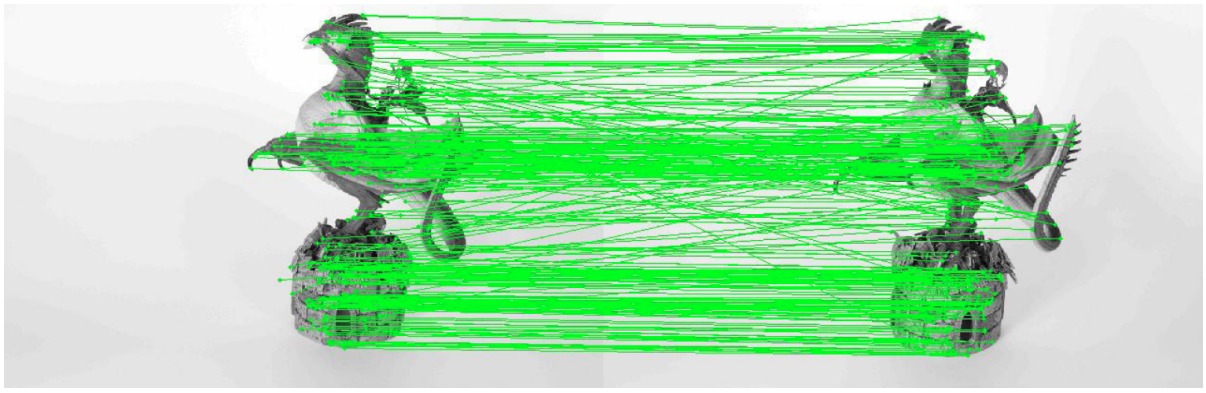


FIG. A.5 – *correspondances obtenues avec le critère de David Lowe.*

La figure A.5 montre un aperçu des correspondances trouvées. Si la plupart des correspondances sont correctes, certaines lignes croisées montrent qu'il est important de récupérer les véritables correspondances de manière robuste. Pour cela on utilise la géométrie des caméras. A cet effet on peut garder les bonnes correspondances entre 2 ou 3 images, et éliminer les mauvaises par RANSAC comme décrit dans les sections 1.3.1 et 1.3.2.

Annexe B

Optimisation

B.1 Solution aux moindres carrés d'équations linéaires

B.1.1 Décomposition en valeurs singulières

La décomposition en valeurs singulières (SVD) est une des décompositions matricielles les plus connues. Son application la plus courante est la résolution de systèmes d'équations surdéterminés.

Etant donné une matrice carrée $\mathtt{A}$, la SVD est une factorisation de $\mathtt{A}$ telle que $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^T$ où $\mathtt{U}$ et $\mathtt{V}$ sont des matrices orthonormales, et $\mathtt{D}$ est une matrice diagonale avec des entrées non négatives. La décomposition peut être faite de sorte que les entrées diagonales de $\mathtt{D}$ sont classées dans l'ordre décroissant, et nous supposerons que ceci est toujours le cas. Ainsi la colonne de $\mathtt{V}$ correspondant à la plus petite valeur singulière est en fait la dernière colonne de $\mathtt{V}$.

La SVD existe également pour les matrices non carrées $\mathtt{A}$. Nous nous intéressons ici au cas où $\mathtt{A}$ a plus de lignes que de colonnes. $\mathtt{A}$ étant une matrice $m \times n$ avec $m \geq n$. Dans ce cas $\mathtt{A}$ peut être factorisée sous la forme $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^T$ où $\mathtt{U}$ est une matrice orthonormale de taille $m \times n$, $\mathtt{D}$ est une matrice diagonale $n \times n$, et $\mathtt{V}$ est une matrice orthonormale de taille $n \times n$. Le fait que $\mathtt{U}$ soit orthonormale signifie que $\mathtt{U}^T\mathtt{U} = \mathtt{I}_{n\times n}$. De plus $\mathtt{U}$ préserve la norme c'est-à-dire pour tout vecteur $\mathbf{x}$, $\|\mathtt{U}\mathbf{x}\| = \|\mathbf{x}\|$.

On peut également définir une décomposition en valeurs singulières pour une matrice ayant plus de colonnes que de lignes. Lorsque l'on souhaite obtenir une SVD de la matrice $\mathtt{A}$ avec $m < n$, il convient de compléter $\mathtt{A}$ avec des lignes contenant des valeurs nulles, pour obtenir une matrice carrée, et alors effectuer la SVD de cette matrice carrée résultante. Les librairies de décomposition en valeurs singulières effectuent généralement cette procédure automatiquement.

Dans ce manuscrit, aucune preuve de l'existence de cette décomposition n'est donnée. Pour une description des preuves liées à cet algorithme, le lecteur peut se référer à [GVL96].

B.1.2 Solution aux moindres carrés d'équations linéaires homogènes

Trouver $\mathbf{x}$ minimisant $\|\mathtt{A}\mathbf{x}\|$ tel que $\|\mathbf{x}\| = 1$ Nous traitons dans un premier temps le cas où $\mathtt{A}$ a un rang plein. En effectuant une décomposition en valeurs singulières, le problème

consiste à minimiser $\left\|\mathtt{U}\mathtt{D}\mathtt{V}^T\mathbf{x}\right\|$ tel que $\|\mathbf{x}\| = 1$. Dans la mesure où $\left\|\mathtt{U}\mathtt{D}\mathtt{V}^T\mathbf{x}\right\| = \left\|\mathtt{D}\mathtt{V}^T\mathbf{x}\right\|$ et $\|\mathbf{x}\| = \left\|\mathtt{V}^T\mathbf{x}\right\|$, le problème est équivalent à minimiser $\left\|\mathtt{D}\mathtt{V}^T\mathbf{x}\right\|$ tel que $\left\|\mathtt{V}^T\mathbf{x}\right\| = 1$. En notant $\mathbf{y} = \mathtt{V}^T\mathbf{x}$, le problème consiste à minimiser $\|\mathtt{D}\mathbf{y}\|$ tel que $\|\mathbf{y}\| = 1$. Maintenant, $\mathtt{D}$ est une matrice diagonale avec ses entrées dans l'ordre décroissant. La solution au problème est évidemment $\mathbf{y} = (0, 0, \cdots, 0, 1)^T$. Finalement, la solution du problème initial est $\mathbf{x} = \mathtt{V}\mathbf{y}$ et est simplement la dernière colonne de $\mathtt{V}$.

Objectif

Etant donné une matrice $\mathtt{A}$ de taille $m \times n$ et de rang au moins n, trouver $\mathbf{x}$ minimisant $\|\mathtt{A}\mathbf{x}\|$ tel que $\|\mathbf{x}\| = 1$.

Solution

$\mathbf{x}$ est la dernière colonne de $\mathtt{V}$, où $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^T$ est la SVD de $\mathtt{A}$.

Dans le cas où $\mathtt{A}$ est de rang $n - (k + 1)$ avec $k \geq 0$, alors toutes les colonnes de $\mathtt{V}$, $\mathbf{V}_n, \cdots, \mathbf{V}_{n-k}$ associées aux $k + 1$ dernières valeurs singulières nulles seront solutions. Et la solution générale est un espace vectoriel de la forme :

$$\left\{\mathbf{V} \text{ tel que } \mathbf{V} = \lambda_0\mathbf{V}_n + \cdots + \lambda_k\mathbf{V}_{n-k}\,,\ (\lambda_0, \cdots, \lambda_k) \in \mathbb{R}^{k+1}\right\} \tag{B.1}$$

B.1.3 Solution aux moindres carrés d'équations linéaires non homogènes

Trouver $\mathbf{x}$ minimisant $\|\mathtt{A}\mathbf{x} - \mathbf{b}\|$ Dans un premier temps nous nous intéressons au cas où $\mathtt{A}$ est de taille $m \times n$ avec $m \geq n$ et la matrice n'est pas dégénérée et est de rang n. Nous cherchons à minimiser $\|\mathtt{A}\mathbf{x} - \mathbf{b}\| = \left\|\mathtt{U}\mathtt{D}\mathtt{V}^T\mathbf{x} - \mathbf{b}\right\|$, dans la mesure où les matrices orthonormales préservent les normes, en multipliant l'équation précédente par $\mathtt{U}^T$ ceci revient à minimiser $\left\|\mathtt{D}\mathtt{V}^T\mathbf{x} - \mathtt{U}^T\mathbf{b}\right\|$. En notant $\mathbf{y} = \mathtt{V}^T\mathbf{x}$ et $\mathbf{b}' = \mathtt{U}^T\mathbf{b}$, le problème consiste à minimiser $\|\mathtt{D}\mathbf{y} - \mathbf{b}'\|$.

Cet ensemble d'équations peut s'écrire sous la forme :

$$\left[\begin{array}{cccc} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & s_n \\ \hline & & & \\ & & 0 & \end{array}\right] \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \left(\begin{array}{c} b'_1 \\ b'_2 \\ \vdots \\ b'_n \\ \hline b'_{n+1} \\ \vdots \\ b'_m \end{array}\right) \tag{B.2}$$

Il est clair que le vecteur $\mathtt{D}\mathbf{y}$ pouvant le mieux approcher $\mathbf{b}'$ est le vecteur $(b'_1, b'_2, \cdots, b'_n, 0, \cdots, 0)^T$, ceci est obtenu en posant $y_i = \frac{b'_i}{d_i}$, pour i allant de 1 à n. On remarque que comme le rang de $\mathtt{A}$ est n, $\forall\, i\ d_i \neq 0$. Finalement on retrouve la solution du

problème initial $\mathbf{x} = \mathtt{V}\mathbf{y}$. Voici un récapitulatif de l'algorithme :

Objectif

Trouver $\mathbf{x}$ minimisant $\|\mathtt{A}\mathbf{x} - b\|$ pour une matrice $\mathtt{A}$ $m \times n$ de rang n avec $m \geq n$.

Algorithme

i. Trouver la SVD $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^T$, où s_i est la $i^{\text{ème}}$ entrée de la matrice diagonale $\mathtt{D}$
ii. Poser $\mathbf{b}' = \mathtt{U}^T\mathbf{b}$
iii. Trouver le vecteur $\mathbf{y}$ définit par $y_i = \frac{b'_i}{s_i}$.
iv. La solution est $\mathbf{x} = \mathtt{V}\mathbf{y}$

Dans la mesure où le rang de $\mathtt{A}$ est $r < n$, la première partie de la résolution du problème reste inchangée, et le problème consiste toujours à minimiser $\|\mathtt{D}\mathbf{y} - \mathbf{b}'\|$.

$$\left[\begin{array}{ccc|c} s_1 & & & \\ & \ddots & & 0 \\ & & s_r & \\ \hline & 0 & & \end{array}\right] \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \left(\begin{array}{c} b'_1 \\ b'_2 \\ \vdots \\ b'_n \\ \hline b'_{n+1} \\ \vdots \\ b'_m \end{array}\right) \tag{B.3}$$

Il est clair que le vecteur $\mathtt{D}\mathbf{y}$ pouvant le mieux approcher $\mathbf{b}'$ est le vecteur $(b'_1, b'_2, \cdots, b'_r, 0, \cdots, 0)^T$, ceci est obtenu par exemple en posant $y_i = \frac{b'_i}{d_i}$, pour i allant de 1 à r. Une solution du problème initial est donc $\mathbf{x} = \mathtt{V}\mathbf{y}$. D'autre part les $n - r$ dernières colonnes de $\mathtt{V}$ sont des vecteurs nuls de $\mathtt{A}$, aussi la solution générale du problème peut s'écrire :

$$\left\{\mathbf{x} \text{ tel que } \mathbf{x} = \mathtt{V}\mathbf{y} + \lambda_{r+1}\mathbf{V}_{r+1} + \cdots + \lambda_n\mathbf{V}_n\,,\ (\lambda_{r+1}, \cdots, \lambda_n) \in \mathbb{R}^{n-r}\right\} \tag{B.4}$$

Finalement on retrouve la solution du problème initial $\mathbf{x} = \mathtt{V}\mathbf{y}$. Voici un récapitulatif de l'algorithme :

Objectif

Trouver $\mathbf{x}$ minimisant $\|\mathtt{A}\mathbf{x} - b\|$ pour une matrice $\mathtt{A}$ $m \times n$ de rang $r < n$.

Algorithme

i. Trouver la SVD $\mathtt{A} = \mathtt{U}\mathtt{D}\mathtt{V}^T$, où s_i est la $i^{\text{ème}}$ entrée de la matrice diagonale $\mathtt{D}$
ii. Poser $\mathbf{b}' = \mathtt{U}^T\mathbf{b}$
iii. Trouver le vecteur $\mathbf{y}$ défini par $y_i = \frac{b'_i}{s_i}$, pour i allant de 1 à r et 0 sinon.
iv. Une solution est $\mathbf{x} = \mathtt{V}\mathbf{y}$
v. La solution générale est $\mathbf{x} = \mathtt{V}\mathbf{y} + \lambda_{r+1}\mathbf{V}_{r+1} + \cdots + \lambda_n\mathbf{V}_n$, où $\mathbf{V}_{r+1}, \cdots, \mathbf{V}_n$ sont les $n - r$ dernières colonnes de $\mathtt{V}$.

Dans le cas où la matrice $\mathtt{A}$ est de rang n, nous allons chercher à résoudre le problème d'une façon différente. Si l'on suppose que $\mathbf{x}$ minimise $\|\mathtt{A}\mathbf{x} - b\|$, il minimise l'expression scalaire $(\mathtt{A}\mathbf{x} - b)^T(\mathtt{A}\mathbf{x} - b)$. Toutes les dérivées partielles de cette expression par rapport aux entrées de $\mathbf{x}$ sont donc forcément nulles. Ceci s'écrit simplement $\mathtt{A}^T\mathtt{A}\mathbf{x} - \mathtt{A}^T b = 0$. La matrice étant de rang n, ce système a une solution unique qui est $\mathbf{x} = (\mathtt{A}^T\mathtt{A})^{-1}\mathtt{A}^T\mathbf{b}$. Il s'agit de la solution au problème initial. L'algorithme est résumé ci-dessous.

Objectif

Trouver $\mathbf{x}$ minimisant $\|\mathtt{A}\mathbf{x} - b\|$ pour une matrice $\mathtt{A}$ $m \times n$ de rang n avec $m > n$.

Algorithme

i. La solution est $\mathbf{x} = (\mathtt{A}^T\mathtt{A})^{-1}\mathtt{A}^T\mathbf{b}$.

Les équations $\mathtt{A}^T\mathtt{A}\mathbf{x} - \mathtt{A}^T b = 0$ sont appelées les *équations normales*. Cet algorithme est proposé ici car lorsque n est petit devant m il devient moins coûteux de calculer la matrice $(\mathtt{A}^T\mathtt{A})^{-1}\mathtt{A}^T$ que d'effectuer une SVD. Lorsque $\mathtt{A}$ est de rang n cette matrice est en fait la pseudo-inverse de $\mathtt{A}$.

B.1.4 Minimisation sous contraintes

Trouver $\mathbf{x}$ minimisant $\|\mathtt{A}\mathbf{x}\|$ sous contraintes $\|\mathbf{x}\| = 1$ et $\mathbf{x} = \mathtt{G}\widehat{\mathbf{x}}$ La contrainte $\mathbf{x} = \mathtt{G}\widehat{\mathbf{x}}$ signifie que $\mathbf{x}$ est dans l'espace formé par les colonnes de $\mathtt{G}$. Effectuons une SVD $\mathtt{G} = \mathtt{U}\mathtt{D}\mathtt{V}^T$, $\mathtt{D}$ a r entrées non nulles (c'est-à-dire $\mathtt{G}$ est de rang r). Si $\mathtt{U'}$ est la matrice formée par les r premières colonnes de $\mathtt{U}$, alors l'espace formé par les colonnes de $\mathtt{U'}$ est le même que l'espace formé par les colonnes de $\mathtt{G}$. Il suffit de résoudre le problème consistant à trouver $\widehat{\mathbf{x}'}$ minimisant $\left\|\mathtt{A}\,\mathtt{U'}\,\widehat{\mathbf{x}'}\right\|$ sous contraintes $\left\|\widehat{\mathbf{x}'}\right\| = 1$. Cette résolution a été vue plus haut dans cette même section. Finalement comme $\mathtt{U'}$ a un rang plein et conserve la norme, la solution du problème

initial est $\mathbf{x} = \mathtt{U}'\widehat{\mathbf{x}}'$. Nous résumons l'algorithme ci-dessous.

Objectif

Trouver $\mathbf{x}$ minimisant $\|\mathtt{A}\mathbf{x}\|$ sous contraintes $\|\mathbf{x}\| = 1$ et $\mathbf{x} = \mathtt{G}\widehat{\mathbf{x}}$.

Algorithme

i. Trouver la SVD $\mathtt{G} = \mathtt{U}\mathtt{D}\mathtt{V}^T$. $\mathtt{D}$ ayant r entrées non nulles.
ii. Former $\mathtt{U}'$ à partir des r premières colonnes de $\mathtt{U}$.
iii. Trouver $\widehat{\mathbf{x}}'$ minimisant $\left\|\mathtt{A}\mathtt{U}'\widehat{\mathbf{x}}'\right\|$ tel que $\left\|\widehat{\mathbf{x}}'\right\| = 1$, vu en B.1.2.
iv. La solution au problème est $\mathbf{x} = \mathtt{U}'\widehat{\mathbf{x}}'$.

Trouver X minimisant $\|\mathtt{A}X\|_2^2$ pour une matrice $\mathtt{A}$ $k \times p$ sous contraintes que les colonnes de $X \in \mathbb{R}^{k\times p}$ soient orthonormées La solution est donnée par $X = [\mathbf{u}_1 \dots \mathbf{u}_p]$ où $\mathbf{u}_j$ est le vecteur propre de $\mathtt{A}^T\,\mathtt{A}$ correspondant à la $j^{ème}$ plus petite valeur propre λ_j.

Nous en proposons une preuve dans le cas où $p = 2$, le cas général se déduit facilement. Si l'on pose $X = [\mathbf{u}_1\ \mathbf{u}_2]$. Les vecteurs propres sont orthonormaux dans la mesure où $\mathtt{A}^T\,\mathtt{A}$ est une matrice symétrique.

$$\|\mathtt{A}X\|_2^2 = \|\mathtt{A}\mathbf{u}_1\|^2 + \|\mathtt{A}\mathbf{u}_2\|^2 = \lambda_1 + \lambda_2$$

.

Il suffit donc de démontrer que pour tout X ayant des colonnes orthonormées alors, $\|\mathtt{A}X\|_2^2 \geq \lambda_1 + \lambda_2$.

Effectuons un raisonnement par l'absurde, supposons qu'il existe une matrice $X = [X_1\ X_2]$ ayant des colonnes orthonormées telle que $\|\mathtt{A}X\|_2^2 < \lambda_1 + \lambda_2$.

Soit $\mathbf{v} \in Span(X_1, X_2) \cap \mathbf{u}_1^{\perp}$. Un tel veteur non nul existe car $dim\,(Span(X_1, X_2)) + dim\left(\mathbf{u}_1^{\perp}\right) = k + 1 > k$. On supposera $\mathbf{v}$ de norme unitaire.

Soit $\mathbf{w} \in \mathbb{R}^k$ de norme unitaire orthogonal à $\mathbf{v}$ et appartenant à $Span(X_1, X_2)$. On a alors pour $\widetilde{X} = [\mathbf{v}, \mathbf{w}]$, $\left\|\mathtt{A}\widetilde{X}\right\|_2^2 = \|\mathtt{A}X\|_2^2$.

D'autre part comme $\|\mathtt{A}\mathbf{w}\|_2^2 \geq \lambda_1$, on a

$$\|\mathtt{A}[\mathbf{u}_1\ \mathbf{v}]\|_2^2 \leq \|\mathtt{A}[\mathbf{w}\ \mathbf{v}]\|_2^2 = \|\mathtt{A}X\|_2^2 < \lambda_1 + \lambda_2$$

.

Or,

$$\|\mathtt{A}[\mathbf{u}_1\ \mathbf{v}]\|_2^2 = \lambda_1 + \|\mathtt{A}\mathbf{v}\|_2^2$$

.

Donc $\exists \mathbf{v} \in \mathbb{R}^k$ tel que $\mathbf{v}$ est orthogonal à $\mathbf{u}_1$ et $\|\mathbb{A}\mathbf{v}\|_2^2 < \lambda_2$. Ceci est absurde car $\mathbf{v}$ appartient à l'espace formé par tous les vecteurs propres sauf celui associé à λ_1 et donc $\|\mathbb{A}\mathbf{v}\|_2^2 \geq \lambda_2$.

B.2 Méthodes itératives

B.2.1 Itérations de Gauss-Newton

Supposons que nous avons une fonctionnelle $\mathbf{f}$, et $\mathbf{P}$ représente le vecteur des paramètres. Nous cherchons $\mathbf{P}$ minimisant $\|\mathbf{f}(\mathbf{P})\|$. La méthode proposée est itérative, nous partons donc d'un estimateur initial $\mathbf{P}_0$, Et à chaque itération nous calculons l'estimateur $\mathbf{P}_{i+1}$ à partir de $\mathbf{P}_i$. En supposant que la fonctionnelle est linéaire et $\mathbb{J}$ étant la matrice jacobienne de $\mathbf{f}$ on a l'équation $\mathbf{f}(\mathbf{P}_{i+1}) = \mathbf{f}(\mathbf{P}_i) + \mathbb{J}(\mathbf{P}_{i+1} - \mathbf{P}_i)$. Nous souhaitons trouver le déplacement permettant d'annuler $\mathbf{f}(\mathbf{P}_{i+1})$, pour cela nous cherchons $\Delta = \mathbf{P}_{i+1} - \mathbf{P}_i$ minimisant $\mathbb{J}\Delta + \mathbf{f}(\mathbf{P}_i)$. A cet effet il suffit de résoudre les *équations normales* décrites en Annexe B.1.3 : $\mathbb{J}^T\mathbb{J}\Delta = -\mathbb{J}^T\mathbf{f}(\mathbf{P}_i)$.

Cette méthode constitue une très bonne approximation lorsque l'on est proche du minimum global et peut converger très rapidement. Néanmoins elle n'offre aucune garantie ni de converger ni de réduire l'erreur de la fonctionnelle.

B.2.2 Descente de gradient

Annuler la fonctionnelle $\mathbf{f}$, revient à chercher le minimum de $\left\|\mathbf{f}(\mathbf{P})^T\mathbf{f}(\mathbf{P})\right\|$. Le développement limité au premier ordre est $\mathbf{f}(\mathbf{P}_{i+1}) = \mathbf{f}(\mathbf{P}_i) + \mathbb{J}\Delta + \circ(\Delta)$. Donc, en choisissant $\lambda\Delta = -\mathbb{J}^T\mathbf{f}(\mathbf{P}_i)$ on a :

$$\begin{array}{rcl} \mathbf{f}(\mathbf{P}_{i+1})^T f(\mathbf{P}_{i+1}) - \mathbf{f}(\mathbf{P}_i)^T \mathbf{f}(\mathbf{P}_i) & = & \mathbf{f}(\mathbf{P}_i)^T\mathbb{J}\Delta + \left(\mathbf{f}(\mathbf{P}_i)^T\mathbb{J}\Delta\right)^T + \circ(\Delta) \\ \mathbf{f}(\mathbf{P}_{i+1})^T f(\mathbf{P}_{i+1}) - \mathbf{f}(\mathbf{P}_i)^T \mathbf{f}(\mathbf{P}_i) & = & 2\mathbf{f}(\mathbf{P}_i)^T\mathbb{J}\Delta + \circ(\Delta) \\ \mathbf{f}(\mathbf{P}_{i+1})^T f(\mathbf{P}_{i+1}) - \mathbf{f}(\mathbf{P}_i)^T \mathbf{f}(\mathbf{P}_i) & = & -2\mathbf{f}(\mathbf{P}_i)^T\mathbb{J}\mathbb{J}^T\mathbf{f}(\mathbf{P}_i) + \circ(\Delta) \end{array} \tag{B.5}$$

Aussi pour λ suffisamment grand les termes d'ordre 2 sont négligeables et nous sommes sûrs de minimiser la fonctionnelle. Pour cet algorithme il faut choisir λ adaptativement de façon à garantir la décroissance. Cette méthode offre plus de garanties que la précédente mais est sujette à rester bloquée dans des minimas locaux.

B.2.3 Itérations de Levenberg-Marquardt

La méthode de Levenberg-Marquardt consiste à utiliser les méthodes de Gauss-Newton et de descente de gradient conjointement. Concrètement à chaque itération le déplacement sera la solution des équations suivantes $\left(\mathbb{J}^T\mathbb{J} + \lambda\mathbb{I}\right)\Delta = -\mathbb{J}^T\mathbf{f}(\mathbf{P}_i)$.

Une valeur initiale généralement choisie pour λ sera 10^{-3} multiplié par la moyenne des termes diagonaux de $\mathbb{J}^T\mathbb{J}$. Si cette équation permet de réduire l'erreur alors on multipliera λ par 10 pour l'itération suivante. En revanche, si l'on augmente l'erreur on refait un essai en

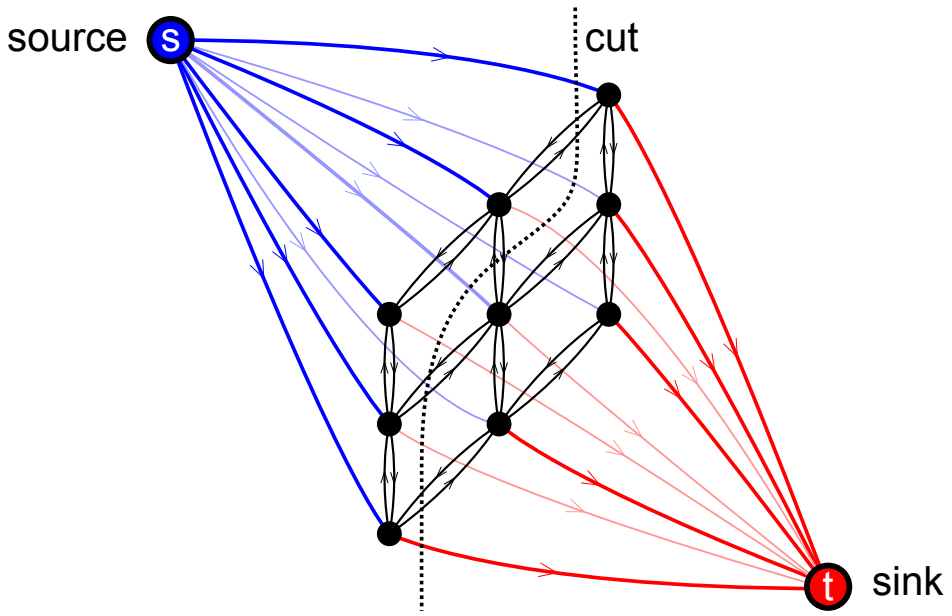


FIG. B.1 – *Une coupe s-t dans un graphe.*

multipliant λ par 10 et en itérant ce procédé jusqu'à ce que l'on parvienne à faire décroître l'erreur.

Ainsi, lorsque λ est petit on se rapproche d'une résolution de type Gauss-Newton, qui est efficace et rapide mais n'offre aucune garantie de convergence. Lorsque pour λ petit, on ne parvient pas à réduire l'erreur on augmente λ et l'équation dominante devient celle correspondant à la descente de gradient $\lambda\Delta = -\mathbb{J}^T\mathbf{f}(\mathbf{P}_i)$, qui pour λ suffisamment grand assure la réduction de l'erreur. Cette méthode permet donc d'allier les avantages de chaque méthode, efficacité d'un côté et garanties de décroissance de l'autre.

Pour des problèmes à grande dimension, et lorsque le jacobien $\mathbb{J}$ a une forte sparsité, on peut utiliser une implémentation sparse de Levenberg-Marquardt disponible sur le web `http://www.ics.forth.gr/~lourakis/sba/` et basée sur un article de M.Lourakis *et al.* [LA09].

B.3 Optimisation discrète : Les Graph-cut

Dans cette section, nous introduisons le problème de la coupe s-t minimale dans un graphe. Soit $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ un graphe orienté avec des noeuds $\mathcal{V}$ et des arêtes $\mathcal{E}$. Il y a également deux noeuds *terminaux* dans $\mathcal{G}$: la source (en anglais source) s et le puits (en anglais sink) t. Nous assignons à chaque arête allant de p vers q une capacité (poids positif) $c(p, q)$.

Une *coupe* de $\mathcal{G}$ est une partition de $\mathcal{V}$ en deux ensembles disjoints $\mathcal{S}$ et $\mathcal{T}$. Elle est appelée coupe s-t $\mathcal{C} = (\mathcal{S}, \mathcal{T})$ quand $s \in \mathcal{S}$ et $t \in \mathcal{T}$. La Figure B.1 présente un exemple d'une coupe s-t dans un graphe. Un ensemble de coupe $\mathcal{C}$ est l'ensemble des arêtes coupées allant depuis un noeud du côté de la source vers un noeud du côté du puits, la capacité ou le coût de $\mathcal{C}$ est la somme des poids des arêtes dans cet ensemble de coupe :

$$c(\mathcal{S},\mathcal{T}) = \sum_{(p,q)\in\mathcal{E},p\in\mathcal{S},q\in\mathcal{T}} c_{pq} \tag{B.6}$$

Trouver la coupe s-t dans un graphe qui a la plus petite capacité est très intéressant dans de nombreux domaines. En effet, en considérant le coût d'une coupe comme une fonction d'énergie, le problème de la coupe minimale est en fait un problème de minimisation binaire. Ceci est particulièrement intéressant lorsque l'énergie est composée d'un terme d'attache aux *données* et d'un terme de *régularisation*. Le premier représenté par les arêtes entre les noeuds et les noeuds terminaux, et le second représenté par les arêtes entre les noeuds non terminaux.

Il a été montré que ce problème est équivalent à trouver le *flux* maximal depuis s vers t (qui est intuitivement le problème de trouver la quantité maximale d'eau qui peut aller de la source vers le puits, en passant au travers des arêtes orientées, sans excéder la capacité de chaque arête) [FF62] : Le flux maximal est égal au coût de la coupe minimale. Des méthodes polynomiales en temps ont été proposées pour trouver l'optimum global. Le lecteur intéressé peut se référer à [PCF05, KZ04] pour une introduction aux algorithmes de graph cuts et une description des types de fonctions pouvant être minimisables globalement par graph cuts.

Une large variété de problèmes de vision par ordinateur peut être formulée en terme de minimisation d'énergie. En particulier, parmi ceux là, beaucoup se réduisent à un problème discret de coupe minimale dans un graphe [GPS]. Souvent, pour ces problèmes, l'image ou l'espace 3D sont représentés par une grille régulière correspondant aux noeuds et aux arêtes du graphe. Cependant, des travaux plus récents utilisent les graph cuts non plus sur des grilles régulières mais sur des complexes [KG04]. Le lecteur peut se référer à l'annexe D pour la définition d'un complexe et un exemple d'une telle minimisation sur un complexe appliquée à la stéréovision. L'idée est de considérer les partitions d'un graphe comme des étiquettes sur les cellules d'un complexe, une surface orientée est identifiée à l'aide d'un vecteur étiquetant : une facette appartient à la surface si elle est entre une cellule étiquetée *intérieure* et une cellule étiquetée *extérieure*.

Annexe C

Preuves spécifiques à nos méthodes de calibration linéaire

C.1 Preuve de la proposition 14

Effectuons la preuve de la proposition 14. Nous commençons par considérer le cas où les 3 lignes des matrices fondamentales $\mathtt{F}^{ij}$ et $\mathtt{F}^{ik}$ sont différentes du vecteur nul de $\mathbb{R}^3$. Ceci implique que les colonnes des matrices $\mathtt{A}_t^{ij}$ et $\mathtt{A}_t^{ij}$ forment deux bases orthogonales de $\mathbb{R}^3$. En effet, l'épipole $\mathbf{e}^{ij}$ est orthogonale aux lignes de $\mathtt{F}^{ij}$ (comme nous l'avons vu en section 1.3.1.1), et $(\mathtt{F}_{t,1:3}^{ij})^\top \times \mathbf{e}^{ij}$ est par construction orthogonal à $\mathtt{F}_{t,1:3}^{ij}$ et $\mathbf{e}^{ij}$. Ainsi, $\mathtt{A}_t^{ik}$ et $\mathtt{A}_t^{ij}$ sont inversibles et le Tenseur Trifocal peut toujours s'écrire

$$\begin{bmatrix} a_t & b_t & c_t \\ d_t & e_t & f_t \\ g_t & h_t & i_t \end{bmatrix} = (\mathtt{A}_t^{ij})^{-1} \mathtt{T}_t^{ijk} (\mathtt{A}_t^{ik})^{-\top}. \tag{C.1}$$

Montrons que $a_t = b_t = c_t = 0$. Rappelons que comme nous l'avons montré dans la section 8, les matrices $\mathtt{T}_t^{ijk}$ lient un point $\mathbf{p} = (p_1, p_2, p_3)$ dans la première image à sa droite épipolaire dans la seconde image $\mathbf{l} = \mathtt{F}^{ij\top}\mathbf{p}$ au travers de l'équation $\mathbf{l}^\top \sum_{s=1}^{3} p_s \mathtt{T}_s^{ijk} = \mathbf{0}^\top$. Choisissant $\mathbf{p}$ comme étant le vecteur $(\delta_{t1}, \delta_{t2}, \delta_{t3})^\top$, où $\delta_{t\ell}$ représente le symbole de Kronecker qui est égal à un si $t = \ell$ et zéro sinon, nous obtenons $\mathtt{F}_{t,1:3}^{ij} \mathtt{T}_t^{ijk} = \mathbf{0}^\top$. En conjonction avec (C.1), la définition de $\mathtt{A}_t^{ij}$ et l'inversibilité de $\mathtt{A}_t^{ik}$, cette équation devient

$$\begin{bmatrix} a_t & d_t & g_t \\ b_t & e_t & h_t \\ c_t & f_t & i_t \end{bmatrix} \begin{bmatrix} \|(\mathtt{F}_{t,1:3}^{ij})^\top\|^2 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Ceci implique $a_t = b_t = c_t = 0$. Par l'argument transposé, avec la seconde relation entre un point et sa droite épipolaire vue en proposition 8, nous obtenons également $d_t = g_t = 0$. Ainsi, le Tenseur trifocal se réduit nécessairement à la forme :

$$\mathtt{T}_t^{ijk} = \mathtt{A}_t^{ij} \begin{bmatrix} 0 & 0 & 0 \\ 0 & e_t & f_t \\ 0 & h_t & i_t \end{bmatrix} (\mathtt{A}_t^{ik})^\top \tag{C.2}$$

Puisque le rang d'une matrice fondamentale est toujours égal à deux, il existe un indice $t' \in \{1,2,3\}$ tel que $\mathtt{F}^{ij}_{t',1:3}$ et $\mathtt{F}^{ij}_{t,1:3}$ ne sont pas colinéaires. Nous avons déjà vérifié que $\mathtt{F}^{ij}_{t,1:3}\mathtt{T}^{ijk}_{t} = \mathbf{0}^\top$ et, de façon similaire, $\mathtt{F}^{ij}_{t',1:3}\mathtt{T}^{ijk}_{t'} = \mathbf{0}^\top$. Ainsi, en substituant $\mathbf{p} = (\delta_{t1}, \delta_{t2}, \delta_{t3})^\top + (\delta_{t'1}, \delta_{t'2}, \delta_{t'3})^\top$ dans l'équation $\mathbf{p}^\top \mathtt{F}^{ij} \sum_{s=1}^{3} p_s \mathtt{T}^{ijk}_{s} = 0$, nous avons

$$\mathtt{F}^{ij}_{t',1:3}\mathtt{T}^{ijk}_{t} + \mathtt{F}^{ij}_{t,1:3}\mathtt{T}^{ijk}_{t'} = \mathbf{0}^\top. \tag{C.3}$$

Maintenant, observons que $(\mathtt{F}^{ij}_{t',1:3})^\top((\mathtt{F}^{ij}_{t,1:3})^\top \times \mathbf{e}^{ij}) = -(\mathtt{F}^{ij}_{t,1:3})^\top((\mathtt{F}^{ij}_{t',1:3})^\top \times \mathbf{e}^{ij}) := \beta_{t,t'}$. De plus, $\beta_{t,t'} \neq \mathbf{0}$ puisque les vecteurs $\mathtt{F}^{ij}_{t,1:3}$ et $\mathtt{F}^{ij}_{t',1:3}$ sont linéairement indépendants et orthogonaux à $\mathbf{e}^{ij}$. Cette observation, en conjonction avec les équations (C.2) et (C.3) mène à

$$\mathtt{A}^{ik}_{t}\begin{bmatrix}0 & 0 & 0\\ 0 & e_t & h_t\\ 0 & f_t & i_t\end{bmatrix}\begin{bmatrix}\alpha\\ \beta\\ 0\end{bmatrix} + \mathtt{A}^{ik}_{t'}\begin{bmatrix}0 & 0 & 0\\ 0 & e_{t'} & h_{t'}\\ 0 & f_{t'} & i_{t'}\end{bmatrix}\begin{bmatrix}\alpha\\ -\beta\\ 0\end{bmatrix} = \mathbf{0},$$

Avec les abréviations $\alpha = \alpha_{t,t'}$ et $\beta = \beta_{t,t'}$. Une transposition et une multiplication donnent
$\beta_{t,t'}\Big(\begin{bmatrix}0, & e_t, & f_t\end{bmatrix}(\mathtt{A}^{ik}_{t})^\top - \begin{bmatrix}0, & e_{t'}, & f_{t'}\end{bmatrix}(\mathtt{A}^{ik}_{t'})^\top\Big) = \mathbf{0}^\top.$
Ce qui revient à
$(e_t\mathtt{F}^{ik}_{t,1:3} - e_{t'}\mathtt{F}^{ik}_{t',1:3})^\top \times \mathbf{e}^{ik} + (f_t - f_{t'})\mathbf{e}^{ik} = \mathbf{0},$
ceci est possible si et seulement si $f_t = f_{t'}$ et $e_t\mathtt{F}^{ik}_{t,1:3} - e_{t'}\mathtt{F}^{ik}_{t',1:3} = \mathbf{0}$. Puisque $\mathtt{F}^{ik}_{t,1:3}$ et $\mathtt{F}^{ik}_{t',1:3}$ sont linéairement indépendants, nous en déduisons que $e_t = e_{t'} = 0$. De plus, en utilisant l'argument transposé, nous obtenons $h_t = h_{t'}$ et donc

$$\mathtt{T}^{ijk}_{t} = \mathtt{A}^{ij}_{t}\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & f\\ 0 & h & i_t\end{bmatrix}(\mathtt{A}^{ik}_{t})^\top, \mathtt{T}^{ijk}_{t'} = \mathtt{A}^{ij}_{t'}\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & f\\ 0 & h & i_{t'}\end{bmatrix}(\mathtt{A}^{ik}_{t'})^\top$$

Posons maintenant t'' comme étant l'élément de l'ensemble $\{1,2,3\}$ qui est différent de t et t'. En reproduisant le même argument où t est remplacé par t'' nous obtenons une formule similaire à (C.14) dans laquelle t' est remplacé par t'' partout. La preuve de la proposition vient en divisant toutes les entrées de $[\mathtt{T}^{ijk}]$ par f, qui est différent de 0. En effet si $f = 0$ alors la matrice fondamentale calculée à partir de ce tenseur trifocal (comme indiqué en section 1.3.2.3) serait la matrice nulle de rang $r < 2$, ce qui est impossible. Donc dans la formulation non homogène il y a uniquement 4 inconnues correspondant aux degrés de liberté restants, et dans la formulation homogène il y a 5 inconnues.

Dans le cas où les matrices $\mathtt{F}^{ij}$ ou $\mathtt{F}^{ik}$ contiennent des lignes nulles, on peut utiliser le fait que l'équation 2.2 caractérise tous les triplets de caméras possibles (à une transformation projective près) qui sont compatibles avec les matrices fondamentales $\mathtt{F}^{ij}$ et $\mathtt{F}^{ik}$. Au vu de [HZ03, Eq. 15.1], le Tenseur Trifocal correspondant à ces caméras coïncide avec celui défini dans la proposition. Ceci complète la preuve. Maintenant on peut également étudier les cas particuliers où une ligne des matrices fondamentales est nulle, de façon similaire et voir que la proposition reste également vraie dans ce cas. En effet, si une ligne est nulle, on peut utiliser pour le changement de base du début, une des lignes non nulles de la matrice fondamentale (au lieu de la ligne nulle qui mène à une matrice dégénérée). Alors, avec des considérations

similaires en utilisant des points et droites épipolaires bien choisies, on peut facilement montrer que des termes supplémentaires s'annulent, et que remplacer la ligne non nulle choisie par la ligne nulle initiale mène à la même formulation.

Dans ce manuscrit on pourra alternativement choisir la formulation non homogène ou homogène, qui s'écrivent :

$$\mathtt{T}_t^{ijk} = \mathtt{A}_t^{ij} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \gamma_0' \\ 0 & 1 & \gamma_t \end{bmatrix} (\mathtt{A}_t^{ik})^\top, \mathtt{T}_t^{ijk} = \mathtt{A}_t^{ij} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \gamma_0' \\ 0 & \gamma_0 & \gamma_t \end{bmatrix} (\mathtt{A}_{t'}^{ik})^\top$$

C.2 Preuves des équations (2.4) et (2.5)

Donnons la preuve de l'équation (2.4). Soient $\mathtt{P}^i$ et $\mathtt{P}^k$ les matrices des caméras calculées à partir du triplet (i, j, k) avec les paramètres $\boldsymbol{\gamma}$, et $\bar{\mathtt{P}}^i$ et $\bar{\mathtt{P}}^k$ les caméras calculées à partir du triplet (k, i, ℓ) avec les paramètres $\underline{\boldsymbol{\gamma}}$. Ainsi, nous cherchons l'homographie $\underline{\mathtt{H}}$ telle que dans la formulation non homogène nous avons $\bar{\mathtt{P}} = \mathtt{P}\underline{\mathtt{H}}$:

$$[\mathtt{I}_{3\times3}|\mathbf{0}]\underline{\mathtt{H}} \cong \text{kron}([\underline{\boldsymbol{\gamma}_{1:3}}, \underline{\gamma_0}], \mathbf{e}^{ki}) - \underline{\gamma_0'}\left[[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik}|\mathbf{0}\right], \tag{C.4}$$

$$\left[[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki}|\mathbf{e}^{ik}\right]\underline{\mathtt{H}} \cong [\mathtt{I}_{3\times3}|\mathbf{0}], \tag{C.5}$$

La première équation mène à $\underline{\mathtt{H}}_{1:3,1:4} = \left[\text{kron}([\underline{\boldsymbol{\gamma}_{1:3}}, \underline{\gamma_0}], \mathbf{e}^{ki}) - \underline{\gamma_0'}[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik}|\mathbf{0}_{3\times1}\right]$. En insérant ce résultat dans l'équation (C.8) et en utilisant le fait que $\mathtt{F}^{ki}\mathbf{e}^{ki} = \mathbf{0}$, nous obtenons

$$-\underline{\gamma_0'}[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki}[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} + \mathbf{e}^{ik}\underline{\mathtt{H}}_{4,1:3} = \alpha \mathtt{I}_{3\times3} \tag{C.6}$$

et $\mathbf{e}^{ik}\underline{\mathtt{H}}_{4,4} = 0$. Ceci implique que $\underline{\mathtt{H}}_{4,4} = 0$. De plus en multipliant les parties gauche et droite de l'équation (C.6) par $(\mathbf{e}^{ik})^\top$, nous avons $\underline{\mathtt{H}}_{4,1:3} = \alpha(\mathbf{e}^{ik})^\top$. Afin de terminer la preuve il reste à déterminer la valeur de α. Ceci est effectué en calculant la trace de chaque côté dans l'équation (C.6).

Maintenant nous pouvons calculer la transformation projective inverse explicitée dans l'équation (2.5), $\mathtt{H}_{\boldsymbol{\gamma},\underline{\boldsymbol{\gamma}}}$ telle que $\mathtt{P} = \bar{\mathtt{P}}\mathtt{H}$:

$$[\mathtt{I}_{3\times3}|\mathbf{0}] \cong \left(\text{kron}([\underline{\boldsymbol{\gamma}_{1:3}}, \underline{\gamma_0}], \mathbf{e}^{ki}) - \underline{\gamma_0'}\left[[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik}|\mathbf{0}\right]\right)\mathtt{H}, \tag{C.7}$$

$$\left[[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki}|\mathbf{e}^{ik}\right] \cong [\mathtt{I}_{3\times3}|\mathbf{0}]\mathtt{H}, \tag{C.8}$$

La première équation mène à $\mathtt{H}_{1:3,1:4} = \left[[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki}|\mathbf{e}^{ik}\right]$, en insérant ceci dans la première équation, nous obtenons à partir de la dernière colonne : $\underline{\boldsymbol{\gamma}_{1:3}} \cdot \mathbf{e}^{ik}\,\mathbf{e}^{ki} + \mathtt{H}_{4,4}\mathbf{e}^{ki} = 0$ c'est-à-dire $\mathtt{H}_{4,4} = -\underline{\boldsymbol{\gamma}_{1:3}} \cdot \mathbf{e}^{ik}$ A partir des 3 premières colonnes nous avons :

$$-\underline{\gamma_0'}[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik}[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} + \mathbf{e}^{ki}\underline{\boldsymbol{\gamma}_{1:3}^T}[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} + \mathbf{e}^{ki}\mathtt{H}_{4,1:3} = \lambda\underline{\gamma_0'}\mathtt{I}_{3\times3} \tag{C.9}$$

En multipliant chaque côté de l'équation par $[\mathbf{e}^{ki}]_\times\,[\mathbf{e}^{ki}]_\times$, et en calculant une fois encore la trace, nous avons : $\lambda = \frac{1}{2}\text{tr}\left([\mathbf{e}^{ki}]_\times[\mathbf{e}^{ki}]_\times[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik}[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki}\right)$ en effet, les épipoles sont de normes unitaires et donc trace $\left([\mathbf{e}^{ki}]_\times[\mathbf{e}^{ki}]_\times\right) = -2$. En multipliant chaque côté de l'équation (C.9) par ${\mathbf{e}^{ki}}^\top$ nous obtenons facilement $\mathtt{H}_{4,1:3}$ ce qui complète la preuve.

C.3 Preuve de la proposition 15

D'après l'équation (2.11) de ce manuscrit, nous avons

$$\mathtt{P}^{2,\gamma_i}\mathtt{H}^{v_i} \cong \mathtt{P}^{i+1,*}, \qquad \mathtt{P}^{3,\gamma_i}\mathtt{H}^{v_i} \cong \mathtt{P}^{i+2,*} \tag{C.10}$$

$$\mathtt{P}^{1,\gamma i+1}\mathtt{H}^{v_{i+1}} \cong \mathtt{P}^{i+1,*}, \qquad \mathtt{P}^{2,\gamma i+1}\mathtt{H}^{v_{i+1}} \cong \mathtt{P}^{i+2,*}. \tag{C.11}$$

Donc,

$$\mathtt{P}^{2,\gamma_i}\mathtt{H}^{v_i} \cong \mathtt{P}^{1,\gamma i+1}\mathtt{H}^{v_{i+1}}, \qquad \mathtt{P}^{3,\gamma_i}\mathtt{H}^{v_i} \cong \mathtt{P}^{2,\gamma i+1}\mathtt{H}^{v_{i+1}}. \tag{C.12}$$

De plus d'après l'équation (2.12) de ce manuscrit,

$$\mathtt{P}^{1,\gamma_{i+1}} \cong \mathtt{P}^{2,\gamma_i}\mathtt{H}^{i,i+1}, \quad \mathtt{P}^{2,\gamma_{i+1}} \cong \mathtt{P}^{3,\gamma_i}\mathtt{H}^{i,i+1}. \tag{C.13}$$

En substituant (C.13) dans (C.12), nous avons

$$\mathtt{P}^{2,\gamma_i}\mathtt{H}^{v_i} \cong \mathtt{P}^{2,\gamma_i}\mathtt{H}^{i,i+1}\mathtt{H}^{v_{i+1}}, \qquad \mathtt{P}^{3,\gamma_i}\mathtt{H}^{v_i} \cong \mathtt{P}^{3,\gamma_i}\mathtt{H}^{i,i+1}\mathtt{H}^{v_{i+1}}. \tag{C.14}$$

Si les centres des caméras $\mathtt{P}^{2,\gamma_i}$ et $\mathtt{P}^{3,\gamma_i}$ ne sont pas confondus, ce qui est équivalent à $\mathtt{P}^{i+1,*}, \mathtt{P}^{i+2,*}$ ont des centres différents, alors l'équation (C.14) peut être satisfaite si et seulement si $\mathtt{H}^{v_i} \cong \mathtt{H}^{i,i+1}\mathtt{H}^{v_{i+1}}$.

Pour prouver i) et ii), nous avons besoin de transformer les relations de proportionnalité en relations d'égalité. Puisque $\mathtt{H}^{v_i} = \mathtt{H}^{i,i+1}\mathtt{H}^{v_{i+1}}$, il existe un nombre réel $\alpha_i \neq 0$ tel que $\mathtt{H}^{v_i} = \alpha_i\mathtt{H}^{i,i+1}\mathtt{H}^{v_{i+1}}$. Donc nous avons

$$\mathtt{H}^{v_1} = \alpha_1\mathtt{H}^{1,2}\mathtt{H}^{v_2}, \tag{C.15}$$

$$\mathtt{H}^{v_2} = \alpha_2\mathtt{H}^{2,3}\mathtt{H}^{v_3}, \tag{C.16}$$

$$\vdots \tag{C.17}$$

$$\mathtt{H}^{v_n} = \alpha_n\mathtt{H}^{n,1}\mathtt{H}^{v_1}. \tag{C.18}$$

Posons $\tilde{\mathtt{H}}^{v_1} = \mathtt{H}^{v_1}$, $\tilde{\mathtt{H}}^{v_2} = \alpha_1\mathtt{H}^{v_2}$, $\tilde{\mathtt{H}}^{v_3} = \alpha_1\alpha_2\mathtt{H}^{v_3}$, …, $\tilde{\mathtt{H}}^{v_n} = \alpha_1 \times \ldots \times \alpha_{n-1}\mathtt{H}^{v_n}$. Chaque $\tilde{\mathtt{H}}^{v_i}$ étant proportionnel à $\mathtt{H}^{v_i}$, ils satisfont l'équation (2.11) du manuscrit. Ceci nous amène à l'affirmation i) de la proposition 15, puisque $\tilde{\mathtt{H}}^{v_i} = \mathtt{H}^{i,i+1}\tilde{\mathtt{H}}^{v_{i+1}}$ pour $i = 1, \ldots, n-1$. De plus nous avons $\tilde{\mathtt{H}}^{v_n} = \left(\prod_{i=1}^{n}\alpha_i\right)\mathtt{H}^{n,1}\tilde{\mathtt{H}}^{v_1} \overset{\text{notation}}{=} \alpha^{-1}\mathtt{H}^{n,1}\tilde{\mathtt{H}}^{v_1}$.

Ce qui implique que

$$\prod_{i=1}^{n}\tilde{\mathtt{H}}^{v_i} = \left(\prod_{i=1}^{n-1}\mathtt{H}^{i,i+1}\tilde{\mathtt{H}}^{v_{i+1}}\right)\left(\prod_{i=1}^{n}\alpha_i\right)\mathtt{H}^{n,1}\tilde{\mathtt{H}}^{v_1} \qquad = \left(\prod_{i=1}^{n}\mathtt{H}^{i,i+1}\right)\left(\prod_{i=1}^{n}\alpha_i\right)\prod_{i=1}^{n}\tilde{\mathtt{H}}^{v_i},$$

qui est équivalent à $\alpha \mathtt{I}_{4\times4} = \prod_{i=1}^{n} \mathtt{H}^{i,i+1}$. En prenant la trace de chaque côté, nous avons l'expression souhaitée pour α ce qui complète la preuve de l'affirmation ii).

Afin de prouver iii), remarquons simplement que toutes les homographies $\mathtt{H}^{v_i}$ sont définies à une transformation projective près. En d'autres termes, nous pouvons remplacer tous les $\mathtt{H}^{v_i}$ par $\mathtt{H}^{v_i}\mathtt{Q}$, où $\mathtt{Q}$ est une matrice inversible de taille 4×4. Soit $\bar{\mathtt{H}}$ la matrice de taille $4n\times4$ résultant des concaténations verticales des matrices $\mathtt{H}^{v_i}$ (satisfaisant les conditions i) et ii) de la proposition). Soit $\bar{\mathtt{H}}^\top\bar{\mathtt{H}} = \mathtt{U}\mathtt{L}\mathtt{U}^\top$ la SVD de $\bar{\mathtt{H}}$. Ainsi, $\mathtt{L}$ est une matrice diagonale de taille 4×4 avec des entrées diagonales strictement positives, et $\mathtt{U}$ est une matrice orthogonale de taille 4×4. Ainsi, $\mathtt{U}$ est une homographie. En fixant $\mathtt{Q} = \mathtt{U}$, nous avons une nouvelle version des homographies $\mathtt{H}^{v_i}$ qui satisfait toutes les conditions de la proposition 15.

C.4 Preuve de la proposition 17

Le triplet de caméras correspondant à $\mathcal{T}^{ijk}$ est :

$$\begin{aligned} \mathtt{P}^i &= [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^k = [[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} \,|\, \mathbf{e}^{ik}], \\ \mathtt{P}^j &= \mathrm{kron}\big([\boldsymbol{\gamma}_{1:3}, 1]; \mathbf{e}^{ij}\big) - \gamma'_0 [[\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} \,|\, \mathbf{0}_{3\times1}]. \end{aligned} \tag{C.19}$$

Il existe une transformation projective telle que $\mathtt{P}'^i = \mathtt{P}^i\mathtt{H}$, $\mathtt{P}'^k = \mathtt{P}^k\mathtt{H}$ avec :

$$\begin{aligned} \mathtt{P}'^i &= [[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} \,|\, \mathbf{e}^{ki}], \qquad \mathtt{P}'^k = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \\ \mathtt{P}'^j &= \big(\mathrm{kron}\big([\boldsymbol{\gamma}_{1:3}, 1]; \mathbf{e}^{ij}\big) - \gamma'_0 [[\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} \,|\, \mathbf{0}_{3\times1}]\big)\, \mathtt{H}. \end{aligned} \tag{C.20}$$

L'équation $[[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} \,|\, \mathbf{e}^{ki}] = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}]\mathtt{H}$, mène à $\mathtt{H}_{1:3,1:4} = \big[[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} | \mathbf{e}^{ki}\big]$.
L'autre équation est :

$$\beta[\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}] = [[\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} \,|\, \mathbf{e}^{ik}] \begin{bmatrix} [\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} & | & \mathbf{e}^{ki} \\ \mathbf{v}_{1\times3} & | & v_4 \end{bmatrix} \tag{C.21}$$

L'égalité sur la dernière colonne implique $v_4 = 0$. Les 3 premières colonnes mènent à $\beta\mathtt{I}_{3\times3} = [\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} [\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} + \mathbf{e}^{ik}\mathbf{v}_{1\times3}$. En multipliant chaque côté de l'équation à gauche par $\mathbf{e}^{ik^T}$ nous avons $\mathbf{v}_{1\times3} = \beta\mathbf{e}^{ik^T}$. L'équation devient, $\beta\mathtt{I}_{3\times3} = [\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} [\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} + \beta\mathbf{e}^{ik}\mathbf{e}^{ik^T}$. En calculant la trace nous avons $\beta = \frac{1}{2}\mathrm{trace}\left([\mathbf{e}^{ik}]_\times \mathtt{F}^{ki} [\mathbf{e}^{ki}]_\times \mathtt{F}^{ik}\right)$. Le triplet de caméras dans un autre espace projectif s'écrit donc :

$$\begin{aligned} \mathtt{P}'^i &= [[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} \,|\, \mathbf{e}^{ki}], \qquad \mathtt{P}'^k = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \\ \mathtt{P}'^j &= \big(\mathrm{kron}\big([\boldsymbol{\gamma}_{1:3}, 1]; \mathbf{e}^{ij}\big) - \gamma'^0{}_0 [[\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} \,|\, \mathbf{0}_{3\times1}]\big) \begin{bmatrix} [\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} & | & \mathbf{e}^{ki} \\ \beta\mathbf{e}^{ik^T} & | & 0 \end{bmatrix}. \end{aligned} \tag{C.22}$$

$\mathtt{P}'^j$ ayant la forme $[\mathtt{M}_{3\times3} \,|\, \mathbf{m}_{3\times1}]$, nous pouvons calculer la matrice fondamentale à un facteur d'échelle près $\mathtt{F}^{jk} = [\mathbf{m}]_\times \mathtt{M}$. Elle est linéaire en fonction de $\boldsymbol{\gamma}$ et s'écrit comme indiqué dans la proposition 17.

C.5 Preuve de la proposition 16

Si l'on considère un ensemble de tenseurs trifocaux cohérents et connectés, en adéquation avec notre proposition 14 :

$$\left(\left[\mathcal{T}^{opn}, \boldsymbol{\gamma}^{-K}\right] \ \ldots \ \left[\mathcal{T}^{ijk}, \boldsymbol{\gamma}^{0}\right] \ \ldots \ \left[\mathcal{T}^{lmh}, \boldsymbol{\gamma}^{N}\right]\right)$$

Nous pouvons calculer les triplets de caméras $\mathtt{P}^i$, $\mathtt{P}^l$, $\mathtt{P}^o$ dans l'espace projectif du Tenseur Trifocal $\left[\mathcal{T}^{ijk}, \boldsymbol{\gamma}^{0}\right]$, nous avons évidemment $\mathtt{P}^i = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}]$ à partir des caméras liées au Tenseur $\mathcal{T}^{ijk}$ (voir équations (2.2)). Et avec la connaissance de la transformation projective entre deux triplets de caméras consécutifs, nous pouvons écrire dans la formulation non homogène :

$$\begin{gathered}\mathtt{P}^i = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^l = [[\mathbf{e}^{ml}]_\times \mathtt{F}^{lm} \,|\, \mathbf{e}^{ml}] \prod_{i=N-2}^{0} \mathtt{H}_{\boldsymbol{\gamma}^i, \boldsymbol{\gamma}^{i+1}}, \\ \mathtt{P}^o = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}] \prod_{i=-K}^{-1} \underline{\mathtt{H}}_{\boldsymbol{\gamma}^i, \boldsymbol{\gamma}^{i+1}}.\end{gathered} \tag{C.23}$$

En effet nous retrouvons $\mathtt{P}^l$ à partir du Tenseur trifocal lié à $\boldsymbol{\gamma}^{N-1}$ et la transportons dans l'espace projectif du tenseur $\left[\mathcal{T}^{ijk}, \boldsymbol{\gamma}^{0}\right]$, avec l'équation de l'homographie (2.5). Et nous retrouvons $\mathtt{P}^o$ à partir du tenseur trifocal $\left[\mathcal{T}^{opn}, \boldsymbol{\gamma}^{-K}\right]$ et la transportons dans l'espace projectif du tenseur $\left[\mathcal{T}^{ijk}, \boldsymbol{\gamma}^{0}\right]$, avec l'équation de l'homographie inverse (2.4).

Comme nous pouvons le constater, $\mathtt{P}^l$ ne dépend pas de $\boldsymbol{\gamma}^{-K}$, $\boldsymbol{\gamma}^0$ et $\boldsymbol{\gamma}^N$. Aussi cette matrice sera simplement notée, $\mathtt{P}^l = [\mathtt{M}_{3\times3} \,|\, \mathbf{m}_{3\times1}]$. Et nous la normalisons de telle sorte que $||\mathbf{m}|| = 1$. En effet celà correspond à l'épipole et notre formulation du Tenseur Trifocal a été effectuée pour des épipoles de norme unitaire.

Nous remarquons également que $[\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}] \prod_{i=-K}^{-2} \underline{\mathtt{H}}_{\boldsymbol{\gamma}^i, \boldsymbol{\gamma}^{i+1}}$ ne dépend pas de $\boldsymbol{\gamma}^{-K}$, $\boldsymbol{\gamma}^0$ et $\boldsymbol{\gamma}^N$. Aussi elle sera simplement notée $[\mathtt{M}'_{3\times3} \,|\, \mathbf{m}'_{3\times1}]$.

Finalement le triplet de caméras de l'équation (C.23) est :

$$\mathtt{P}^i = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^l = [\mathtt{M}_{3\times3} \,|\, \mathbf{m}_{3\times1}], \qquad \mathtt{P}^o = [\mathtt{M}'_{3\times3} \,|\, \mathbf{m}'_{3\times1}] \, \underline{\mathtt{H}}_{\boldsymbol{\gamma}^{-1}, \boldsymbol{\gamma}^0}.$$

Comme établi dans [HZ03] et montré également dans ce manuscrit en section 1.3.1 , la matrice fondamentale associée aux caméras $\mathtt{P}^i$ et $\mathtt{P}^l$ est $\mathtt{F}^{li} = [\mathbf{m}]_\times \mathtt{M}$ et l'épipole est $\mathbf{e}^{il} = \mathbf{m}$.

Nous pouvons écrire le triplet de caméras dans un espace projectif tel que : $\mathtt{P}'^i = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}]$ and $\mathtt{P}'^l = [[\mathbf{e}^{il}]_\times \mathtt{F}^{li} \,|\, \mathbf{e}^{il}]$, avec $\mathtt{P}'^i = \mathtt{P}^i\mathtt{H}$ et $\mathtt{P}'^l = \mathtt{P}^l\mathtt{H}$. Grâce à la première équation nous savons que $\mathtt{H}$ a la forme simplifiée :

$$\mathtt{H} = \begin{bmatrix} \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ \mathbf{a}_{1:3} & | & a_4 \end{bmatrix}. \tag{C.24}$$

L'équation restante est $[[\mathbf{e}^{il}]_\times \mathtt{F}^{li} \,|\, \mathbf{e}^{il}] = [[\mathbf{m}]_\times [\mathbf{m}]_\times \mathtt{M} \,|\, \mathbf{m}] \cong [\mathtt{M}_{3\times3} \,|\, \mathbf{m}_{3\times1}]\mathtt{H}$. En utilisant l'équation du produit vectoriel triple $[\mathbf{m}]_\times [\mathbf{m}]_\times \mathtt{M}_{:,i} = \mathbf{m}\,(\mathbf{m} \cdot \mathtt{M}_{:,i}) - \mathtt{M}_{:,i}$ nous déduisons évidemment que $a_4 = -1$ et $\mathbf{a}_{1:3} = -\mathbf{m}^T\mathtt{M}$. Le triplet de caméras peut s'écrire :

$$\begin{gathered}\mathtt{P}^i = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^l = [[\mathbf{e}^{il}]_\times \mathtt{F}^{li} \,|\, \mathbf{e}^{il}], \\ \mathtt{P}^o = [\mathtt{M}'_{3\times3} \,|\, \mathbf{m}'_{3\times1}] \, \underline{\mathtt{H}}_{\boldsymbol{\gamma}^{-1}, \boldsymbol{\gamma}^0} \begin{bmatrix} \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ \mathbf{a}_{1\times3} & | & a_4 \end{bmatrix}.\end{gathered} \tag{C.25}$$

Nous allons maintenant retrouver la matrice fondamentale $\mathtt{F}^{oi}$ et montrer qu'elle ne dépend pas de $\boldsymbol{\gamma}^{-K}$, $\boldsymbol{\gamma}^0$ et $\boldsymbol{\gamma}^N$. Cette matrice fondamentale correspond à la paire de caméras :

$$\mathtt{P}^i = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^o = [\mathtt{M}'_{3\times3} \,|\, \mathbf{m}'_{3\times1}]\, \underline{\mathtt{H}}_{\boldsymbol{\gamma}^{-1},\boldsymbol{\gamma}^0} \tag{C.26}$$

C'est-à-dire,

$$\begin{aligned} \mathtt{P}^i &= [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \\ \mathtt{P}^o &= [\mathtt{M}'_{3\times3} \,|\, \mathbf{m}'_{3\times1}] \begin{bmatrix} [\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} & | & \mathbf{e}^{ij} \\ -\mu \mathbf{e}^{ji} & | & 0 \end{bmatrix} \begin{bmatrix} -{\boldsymbol{\gamma}'^0}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ {\boldsymbol{\gamma}^0}_{1:3} & | & 1 \end{bmatrix} \end{aligned} \tag{C.27}$$

Les deux matrices gauches de la décomposition de la caméra $\mathtt{P}^o$ ne dépendent pas de $\boldsymbol{\gamma}^{-K}$, $\boldsymbol{\gamma}^0$ et $\boldsymbol{\gamma}^N$, aussi nous pouvons fusionner ces deux matrices sous la forme fixe $[\mathtt{M}''_{3\times3} \,|\, \mathbf{m}''_{3\times1}]$. Nous en déduisons que le triplet de caméras dans l'équation (C.25) peut s'écrire :

$$\begin{aligned} \mathtt{P}^i = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^l &= [[\mathbf{e}^{il}]_\times \mathtt{F}^{li} \,|\, \mathbf{e}^{il}], \\ \mathtt{P}^o = [\mathtt{M}''_{3\times3} \,|\, \mathbf{m}''_{3\times1}] \begin{bmatrix} -{\boldsymbol{\gamma}'^0}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ {\boldsymbol{\gamma}^0}_{1:3} & | & 1 \end{bmatrix} &\begin{bmatrix} \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ \mathbf{a}_{1\times3} & | & a_4 \end{bmatrix} \end{aligned} \tag{C.28}$$

De plus en normalisant $[\mathtt{M}''_{3\times3} \,|\, \mathbf{m}''_{3\times1}]$ de sorte que $||\mathbf{m}''|| = 1$, nous obtenons la matrice fondamentale $\mathtt{F}^{oi} = [\mathbf{m}'']_\times \mathtt{M}''$ et l'épipole de norme unitaire $\mathbf{e}^{io} = \mathbf{m}''$. De plus avec des calculs similaires à ceux menant à l'équation (C.24), et en remarquant que $\mathtt{H}^{-1} = \mathtt{H}$, nous pouvons écrire le triplet de caméras :

$$\begin{aligned} \mathtt{P}^i = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^l &= [[\mathbf{e}^{il}]_\times \mathtt{F}^{li} \,|\, \mathbf{e}^{il}], \\ \mathtt{P}^o = [[\mathbf{e}^{io}]_\times \mathtt{F}^{oi} \,|\, \mathbf{e}^{io}] \begin{bmatrix} \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ \mathbf{a}'_{1\times3} & | & a'_4 \end{bmatrix} &\begin{bmatrix} -{\boldsymbol{\gamma}'^0}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ {\boldsymbol{\gamma}^0}_{1:3} & | & 1 \end{bmatrix} \begin{bmatrix} \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ \mathbf{a}_{1\times3} & | & a_4 \end{bmatrix} \end{aligned} \tag{C.29}$$

Avec $a'_4 = -1$ et $\mathbf{a}'_{1\times3} = -\mathbf{m}''^T \mathtt{M}''$. Finalement nous souhaitons exprimer, par identification, ce triplet dans la formulation de la proposition 14. $\mathcal{T}^{iol}$ devrait être équivalent au triplet :

$$\begin{aligned} \mathtt{P}^i = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^l &= [[\mathbf{e}^{il}]_\times \mathtt{F}^{li} \,|\, \mathbf{e}^{il}], \\ \mathtt{P}^o = \mathrm{kron}\big([\boldsymbol{\gamma}^s{}_{1:3}, 1]; \mathbf{e}^{io}\big) - {\gamma'^s}_0 &[[\mathbf{e}^{io}]_\times \mathtt{F}^{oi} \,|\, \mathbf{0}_{3\times1}], \end{aligned} \tag{C.30}$$

Nous avons donc juste à résoudre,

$$\begin{bmatrix} -{\boldsymbol{\gamma}'^s}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ {\boldsymbol{\gamma}^s}_{1:3} & | & 1 \end{bmatrix} \cong \begin{bmatrix} \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ \mathbf{a}'_{1\times3} & | & -1 \end{bmatrix} \begin{bmatrix} -{\boldsymbol{\gamma}'^0}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ {\boldsymbol{\gamma}^0}_{1:3} & | & 1 \end{bmatrix} \begin{bmatrix} \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ \mathbf{a}_{1\times3} & | & -1 \end{bmatrix}. \tag{C.31}$$

Aussi nous retrouvons facilement le changement de variable linéaire explicité dans l'équation (2.19) de la proposition. Ceci complète la preuve.

C.6 Preuve de la proposition 18

Nous considérons la boucle faite de 4 Tenseurs Trifocaux, $\left[\mathcal{T}^{ijk}, \boldsymbol{\gamma}^0\right]$, $\left[\mathcal{T}^{kil}, \boldsymbol{\gamma}^1\right]$, $\left[\mathcal{T}^{lkj}, \boldsymbol{\gamma}^2\right]$ et $\left[\mathcal{T}^{jli}, \boldsymbol{\gamma}^3\right]$. Une matrice fondamentale ayant 7 degrés de liberté, en considérant les Tenseurs $\mathcal{T}^{ijk}$ et $\mathcal{T}^{lkj}$, nous avons 7 contraintes de cohérence exprimant le fait que ces tenseurs ont la même matrice fondamentale $\mathtt{F}^{jk}$ à un facteur multiplicatif près. De plus $\mathcal{T}^{kil}$ et $\mathcal{T}^{jli}$ doivent également partager la même matrice fondamentale $\mathtt{F}^{il}$, nous avons donc 14 contraintes de cohérence indépendantes.

A-partir du tenseur trifocal $\mathcal{T}^{ijk}$, nous déduisons de la proposition 17 que la matrice fondamentale $\mathtt{F}^{jk}$ s'écrit :

$$\begin{aligned}\mathtt{F}^{jk} \;=\; & \left(\boldsymbol{\gamma}^0{}_{1:3} \cdot \mathbf{e}^{ki}\right) [\mathbf{e}^{ij}]_\times [\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} [\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} + \\ & \left[[\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} \mathbf{e}^{ki}\right]_\times \left[\left(\mathbf{e}^{ij} \boldsymbol{\gamma}^0{}_{1:3}{}^T - \gamma'^0{}_0 [\mathbf{e}^{ij}]_\times \mathtt{F}^{ji}\right) [\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} + \beta \mathbf{e}^{ij} \mathbf{e}^{ikT}\right]\end{aligned} \tag{C.32}$$

A-partir du tenseur $\mathcal{T}^{lkj}$ nous obtenons la matrice fondamentale $\mathtt{F}^{kj}$:

$$\begin{aligned}\mathtt{F}^{kj} \;=\; & \left(\boldsymbol{\gamma}^2{}_{1:3} \cdot \mathbf{e}^{jl}\right) [\mathbf{e}^{lk}]_\times [\mathbf{e}^{lk}]_\times \mathtt{F}^{kl} [\mathbf{e}^{jl}]_\times \mathtt{F}^{lj} + \\ & \left[[\mathbf{e}^{lk}]_\times \mathtt{F}^{kl} \mathbf{e}^{jl}\right]_\times \left[\left(\mathbf{e}^{lk} \boldsymbol{\gamma}^2{}_{1:3}{}^T - \gamma'^2{}_0 [\mathbf{e}^{lk}]_\times \mathtt{F}^{kl}\right) [\mathbf{e}^{jl}]_\times \mathtt{F}^{lj} + \beta_2 \mathbf{e}^{lk} \mathbf{e}^{ljT}\right]\end{aligned} \tag{C.33}$$

Les 7 premières équations de cohérence s'écrivent $\eta_1 \mathtt{F}^{jk} = \mathtt{F}^{kj}{}^T$ où les matrices fondamentales sont calculées à partir des tenseurs $\mathcal{T}^{ijk}$ et $\mathcal{T}^{lkj}$. De la même façon nous pouvons écrire les autres équations de cohérence $\eta_2 \mathtt{F}^{il} = \mathtt{F}^{li}{}^T$ à partir des tenseurs $\mathcal{T}^{kil}$ et $\mathcal{T}^{jli}$.

En multipliant à droite par $\mathbf{e}^{ik}$ et à gauche par $\mathbf{e}^{lj}$, les équations (C.32) et (C.33), nous avons : $\eta_1 \beta \mathbf{e}^{ljT} \left[[\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} \mathbf{e}^{ki}\right]_\times \mathbf{e}^{ij} = \beta_2 \mathbf{e}^{lkT} \left[[\mathbf{e}^{lk}]_\times \mathtt{F}^{kl} \mathbf{e}^{jl}\right]_\times{}^T \mathbf{e}^{ik}$. Aussi nous retrouvons le facteur multiplicatif η_1. De la même façon nous retrouvons également le facteur multiplicatif η_2.

Nous obtenons un système matriciel de $9 \times 2 = 18$ équations de rang 14. Il ne reste plus qu'à trouver une contrainte de cohérence indépendante, ou en d'autres termes une contrainte de cyclicité. Remarquons que dans la mesure où $\mathcal{T}^{ijk}$ et $\mathcal{T}^{lkj}$ partagent la même matrice fondamentale $\mathtt{F}^{jk}$, il existe une transformation projective permettant de passer de l'espace projectif d'un triplet à celui de l'autre triplet, et nous pouvons ainsi déduire la formulation du tenseur $\mathcal{T}^{kil}$. En effet dans l'équation (C.22) de l'annexe C.4, nous voyons que le triplet de caméras correspondant au tenseur $\mathcal{T}^{ijk}$ peut s'écrire dans un espace projectif :

$$\begin{aligned}\mathtt{P}'^i = [[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} \,|\, \mathbf{e}^{ki}], \qquad \mathtt{P}'^k = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \\ \mathtt{P}'^j = [[\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} \,|\, \mathbf{e}^{ij}] \begin{bmatrix} -\gamma'^0{}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ \boldsymbol{\gamma}^0{}_{1:3} & | & 1 \end{bmatrix} \begin{bmatrix} [\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} & | & \mathbf{e}^{ki} \\ \beta \mathbf{e}^{ikT} & | & 0 \end{bmatrix},\end{aligned} \tag{C.34}$$

Maintenant si nous considérons le tenseur $\mathcal{T}^{kil}$, il correspond au triplet de caméras :

$$\begin{aligned}\mathtt{P}^k = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^l = [[\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} \,|\, \mathbf{e}^{kl}], \\ \mathtt{P}^i = [[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} \,|\, \mathbf{e}^{ki}] \begin{bmatrix} -\gamma'^1{}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ \boldsymbol{\gamma}^1{}_{1:3} & | & 1 \end{bmatrix},\end{aligned} \tag{C.35}$$

En considérant les caméras k et i dans les équations (C.34) et (C.35), nous remarquons que la transformation projective entre ces deux triplets est

$$\mathtt{H} = \begin{bmatrix} -{\gamma'^1}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ {\boldsymbol{\gamma}^1}_{1:3} & | & 1 \end{bmatrix}.$$

Finalement, le tenseur $\mathcal{T}^{ijk}$ dans l'espace projectif de $\mathcal{T}^{kil}$ est :

$$\begin{aligned} \mathtt{P}'^i &= [[\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} \,|\, \mathbf{e}^{ki}] \begin{bmatrix} -{\gamma'^1}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ {\boldsymbol{\gamma}^1}_{1:3} & | & 1 \end{bmatrix}, \qquad \mathtt{P}'^k = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \\ \mathtt{P}'^j &= [[\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} \,|\, \mathbf{e}^{ij}] \begin{bmatrix} -{\gamma'^0}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ {\boldsymbol{\gamma}^0}_{1:3} & | & 1 \end{bmatrix} \begin{bmatrix} [\mathbf{e}^{ki}]_\times \mathtt{F}^{ik} & | & \mathbf{e}^{ki} \\ \beta \mathbf{e}^{ik^T} & | & 0 \end{bmatrix} \begin{bmatrix} -{\gamma'^1}_0 \mathtt{I}_{3\times3} & | & \mathbf{0}_{3\times1} \\ {\boldsymbol{\gamma}^1}_{1:3} & | & 1 \end{bmatrix}, \end{aligned} \tag{C.36}$$

Maintenant nous allons également calculer le tenseur $\mathcal{T}^{lkj}$ dans l'espace projectif de $\mathcal{T}^{kil}$. Le triplet de caméras correspondant à $\mathcal{T}^{lkj}$ est :

$$\begin{aligned} \mathtt{P}^l &= [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \qquad \mathtt{P}^j = [[\mathbf{e}^{lj}]_\times \mathtt{F}^{jl} \,|\, \mathbf{e}^{lj}], \\ \mathtt{P}^k &= \mathrm{kron}\big([{\boldsymbol{\gamma}^2}_{1:3}, 1]; \mathbf{e}^{lk}\big) - {\gamma'^2}_0 [[\mathbf{e}^{lk}]_\times \mathtt{F}^{kl} \,|\, \mathbf{0}_{3\times1}], \end{aligned} \tag{C.37}$$

Nous cherchons la transformation projective $\mathtt{H}$ telle que : $\mathtt{P}''^l = \mathtt{P}^l\mathtt{H} = [[\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} \,|\, \mathbf{e}^{kl}]$ et $\mathtt{P}''^k = \mu \mathtt{P}^k \mathtt{H} = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}]$. Grâce à la première équation $\mathtt{H}_{1:3,1:4} = [[\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} \,|\, \mathbf{e}^{kl}]$. En insérant cette égalité dans la seconde équation nous obtenons :

$$\mathbf{e}^{lk} \left({\boldsymbol{\gamma}^2}_{1:3} \cdot \mathbf{e}^{kl}\right) + \mathtt{H}_{4,4} \mathbf{e}^{lk} = \mathbf{0}_{3\times1} \tag{C.38}$$

$$\mu \mathtt{I} = \mathbf{e}^{lk} {\boldsymbol{\gamma}^2}_{1:3}^T [\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} - {\gamma'^2}_0 [\mathbf{e}^{lk}]_\times \mathtt{F}^{kl} [\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} + \mathbf{e}^{lk} \mathtt{H}_{4,1:3} \tag{C.39}$$

En multipliant à gauche par $[\mathbf{e}^{lk}]_\times [\mathbf{e}^{lk}]_\times$ nous pouvons calculer μ de façon explicite en utilisant la trace. Une fois que $\mu = \frac{1}{2}\mathrm{trace}\left([\mathbf{e}^{lk}]_\times [\mathbf{e}^{lk}]_\times [\mathbf{e}^{lk}]_\times \mathtt{F}^{kl} [\mathbf{e}^{kl}]_\times \mathtt{F}^{lk}\right)$ est connu nous pouvons retrouver $\mathtt{H}_{4,1:3}$ en multipliant à gauche la deuxième équation par $\mathbf{e}^{lk^T}$.

$$\mathtt{H} = \begin{bmatrix} [\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} & | & \mathbf{e}^{kl} \\ \mu {\gamma'^2}_0 \mathbf{e}^{lk^T} - {\boldsymbol{\gamma}^2}_{1:3}^T [\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} & | & -{\boldsymbol{\gamma}^2}_{1:3} \cdot \mathbf{e}^{kl} \end{bmatrix}$$

Ainsi le tenseur $\mathcal{T}^{lkj}$ s'écrit dans l'espace projectif du tenseur $\mathcal{T}^{kil}$:

$$\begin{aligned} \mathtt{P}''^l &= [[\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} \,|\, \mathbf{e}^{kl}], \qquad \mathtt{P}''^k = [\mathtt{I}_{3\times3} \,|\, \mathbf{0}_{3\times1}], \\ \mathtt{P}''^j &= [[\mathbf{e}^{lj}]_\times \mathtt{F}^{jl} \,|\, \mathbf{e}^{lj}] \begin{bmatrix} [\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} & | & \mathbf{e}^{kl} \\ \mu {\gamma'^2}_0 \mathbf{e}^{lk^T} - {\boldsymbol{\gamma}^2}_{1:3}^T [\mathbf{e}^{kl}]_\times \mathtt{F}^{lk} & | & -{\boldsymbol{\gamma}^2}_{1:3} \cdot \mathbf{e}^{kl} \end{bmatrix}, \end{aligned} \tag{C.40}$$

Comme nous pouvons le voir dans les équations (C.36) et (C.40), nous avons les triplets de caméras associés aux tenseurs $\mathcal{T}^{ijk}$ et $\mathcal{T}^{lkj}$ dans le même espace projectif de $\mathcal{T}^{kil}$. Aussi nous pouvons écrire les dernières contraintes de cohérence comme étant $\mathtt{P}''^j \cong \mathtt{P}'^j$, ou plus particulièrement $[\mathbf{e}^{ij}]_\times \mathtt{P}''^j = \eta_3 [\mathbf{e}^{ij}]_\times \mathtt{P}'^j$. Nous souhaitons retrouver le facteur multiplicatif η_3. En utilisant la dernière colonne et en multipliant à gauche par $\left[[\mathbf{e}^{ij}]_\times \mathbf{e}^{lj}\right]_\times$ nous avons :

$$\left[[\mathbf{e}^{ij}]_\times \mathbf{e}^{lj}\right]_\times [\mathbf{e}^{ij}]_\times [\mathbf{e}^{lj}]_\times \mathtt{F}^{jl} \mathbf{e}^{kl} = -\eta_3 {\gamma'^0}_0 \left[[\mathbf{e}^{ij}]_\times \mathbf{e}^{lj}\right]_\times [\mathbf{e}^{ij}]_\times [\mathbf{e}^{ij}]_\times \mathtt{F}^{ji} \mathbf{e}^{ki} \tag{C.41}$$

Ceci permet d'obtenir la valeur de ${\eta_3\gamma'^0}_0$, et l'équation $[\mathbf{e}^{ij}]_\times \mathtt{P}''^j = \eta_3 [\mathbf{e}^{ij}]_\times \mathtt{P}'^j$ devient simple et linéaire. Nous avons 12 contraintes de cohérence supplémentaires. Finalement les contraintes de cyclicité s'écrivent $\mathtt{C}_{30\times 16}\Gamma = \mathbf{d}$. Cette matrice est de rang 15 et est équivalente à $[\mathtt{C} \; -\mathbf{d}][\Gamma; 1] = \mathbf{0}$. En calculant la SVD il est facile de transformer le système en $\mathtt{C}'_{15\times 16}\Gamma = \mathbf{d}'$.

Annexe D

Transformation de maillages et Applications

D.1 Introduction à la Triangulation de Delaunay

D.1.1 Enveloppe convexe, Polytopes, Simplexes, et Complexes

Soit $\mathcal{M} = \{M_1, \ldots, M_n\}$ un ensemble de points dans $\mathbb{R}^d$. L' *enveloppe convexe* de $\mathcal{M}$, $conv(\mathcal{M})$, est le plus petit ensemble convexe contenant $\mathcal{M}$. C'est un *polytope*, l'enveloppe convexe d'un ensemble *fini* de points (voir Figure D.1 pour un exemple en dimension deux). Un plan H est un plan support d'un polytope $\mathcal{P}$ si il intersecte $\mathcal{P}$ et si $\mathcal{P}$ est entièrement contenu dans un des deux demi-espaces fermés liés au plan H. Les *faces* de $\mathcal{P}$ sont ses intersections avec ses *plans support*.

Nous appelons *k-simplexe* l'enveloppe convexe de $k+1$ points affines indépendants. Par exemple, un 0-simplexe est un point, un 1-simplexe est un segment de ligne, un 2-simplexe est un triangle et un 3-simplexe est un tétraèdre. Nous considèrerons également des 4-simplexes dans $\mathbb{R}^4$: ils sont nommés *pentachorones* ou *pentatopes*. Un simplexe défini par un sous-ensemble de sommets d'un autre simplexe est une *face* de ce dernier.

Un *complexe simplicial* $\mathcal{C}$ est un ensemble fini de simplexes satisfaisant les conditions suivantes :

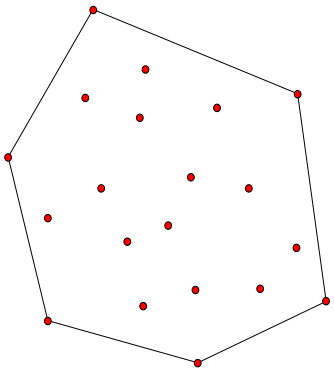


FIG. D.1 – *Enveloppe convexe d'un ensemble de points.*

1. chaque face d'un simplexe dans $\mathcal{C}$ est aussi un simplexe dans $\mathcal{C}$
2. deux simplexes dans $\mathcal{C}$ ne s'intersectent pas, ou leur intersection est un simplexe de plus petite dimension qui est leur face commune de plus grande dimension.

Les *faces* d'un complexe simplicial sont ses simplexes le constituant. En dimension d, les faces de dimensions 0, 1, $d-1$, et d sont appelées respectivement les *sommets*, *arêtes*, *facettes*, et *cellules* du complexe. Un k-complexe est un complexe dont la dimension maximale des simplexes est k.

Un *complexe de cellules* est défini comme un complexe dont les faces (non nécessairement bornées) sont des polytopes. Des propriétés similaires doivent être satisfaites :

1. chaque face d'un polytope de $\mathcal{C}$ est également un polytope dans $\mathcal{C}$
2. deux polytopes dans $\mathcal{C}$ ne s'intersectent pas, ou leur intersection est un polytope de plus petite dimension qui est leur face commune de plus grande dimension.

Deux faces d'un complexe sont *incidentes* si l'une contient l'autre et leur dimension diffère de un. Deux sommets (cellules) d'un complexe sont *adjacentes* si elles partagent une arête incidente commune (respectivement facette). En accord avec ces relations entre les faces d'un complexe, les deux graphes suivant sont définis :

– *Le graphe incident* d'un complexe enregistre un noeud pour chaque face du complexe et une arête pour chaque paire de faces incidentes.
– *Le graphe d'adjacence* d'un complexe enregistre un noeud pour chaque cellule du complexe et une arête pour chaque paire de cellules adjacentes.

Deux complexes $\mathcal{C}$ et $\mathcal{C}^*$ sont appelés *duaux* entre eux, s'il existe une bijection entre les faces de $\mathcal{C}$ et celles de $\mathcal{C}^*$ qui inverse les relations d'inclusion. Un exemple bien connu de tels complexes est décrit dans les sections suivantes.

D.1.2 Diagramme de Voronoi

Les diagrammes de Voronoi sont des structures versatiles qui encodent les relations de proximité entre les objets. Ils sont particulièrement intéressants pour effectuer des recherches de plus proches voisins, et pour modéliser des processus de croissance (e.g. croissance de cristal en sciences des matériaux).

Soit $\mathcal{M} = \{M_1, \ldots, M_n\}$ un ensemble de points dans $\mathbb{R}^d$. La *région de Voronoi*, ou *cellule de Voronoi*, notée $V(p_i)$, associée à un point p_i est la région de l'espace plus proche de p_i que de tous les autres points dans $\mathcal{M}$:

$$V(M_i) = \{X \in \mathbb{R}^d \ : \ \forall j, \ \|X - M_i\| \leq \|X - M_j\|\} \ . \tag{D.1}$$

$V(M_i)$ est l'intersection de $n-1$ demi-espaces formés par les plans bissecteurs des points $[M_i M_j], \ j \neq i$. $V(M_i)$ est donc un polytope convexe, non nécessairement borné. Le *diagramme de Voronoi* de $\mathcal{M}$, noté $\mathcal{V}or(\mathcal{M})$, est la partition de l'espace induite par les cellules de Voronoi $V(M_i)$.

Voir Figure D.3*(a)* pour un exemple en dimension deux d'un diagramme de Voronoi. En dimension deux, les arêtes partagées par deux cellules de Voronoi sont appelées *arêtes de Voronoi*

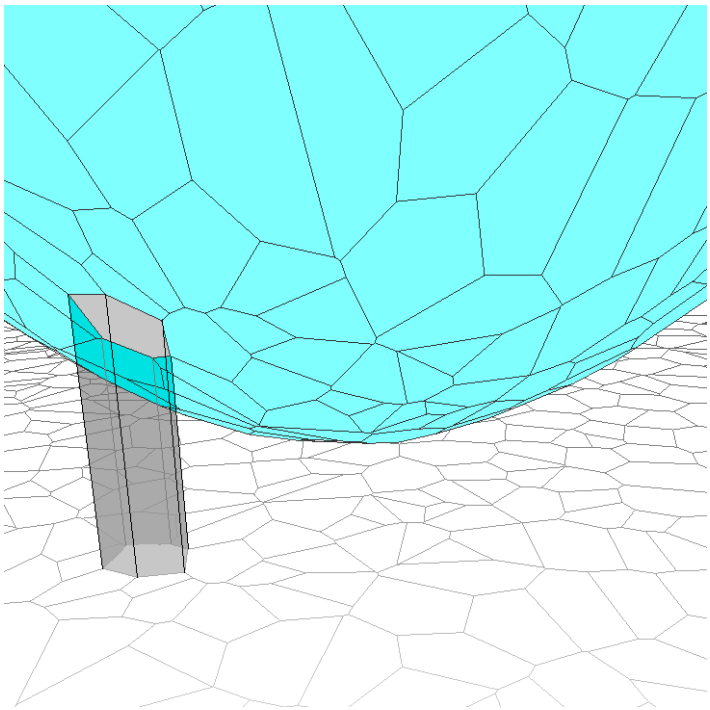


FIG. D.2 – *Le polytope de Voronoi est les faces reprojetées sur le plan.*

et les points partagés par trois cellules de Voronoi sont appelés *sommets de Voronoi*. De façon similaire, en dimension trois, nous appelons *facettes de Voronoi, arêtes de Voronoi* et *sommets de Voronoi* les objets géométriques partagés par respectivement une, deux et trois cellules de Voronoi. Le diagramme de Voronoi est la collection de tous ces objets k-dimensionnel, avec $0 \leq k \leq d$, que nous appelons *objets de Voronoi*. En particulier, remarquons que les cellules de Voronoi $V(M_i)$ correspondent à des objets de Voronoi de dimension d. Le diagramme de Voronoi est un complexe de cellules dont le domaine est $\mathbb{R}^d$.

Soit $\mathcal{P}$ la paraboloïde dans $\mathbb{R}^{d+1}$ défini par l'équation

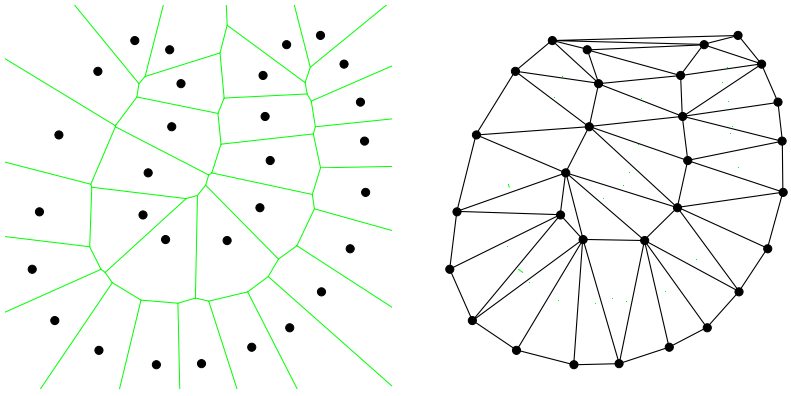
$$x_{d+1} = \sum_{i=1}^{d} x_i^2 \tag{D.2}$$

où $(x_1, x_2, \cdots, x_{d+1})$ sont les coordonnées d'un point dans $\mathbb{R}^{d+1}$. Le *polytope de Voronoi* comme est l'intersection de n demi-espaces dans $\mathbb{R}^{d+1}$, formés comme l'espace au-dessus de n hyperplans tangents à $\mathcal{P}$ au niveau des projections verticales des points de $\mathcal{M}$ sur $\mathcal{P}$ (voir figure D.2). Il peut être montré que le diagramme de Voronoi de $\mathcal{M}$ peut être calculé en projetant les faces du polytope de Voronoi sur $\mathbb{R}^d$. Le problème de calculer le diagramme de Voronoi peut donc se réduire au problème d'intersections d'hyper espaces.

Le corollaire suivant[1] est un résultat de ce théorème. Le lecteur intéressé peut se référer à [BY98] pour la preuve et une description exhaustive du problème de l'intersection de demi-espaces.

La complexité (c'est-à-dire le nombre de facettes) du diagramme de Voronoi de n points

[1] [BY98]



(a) (b)

FIG. D.3 – *(a) diagramme de Voronoi d'un ensemble de points dans le plan. (b) Sa triangulation duale de Delaunay.*

dans $\mathbb{E}^d$ est $\Theta\left(n^{\lceil \frac{d}{2} \rceil}\right)$. Nous pouvons calculer un tel diagramme en temps $O\left(nlogn + n^{\lceil \frac{d}{2} \rceil}\right)$, qui est optimal dans le pire des cas.

D.1.3 Triangulation de Delaunay

Les complexes de Delaunay sont un outil classique dans le champ de la génération et la transformation de maillage grâce à ses propriétés d'optimalité.

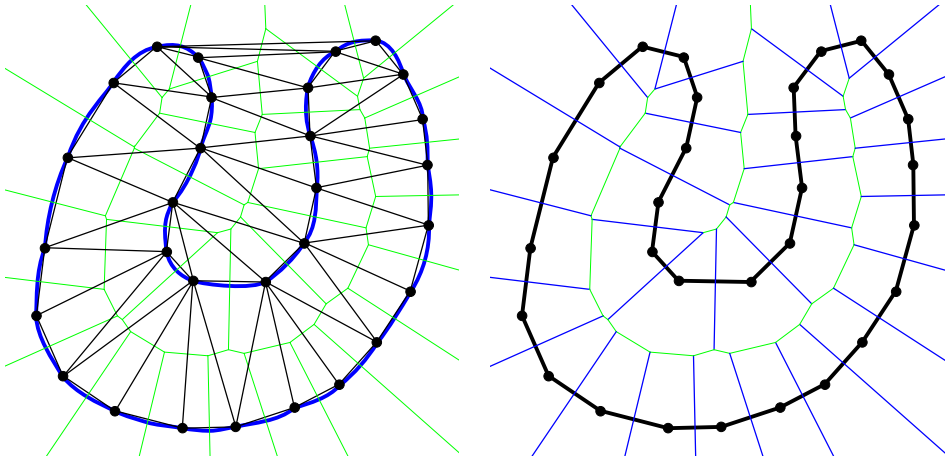
Soit $\mathcal{M} = \{M_1, \ldots, M_n\}$ un ensemble de points dans $\mathbb{R}^d$. Le complexe de Delaunay de $\mathcal{M}$, noté $\mathcal{D}el(\mathcal{M})$, est obtenu comme étant le dual du diagramme de Voronoi, il existe une bijection, inversant les relations d'inclusion, entre les k-faces du diagramme de Voronoi et les $(d-k)$-faces du complexe de Delaunay. Comme on le voit sur la figure D.3, le complexe de Delaunay est obtenu en enregistrant un noeud pour chaque point de $\mathcal{M}$, sachant que chaque point de $\mathcal{M}$ est lié à une unique cellule du diagramme de Voronoi. Ensuite, une arête est enregistrée entre deux noeuds, si les deux cellules correspondantes du diagramme de Voronoi sont adjacentes. $\mathcal{D}el(\mathcal{M}))$ est un complexe simplicial, appelé triangulation de Delaunay.

La propriété essentielle de la triangulation de Delaunay est appelée la propriété du *cercle vide* (respectivement *sphère vide* en 3D, *hypersphère vide* en dimension plus grande) : en 2D (respectivement en 3D, 4D), un triangle (respectivement tétraèdre, pentatope) appartient à la triangulation de Delaunay si et seulement si son cercle circonscrit (respectivement sphère circonscrite, hypersphère circonscrite) ne contient aucun autre point de $\mathcal{M}$. Le théorème suivant[2] est valide pour les complexes de Delaunay :

Proposition 19 *Soit $\mathcal{M}$ un ensemble de n points $M_1, \cdots, M_n$ dans $\mathbb{E}^d$. Chaque d-face du complexe de Delaunay peut être circonscrite par une sphère qui passe par tous ses sommets, et dont l'intérieur ne contient aucun point de $\mathcal{M}$.*

Le calcul du complexe de Delaunay dans $\mathbb{R}^d$ peut se réduire au calcul de l'enveloppe convexe dans $\mathbb{R}^{d+1}$. Le corollaire suivant [3] est un résultat du théorème reproduit ci-dessus. Le lecteur

[2] [BY98]
[3] [BY98]



(a) (b)

FIG. D.4 – *(a) diagramme de Voronoi (vert) et triangulation de Delaunay (noir) d'un échantillon de points restreints à une courbe de dimension 2 (bleu). (b) La triangulation de Delaunay (noir) de l'échantillon de points réduits à la courbe approxime la forme de la courbe. Seuls les segments de la triangulation de Delaunay, dont les duaux (bleu) intersectent la courbe sont laissés dans la triangulation de Delaunay restreinte.*

intéressé peut se référer à [BY98] pour une description exhaustive du calcul des enveloppes convexes.

Le complexe de Delaunay de n points dans $\mathbb{E}^d$ peut être calculé en temps $O\left(n\log n + n^{\lceil \frac{d}{2} \rceil}\right)$, et est optimal dans le pire des cas.

De plus, comme prouvé récemment dans [ABL03], la complexité en 3D n'est que de $O(n \log n)$ lorsque les points sont distribués sur une surface lisse.

D.2 Une application à la transformation de maillage

Chaque k-simplexe dans la triangulation de Delaunay est dual d'un objet de Voronoi $(d-k)$-dimensionnel. En 3D, le dual d'une cellule de Delaunay (un tétraèdre) est le sommet de Voronoi qui coïncide avec le centre circonscrit au tétraèdre, le dual d'une facette de Delaunay (un triangle) est une arête de Voronoi, le dual d'une arête de Delaunay est une facette de Voronoi, et le dual d'un sommet de Delaunay M_i est la cellule de Voronoi $V(M_i)$.

Soit $\mathcal{M} = \{M_1, \ldots, M_n\}$ un ensemble de points dans $\mathbb{R}^d$. Etant donné un ensemble $\Omega \in \mathbb{R}^d$, typiquement une variété de dimension $k \leq d$, nous appelons la *triangulation de Delaunay de $\mathcal{M}$ restreinte à Ω*, notée $\mathcal{D}el|_\Omega(\mathcal{M})$ le sous-complexe de $\mathcal{D}el(\mathcal{M})$ composé des simplexes de Delaunay dont les objets de Voronoi duaux intersectent Ω. Par exemple, en 2D, comme illustré en Figure D.4*(b)*, la triangulation de Delaunay d'un ensemble de points restreinte à une courbe $\mathcal{S}$ est composée des arêtes de Delaunay dont les arêtes duales de Voronoi intersectent $\mathcal{S}$. De façon similaire, la triangulation de Delaunay restreinte à une région Ω est composée des triangles de Delaunay dont les centres circonscrits sont contenus dans Ω. Le lecteur attentif a sans doute remarqué que dans les deux cas la triangulation de Delaunay restreinte à un objet forme une

bonne approximation de l'objet.

On peut montrer que, sous certaines hypothèses, et particulièrement si $\mathcal{M}$ est un échantillon "suffisamment dense" dans Ω, au sens défini dans [AB99], $\mathcal{D}el|_\Omega(\mathcal{M})$ est une bonne approximation de Ω, à la fois d'un point de vue topologique et d'un point de vue géométrique : par rapport à la topologie, $\mathcal{D}el|_\Omega(\mathcal{M})$ est homéomorphique à Ω ; vis-à-vis de la géométrie, la distance de Hausdorff entre $\mathcal{D}el|_\Omega(\mathcal{M})$ et Ω peut être rendue arbitrairement aussi petite qu'on le souhaite suivant les points échantillonnant ; les normales et courbures de Ω peuvent être approximées à partir de $\mathcal{D}el|_\Omega(\mathcal{M})$.

Basés sur ces propriétés d'approximation, un ensemble d'algorithmes mathématiquement corrects pour la génération et la transformation de maillage à partir d'un nuage de points a été proposé cette dernière décennie. Le lecteur peut se référer à [BO05] pour plus de détails et de références.

Cette triangulation de Delaunay restreinte est utilisée pour remailler à partir d'un nuage de points enrichi ou appauvri et d'une forme initiale avant modification du nuage de points. Une application de ce principe est proposée dans [PB07] et est utilisée dans le cadre de la seconde partie de ce manuscrit consacrée à la reconstruction spatio-temporelle, lorsque nous souhaitons changer la résolution du maillage et éventuellement changer la topologie.

D.3 Une application à la stéréovision spatio-temporelle

Nous décrivons ici la première étape de la méthode proposée dans la section 4.5. Pour de plus amples détails le lecteur intéressé pourra se référer à [APK09].

D.3.1 Génération d'un nuage de points 4D, triangulation de Delaunay 4D

Etant donné plusieurs séquences vidéos d'une même scène dynamique, nous générons dans un premier temps un nuage dense de points 3D pour chaque instant. Soit $I_k^t, k \in \{1, \cdots, n\}, t \in [0, T]$ les séquences vidéos en entrée : I_k^t est l'image capturée par la caméra k au temps t. Pour chaque image nous extrayons des points d'intérêts de plusieurs types $P_{k,i}^t$ (en pratique points de Harris et DOGs). Pour chaque instant t, nous prenons toutes les paires d'images $(I_k^t, I_{k'}^t)$ et triangulons les meilleures correspondances possibles $(P_{k,i}^t, P_{k',j}^t)$ entre les deux ensembles de points d'intérêt vérifiant les contraintes épipolaires : $P_{k',j}^t$ doit être à l'intérieur de la reprojection d'une bande autour de la droite 3D passant par le centre de la caméra i et le point $P_{k,i}^t$ sur le plan image de la caméra i (Figure D.5). Nous notons $X_{kk',ij}^t$ le point 3D obtenu par triangulation des points $P_{k,i}^t$ et $P_{k',j}^t$.

Pour trouver la meilleure correspondance pour $P_{k,i}^t$, la partie de l'image $I_{k'}^t$ dans une fenêtre autour de la correspondance candidate $P_{k',j}^t$ est reprojetée sur la caméra k par le biais d'un plan passant par $X_{kk',ij}^t$ et normal au rayon optique de $P_{k,i}^t$. Le score de ressemblance est estimé par corrélation NCC entre l'image dans la fenêtre initiale et l'image dans la fenêtre reprojetée dans l'autre caméra. La correspondance est acceptée si la NCC est supérieure à un seuil paramétrisable. Finalement nous fusionnons les points provenant de différentes paires de caméras si leur distance est inférieure à un seuil. Ainsi chaque point 3D est associé à deux vues ou plus.

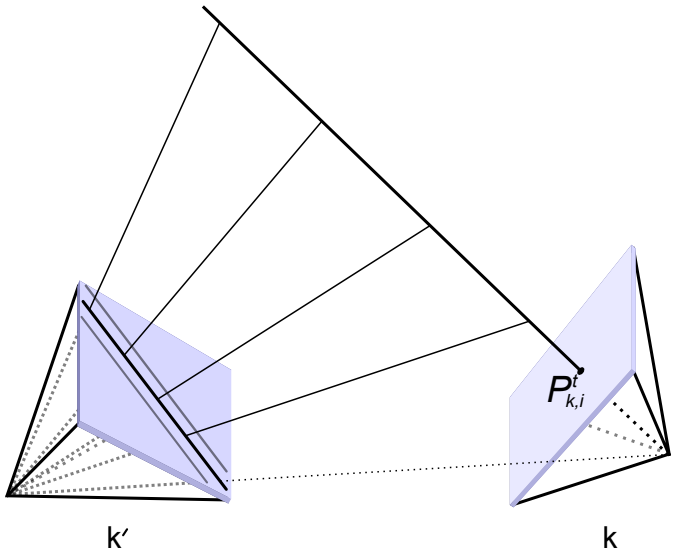


FIG. D.5 – *Le rayon optique d'une correspondance pour un point d'intérêt P doit être contenu dans une bande autour du rayon optique de P dans l'autre caméra.*

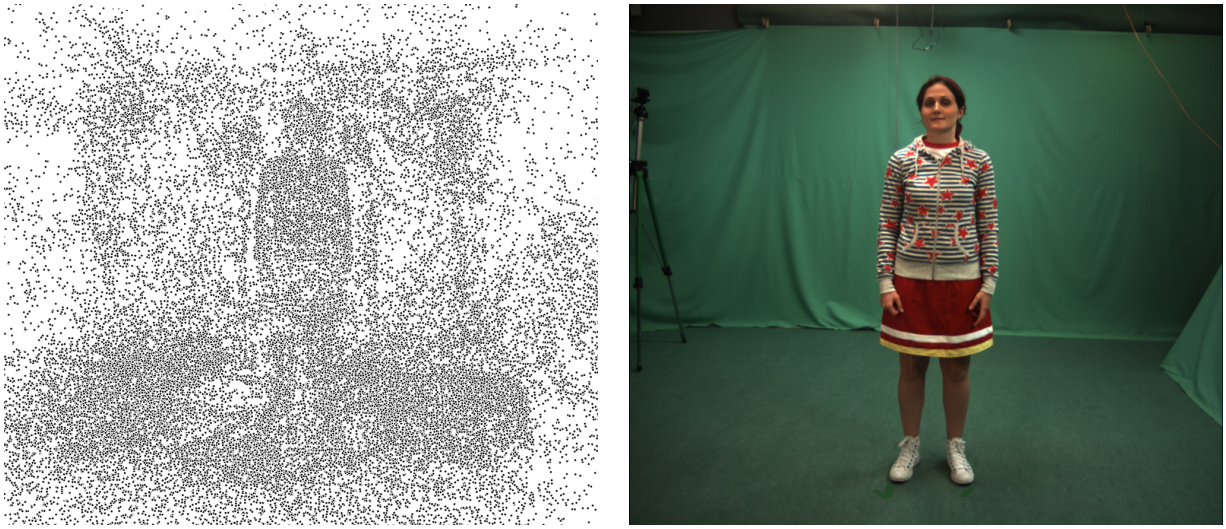


FIG. D.6 – *Le nuage de points 3D extrait à partir de 14 images. Il y a un grand nombre de points outliers.*

Nous construisons un nuage “global” spatio-temporel de points en ajoutant le temps comme quatrième coordonnée. Enfin, nous calculons la triangulation de Delaunay 4D du nuage de points spatio-temporel, et nous enregistrons pour chaque sommet les vues et points d'intérêt à partir desquels il a été généré.

D.3.2 Extraction de l'hyper-surface 4D

Au cours de cette étape nous calculons une représentation 4D de la scène en extrayant une surface simpliciale 4D à partir de la triangulation de Delaunay. Dans ce but, nous souhaitons étiqueter les pentatopes de Delaunay comme intérieurs ou extérieurs. L'ensemble des facettes situées entre deux cellules étiquetées différemment forme une hypersurface, naturellement orientée dans l'espace 4D. Une facette de cette hypersurface est un tétraèdre orienté depuis le voisin intérieur vers le voisin extérieur. Nous notons $\mathcal{D}el(\mathcal{M})$ le complexe de Delaunay 4D de $\mathcal{M}$ (nuage de points spatio-temporel). $\mathcal{D}el(\mathcal{M})$ est l'ensemble des pentatopes, tétraèdres, triangles, segments, et points dans le complexe de Delaunay. Nous notons $\mathcal{C} = \{C_1, \cdots, C_m\} \subset \mathcal{D}el(\mathcal{M})$ l'ensemble des m cellules de $\mathcal{D}el(\mathcal{M})$. Nous définissons $\vec{l} = \{l_1, \cdots, l_m\} \in \{I, O\}^m$ un vecteur étiquetant qui décrit l'état des cellules de Delaunay. Une cellule est étiquetée I si elle est à l'intérieure, ou O si elle est à l'extérieur de la scène 4D notée $\mathcal{S}$:

$$S(\mathcal{M}, \vec{l}) = \left\{ F \in \mathcal{D}el(\mathcal{M}) \;\middle|\; \begin{array}{l} F \text{ est une facette de } \mathcal{D}el(\mathcal{M}) \\ \exists C_i, C_j \in \mathcal{C}(\mathcal{M}) \text{ tel que } \left\{ \begin{array}{l} C_i \cap C_j = F \\ l_i = I \text{ et } l_j = O \end{array} \right. \end{array} \right\} \tag{D.3}$$

La facette F est orientée depuis le demi-espace contenant la cellule intérieure C_i vers le demi-espace contenant la cellule extérieure C_j. L'hypersurface simpliciale $\mathcal{S}$ est alors l'union de toutes les facettes dans S,

$$\mathcal{S} = \bigcup_{F \in S} F \tag{D.4}$$

Dans la suite nous décrivons comment nous calculons cet étiquetage optimal en minimisant une énergie tenant compte de la visibilité, de la photométrie, et de la régularité de l'hypersurface.

Dans un premier temps, nous définissons un graphe de voisinage $\mathcal{G} := < \mathcal{V}, \mathcal{E} >$ entre toutes les cellules de Delaunay $C_i \in \mathcal{C}(\mathcal{M})$. Ces dernières, qui sont des noeuds du graphe, sont liées par une arête dans le graphe. Elles partagent des faces de dimensions 2 ou 3 dans la triangulation de Delaunay. De plus tous les noeuds sont connectés à un noeud source (anglais : source) et à un noeud puits (sink) :

$$\begin{array}{rl} \mathcal{G} := & < \mathcal{V}, \mathcal{E} > \\ \mathcal{V} := & \{v_1, \cdots, v_m\} \cup \{s, t\} \\ \mathcal{E} := & \{(s, v_1), \cdots, (s, v_m), (t, v_1), \cdots, (t, v_m)\} \cup \\ & \{(v_i, v_j) \,|C_i \cap C_j \text{ is a (2 or 3)-simplex}\} \end{array} \tag{D.5}$$

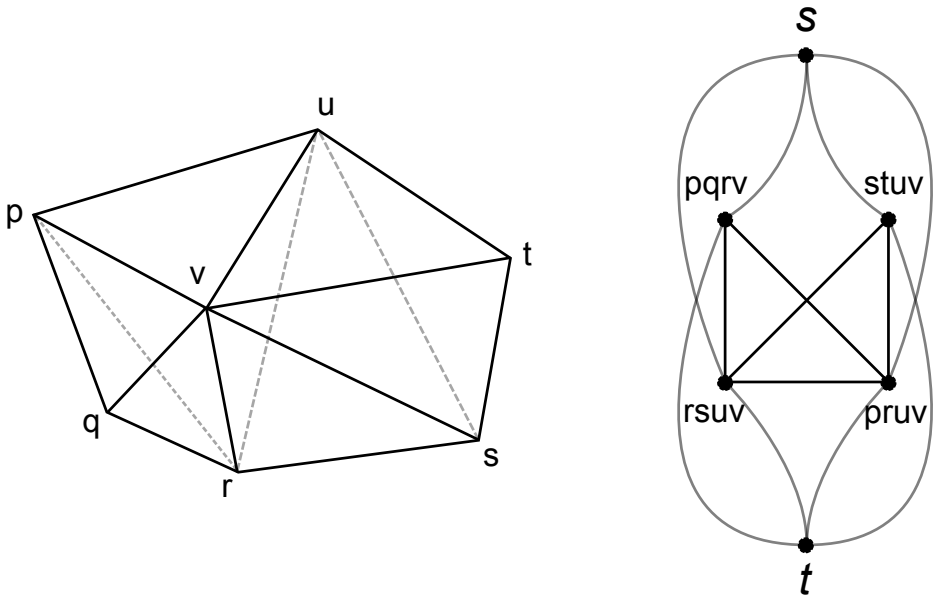


FIG. D.7 – *Un exemple en 3D du graphe de voisinage(à droite) construit à partir d'une triangulation(à gauche). Tous les sommets sont connectés à la source et au puits.*

La Figure D.7 montre un exemple en 3D du graphe de voisinage construit. En 3D, seuls les noeuds dont les cellules de Delaunay correspondantes partagent une face de dimension un ou deux sont connectés.

Dans un second temps nous définissons une énergie à minimiser, des considérations photométriques et de visibilité sont utilisées pour donner des poids appropriés à chaque arête du graphe $\mathcal{G}$. Un étiquetage optimal est alors calculé en trouvant la coupe de poids minimal dans ce graphe. Pour cela nous utilisons une optimisation par Graph cuts présentée en Annexe B.3. Si la source représente l'extérieur et le puits l'intérieur, alors la coupe permet d'avoir une partition en deux de l'espace entre les noeuds du côté de l'intérieur, et les noeuds du graphe du côté de l'extérieur. Ainsi les cellules de Delaunay sont étiquetées intérieures ou extérieures, ce qui permettra d'obtenir l'hypersurface 4D souhaitée.

Nous cherchons donc la coupe dans le graphe, minimisant l'énergie composée de trois termes tenant compte de la photométrie, de la visibilité, et de la régularité,

$$\mathrm{E}(\mathcal{S}) = \mathrm{E}_{\mathrm{photo}}(\mathcal{S})\mathrm{E}_{\mathrm{visibilit}}(\mathcal{S}) + \mathrm{E}_{\mathrm{rgularit}}(\mathcal{S}) \tag{D.6}$$

Le premier terme de photométrie n'est pas détaillé et est très simple, on donne un score de similarité à chaque face résultant de l'intersection de deux cellules de Delaunay. Ce score est lié à la corrélation dans les images à partir desquelles la cellule a été générée. Chaque face est échantillonnée en N points et est agrandie ou réduite, puis on calcule la corrélation dans les images à partir des reprojections de ces points échantillonnés dans les différentes caméras. Ce score ou poids est alloué aux arêtes du graphe correspondantes. De cette façon la coupe dans le

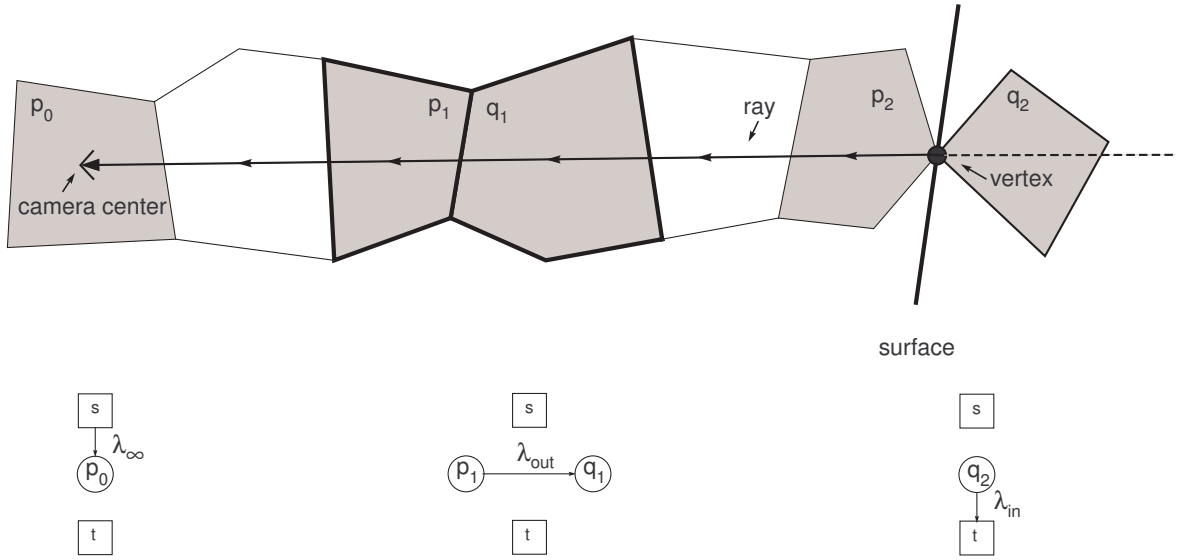


FIG. D.8 – *En haut : Une coupe 3D de la triangulation 4D. Un rayon provenant du sommet vers le centre de la caméra intersecte des cellules 3D. En bas : Les termes correspondants dans l'énergie, pénalisant les grands nombres d'intersections entre le rayon et les pentatopes. Ces intersections sont représentées par des poids dans le graphe.*

graphe aura un côut plus élevé si elle traverse des arêtes à faible photo-consistance, plutôt que si elle traverse des arêtes (faces dans le complexe de Delaunay) présentant une bonne similarité dans les images.

Le rôle de $\mathrm{E}_{\text{visibilit}}$ est de pénaliser les facettes produisant un grand nombre d'occlusions et donc inconsistantes avec les observations. L'idée est qu'un point ne peut être inlier (correct) s'il n'est pas visible dans la caméra à partir de laquelle il a été généré.

D.3.2.1 Terme de Visibilité

L'idée étant qu'un point ne peut être inlier (correct) s'il n'est pas visible dans la caméra à partir de laquelle il a été généré. Dans le cas 3D, afin d'approximer cette idée de façon simple et possible à résoudre par Graph cuts, nous pénalisons pour chaque point toutes les facettes intersectant les parties des rayons entre le point et les centres des caméras à partir desquelles le point a été triangulé précédemment.

Dans le cas 4D nous utilisons un argument similaire. Un point appartenant à l'hypersurface, devrait être visible dans toutes les vues à partir desquelles il a été généré. Tous les pentatopes 4D qui intersectent un rayon dans l'espace 4D entre le point 4D et le centre 4D d'une image à partir de laquelle le point à été généré devraient être étiquetés extérieurs. Le centre spatio-temporel de la caméra pour une image à l'instant t, est bien entendu le centre 3D de la caméra auquel on ajoute t comme quatrième coordonnée. En revanche le pentatope situé derrière le point devrait être étiqueté intérieur. Finalement en considérant tous les points nous accumulons les scores pour les pentatopes.

Sur la Figure D.8(en haut) nous voyons l'objet 3D et un rayon intersectant les cellules.

Les noeuds p_0, p_1, q_1, p_2 et q_2 montrés sur la Figure D.8(en bas) sont les noeuds du graphe correspondant aux pentatopes le long du rayon. Plusieurs termes de visibilité sont liés au rayon provenant d'un point. La cellule contenant la caméra devrait être étiquetée extérieure : un terme λ_∞ est ajouté pour l'arête allant de la source vers p_0. Une facette 3D traversée par le rayon depuis l'intérieur vers l'extérieur devrait être pénalisée : un terme λ_{out} est ajouté à l'arête allant de p_1 vers q_1. La cellule derrière le rayon devrait être intérieure : un terme λ_{in} est ajouté à l'arête allant de q_2 vers le puits.

Les poids positifs λ_{in}, λ_{out} et λ_∞ prennent en compte le degré de confiance du point reconstruit. En agrégeant ces termes de visibilité pour tous les points et tous les instants nous obtenons un "vote" complexe.

D.3.2.2 Régularité spatio-temporelle

Afin de prendre en compte la régularité spatio-temporelle, nous minimisons l'aire de l'hypersurface 4D dans $\mathbb{R}^4$. C'est la somme du volume des tétraèdres 4D entre les pentatopes intérieurs et extérieurs,

$$\mathrm{E}_{\text{rgularit}}(\mathcal{S}) = A(\mathcal{S}) = \int_{\mathcal{S}} \mathrm{d}\mathcal{S} = \sum_{F \in S} A(F) \tag{D.7}$$

où $\mathcal{S}$ est l'hypersurface à reconstruire, F est un tétraèdre 4D, et $A(F)$ le volume du tétraèdre F dans $\mathbb{R}^4$. En minimisant ce terme on encourage la régularité dans le temps et dans l'espace. Concrètement, pour chaque paire de pentatopes (partageant un tétraèdre) représentée par les noeuds p et q dans le graphe, un terme $A(F)$ est ajouté à l'arête $p \rightarrow q$ et à l'arête opposée $q \rightarrow p$.

Pour extraire un maillage 3D correspondant à un instant t, à partir de l'hypersurface 4D obtenue en sortie, nous intersectons l'hypersurface avec le plan temporel à l'instant t. Cette tâche peut être effectuée efficacement sur carte graphique, puisqu'elle se réduit à un algorithme de *marching tétraèdre* [GH95] sur les tétraèdres du maillage 4D.

Publications

Dense and Accurate Spatio-Temporal Multi-View Stereovision.
Courchay, J., Pons, J-P., Monasse, P., Keriven, R.
Asian Conference on Computer Vision, ACCV 2009.

A global camera network calibration method with Linear Programming.
Courchay, J., Dalalyan, A., Keriven, R., Sturm, P.
Conference on 3D Data Processing, Visualization, and Transmission. 3DPVT 2010.

Exploiting loops in the graph of trifocal tensors for calibrating a network of cameras.
Courchay, J., Dalalyan, A., Keriven, R., Sturm, P.
European conference on computer vision, ECCV 2010.

On Camera calibration with linear programming and Loop constraints linearization.
Courchay, J., Dalalyan, A., Keriven, R., Sturm, P.
The international journal of computer vision, IJCV SI : 3DPVT 2010.

Bibliographie

[AB99] N. Amenta and M. Bern. Surface reconstruction by Voronoi filtering. *Discrete and Computational Geometry*, 22 :481–504, 1999. 120

[ABL03] D. Attali, J.-D. Boissonnat, and A. Lieutier. Complexity of the Delaunay triangulation of points on surfaces : the smooth case. In *Annual Symposium on Computational Geometry*, pages 201–210, 2003. 119

[AdAT+05] Naveed Ahmed, Edilson de Aguiar, Christian Theobalt, Marcus Magnor, and Hans-Peter Seidel. Automatic generation of personalized human avatars from multi-view video. In *Proc. VRST'05*, pages 257–260, 2005. 66

[APK09] E. Aganj, J.-P. Pons, and R. Keriven. Globally optimal spatio-temporal reconstruction from cluttered videos. In *Asian Conference on Computer Vision*, 2009. 85, 120

[APSK07] Ehsan Aganj, Jean-Philippe Pons, Florent Ségonne, and Renaud Keriven. Spatio-temporal shape from silhouette using four-dimensional Delaunay meshing. In *IEEE International Conference on Computer Vision*, 2007. 66, 67

[AS01] S. Avidan and A. Shashua. Threading fundamental matrices. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(1) :73–77, 2001. 30

[ASS+09] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *ICCV*, 2009. 30

[ATR+08] Naveed Ahmed, Christian Theobalt, Christian Rössl, Sebastian Thrun, and Hans-Peter Seidel. Dense correspondence finding for parametrization-free animation reconstruction from video. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 66, 68

[BBH08] D. Bradley, T. Boubekeur, and W. Heidrich. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 67, 87

[BBK06] A. Bronstein, M. Bronstein, and R. Kimmel. Efficient computation of isometry-invariant distances between surfaces. In *SIAM J. Sci. Comput.*, volume 28, 2006. 86, 87

[BF81] Robert C. Bolles and Martin A. Fischler. A ransac-based approach to model fitting and its application to finding cylinders in range data. In *IJCAI'81 : Proceedings of the 7th international joint conference on Artificial intelligence*, pages 637–643, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc. 10

[BKP09] Martin Bujnak, Zuzana Kukelova, and Tomas Pajdla. 3D reconstruction from image collections with a single known focal length. In *ICCV*, pages 351–358, 2009. 30

[BO05] J.-D. Boissonnat and S. Oudot. Provably good sampling and meshing of surfaces. *Graphical Models*, 67 :405–451, 2005. 120

[BY98] J.-D. Boissonnat and M. Yvinec. *Algorithmic Geometry*. Cambridge University Press, 1998. 117, 118, 119

[Car95] Stefan Carlsson. Duality of reconstruction and positioning from projective views. In *In Proceedings of the workshop on Scene Representations*, pages 85–92, 1995. 27

[CCVG06] N. Cornelis, K. Cornelis, and L.J. Van Gool. Fast compact city modeling for navigation pre-visualization. In *CVPR*, 2006. 31

[CM05] O. Chum and J. Matas. Matching with PROSAC : Progressive sample consensus. In *CVPR*, pages I : 220–226, 2005. 31, 52

[CWM05] O. Chum, T. Werner, and J. Matas. Two-view geometry estimation unaffected by a dominant plane. In *CVPR*, pages I : 772–779, 2005. 52

[dAST+08] Edilson de Aguiar, Carsten Stoll, Christian Theobalt, Naveed Ahmed, Hans-Peter Seidel, and Sebastian Thrun. Performance capture from sparse multi-view video. In *ACM SIGGRAPH*, 2008. 66, 67, 68

[dATSS07] E. de Aguiar, C. Theobalt, C. Stoll, and H.-P. Seidel. Marker-less deformable mesh tracking for human shape and motion capture. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 66, 67, 68

[DPG+08] A. Delaunoy, E. Prados, P. Gargallo, J.-P. Pons, and P. Sturm. Minimizing the multi-view stereo reprojection error for triangular surface meshes. In *British Machine Vision Conference*, 2008. 70

[FF62] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962. 104

[FLP01] O. Faugeras, Quang-Tuan Luong, and T. Papadopoulou. *The Geometry of Multiple Images : The Laws That Govern The Formation of Images of A Scene and Some of Their Applications*. MIT Press, Cambridge, MA, USA, 2001. 18, 30

[FP08] Y. Furukawa and J. Ponce. Dense 3D motion capture from synchronized video streams. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 66, 67, 68

[FP09] Yasutaka Furukawa and Jean Ponce. Accurate camera calibration from multi-view stereo and bundle adjustment. *International Journal of Computer Vision*, 84(3) :257–268, 2009. v, 30

[FZ98] Andrew W. Fitzgibbon and Andrew Zisserman. Automatic camera recovery for closed or open image sequences. In *ECCV*, pages 311–326, 1998. 30

[GH95] A. Gueziec and R. Hummel. Exploiting triangulated surface extraction using tetrahedral decomposition. *IEEE Transactions on Visualization and Computer Graphics*, 1(4), 1995. 125

[GM04] B. Goldlücke and M. Magnor. Space-time isosurface evolution for temporally coherent 3D reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 350–355, 2004. 66, 67, 84

[GPS] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. 104

[GVL96] Gene H. Golub and Charles F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996. 44, 97

[HD07] F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *IEEE International Conference on Computer Vision*, 2007. 68, 70

[HTKP09] M. Havlena, A. Torii, J. Knopp, and T. Pajdla. Randomized structure from motion based on atomic 3d models from camera triplets. In *CVPR*, 2009. 30

[HZ03] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision.* Cambridge University, 2nd edition,, 2003. 9, 10, 30, 55, 106, 110

[Jac97] David Jacobs. Linear fitting with missing data : Applications to structure-from-motion and to characterizing intensity images. In *CVPR*, page 206, 1997. 30

[KCVS98] Leif Kobbelt, Swen Campagna, Jens Vorsatz, and Hans-Peter Seidel. Interactive multi-resolution modeling on arbitrary meshes. In *International Conference on Computer Graphics and Interactive Techniques*, pages 105–114, 1998. 72

[KG04] D. Kirsanov and S.J. Gortler. A discrete global minimization algorithm for continuous variational problems. harvard computer science technical report : Tr-14-04. Technical report, Cambridge, MA, 07/2004 2004. 104

[KZ04] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts ? *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(2) :147–159, 2004. 104

[KZIS08] M. Klopschitz, C. Zach, A. Irschara, and D. Schmalstieg. Generalized detection and merging of loop closures for video sequences. In *3DPVT*, 2008. 31

[LA09] Manolis I. A. Lourakis and Antonis A. Argyros. SBA : A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software*, 36(1) :2 :1–2 :30, March 2009. 14, 103

[Lin94] Tony Lindeberg. Scale-space theory : A basic tool for analysing structures at different scales. *Journal of Applied Statistics*, pages 224–270, 1994. 91

[Low04a] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2) :91–110, 2004. 52, 91

[Low04b] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2) :91–110, 2004. 86

[LPK07] P. Labatut, J.-P. Pons, and R. Keriven. Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In *ICCV*, 2007. 85

[MBG09] J Michot, A. Bartoli, and F. Gaspard. Algebraic line search for bundle adjustment. In *BMVC*, 2009. 14

[Mik02] K. Mikolajczyk. Detection of local features invariant to affine transformations, application to matching and recognition. *PhD Thesis, Institut National de Polytechniques de Grenoble, France*, 2002. 91

[MLD+09] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Generic and real-time sfm using local bundle adjustment. *Image Vision Comput.*, 27(8) :1178–1193, July 2009. 30

[MP05] D. Martinec and T. Pajdla. 3D reconstruction by fitting low-rank matrices with missing data. In *CVPR*, pages I : 198–205, 2005. 30

[MP07] Daniel Martinec and Tomás Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *CVPR*, 2007. 30, 55

[MWS95] Daphna Weinshall Michael, Michael Werman, and Amnon Shashua. Shape tensors for efficient and learnable indexing. In *In Proceedings of the IEEE Workshop on Representations of Visual Scenes*, pages 58–65, 1995. 27

[NA02] J. Neumann and Y. Aloimonos. Spatio-temporal stereo using multi-resolution subdivision surfaces. *The International Journal of Computer Vision*, 47(1–3) :181–193, 2002. 66, 67, 84

[PB07] Jean-Philippe Pons and Jean-Daniel Boissonnat. Delaunay deformable models : Topology-adaptive meshes based on the restricted Delaunay triangulation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 70, 71, 81, 120

[PCF05] N. Paragios, Y. Chen, and O. Faugeras. *Handbook of Mathematical Models in Computer Vision*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. 104

[PKF07] J.-P. Pons, R. Keriven, and O. Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *The International Journal of Computer Vision*, 72(2) :179–193, 2007. 66, 68, 70, 78, 82

[PKFH] J.-P. Pons, R. Keriven, O. Faugeras, and G. Hermosillo. Variational stereovision and 3D scene flow estimation with statistical similarity measures. In *IEEE International conference on Computer Vision*, volume 2, pages 597–602. 68, 70

[PMP+05] J. Ponce, K. McHenry, T. Papadopoulo, M. Teillaud, and B. Triggs. On the absolute quadratic complex and its application to autocalibration. In *CVPR*, pages 780–787, 2005. 9, 33, 53

[Pon10] J. Ponce. What is a camera ? In *Conference on Computer Vision and Pattern Recognition*, 2010. 59

[PSDB+10] P. Popa, I. South-Dickinson, D. Bradley, A. Sheffer, and W. Heidrich. Globally consistent space-time reconstruction. *Computer Graphics Forum (Proc. Eurographics Symposium on Geometry Processing)*, 29(5), 2010. 67, 87

[PVGV+04] Marc Pollefeys, Luc Van Gool, Maarten Vergauwen, Frank Verbiest, Kurt Cornelis, Jan Tops, and Reinhard Koch. Visual modeling with a hand-held camera. *Int. J. Comput. Vision*, 59(3) :207–232, 2004. 30, 31

[QL02] L. Quan and M. Lhuillier. Dense 3D motion capture from synchronized video streams. In *European Conference on Computer Vision*, 2002. 30

[Qua95] Long Quan. Invariants of six points and projective reconstruction from three uncalibrated images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(1) :34–46, 1995. 27

[SFP10] D. Scaramuzza, F. Fraundorfer, and M. Pollefeys. Closing the loop in appearance-guided omnidirectional visual odometry by using vocabulary trees. *Robot. Auton. Syst.*, page to appear, 2010. 31

[SH07] J. Starck and A. Hilton. Surface capture for performance based animation. *IEEE Computer Graphics and Applications*, 27(3) :21–31, 2007. 66, 68

[SLK09] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Scramsac : Improving ransac's efficiency with a spatial consistency filter. In *ICCV*, 2009. 52

[SPM04] S. N. Sinha, M. Pollefeys, and L. McMillan. Camera network calibration from dynamic silhouettes. In *CVPR*, 2004. 30

[SSS06] Noah Snavely, Steven M. Seitz, and Richard Szeliski. *Photo tourism : Exploring photo collections in 3D*. ACM Press, New York, NY, USA, 2006. 30

[SSS08] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from Internet photo collections. *Int. J. Comput. Vision*, 80 :189–210, 2008. 30, 36, 53, 54, 56

[ST96] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In *ECCV (2)*, pages 709–720, 1996. 30

[SU05] Gregory G. Slabaugh and Gozde B. Unal. Active polyhedron : Surface evolution theory applied to deformable meshes. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 84–91, 2005. 70

[SvHVG+08] Christoph Strecha, Wolfgang von Hansen, Luc Van Gool, Pascal Fua, and Ulrich Thoennessen. On Benchmarking Camera Calibration and Multi-View Stereo for High Resolution Imagery. In *CVPR*, 2008. v, 54, 55

[SWG08] Jochen Süßmuth, Marco Winter, and G"unther Greiner. Reconstructing animated meshes from time-varying point clouds. *Computer Graphics Forum (Proc. Eurographics Symposium on Geometry Processing)*, 27(5) :1469–1476, 2008. 68

[TAL+07] Christian Theobalt, Naveed Ahmed, Hendrik Lensch, Marcus Magnor, and Hans-Peter Seidel. Seeing people in different light-joint shape, motion, and reflectance capture. *IEEE Transactions on Visualization and Computer Graphics*, 13(4) :663–674, 2007. 66, 68

[TBT+07] J-P Tardif, A. Bartoli, M. Trudeau, M. Guilbert, and S. Roy. Algorithms for batch matrix factorization with application to structure-from-motion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 30

[THP09] A. Torii, M. Havlena, and T. Pajdla. From google street view to 3d city models. In *OMNIVIS*, 2009. 31

[TK92] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography : a factorization method. *Int. J. Comput. Vision*, 9(2) :137–154, 1992. 30

[TK09] Nicolas Thorstensen and R. Keriven. Non-rigid shape matching using geometry and photometry. 2009. 86

[TMHF99] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Workshop on Vision Algorithms*, pages 298–372, 1999. 30

[TPD08] J.P. Tardif, Y. Pavlidis, and K. Daniilidis. Monocular visual odometry in urban environments using an omnidirectional camera. In *IROS*, pages 2531–2538, 2008. 31

[VB05] S. Vedula and S. Baker. Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3) :475–480, 2005. 66

[VBK05] Sundar Vedula, Simon Baker, and Takeo Kanade. Image-based spatio-temporal modeling and view interpolation of dynamic events. *ACM Transactions on Graphics*, 24(2) :240–261, 2005. 66

[VBMP08] Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. Articulated mesh animation from multi-view silhouettes. *ACM Transactions on Graphics*, 27(3), 2008. 66, 68

[VBSK00] S. Vedula, S. Baker, S. Seitz, and T. Kanade. Shape and motion carving in 6D. *IEEE Conference on Computer Vision and Pattern Recognition*, 2000. 66, 67

[VKLP09] H. Vu, R. Keriven, P. Labatut, and J.-P Pons. Towards high-resolution large-scale multi-view stereo. In *CVPR*, 2009. v, 30, 55, 70

[VZBH08] Kiran Varanasi, Andrei Zaharescu, Edmond Boyer, and Radu P. Horaud. Temporal surface tracking using mesh evolution. In *European Conference on Computer Vision*, volume 2, pages 30–43, 2008. 66, 70

[WCF07] R. White, K. Crane, and D. Forsyth. Capturing and animating occluded cloth. In *SIGGRAPH*, 2007. 82

[WJH+07] Michael Wand, Philipp Jenke, Qixing Huang, Martin Bokeloh, Leonidas Guibas, and Andreas Schilling. Reconstruction of deforming geometry from time-varying point clouds. In *Eurographics Symposium on Geometry Processing*, pages 49–58, 2007. 68

[WRV+08] Andreas Wedel, Clemens Rabe, Tobi Vaudrey, Thomas Brox, Uwe Franke, and Daniel Cremers. Efficient dense scene flow from sparse or dense stereo data. In *European Conference on Computer Vision*, pages 739–751, 2008. 68

[ZK00] Y. Zhang and C. Kambhamettu. Integrated 3D scene flow and structure recovery from multiview image sequences. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 674–681, 2000. 66, 68, 70

[ZK01] Y. Zhang and C. Kambhamettu. On 3D scene flow and structure estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 778–785, 2001. 68, 70