

Une méthode d'auto-étalonnage pince-caméra

Nicolas Andreff, Bernard Espiau, Radu Horaud

▶ **To cite this version:**

Nicolas Andreff, Bernard Espiau, Radu Horaud. Une méthode d'auto-étalonnage pince-caméra. [Rapport de recherche] RR-3507, INRIA. 1998. inria-00073177

HAL Id: inria-00073177
https://inria.hal.science/inria-00073177

Submitted on 24 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Une méthode d'auto-étalonnage pince-caméra

N. Andreff, B. Espiau et R. Horaud

No 3507

Octobre 1998

THÈME 3



ISSN 0249-6399

INRIA
RHÔNE-ALPES

Une méthode d'auto-étalonnage pince-caméra

N. Andreff*, B. Espiau† et R. Horaud‡

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projets BIP et MOVI

Rapport de recherche n° 3507 — Octobre 1998 — 38 pages

Résumé : Ce travail est lié à l'asservissement visuel d'une caméra montée sur un bras robotique. La loi de commande d'un tel système dépend de la position/orientation relative de la caméra et du repère de commande du robot. L'étalonnage pince-caméra consiste à déterminer cette transformation rigide, composée d'une rotation et d'une translation. Ce rapport expose une méthode de résolution linéaire du problème. Elle est basée sur la ressemblance du problème d'étalonnage pince-caméra avec l'équation de Sylvester. Elle permet ainsi l'analyse algébrique du problème, grâce aux propriétés du produit tensoriel. Cette méthode est à la base de la contribution principale de ce travail qui réside en une méthode d'auto-étalonnage pince-caméra qui permet de s'affranchir, en pratique, de mire de calibration. Pour cela, les mouvements de la caméra sont obtenus par reconstruction euclidienne plutôt que par calcul de pose.

Mots-clé : étalonnage pince-caméra, auto-étalonnage pince-caméra reconstruction euclidienne, équation de Sylvester, produit tensoriel (Kronecker)

(Abstract: pto)

* Nicolas.Andreff@inrialpes.fr
† Bernard.Espiau@inrialpes.fr
‡ Radu.Horaud@inrialpes.fr

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN (France)
Téléphone : 04 76 61 52 00 - International: +33 4 76 61 52 00
Télécopie : 04 76 61 52 52 - International: +33 4 76 61 52 52

Hand-Eye Autocalibration

Abstract: This work is related to the visual servoing of an arm mounted camera. The control law of such a system involves the relative position/orientation of the camera with respect to the robot effector. Hand-eye calibration is the determination of this rigid transformation, composed of a rotation and a translation. This report presents a new linear method for hand-eye calibration. It is based on the similarity of the problem with the Sylvester equation. It ths allows an algebraic analysis thanks to the tensor (or Kronecker) product. However, the main contribution of this work lies in the fundamentally new hand-eye autocalibration method which is built upon the linear method. Such an autocalibration allows to get rid of any calibration block and uses Euclidean reconstruction rather than pose estimation to compute the camera motions.

Key-words: hand-eye calibration, hand-eye autocalibration, Euclidean reconstruction, Sylvester equation, tensor (Kronecker) product

Table des matières

1 Introduction **5**
1.1 Introduction . . . 5
1.2 Transformation pince-caméra et asservissement visuel . . . 5
1.3 Transformation pince-caméra et résolution . . . 6
1.4 Pose ou reconstruction? . . . 6
1.5 Petits mouvements . . . 6
1.6 Composition du rapport . . . 7

2 Bref état de l'art **8**
2.1 Introduction . . . 8
2.2 Méthodes principales . . . 8
2.2.1 Axe/angle . . . 8
2.2.2 Quaternions duaux unitaires . . . 8
2.2.3 Minimisation non linéaire . . . 9

3 Méthode linéaire avec calcul de pose **10**
3.1 Introduction . . . 10
3.2 Formulation du problème . . . 10
3.2.1 Construction du système linéaire . . . 10
3.2.2 Étude de rang . . . 11
3.2.3 Interprétation . . . 15
3.2.4 Résolution effective . . . 16
3.3 Transformation "moyenne" . . . 16
3.3.1 Angles Roulis-Tangage-Lacet . . . 16
3.3.2 Axe/angle . . . 17
3.3.3 Calcul robuste . . . 18
3.4 Mesure d'erreur . . . 19
3.5 Simulations . . . 20
3.5.1 Influence du bruit . . . 20
3.5.2 Influence du nombre de mouvements . . . 20
3.6 Résultats expérimentaux . . . 23
3.6.1 Trajectoire 1 . . . 23
3.6.2 Trajectoire 2 . . . 23
3.7 Conclusion . . . 26

4 Méthode linéaire avec reconstruction **27**
4.1 Introduction . . . 27
4.2 Nouvelle formulation . . . 27
4.3 Résultats expérimentaux . . . 28
4.4 Conclusion . . . 29

5 Conclusion **32**

A Annexe **35**
A.1 Contraintes orthogonales 35
A.2 Produit de Kronecker et rotations 35
A.3 Démonstration de la Proposition 2 36

Chapitre 1

Introduction

1.1 Introduction

Pour commander un système robotique, il existe deux grandes familles de méthodes : les méthodes de planification qui déterminent à l'avance la trajectoire que doit suivre le robot et les méthodes de commande référencée capteur qui donnent un but à atteindre au robot en terme d'un signal en provenance du capteur. Lorsque l'on veut commander un robot en commande référencée capteur, il est nécessaire de pouvoir relier les informations en provenance du capteur au repère de commande effectif du robot. En particulier, dans le cadre de l'asservissement visuel, qui est un cas de commande référencée capteur où le capteur est un système de vision artificielle, on veut connaître la position/orientation (ou *transformation rigide*) qui relie le repère de commande du robot au repère dans lequel sont effectuées les opérations de vision (calcul de pose, reconstruction, etc.). Ce problème est traditionnellement appelé *étalonnage pince-caméra.*

Nous nous plaçons ici dans le cas d'un système composé d'un bras robotique sur lequel est fixée une caméra. Nous appellerons repère pince, le repère de commande du robot et, tout naturellement, repère caméra, le repère associé à la caméra.

Le reste de ce chapitre introductif se décompose ainsi : d'abord, nous ferons apparaître la transformation pince-caméra dans un exemple très simple d'asservissement visuel; puis, nous montrerons comment la transformation pince-caméra relie les déplacements de la pince et de la caméra; ensuite, nous verrons qu'il existe plusieurs moyens d'obtenir les déplacements de la caméra et enfin, nous présenterons la composition du rapport qui découle de ces remarques préliminaires.

1.2 Transformation pince-caméra et asservissement visuel

Afin de mettre en évidence la nécessité de connaître la transformation pince-caméra lors de l'asservissement visuel, prenons comme exemple simple, l'asservissement en position avec convergence exponentielle par *fonction de tâche.* Amplement développée dans [SLBE91], la notion de fonction de tâche est présentée dans [Cha90] et [ECR92] avec son application à l'asservissement visuel.

Dans cet exemple, nous avons :

$$\left\{ \begin{array}{rcl} \vec{\mathbf{T}}_c & = & -\lambda \vec{\mathbf{e}} \\ \vec{\mathbf{e}} & = & \mathbf{L}^{T^+}(\vec{\mathbf{s}} - \vec{\mathbf{s}}^*) \end{array} \right.$$

où $\vec{\mathbf{T}}_c$ est le torseur cinématique appliqué au robot et exprimé dans le repère pince; λ est un facteur de gain; $\vec{\mathbf{e}}$ est la fonction de tâche que l'on veut réguler à 0; $\mathbf{L}^T$ est appelée matrice d'interaction et est exprimée dans le repère pince; enfin, $\vec{\mathbf{s}}$ (resp. $\vec{\mathbf{s}}^*$) représente la primitive image courante (resp. désirée).

La matrice d'interaction $\mathbf{L}^T$ est fonction des paramètres intrinsèques de la caméra (α_u et α_v, les facteurs d'échelle des axes de l'image et θ l'angle entre ces axes), d'une matrice d'interaction canonique exprimée dans le repère caméra ($\mathbf{L}_c^T$) et de la transformation pince-caméra ($\mathbf{X} = (\mathbf{R}_X, \vec{\mathbf{t}_X})$) :

$$\mathbf{L}^T = \begin{pmatrix} -\alpha_u & \alpha_u cotg\theta \\ 0 & -\frac{\alpha_v}{sin\theta} \end{pmatrix} \cdot \mathbf{L}_c^T \cdot \begin{pmatrix} \mathbf{R}_X & -\mathbf{R}_X \mathbf{A}_s(-\mathbf{R}_X{}^T \vec{\mathbf{t}_X}) \\ 0 & \mathbf{R}_X \end{pmatrix}$$

où $\mathbf{A}_s(\bullet)$ est la matrice antisymétrique associée au produit vectoriel et $\mathbf{L}_C^T$ dépend de la position de la caméra et du type de primitive image utilisé (points, droites, etc.).

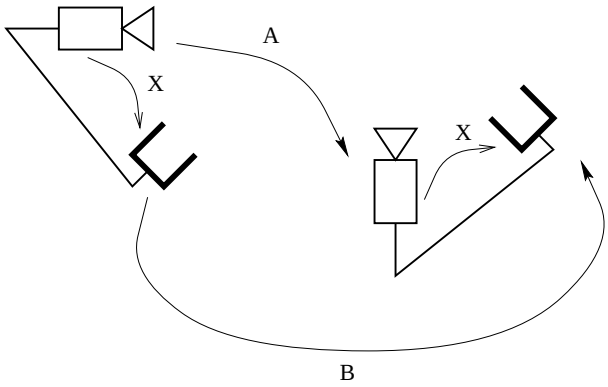


FIG. 1.1 – *Les deux positions d'un même système pince-caméra séparés par le déplacement* **A** *de la caméra, auquel est associé le déplacement* **B** *du robot au moyen de la transformation pince-caméra* **X**.

1.3 Transformation pince-caméra et résolution

Pour faire apparaître la transformation pince-caméra, on peut considérer le déplacement d'un système constitué d'une pince et d'une caméra fixées l'une à l'autre (Figure 1.1)

Lorsque la pince se déplace selon la transformation rigide $\mathbf{B} = (\mathbf{R}_B, \vec{\mathbf{t}}_B)$, la caméra se déplace, elle, selon la transformation rigide $\mathbf{A} = (\mathbf{R}_A, \vec{\mathbf{t}}_A)$. **A** et **B** sont liées par :

$$\mathbf{AX} = \mathbf{XB}$$

C'est cette équation matricielle qu'il faut maintenant résoudre après avoir obtenu **B** grâce au modèle cinématique du robot et **A** par des calculs de pose ou par reconstruction tridimensionnelle.

1.4 Pose ou reconstruction ?

Pour connaître les déplacements **A**, on peut soit procéder à des calculs de pose, soit à une reconstruction tridimensionnelle.

Dans un calcul de pose, on observe une mire dont on connaît parfaitement la forme et la taille. On en déduit la pose, c'est-à-dire la position/orientation de la mire par rapport au repère caméra. On obtient le déplacement de la caméra par composition de changements de repère entre 2 poses successives.

Lors d'un calcul de reconstruction tridimensionnelle, on observe une scène inconnue de plusieurs points de vue. On en déduit la structure géométrique de la scène et sa position/orientation par rapport au repère caméra et le déplacement de la caméra entre les points de vue successifs. Cependant, comme la scène est inconnue, il reste une ambiguïté impossible à lever : "La caméra voit-elle une scène de grande taille depuis un point de vue éloigné ou est-elle proche d'une scène de petite taille?". Cela se traduit par l'apparition d'un facteur d'échelle inconnu dans les positions et les déplacements de la caméra.

1.5 Petits mouvements

Nous resterons attentifs au long de ce rapport au cas où les mouvements du robot sont restreints à de faible amplitude. En effet, il est attrayant de pouvoir utiliser les mouvements asservis du robot pour faire (ou mettre à jour) l'étalonnage pince-caméra.

Un autre intérêt à ces petits mouvements apparaît si l'on considère que la transformation pince-caméra est complètement inconnue. Dans ce cas, il est difficile de déplacer la caméra de manière à ce qu'elle voie toujours la même mire ou scène. En revanche, si l'on se restreint à des mouvements de faible amplitude, on augmente les chances de garder l'objet dans le champ de vision. Cela permet alors de simplifier la mise en oeuvre pratique de l'étalonnage.

En revanche, l'utilisation de petits mouvements présente l'inconvénient de ne pas pouvoir utiliser de représentation réduite des rotations (axe/angle ou quaternions) car alors les axes de rotation sont mal définis.

1.6 Composition du rapport

Le chapitre 2 fait un bref état de l'art des méthodes existantes. Le chapitre 3 présente une méthode linéaire d'étalonnage pince-caméra avec calcul de pose, qui permet une analyse algébrique du problème et des mouvements nécessaires à sa résolution. Le chapitre 4 étend cette méthode au cas où les déplacements de la caméra sont obtenus par reconstruction et ne sont donc connus qu'à un facteur d'échelle près (auto-étalonnage pince-caméra).

Chapitre 2

Bref état de l'art

2.1 Introduction

Il ne s'agit ici que d'un bref état de l'art. Nous n'y présentons que les méthodes qui nous ont semblé les plus courantes. Ces méthodes ont toutes trait à l'étalonnage pince-caméra avec mire. En effet, il n'existe pas à notre connaissance de méthode d'auto-étalonnage pince-caméra.

Ces méthodes serviront de base de comparaison pour nos méthodes.

2.2 Méthodes principales

2.2.1 Axe/angle

Cette méthode, présentée dans [TL89], résout le problème d'étalonnage pince-caméra en 2 étapes. On détermine d'abord l'axe $\vec{\mathbf{n}}$ et l'angle θ de la rotation pince-caméra à l'aide des équations suivantes :

$$As(\vec{\mathbf{n}}_A + \vec{\mathbf{n}}_B)\vec{\mathbf{n}} = \vec{\mathbf{n}}_A - \vec{\mathbf{n}}_B \tag{2.1}$$

$$\theta = 2\arctan(\|\vec{\mathbf{n}}\|) \tag{2.2}$$

où $\vec{\mathbf{n}}_A$ (resp. $\vec{\mathbf{n}}_B$) est l'axe de rotation du mouvement effectué par la caméra (resp. par le robot) et $As(.)$ est la matrice antisymétrique associée au produit vectoriel. Puis, l'on construit la matrice de rotation $\mathbf{R}$ associée à $\vec{\mathbf{n}}$ et θ afin de trouver la translation pince-caméra $\vec{\mathbf{t}}$ par la méthode des moindres carrés appliquée au système :

$$(\mathbf{R}_B - \mathbf{I})\vec{\mathbf{t}} = \mathbf{R}\vec{\mathbf{t}}_A - \vec{\mathbf{t}}_B$$

Cette méthode présente l'avantage d'être simple à mettre en œuvre mais elle repose essentiellement sur l'hypothèse que les mouvements re rotation sont suffisamment amples pour pouvoir en extraire les axes. De plus, l'axe $\vec{\mathbf{n}}$ est obtenu par résolution aux moindres carrés, ce qui rend les résultats sensibles aux mesures aberrantes.

2.2.2 Quaternions duaux unitaires

Cette méthode, présentée dans [DBC96], est basée sur la représentation des déplacements rigides par des quaternions duaux unitaires, une extension des quaternions, de la forme :

$$(\mathbf{R}, \vec{\mathbf{t}}) \longleftrightarrow \tilde{q} \triangleq \begin{pmatrix} \mathbf{q}^T \\ \mathbf{q}'^T \end{pmatrix} \triangleq \begin{pmatrix} q_0 & \vec{\mathbf{q}}^T \\ q'_0 & \vec{\mathbf{q}}'^T \end{pmatrix} \text{ avec } \|\tilde{q}\| = 1$$

où $\mathbf{q}$ et $\mathbf{q}'$ sont des quaternions unitaires. Quant à q_0 et $\vec{\mathbf{q}}$, ce sont les parties scalaire et vectorielle de $\mathbf{q}$.

Avec cette représentation, l'équation $\mathbf{AX} = \mathbf{XB}$ s'écrit :

$$\tilde{a} = \tilde{q}\tilde{b}\bar{\tilde{q}} \tag{2.3}$$

où $\tilde{a}$ est associé à la transformation rigide $\mathbf{A}$, $\tilde{b}$ à $\mathbf{B}$ et $\tilde{q}$ à $\mathbf{X}$, et mène à :

$$\begin{pmatrix} \vec{\mathbf{a}} - \vec{\mathbf{b}} & As(\vec{\mathbf{a}} + \vec{\mathbf{b}}) & 0_{3\times1} & 0_{3\times3} \\ \vec{\mathbf{a}}' - \vec{\mathbf{b}}' & As(\vec{\mathbf{a}}' + \vec{\mathbf{b}}') & \vec{\mathbf{a}} - \vec{\mathbf{b}} & As(\vec{\mathbf{a}} + \vec{\mathbf{b}}) \end{pmatrix} \begin{pmatrix} \mathbf{q} \\ \mathbf{q}' \end{pmatrix} = 0 \tag{2.4}$$

En empilant $n \geq 2$ matrices de ce type, correspondant à n mouvements du système pince-caméra, on obtient une matrice $\mathbf{T}$ de dimension $6n \times 8$ et de rang 6 (en l'absence de bruit). Alors, la solution du système est une combinaison linéaire de 2 vecteurs indépendants du noyau. Ces vecteurs sont obtenus par SVD. Ils sont, dans ce cas, les 2 derniers vecteurs singuliers droits (notés $\vec{\mathbf{v}}_7$ et $\vec{\mathbf{v}}_8$) correspondant aux 2 plus petites valeurs singulières de T :

$$\begin{pmatrix} \mathbf{q} \\ \mathbf{q}' \end{pmatrix} = \lambda \vec{\mathbf{v}}_7 + \mu \vec{\mathbf{v}}_8$$

Comme le résultat doit être un quaternion dual unitaire, nous obtenons deux équations en λ et μ dont la solution détermine la transformation pince-caméra :

$$\lambda^2 \vec{\mathbf{u}}_1^T \vec{\mathbf{u}}_1 + 2\lambda\mu \vec{\mathbf{u}}_1^T \vec{\mathbf{u}}_2 + \mu^2 \vec{\mathbf{u}}_2^T \vec{\mathbf{u}}_2 = 1 \quad (2.5)$$

$$\lambda^2 \vec{\mathbf{u}}_1^T \vec{\mathbf{v}}_1 + \lambda\mu \left(\vec{\mathbf{u}}_1^T \vec{\mathbf{v}}_2 + \vec{\mathbf{u}}_2^T \vec{\mathbf{v}}_1 \right) + \mu^2 \vec{\mathbf{u}}_2^T \vec{\mathbf{v}}_2 = 0 \quad (2.6)$$

avec les conventions $\vec{\mathbf{v}}_7 = (\vec{\mathbf{u}}_1^T, \vec{\mathbf{v}}_1^T)^T$ et $\vec{\mathbf{v}}_8 = (\vec{\mathbf{u}}_2^T, \vec{\mathbf{v}}_2^T)^T$.

Cette méthode permet le calcul simultané de la rotation et de la translation pince-caméra. Cependant, elle repose elle aussi sur l'hypothèse que les rotations sont conséquentes. Enfin, l'application des contraintes pour trouver λ et μ nécessite de trouver une solution réelle à une équation polynomiale du second degré dont il n'est pas prouvé que le déterminant est positif.

2.2.3 Minimisation non linéaire

Cette méthode, présentée dans [HD95], réalise une minimisation non linéaire, selon l'algorithme de Levenberg-Marquardt [PTVF92], de la fonction de coût suivante qui utilise la représentation des rotations par des quaternions unitaires [1] :

$$\begin{aligned} f(\mathbf{q}, \vec{\mathbf{t}}) = \; & \lambda_1 \Sigma_{i=1}^n \| \vec{\mathbf{n}}_{A_i} - \mathbf{q} * \vec{\mathbf{n}}_{B_i} * \bar{\mathbf{q}} \|^2 \\ & + \lambda_2 \Sigma_{i=1}^n \| \mathbf{q} * \vec{\mathbf{t}}_{B_i} * \bar{\mathbf{q}} - (\mathbf{R}_{A_i} - \mathbf{I}) \vec{\mathbf{t}} - \vec{\mathbf{t}}_{A_i} \|^2 \\ & + \lambda (1 - \mathbf{q}^T \mathbf{q})^2 \end{aligned}$$

où n est le nombre de mouvements, λ est un multiplicateur de Lagrange, λ_1 et λ_2 sont deux poids au choix de l'utilisateur et les notations restantes sont identiques à celles des méthodes précédentes.

Cette méthode non-linéaire nécessite une initialisation proche de la solution sans quoi la convergence est longue. De plus, elle nécessite aussi des rotations de grande amplitude, ainsi le choix de poids par l'utilisateur. Elle permet cependant une grande précision de résolution.

1. Les quaternions unitaires utilisés ici sont la partie non duale des quaternions **duaux** unitaires vus dans la méthode précédente.

Chapitre 3

Méthode linéaire avec calcul de pose

3.1 Introduction

Dans ce chapitre, nous présentons une méthode linéaire d'étalonnage pince-caméra. Cette méthode est inspirée par la ressemblance du problème d'étalonnage pince-caméra avec l'équation matricielle dite de Sylvester. Elle a, notamment, pour but d'aider à l'analyse algébrique des mouvements nécessaires à l'étalonnage pince-caméra.

3.2 Formulation du problème

3.2.1 Construction du système linéaire

Pour faire apparaître la transformation pince-caméra, on peut considérer le déplacement d'un système constitué d'une pince et d'une caméra fixées l'une à l'autre (Figure 1.1)

Lorsque la pince se déplace selon la transformation rigide $\mathbf{B} = (\mathbf{R}_B, \vec{\mathbf{t}}_B)$ (exprimée dans le repère pince), la caméra se déplace, elle, selon la transformation rigide $\mathbf{A} = (\mathbf{R}_A, \vec{\mathbf{t}}_A)$ (exprimée dans le repère caméra). $\mathbf{A}$ et $\mathbf{B}$ sont alors liées par :

$$\mathbf{AX} = \mathbf{XB} \tag{3.1}$$

qui se décompose en :

$$\mathbf{R}_A\mathbf{R}_X - \mathbf{R}_X\mathbf{R}_B = \mathbf{0} \tag{3.2}$$

$$\mathbf{R}_X\vec{\mathbf{t}}_B + \vec{\mathbf{t}}_X - \mathbf{R}_A\vec{\mathbf{t}}_X = \vec{\mathbf{t}}_A \tag{3.3}$$

L'équation (3.2) peut s'écrire comme une équation de Sylvester, c'est-à-dire une équation matricielle de la forme :

$$\mathbf{UV} + \mathbf{VW} = \mathbf{T} \tag{3.4}$$

avec dans notre cas, $\mathbf{U} = \mathbf{R}_A$, $\mathbf{W} = -\mathbf{R}_B$ et $\mathbf{T} = \mathbf{0}$.

Les études classiques portant sur la résolution d'une telle équation [BS72, GNL79, RB89, HR92, DSH95, Dua96] mettent en évidence une condition nécessaire d'unicité de la solution : les matrices $\mathbf{U}$ et $\mathbf{W}$ ne doivent pas avoir de valeurs propres opposées. Ici, $\mathbf{U}$ est une matrice de rotation et a donc une valeur propre égale à 1 et $\mathbf{W}$ est le produit d'une matrice de rotation par -1 et a donc -1 pour valeur propre. La condition d'unicité n'est donc pas vérifiée et ces méthodes ne sont pas applicables.

Cependant, certains auteurs [RB89, HR92, DSH95], formulent le problème d'une manière intéressante : en réordonnant les coefficients de la matrice $\mathbf{V}$ sous la forme d'un vecteur (une ligne après l'autre), on obtient un système équivalent à (3.4) de la forme :

$$(\mathbf{U} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{W})vec(\mathbf{V}) = vec(\mathbf{T}) \tag{3.5}$$

où

- *vec* est l'opérateur introduit par Neudecker [Neu69] transformant une matrice $\mathbf{M}$ de dimension $(m \times n)$ et de terme général M_{ij} en le vecteur

$$vec(\mathbf{M}) = (M_{11}, M_{12}, \ldots, M_{1n}, M_{21}, M_{22}, \ldots, M_{mn})^T$$

– le symbole $\otimes$ représente le produit de Kronecker [1] (voir [Bel60] pour les premiers théorèmes et [Bre78] pour une liste exhaustive de résultats théoriques) défini pour les matrices $\mathbf{M}$ et $\mathbf{N}$ de dimensions respectives $(m \times n)$ et $(o \times p)$ par la matrice de dimension $(mo \times np)$:

$$\mathbf{M} \otimes \mathbf{N} = \begin{pmatrix} M_{11}\mathbf{N} & \dots & M_{1n}\mathbf{N} \\ \vdots & \ddots & \vdots \\ M_{m1}\mathbf{N} & \dots & M_{mn}\mathbf{N} \end{pmatrix}$$

En s'inspirant de cette formulation, on obtient, après application de diverses propriétés du produit de Kronecker, un système équivalent au système formé par les équations (3.2) et (3.3), exprimé ici sous forme de blocs :

$$\left(\begin{array}{c|c} \mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B & \mathbf{0}_{9\times 3} \\ \hline \mathbf{I}_3 \otimes (\vec{\mathbf{t}}_B^T) & \mathbf{I}_3 - \mathbf{R}_A \end{array}\right) \left(\begin{array}{c} vec(\mathbf{R}_X) \\ \hline \vec{\mathbf{t}}_X \end{array}\right) = \left(\begin{array}{c} \vec{\mathbf{0}}_{9\times 1} \\ \hline \vec{\mathbf{t}}_A \end{array}\right) \tag{3.6}$$

Parmi les manipulations mises en jeu par la création de ce système, on notera plus particulièrement la réécriture de (3.2) en

$$\mathbf{R}_X - \mathbf{R}_A \mathbf{R}_X \mathbf{R}_B^T = 0$$

en utilisant la propriété fondamentale des rotations ($\mathbf{R}^{-1} = \mathbf{R}^T$) plutôt que

$$\mathbf{R}_X - \mathbf{R}_A \mathbf{R}_X \mathbf{R}_B^{-1} = 0$$

Cela permet de garantir que la solution du système est bien une matrice de transformation rigide (voir Annexe A.1) et ainsi d'améliorer le comportement numérique de notre méthode.

Nous noterons par la suite $\mathbf{C}$, la matrice de ce système.

3.2.2 Étude de rang

Cas d'une translation pure

Dans le cas d'un mouvement de translation pure, on a la double égalité suivante :

$$\mathbf{R}_A = \mathbf{R}_B = \mathbf{I}$$

La matrice $\mathbf{C}$ du système (3.6) devient alors :

$$\mathbf{C}_{trans} = \left(\begin{array}{c|c} \mathbf{0}_{9\times 9} & \mathbf{0}_{9\times 3} \\ \hline \mathbf{I}_3 \otimes (\vec{\mathbf{t}}_B^T) & \mathbf{0}_{3\times 3} \end{array}\right) \tag{3.7}$$

Son rang est donc celui de la sous-matrice $\mathbf{I}_3 \otimes \vec{\mathbf{t}}_B^T$ [2]. Cette matrice de dimension (3×9) s'écrit :

$$\mathbf{I}_3 \otimes \vec{\mathbf{t}}_B^T = \begin{pmatrix} \vec{\mathbf{t}}_B^T & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{1\times 3} & \vec{\mathbf{t}}_B^T & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \vec{\mathbf{t}}_B^T \end{pmatrix}$$

qui est de rang nul si $\vec{\mathbf{t}}_B = 0$ (cas inutile) et de rang 3 sinon.

De plus, on remarquera que dans le cas d'une translation pure, le système (3.2)-(3.3) se simplifie en :

$$\begin{aligned} \mathbf{R}_X &= \mathbf{R}_X \\ \mathbf{R}_X \vec{\mathbf{t}}_B &= \vec{\mathbf{t}}_A \end{aligned}$$

Cela est cohérent avec la simplification du système (3.7) en :

$$(\mathbf{I}_3 \otimes \vec{\mathbf{t}}_B^T) vec(\mathbf{R}_X) = \vec{\mathbf{t}}_A$$

et signifie qu'il n'est pas possible de retrouver $\vec{\mathbf{t}}_X$ à partir de simples translations.

1. Cet opérateur est aussi appelé produit tensoriel.
2. On peut abandonner le parenthèsage car $(\mathbf{M} \otimes \mathbf{N})^T = (\mathbf{M}^T) \otimes (\mathbf{N}^T)$.

Cas de n translations pures Dans le cas de $n > 1$ translations pures, on peut empiler n systèmes simplifiés et obtenir le système de $3n$ équations à 9 inconnues :

$$\begin{pmatrix} \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{B_1}^T \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{B_2}^T \\ \vdots \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{B_n}^T \end{pmatrix} vec(\mathbf{R}_X) = \begin{pmatrix} \vec{\mathbf{t}}_{A_1} \\ \vec{\mathbf{t}}_{A_2} \\ \vdots \\ \vec{\mathbf{t}}_{A_n} \end{pmatrix}$$

Ce système est de rang 9 si l'on dispose de 3 translations indépendantes.

Conclusion Dans le cas d'une translation pure (non nulle), le rang du système est de rang 3. La simplification du système met en évidence qu'avec 3 translations indépendantes, on peut retrouver $\mathbf{R}_X$, l'orientation relative de la pince vis-à-vis de la caméra. En revanche, il est impossible de retrouver la translation séparant la pince de la caméra.

Cas d'une rotation pure

Il faut d'abord se rappeler que lorsque deux repères rigidement liés se déplacent, ils correspondent au même vissage et possèdent donc 2 valeurs communes : ils ont le même angle de rotation et les produits scalaires de leur vecteur de translation avec leur axe de rotation respectifs sont égaux.

Il faut ensuite remarquer que les déplacements de la caméra et du robot ne peuvent pas être simultanément des rotations pures, à moins que leurs repères respectifs n'aient la même origine (ce qui signifierait alors que la translation pince-caméra $\vec{\mathbf{t}}_X$ était nulle) ou que l'axe de la rotation du repère caméra ne soit parallèle à la translation pince-caméra.

Pour cela, réécrivons l'équation (3.3) :

$$\vec{\mathbf{t}}_A = \mathbf{R}_x \vec{\mathbf{t}}_B + (\mathbf{I} - \mathbf{R}_A)\vec{\mathbf{t}}_X$$

Si nous mettons $\vec{\mathbf{t}}_B = 0$ dans cette équation, nous obtenons :

$$\vec{\mathbf{t}}_A = (\mathbf{I} - \mathbf{R}_A)\vec{\mathbf{t}}_X$$

Ainsi, $\vec{\mathbf{t}}_A$ ne peut être nul que dans les deux cas énoncés précédemment : $\vec{\mathbf{t}}_X$ est l'axe de rotation de $\mathbf{R}_A$ ou $\vec{\mathbf{t}}_X = 0$. En effet, le cas où $\mathbf{I} = \mathbf{R}_A$ ne peut se produire que lorsque $\mathbf{R}_B$ est aussi l'identité (en vertu de la première contrainte de rigidité), ce qui signifierait que ni le robot, ni la caméra n'ont bougé.

Inversement, si nous mettons $\vec{\mathbf{t}}_A = 0$, nous obtenons :

$$\vec{\mathbf{t}}_B = -\mathbf{R}_x^T(\mathbf{I} - \mathbf{R}_A)\vec{\mathbf{t}}_X$$

qui s'annule dans les mêmes cas que précédemment.

Supposons donc que $\vec{\mathbf{t}}_B = 0$, c'est-à-dire que l'on demande au robot d'effectuer une rotation pure. Dans ce cas, nous obtenons une autre expression pour $\mathbf{C}$:

$$\mathbf{C}_{rot} = \left(\begin{array}{c|c} \mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B & \mathbf{0}_{9\times 3} \\ \hline \mathbf{0}_{3\times 9} & \mathbf{I}_3 - \mathbf{R}_A \end{array}\right) \tag{3.8}$$

La matrice $\mathbf{C}_{rot}$ est donc diagonale par blocs. Son rang est donc :

$$rg(\mathbf{C}_{rot}) = rg(\mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B) + rg(\mathbf{I}_3 - \mathbf{R}_A)$$

Rang de $\mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B$ Calculer ce rang revient à étudier les valeurs propres de $\mathbf{R}_A \otimes \mathbf{R}_B$.

Soient $\lambda_i (i \in \{1,2,3\})$ (resp. $\mu_j (j \in \{1,2,3\})$), les 3 valeurs propres de $\mathbf{R}_A$ (resp. $\mathbf{R}_B$)). On sait que dans les 2 cas, ces valeurs propres sont $\{e^{i\theta}, e^{-i\theta}, 1\}$ où θ est l'angle commun de rotation.

De plus, une propriété du produit de Kronecker est que les valeurs propres du produit $\mathbf{A} \otimes \mathbf{B}$ sont les produits des valeurs propres de $\mathbf{A}$ par celles de $\mathbf{B}$ [Bel60].

Dans le cas présent, ces produits sont par conséquent :

$$\{1, 1, 1, e^{i\theta}, e^{i\theta}, e^{-i\theta}, e^{-i\theta}, e^{2i\theta}, e^{-2i\theta}\}$$

On distingue 3 cas :

- quand $\theta = 2k\pi$, k étant un entier quelconque, alors toutes les valeurs propres sont égales à 1 (ce qui est normal car alors les deux rotations sont nulles);
- quand $\theta = (2k+1)\pi$, il n'y a plus que 2 valeurs propres distinctes, 1 et -1, de multiplicités respectives 5 et 4;
- quand $\theta \neq k\pi$, la valeur propre 1 est de multiplicité 3.

Le rang de $\mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B$ est donc égal à 6 dans le cas général, 4 dans le cas d'une rotation de $(2k+1)\pi$ et 0 dans le cas dégénéré d'un mouvement nul. De plus, le rang de $\mathbf{I}_3 - \mathbf{R}_A$ est toujours 2, dans le cas d'une rotation pure non nulle.

En conclusion, en excluant le cas dégénéré et en considérant la mesure principale de l'angle de rotation θ (i.e. ramenée dans l'intervalle $[0, 2\pi[$) :

$$\ker(\mathbf{C}_{rot}) = \underbrace{\ker(\mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B)}_{\begin{array}{l}=6 \text{ si } \theta \neq \pi \\ =4 \text{ sinon}\end{array}} + \underbrace{\ker(\mathbf{I}_3 - \mathbf{R}_A)}_{=2} = \begin{cases} =8 & \text{si } \theta \neq \pi \\ =6 & \text{sinon} \end{cases}$$

Cas de n rotations pures Dans le cas de $n > 1$ rotations pures du robot (c'est-à-dire que les $\vec{\mathbf{t}}_{B_i}$ sont nuls), on peut former le système de $12n$ équations et 12 inconnues suivant :

$$\begin{pmatrix} \mathbf{I}_9 - \mathbf{R}_{A_1} \otimes \mathbf{R}_{B_1} & \mathbf{0}_{9\times 3} \\ \mathbf{0}_{3\times 9} & \mathbf{I}_3 - \mathbf{R}_{A_1} \\ \mathbf{I}_9 - \mathbf{R}_{A_2} \otimes \mathbf{R}_{B_2} & \mathbf{0}_{9\times 3} \\ \mathbf{0}_{3\times 9} & \mathbf{I}_3 - \mathbf{R}_{A_2} \\ \vdots & \vdots \\ \mathbf{I}_9 - \mathbf{R}_{A_n} \otimes \mathbf{R}_{B_n} & \mathbf{0}_{9\times 3} \\ \mathbf{0}_{3\times 9} & \mathbf{I}_3 - \mathbf{R}_{A_n} \end{pmatrix} vec(\mathbf{X}) = \begin{pmatrix} \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{t}}_{A_1} \\ \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{t}}_{A_2} \\ \vdots \\ \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{t}}_{A_n} \end{pmatrix} \tag{3.9}$$

Ce système se découple en :

$$\begin{pmatrix} \mathbf{I}_3 - \mathbf{R}_{A_1} \\ \mathbf{I}_3 - \mathbf{R}_{A_2} \\ \vdots \\ \mathbf{I}_3 - \mathbf{R}_{A_n} \end{pmatrix} \vec{\mathbf{t}}_X = \begin{pmatrix} \vec{\mathbf{t}}_{A_1} \\ \vec{\mathbf{t}}_{A_2} \\ \vdots \\ \vec{\mathbf{t}}_{A_n} \end{pmatrix} \tag{3.10}$$

et

$$\begin{pmatrix} \mathbf{I}_9 - \mathbf{R}_{A_1} \otimes \mathbf{R}_{B_1} \\ \mathbf{I}_9 - \mathbf{R}_{A_2} \otimes \mathbf{R}_{B_2} \\ \vdots \\ \mathbf{I}_9 - \mathbf{R}_{A_n} \otimes \mathbf{R}_{B_n} \end{pmatrix} vec(\mathbf{R}_X) = \begin{pmatrix} \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{0}}_{9\times 1} \\ \vdots \\ \vec{\mathbf{0}}_{9\times 1} \end{pmatrix} \tag{3.11}$$

Or, le système (3.10) est de rang plein dès qu'il y a 2 rotations pures d'axes distincts; d'où la proposition suivante.

Proposition 1 *Si l'on dispose d'au moins 2 rotations pures d'axes distincts, alors le vecteur de translation $\vec{\mathbf{t}}_X$ de la transformation pince-caméra est la solution du système de rang plein (3.10).*

En ce qui concerne la partie rotationnelle, on a la proposition suivante, dont la preuve se trouve en Annexe A.

Proposition 2 *Si l'on dispose d'au moins 2 rotations pures d'axes distincts, alors le système (3.11) est de rang 8, son noyau $\mathcal{K}$ est de dimension 1 et la matrice de rotation $\mathbf{R}_X$ de la transformation pince-caméra est obtenue par :*

$$\mathbf{R}_X = \frac{sgn(\det(\mathbf{V}))}{|\det(\mathbf{V})|^{\frac{1}{3}}}\mathbf{V}$$

où sgn est l'opérateur qui renvoie le signe de son argument, $\mathbf{V} = vec^{-1}(\vec{\mathbf{v}})$ et $\vec{\mathbf{v}}$ est un vecteur directeur de $\mathcal{K}$.

Remarque Si, au lieu de disposer de rotations pures du robot, on dispose de rotations pures de la caméra, alors on peut se ramener au cas précédent en inversant les rôles de $\mathbf{A}$ et $\mathbf{B}$. Cela nous donnera la transformation inverse de celle que nous cherchons (au lieu de la transformation pince-caméra, nous aurons la transformation caméra-pince).

Conclusion Avec 2 rotations pures d'axes différents, on peut retrouver entièrement la transformation pince-caméra.

Cas général

Dans le cas général, ni les rotations, ni les translations ne sont nulles. De (3.6), (3.7) et (3.8), on obtient que la matrice $\mathbf{C}$ du système linéaire (3.6) est égale à la somme de $\mathbf{C}_{trans}$ et de $\mathbf{C}_{rot}$:

$$\mathbf{C} = \mathbf{C}_{trans} + \mathbf{C}_{rot}$$

On a donc l'inégalité :

$$\ker(\mathbf{C}) \leq \ker(\mathbf{C}_{trans}) + \ker(\mathbf{C}_{rot}) \leq 11$$

Un seul mouvement général ne suffit donc pas pour résoudre le problème. Cela est cohérent avec [TL89].

En fait, le rang de $\mathbf{C}$ est 9 (hors cas d'une rotation de π). En effet, on se souviendra que le rang du bloc supérieur gauche de $\mathbf{C}$ ($\mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B$) est 6. Quant au rang de la partie inférieure de $\mathbf{C}$, de dimension $(3 \otimes 12)$, il est égal à 3, car dans le cas présent d'une translation $\vec{\mathbf{t}}_B$ non nulle, le bloc inférieur gauche est de rang plein et d'après ses dimensions, de rang 3.

Cas de n déplacements généraux Dans le cas de $n > 1$ déplacements généraux, nous formons le système suivant :

$$\begin{pmatrix} \mathbf{I}_9 - \mathbf{R}_{A_1} \otimes \mathbf{R}_{B_1} & \mathbf{0}_{9\times 3} \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{B_1}^T & \mathbf{I}_3 - \mathbf{R}_{A_1} \\ \mathbf{I}_9 - \mathbf{R}_{A_2} \otimes \mathbf{R}_{B_2} & \mathbf{0}_{9\times 3} \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{B_2}^T & \mathbf{I}_3 - \mathbf{R}_{A_2} \\ \vdots & \vdots \\ \mathbf{I}_9 - \mathbf{R}_{A_n} \otimes \mathbf{R}_{B_n} & \mathbf{0}_{9\times 3} \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{B_n}^T & \mathbf{I}_3 - \mathbf{R}_{A_n} \end{pmatrix} vec(\mathbf{X}) = \begin{pmatrix} \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{t}}_{A_1} \\ \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{t}}_{A_2} \\ \vdots \\ \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{t}}_{A_n} \end{pmatrix} \tag{3.12}$$

La partie droite de la matrice du système est de rang 3 dès que l'on dispose de 2 rotations d'axes distincts. La partie gauche est de rang 9 dès que l'on dispose de 2 rotations d'axes distincts et d'une translation.

Donc, avec un minimum d'une rotation pure et d'un mouvement général, le système est de rang plein et a une solution unique.

Cas d'un mouvement général et de n translations pures On voit immédiatement que si l'on ne dispose que d'une seule rotation, le système (3.12) n'est pas de rang plein car la partie droite de sa matrice est de rang 2. Il n'y a donc pas de solution unique. Cependant, nous allons voir que la partie rotationnelle peut être calculée et que la partie translationnelle appartient à une variété affine de dimension 1.

En supposant que le premier mouvement est un mouvement général et que le second est une translation pure, on peut réécrire le système (3.12) :

$$\begin{pmatrix} \mathbf{I}_9 - \mathbf{R}_{A_1} \otimes \mathbf{R}_{B_1} & \mathbf{0}_{9\times 3} \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{B_1}^T & \mathbf{I}_3 - \mathbf{R}_{A_1} \\ \mathbf{0}_9 & \mathbf{0}_{9\times 3} \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{B_2}^T & \mathbf{0}_3 \end{pmatrix} vec(\mathbf{X}) = \begin{pmatrix} \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{t}}_{A_1} \\ \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{t}}_{A_2} \end{pmatrix}$$

qui est équivalent à :

$$\begin{cases} \begin{pmatrix} \mathbf{I}_9 - \mathbf{R}_{A_1} \otimes \mathbf{R}_{B_1} & \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{B_2}^T \end{pmatrix} vec(\mathbf{R}_X) = \begin{pmatrix} \vec{\mathbf{0}}_{9\times 1} \\ \vec{\mathbf{t}}_{A_2} \end{pmatrix} \\ (\mathbf{I}_3 - \mathbf{R}_{A_1})\vec{\mathbf{t}}_X = \vec{\mathbf{t}}_{A_1} - \mathbf{R}_X \vec{\mathbf{t}}_{B_1} \end{cases}$$

Si les deux mouvements sont indépendants, (c'est-à-dire que $\vec{\mathbf{t}}_{B_2}$, l'axe $\vec{\mathbf{n}}_{B_1}$ de $\mathbf{R}_{B_1}$ et l'axe $\vec{\mathbf{n}}_{A_1}$ de $\mathbf{R}_{A_1}$ sont indépendants), alors la première équation admet une solution unique pour $\mathbf{R}_X$. En revanche, la deuxième équation admet une infinité de solutions. En effet, c'est un système à 3 inconnues et 3 équations de rang 2. On peut donc ajouter à toute solution un vecteur pris dans le noyau (c.-à-d. proportionnel à $\vec{\mathbf{n}}_{A_1}$) et obtenir une nouvelle solution. L'ensemble des solutions est donc la variété affine de dimension 1 définie par :

$$\left\{\vec{\mathbf{t}}_\perp + \alpha\vec{\mathbf{n}}_{A_1} | \alpha \in \Re\right\}$$

où $\vec{\mathbf{t}}_\perp$ est une solution du système perpendiculaire à $\vec{\mathbf{n}}_{A_1}$. Elle est unique car alors nous résolvons un système de rang 2 dans un espace de dimension 2 (le plan perpendiculaire à $\vec{\mathbf{n}}_{A_1}$ donc invariant par $\mathbf{R}_{A_1}$). En pratique, cette solution peut être déterminée par une Décomposition en Valeurs Singulières [PTVF92, §2.6] de $\mathbf{I}_3 - \mathbf{R}_{A_1}$.

Cet ensemble de solution signifie que nous ne pouvons retrouver que la projection de la translation pince-caméra sur le plan invariant associé à l'unique rotation de la caméra. Par exemple, si l'on monte une caméra sur un véhicule routier automatisé, on pourra connaître la position de celle-là par rapport à celui-ci à un unique paramètre près, la hauteur, qui n'est en général pas critique pour ce genre d'application.

On vérifie aisément que ce résultat reste valable quand on remplace le mouvement général par une rotation pure et/ou qu'il y a plus d'une translation.

3.2.3 Interprétation

Les résultats théoriques précédents sont rassemblés dans le Tableau 3.1. Il en ressort que tout type de mouvement contribue à la détermination de l'orientation relative de la pince par rapport à la caméra alors que seules les rotations participent à la détermination de sa position relative. Ceci explique que cette dernière est en général difficile à obtenir précisément.

Une autre remarque intéressante est que cette étude algébrique confirme l'analyse géométrique du problème présenté dans [Che91].

	Translation $\mathbf{R}_B = \mathbf{I}$ $\vec{\mathbf{t}}_B \neq 0$	Rotation $\mathbf{R}_B \neq \mathbf{I}$ $\vec{\mathbf{t}}_B = 0$	Mouvement général $\mathbf{R}_B \neq \mathbf{I}$ $\vec{\mathbf{t}}_B \neq 0$
Translation $\mathbf{R}_B = \mathbf{I}$ $\vec{\mathbf{t}}_B \neq 0$	$\mathbf{R}_X$ (si l'on dispose d'une troisième translation indépendante)	$\mathbf{R}_X$ $\vec{\mathbf{t}}_X(\alpha)$	$\mathbf{R}_X$ $\vec{\mathbf{t}}_X(\alpha)$
Rotation $\mathbf{R}_B \neq \mathbf{I}$ $\vec{\mathbf{t}}_B = 0$	$\mathbf{R}_X$ $\vec{\mathbf{t}}_X(\alpha)$	$\mathbf{R}_X, \vec{\mathbf{t}}_X$ Solution découplée	$\mathbf{R}_X, \vec{\mathbf{t}}_X$ Solution du cas général
Mouvement général $\mathbf{R}_B \neq \mathbf{I}$ $\vec{\mathbf{t}}_B \neq 0$	$\mathbf{R}_X$ $\vec{\mathbf{t}}_X(\alpha)$	$\mathbf{R}_X, \vec{\mathbf{t}}_X$ Solution du cas général	$\mathbf{R}_X, \vec{\mathbf{t}}_X$ Solution du cas général

TAB. 3.1 – *Résumé des résultats pour 2 mouvements indépendants*

3.2.4 Résolution effective

Séparons le système (3.12) en

$$\begin{pmatrix} \mathbf{I}_9 - \mathbf{R}_{A_1} \otimes \mathbf{R}_{B_1} \\ \mathbf{I}_9 - \mathbf{R}_{A_2} \otimes \mathbf{R}_{B_2} \\ \vdots \\ \mathbf{I}_9 - \mathbf{R}_{A_n} \otimes \mathbf{R}_{B_n} \end{pmatrix} vec(\mathbf{R}_X) = 0 \tag{3.13}$$

et

$$\begin{pmatrix} \mathbf{I}_3 - \mathbf{R}_{A_1} \\ \mathbf{I}_3 - \mathbf{R}_{A_2} \\ \vdots \\ \mathbf{I}_3 - \mathbf{R}_{A_n} \end{pmatrix} \vec{\mathbf{t}}_X = \begin{pmatrix} \vec{\mathbf{t}}_{A_1} - \mathbf{R}_X \vec{\mathbf{t}}_{B_1} \\ \vec{\mathbf{t}}_{A_2} - \mathbf{R}_X \vec{\mathbf{t}}_{B_2} \\ \vdots \\ \vec{\mathbf{t}}_{A_n} - \mathbf{R}_X \vec{\mathbf{t}}_{B_n} \end{pmatrix} \tag{3.14}$$

On peut alors résoudre le problème en 2 phases : on recouvre d'abord la rotation selon la Proposition 2, puis la translation connaissant la rotation.

Cette solution possède plusieurs avantages. Le premier est un avantage par rapport à une résolution obtenue par moindres carrés linéaires du système (3.12). En effet, en présence de bruit, la solution par moindres carrés linéaires va nous donner un vecteur de dimension 12, dont les 9 coefficients associés à la rotation ne satisferont pas les contraintes d'orthogonalité (cf. Annexe A). On pourra certes les appliquer *a posteriori* mais, comme ces contraintes ne sont pas linéaires, nous ne saurons pas quelle correction apporter à la translation. En revanche, la solution en 2 pas garantit l'obtention d'une rotation "propre". De plus, cette rotation est obtenue par l'intermédiaire d'un vecteur directeur d'un noyau de dimension 1 d'une application linéaire qui se calcule très robustement par Décomposition en Valeurs Singulières. Ainsi donc, la translation sera obtenue par moindres carrés linéaires sur un espace linéaire de dimension 3, qui est optimale.

Un autre avantage de la résolution en 2 étapes est qu'elle tient compte du cas particulier où le robot n'effectue que des rotations pures. En effet, elle devient alors équivalente à la solution découplée que nous avons vu dans ce cas particulier.

Enfin, elle permet de détecter le cas particuliers où le robot n'effectue que des translations pures car alors l'équation matricielle (3.13) devient : $\mathbf{0}\ vec(\mathbf{R}_X) = 0$.

3.3 Transformation "moyenne"

En effectuant n tirages aléatoires de k images parmi la séquence disponible, on peut calculer n estimations de la transformation pince-caméra. À partir de ces n estimations, on veut calculer une transformation pince-caméra "moyenne".

La définition du sens de "moyenne" est ardue car on ne peut pas utiliser la moyenne arithmétique de matrices de déplacement rigide pour obtenir une matrice de déplacement rigide. En effet, la somme de $n > 1$ matrices de rotation n'est pas une matrice de rotation. Si l'on utilise les quaternions unitaires pour représenter la rotation, et *a fortiori* les quaternions duaux unitaires pour représenter les déplacements rigides, le problème est le même : la moyenne arithmétique ne conserve pas les contraintes associées à chaque représentation.

On ira voir [Pen96] pour une étude mathématique poussée du problème. Nous préférons cependant nous contenter d'une approche plus simple, mais d'une efficacité comparable. Nous travaillerons donc sur des représentations réduites des rotations : angles Roulis-Tangage-Lacet ou représentation par axe et angle, par exemple.

3.3.1 Angles Roulis-Tangage-Lacet

L'avantage d'une telle représentation est qu'elle est minimale. En revanche, il y a des ambiguïtés qui apparaissent dans l'extraction de ces angles de la matrice de rotation. En dehors de ces ambiguïtés, aisément détectables, on obtient une représentation unique de la rotation. On peut alors calculer une moyenne sur les n triplets d'angles extraits des n estimations de la rotation pince-caméra et reconstruire la rotation "moyenne" à partir de ces angles moyens.

La non-unicité de la mesure de l'angle peut, si l'on n'y prend garde, biaiser le calcul. En effet, lorsque l'on considère la mesure principale des angles (c'est-à-dire prise sur $[0, 2\pi[$) et que l'angle réel que l'on estime est proche de 0, on peut avoir des estimations de cet angle proches de 2π (cf Figure 3.1) qui vont tirer, à tort,

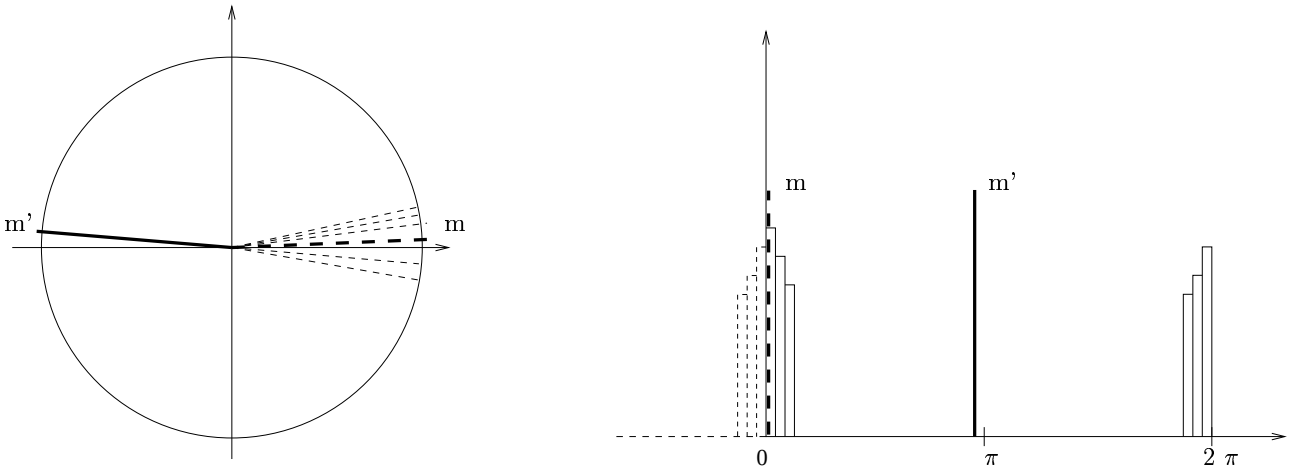


FIG. 3.1 – *La moyenne d'une distribution d'angles centrée sur 0 peut donner 2 résultats selon si l'on se place sur* $]-\pi,\pi]$ *(m) ou sur* $[0,2\pi[$ *(m').*

la moyenne vers π. Un problème similaire interviendra lorsque l'angle réel estimé sera proche de π et que l'on exprimera les angles dans $]-\pi,\pi]$. On en déduit donc la méthode suivante pour calculer la moyenne d'angles.

Soient $\alpha \in [0,2\pi[$, l'angle réel à estimer et α' son expression dans $]-\pi,\pi]$. Soient encore $\alpha_i \in [0,2\pi[$, ses estimations et α'_i, l'expression des α_i dans $]-\pi,\pi]$.

1. Calculer les moyennes $\bar{\alpha} = \frac{1}{n}\sum_{i=1}^{n} \alpha_i$ et $\bar{\alpha'} = \frac{1}{n}\sum_{i=1}^{n} \alpha'_i$
2. Calculer les variances $\sigma_\alpha^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\alpha_i - \bar{\alpha})^2$ et $\sigma_{\alpha'}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\alpha'_i - \bar{\alpha'})^2$
3. Si $\sigma_\alpha > \sigma_{\alpha'}$, alors $\alpha' \approx \bar{\alpha'}$; sinon, $\alpha \approx \bar{\alpha}$
4. Ramener le résultat dans l'intervalle souhaité.

Pour retrouver la transformation "moyenne", on fait alors la moyenne des angles Roulis-Tangage-Lacet pour en déduire la rotation associée, puis on fait la moyenne des translations.

3.3.2 Axe/angle

Cette représentation n'est pas minimale mais presque. En effet, la contrainte qui impose que la norme de l'axe de rotation soit unitaire peut être supprimée en incorporant la valeur de l'angle dans l'axe : en multipliant le vecteur unitaire par l'angle, on obtient un vecteur de dimension 3 qui représente à lui seul la rotation.

De plus, cette contrainte n'est pas aussi gênante que celles intervenant dans les représentations matricielle et par quaternion. En effet, si la somme de vecteurs unitaires n'est pas un vecteur unitaire, il est cependant trivial de trouver le vecteur unitaire associé à cette somme.

Cette représentation est ambiguë car $(\theta, \vec{\mathbf{n}})$ et $(-\theta, -\vec{\mathbf{n}})$ sont deux couples axe/angle qui représentent la même rotation. Pour éviter cette ambiguïté, on choisit de prendre θ dans l'intervalle $[0,2\pi[$. Cependant, lorsque $\theta \approx \pi$, l'ambiguïté n'est pas levée par cette convention car les couples $(\pi, \vec{\mathbf{n}})$ et $(\pi, -\vec{\mathbf{n}})$ représentent la même rotation.

Lorsque l'on détecte que l'on est dans un voisinage de π (en calculant la moyenne des angles sur $[0,2\pi[$), on peut alors lever l'ambiguïté afférente en forçant un coefficient de l'axe de rotation à être positif. À cause de l'ambiguïté autour de π, il est difficile de savoir si l'on doit propager le changement de signe vers l'angle. On va donc choisir entre la moyenne des angles avec et sans propagation du signe selon la plus petite variance. On obtient alors la méthode suivante :

Soient $(\theta_i, \vec{\mathbf{n}}_i)$, les couples axe/angles estimés avec $\theta_i \in [0,2\pi[$

1. Calculer $\bar{\theta}$, l'angle moyen de la rotation à partir des angles θ_i.
2. Calculer σ_θ, l'écart-type des angles θ_i par rapport à $\bar{\theta}$

3. Si $\bar{\theta} \approx \pi$, alors
   (a) Choisir un coefficient c de l'axe de rotation
   (b) Pour tout i
      - $\vec{\mathbf{n}}'_i = sgn(\vec{\mathbf{n}}^c_i)\vec{\mathbf{n}}_i$
      - $\theta'_i = sgn(\vec{\mathbf{n}}^c_i)\theta_i$
   (c) Calculer $\bar{\theta}'$
   (d) Calculer $\sigma_{\theta'}$, l'écart-type des angles θ'_i par rapport à $\bar{\theta}'$
   (e) Si $\sigma_\theta > \sigma_{\theta'}$, alors $\bar{\theta}'$ est le véritable angle moyen.
   (f) Calculer $\bar{\vec{\mathbf{n}}}' = \frac{1}{n}\sum_{i=1}^n \vec{\mathbf{n}}'_i$
   (g) Normaliser $\bar{\vec{\mathbf{n}}}'$

   sinon
   (a) Calculer $\bar{\vec{\mathbf{n}}} = \frac{1}{n}\sum_{i=1}^n \vec{\mathbf{n}}_i$
   (b) Normaliser $\bar{\vec{\mathbf{n}}}$

3.3.3 Calcul robuste

On peut rendre plus robustes les deux méthodes précédentes en remplaçant la moyenne arithmétique par des estimateurs plus robustes. Le premier qui vient à l'esprit est l'opérateur médian, mais l'on peut aussi utiliser un M-estimateur, cf. [Rou97] pour une courte introduction et [Hub81, RL87, HRRS86] pour plus de détails.

Calcul du M-estimateur

On donne ici les deux méthodes de calcul du M-estimateur (présentées dans [Hub81], § 6.7 modifiées selon [HRRS86], § 2.3a).

On note, selon la tradition statistique, T, la moyenne et S, l'écart-type et x_i, les échantillons. Le principe est de remplacer le calcul de la moyenne $T = \frac{1}{n}\sum_{i=1}^n x_i$, très sensible aux mesures aberrantes par une estimation de celle-ci selon une méthode dérivée de l'estimation au maximum de vraisemblance (d'où l'appellation M-estimateur). On va donc minimiser[3]

$$\sum_{i=1}^{n} \rho(x_i - T) \tag{3.15}$$

où ρ est une fonction positive qui décroît lorsque $\|x_i - T\|$ croît, plutôt que $\sum(x_i - T)^2$, critère sous-jacent au calcul de la moyenne arithmétique.

La solution du problème de minimisation (3.15) peut être obtenue comme la solution de l'équation suivante, faisant intervenir la dérivée (ψ) de ρ:

$$\sum_{i=1}^{n} \psi(x_i - T) = 0 \tag{3.16}$$

qui, si l'on pose $w_i(x_i, T) = \frac{\psi(x_i - T)}{x_i - T}$, peut aussi s'écrire

$$\sum_{i=1}^{n} w_i(x_i, T) \cdot (x_i - T) = 0$$

On obtient ainsi une solution formelle, quoiqu'implicite, pour T sous la forme d'une moyenne pondérée:

$$T = \frac{\sum w_i(x_i, T) xi}{\sum w_i(x_i, T)} \tag{3.17}$$

On peut donc obtenir T en résolvant soit (3.16), soit (3.17). Dans le premier cas, on utilisera l'algorithme de Newton-Raphson [AW86] qui va réduire itérativement les résidus $x_i - T$ (méthode des résidus); dans le second, on va raffiner les poids $w_i(x_i, T)$ jusqu'à convergence (méthode des poids).

3. On notera la similarité avec la méthode du maximum de vraisemblance qui minimise $\sum_{i=1}^n -\ln f(x_i, T)$.

Dans les deux cas, il est nécessaire de normaliser les résidus par l'écart-type de l'échantillon (l'échelle, selon la terminologie statistique) car les M-estimateurs ne sont pas invariants par changement d'échelle. Cet écart-type dépend, bien entendu, de la moyenne et peut être mis à jour en même temps que celle-ci [Hub81]. Cependant, nous suivrons les recommandations de [HRRS86] et les estimerons séparément.

En pratique, nous avons utilisé la fonction $\psi(x) = \begin{cases} k\ sgn(x) & si\ |x| > k \\ x & si\ |x| \leq k \end{cases}$ avec $k = 1.345$ [Hub81].

Méthode des résidus

1. Calculer $T_0 = med(x_i)$
2. Calculer $S_0 = 1.483\ med(|x_i - T_0|)$
3. Itérer jusqu'à convergence:
$$T_{k+1} = T_k + \frac{\sum \psi\left[(x_i - T_k)/S_0\right] S_0}{\sum \psi'\left[(x_i - T_k)/S_0\right]}$$
4. $T = T_{k+1}$
5. $S = 1.483\ med(|x_i - T|)$

Méthode des poids

1. Calculer $T_0 = med(x_i)$
2. Calculer $S_0 = 1.483\ med(|x_i - T_0|)$
3. Itérer
$$T_{k+1} = T_k + \frac{\sum w_i^{(k)} x_i}{\sum w_i^{(k)}} \quad \text{avec} \quad w_i^{(k)} = \frac{\psi\left[(x_i - T_k)/S_0\right]}{(x_i - T_k)/S_0}$$
jusqu'à convergence.
4. $T = T_{k+1}$
5. $S = 1.483\ med(|x_i - T|)$

3.4 Mesure d'erreur

Les erreurs ont été obtenues en faisant la moyenne des carrés des erreurs relatives (RMS).

L'erreur relative en rotation a été calculée par: $\|\hat{\mathbf{q}} - \mathbf{q}\|$ où $\mathbf{q}$ est le quaternion associé à la rotation pince-caméra obtenue par étalonnage selon la méthode utilisée et $\hat{\mathbf{q}}$ est le quaternion associé à la rotation pince-caméra de référence. On notera que si les quaternions $\hat{\mathbf{q}}$ et $\mathbf{q}$ sont unitaires, alors

$$\begin{aligned} \|\hat{\mathbf{q}} - \mathbf{q}\| &= \|\mathbf{1} - \hat{\mathbf{q}}^{-1}\mathbf{q}\| \\ &= (\mathbf{1} - \hat{\mathbf{q}}^{-1}\mathbf{q})\overline{(\mathbf{1} - \hat{\mathbf{q}}^{-1}\mathbf{q})} \\ &= (\mathbf{1} - \hat{\mathbf{q}}^{-1}\mathbf{q})(\mathbf{1} - \overline{\hat{\mathbf{q}}^{-1}\mathbf{q}}) \\ &= \mathbf{1} + (\hat{\mathbf{q}}^{-1}\mathbf{q})(\overline{\hat{\mathbf{q}}^{-1}\mathbf{q}}) \\ &\quad -(\hat{\mathbf{q}}^{-1}\mathbf{q} + \overline{\hat{\mathbf{q}}^{-1}\mathbf{q}}) \\ &= 2 - 2\cos(\frac{\alpha}{2}) \end{aligned}$$

où α est l'angle de la rotation résiduelle $\hat{\mathbf{q}}^{-1}\mathbf{q}$. Cette fonction est strictement croissante et vaut 0 au voisinage de $\alpha = 0$. De plus, elle permet d'éviter l'indétermination apparaîssant pour $\alpha = \pi$ lors de l'utilisation d'une distance géodésique (voir [SLBE91], p.35). Enfin, l'étude de sa fonction réciproque ($\epsilon \mapsto 2\arccos(1 - \frac{\epsilon}{2}) = 2\sqrt{\epsilon} + \mathcal{O}(\epsilon^{3/2})$) permet de rendre un peu plus tangible le résultat d'une telle mesure d'erreur (voir Tableau 3.2).

L'erreur relative en translation a été obtenue par: $\|\vec{\mathbf{t}} - \vec{\mathbf{t}}^*\|/\|\vec{\mathbf{t}}^*\|$ où $\vec{\mathbf{t}}$ et $\vec{\mathbf{t}}^*$ représentent respectivement les translations pince-caméra estimée et réelle.

erreur	10^{-5}	10^{-4}	10^{-3}	10^{-2}
angle (deg)	0.3624	1.1459	3.6239	11.4639
angle (deg)	5	10	15	20
erreur	1.9036 10^{-3}	7.6106 10−3	1.7110 10−2	3.0384 10−2

TAB. 3.2 – *Quelques couples (erreur en rotation, angle résiduel).*

3.5 Simulations

Pour chaque série de simulations, la démarche a été similaire: pour chaque valeur du paramètre testé, nous avons tiré aléatoirement une transformation pince-caméra (par tirage aléatoire des angles Roulis-Tangage-Lacet de la matrice de rotation et des coefficients de translation selon des lois normales), une séquence de mouvements conjugués du robot et de la caméra, puis nous avons bruité les déplacements de la caméra, effectué l'étalonnage avec 3 méthodes différentes et comparé leur résultat avec la transformation initiale.

Les méthodes utilisées sont la méthode axe/angle [TL89], la méthode des quaternions duaux unitaires [DBC96], la méthode d'optimisation non-linéaire [HD95] et la méthode linéaire présentée dans ce rapport.

3.5.1 Influence du bruit

Description Dans cette simulation, nous avons bruité les mouvements de la caméra $(\mathbf{R}_{A_i}, \vec{\mathbf{t}}_{A_i})$ et obtenu les mouvements bruités $(\tilde{\mathbf{R}_{A_i}}, \tilde{\vec{\mathbf{t}}_{A_i}})$.

Les translations $\tilde{\vec{\mathbf{t}}_{A_i}}$ ont été déterminées par:

$$\tilde{\vec{\mathbf{t}}_{A_i}} = \vec{\mathbf{t}}_{A_i} + bruit * \|\vec{\mathbf{t}}_{A_i}\| * \vec{\mathbf{n}}$$

où *bruit* est un facteur variant de 0 à 0.1 et $\vec{\mathbf{n}}$ est une réalisation d'un vecteur aléatoire de loi normale centrée réduite.

Quant aux rotations, nous les avons bruitées en ajoutant une erreur relative sur leurs angles Roulis-Tangage-Lacet:

$$\tilde{\alpha} = \alpha(1 + bruit * r)$$

où α est un angle de la rotation exacte, $\tilde{\alpha}$ l'angle correspondant de la rotation bruitée, *bruit* est le même facteur que pour la translation. Enfin, r est une réalisation d'une variable aléatoire de loi normale centrée réduite.

Étalonnage avec 2 mouvements quelconques Nous avons effectué 100 choix différents de transformations pince-caméra et de couples de mouvements. Pour chaque choix, nous avons bruité les mouvements de la caméra avec des *bruit* variant de 0 à 0.2. La Figure 3.2 présente les erreurs d'étalonnage obtenues selon les 4 méthodes retenues avec 2 mouvements et la figure 3.3 les temps de calculs moyens respectifs.

Avec seulement 2 mouvements, les méthodes linéaires sont plus précises en rotation mais moins robustes en translation que les méthodes plus complexes.

Étalonnage avec 2 mouvements de faible amplitude Cette simulation est en tous points semblable avec la précédente, exception faite de l'amplitude des mouvements d'étalonnage que nous avons limitée en rotation et translation. Les erreurs d'étalonnage sont présentées en Figure 3.4.

Les résultats obtenus sont sensiblement meilleurs que pour les méthodes complexes (quaternions duaux et minimisation non-linéaire) et légèrement meilleurs que pour l'autre méthode linéaire: méthode axe/angle.

3.5.2 Influence du nombre de mouvements

Description Nous avons ici fait varier le nombre de mouvements de 2 à 15 en gardant un niveau de bruit constant ($bruit = 0.01$, selon la définition du paragraphe précédent). Nous nous sommes concentrés ici sur des mouvements de faible amplitude. Les mouvements de caméra ont été bruités de la même manière que pour la simulation précédente et 100 tirages aléatoires ont été fait pour chaque nombre de mouvements.

La figure 3.5 présente les erreurs d'étalonnage obtenues selon les 3 méthodes retenues et la figure 3.6, les temps de calculs moyens respectifs.

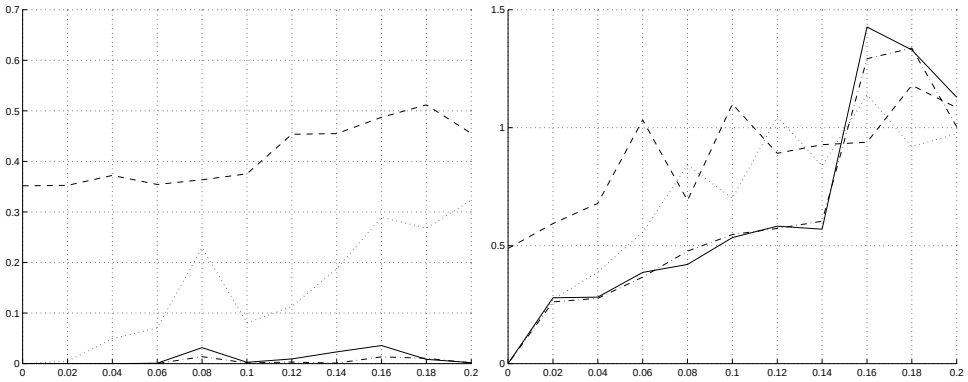


FIG. 3.2 – *Erreurs en rotation (gauche) et en translation (droite) pour l'étalonnage en fonction du niveau de bruit dans le cas de 2 mouvements du robot selon les différentes méthodes : axe/angle (—), quaternions (· · ·), non-linéaire (- -), linéaire (– ·).*

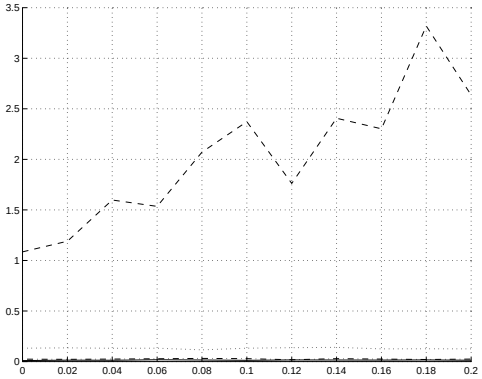


FIG. 3.3 – *Temps moyen de calcul (en secondes CPU) associés aux résultats de la Figure 3.2.*

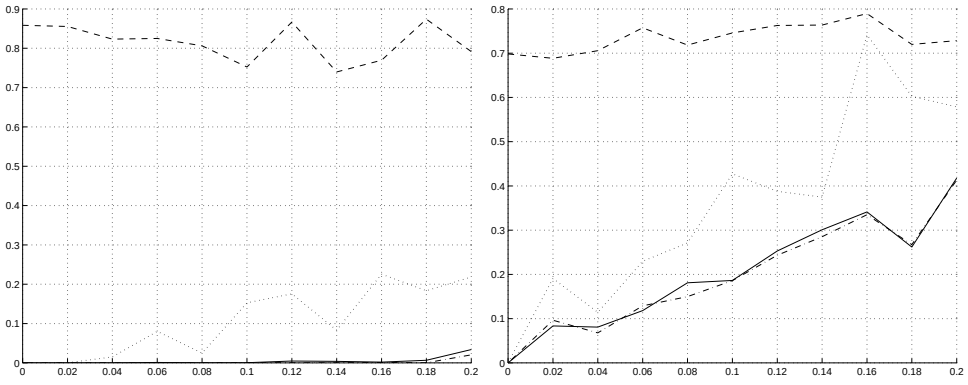


FIG. 3.4 – *Erreurs en rotation (gauche) et en translation (droite) pour l'étalonnage en fonction du niveau de bruit dans le cas de 2 mouvements* **de faible amplitude** *du robot selon les différentes méthodes : axe/angle (—), quaternions (· · ·), non-linéaire (- -), linéaire (– ·).*

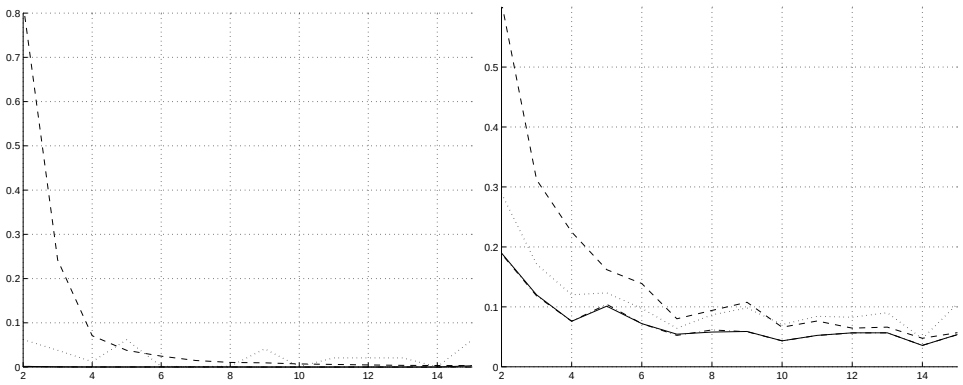


FIG. 3.5 – *Erreurs en rotation (gauche) et en translation (droite) pour l'étalonnage en fonction du nombre de mouvements du robot (rotations pures de faible amplitude) selon les différentes méthodes : axe/angle(—), quaternions (· · ·), non-linéaire (- -), linéaire (– ·).*

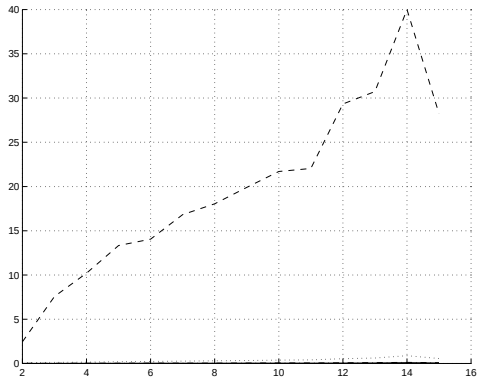


FIG. 3.6 – *Temps moyen de calcul (en secondes CPU) pour l'étalonnage en fonction du nombre de mouvements du robot (rotations pures de faible amplitude) selon les différentes méthodes : axe/angle (—), quaternions (· · ·), non-linéaire (- -), linéaire (– ·).*

Les résultats sont semblables à ce que l'on a vu pour l'influence du bruit : une erreur bien plus faible en rotation, une erreur très faible (et inférieure à celles des autres méthodes) en translation et un temps de calcul constant et réduit.

3.6 Résultats expérimentaux

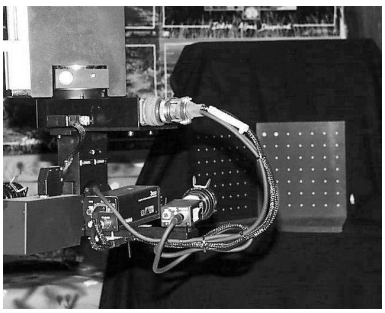


FIG. 3.7 – *Dispositif expérimental : une caméra montée sur un robot cartésien à 5 ddl regarde une mire*

Nous avons utilisé un robot cartésien à 5 degrés de liberté portant une caméra (Figure 3.7). À l'aide d'une connaissance *a priori* mais néanmoins imprécise de la transformation pince-caméra, nous avons déplacé le robot de manière à ce que la caméra conserve la mire dans son champ de vision. En chacune des positions du robot, nous avons enregistré la valeur de ses coordonnées articulaires et pris une image de la mire.

Munis de ces précieuses informations, nous avons fastidieusement procédé à l'étalonnage pince-caméra. Pour cela, nous avons transformé les coordonnées articulaires du robot en positions dans l'espace puis en déplacements. Puis, nous avons détecté les points de la mire dans l'image et les avons mis en correspondance avec le modèle géométrique que nous possédons. Nous avons alors déterminé les paramètres de formation de l'image, parmi lesquels on trouve la pose de la caméra par rapport à la mire. De ces poses, nous avons enfin pu déterminer les déplacements de la caméra. Ce fut alors un jeu d'enfant de déterminer la transformation pince-caméra.

Afin de mesurer l'erreur d'étalonnage et comme nous ne possédons pas de valeur de référence, nous avons calculé, pour chaque couple de déplacement $(\mathbf{A}, \mathbf{B})$, l'écart entre les transformations rigides $\mathbf{AX}$ et $\mathbf{XB}$ selon les mesures définies au paragraphe 3.5.

3.6.1 Trajectoire 1

Dans cette expérience, nous avons utilisé une trajectoire de 33 positions les plus éloignées possibles (au sens de [TL89]). La Figure 3.8 montre la moyenne des erreurs en rotation (gauche) et en translation (droite) obtenues en étalonnant selon **1)** la méthode axe/angle [TL89], **2)** la méthode des quaternions duaux [DBC96], **3)** la minimisation non linéaire [HD95] et **4)** la méthode linéaire présentée ici.

Nous avons utilisé cette séquence d'images pour calculer de façon robuste (§ 3.3) une transformation pince-caméra qui nous servira d'étalon pour les expériences suivantes.

3.6.2 Trajectoire 2

Dans cette deuxième expérience, nous avons utilisé une trajectoire composée de 2 mouvements initiaux de grande amplitude et d'axes de rotation différents puis de 60 mouvements de faible amplitude convergeant exponentiellement vers 0 selon les coordonnées articulaires du robot. La Figure 3.9 montre les erreurs en rotation et translation (mêmes conventions qu'en Figure 3.8). Quant à la Figure 3.10, elle montre les erreurs lorsque l'on n'utilise que les mouvements de faible amplitude (trajectoire partielle **expo**). Enfin, la Figure 3.11 montre les erreurs lorsque l'on n'utilise que les 2 mouvements initiaux (trajectoire partielle **init**). Les erreurs par rapport à la transformation pince-caméra moyenne obtenue avec les images de la séquence précédente ont été collectées dans le Tableau 3.3.

Les deux types d'erreurs donnent lieu à une conclusion identique si l'on compare les deux trajectoires partielles : la méthode non-linéaire n'est pas efficace avec peu de mouvements (même grands) mais l'est avec de nombreux mouvements (mêmes petits); la méthode linéaire que nous proposons est efficace dans les deux cas et les deux autres méthodes perdent leur efficacité lorsque les mouvements sont petits.

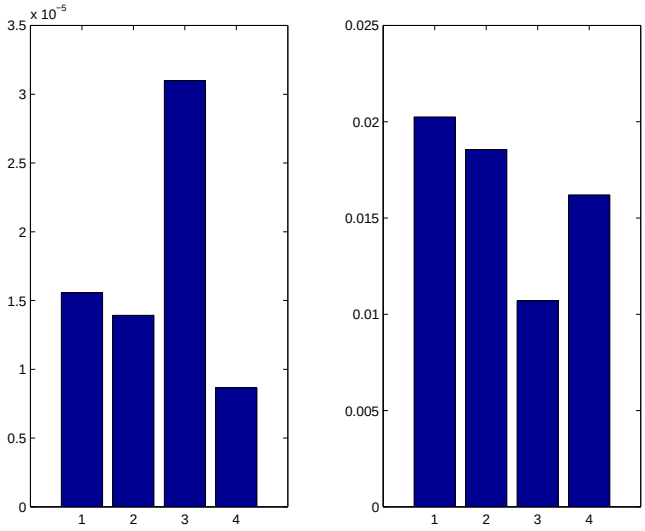


FIG. 3.8 – *Moyenne des erreurs en rotation (gauche) et en translation (droite) pour une trajectoire de 33 positions. Méthodes utilisées :* **1)** *axe/angle [TL89],* **2)** *quaternions duaux [DBC96],* **3)** *non linéaire [HD95],* **4)** *linéaire.*

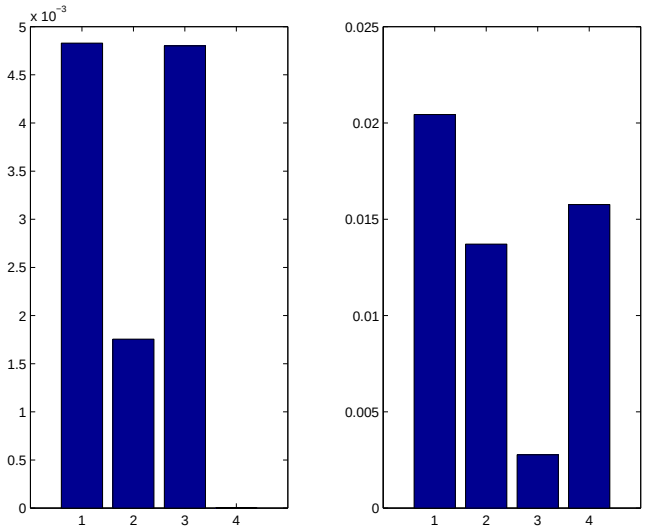


FIG. 3.9 – *Moyenne des erreurs en rotation (gauche) et en translation (droite) pour une trajectoire composée de 2 grands mouvements puis 60 petits et exponentiellement décroissants. Mêmes conventions qu'en Figure 3.8.* **NB:** *l'erreur en rotation selon* **4)** *n'a pas été oubliée !*

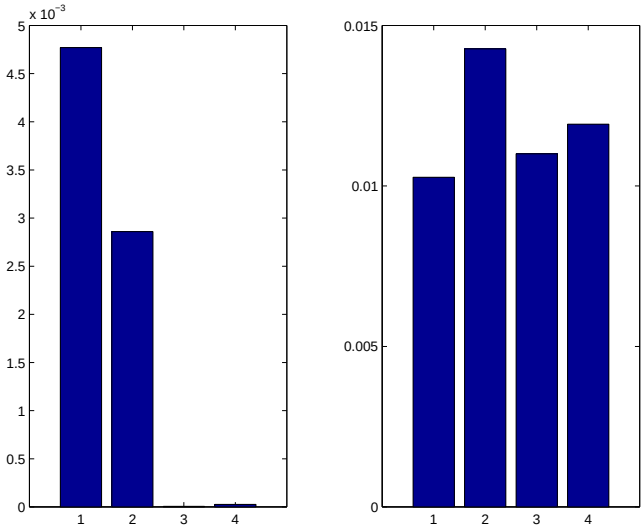


FIG. 3.10 – *Moyenne des erreurs en rotation (gauche) et en translation (droite) pour une trajectoire de 60 mouvements, petits et exponentiellement décroissants. Mêmes conventions qu'en Figure 3.8.*

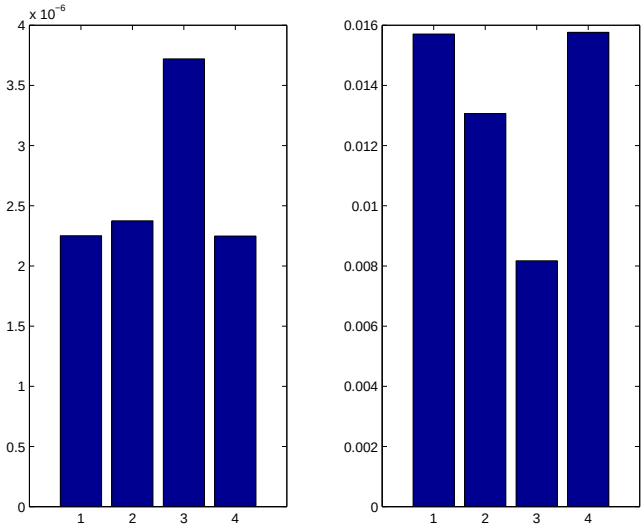


FIG. 3.11 – *Moyenne des erreurs en rotation (gauche) et en translation (droite) avec 2 mouvements de grande amplitude. Mêmes conventions qu'en Figure 3.8.* **NB:** *L'erreur en rotation de* **3)** *a été divisée par 1000 pour des raisons graphiques.*

	Figure 3.9 : complète		Figure 3.10 : **expo**		Figure 3.11 : **init**	
	rot.	trans.	rot.	trans.	rot.	trans.
1	1.75	1.06	1.75	1.01	$1.48\ 10^{-5}$	0.56
2	1.60	1.01	1.42	1.00	$4.94\ 10^{-5}$	0.41
3	1.81	1.00	$4.91\ 10^{-4}$	1.01	1.39	1.01
4	$1.00.10^{-5}$	0.57	$6.23\ 10^{-3}$	0.87	$1.04\ 10^{-5}$	0.56

TAB. 3.3 – *Erreurs par rapport à la transformation moyenne*

Enfin, ce manque de précision sur une partie de la trajectoire pour les méthodes classiques se répercute sur les résultats obtenus avec la trajectoire complète.

3.7 Conclusion

On tirera plusieurs conclusions de ce chapitre. La première est que la méthode linéaire que nous y avons présentée permet une analyse algébrique très simple du problème d'étalonnage pince-caméra.

La seconde conclusion est qu'il faut se méfier des simulations car nous obtenons ici des résultats comparatifs inverses de ce que l'on peut trouver dans la littérature. L'explication d'un tel phénomène nous semble résider dans le nombre élevé de paramètres dont dépend l'étalonnage pince-caméra : manières de bruiter, choix des mouvements et choix de la transformation pince-caméra (quelle relation y'a-t-il entre ces choix ?), mesures d'erreurs, etc.

Enfin, la dernière conclusion, est que simulation ou expérimentation, la méthode linéaire donne les meilleurs résultats en rotation. En ce qui concerne la translation, ce n'est pas pire avec notre méthode qu'avec les autres.

Chapitre 4

Méthode linéaire avec reconstruction

4.1 Introduction

Dans ce chapitre, nous allons modifier la méthode linéaire du chapitre précédent de manière à pouvoir effectuer l'étalonnage pince-caméra en se passant d'une mire de calibration. L'intérêt d'une telle méthode est qu'elle autorise une plus grande autonomie du robot. Pour que cette autonomie soit totale, il faut pouvoir étalonner la transformation pince-caméra sans connaissance *a priori* sur sa valeur. Dans ce cadre, on ne peut donc pas effectuer de grands mouvements car on risquerait de n'avoir jamais la même scène dans le champ de la caméra, ce qui empêcherait tout calcul de reconstruction tridimensionnelle. Il faut donc que la méthode soit robuste aux petits mouvements.

En l'absence de mire, nous ne pouvons calculer les déplacements de caméra que par le biais d'une reconstruction tridimensionnelle et à un facteur d'échelle près. Dans ces conditions, l'équation (3.3) devient :

$$(\mathbf{I}_3 - \mathbf{R}_A)\vec{\mathbf{t}}_X = \lambda\widetilde{\vec{\mathbf{t}}_A} - \mathbf{R}_X\vec{\mathbf{t}}_B \tag{4.1}$$

où λ est le facteur d'échelle inconnu, $\widetilde{\vec{\mathbf{t}}_A}$ est la translation reconstruite de la caméra et $\vec{\mathbf{t}}_A = \lambda\widetilde{\vec{\mathbf{t}}_A}$ est la translation réelle effectuée par la caméra.

Il est bien évident que l'utilisation d'une méthode classique d'étalonnage pince-caméra donnerait lieu à une solution biaisée pour la translation pince-caméra $\vec{\mathbf{t}}_X$. En effet, l'équation précédente se réécrit :

$$\underbrace{(\mathbf{I}_3 - \mathbf{R}_A)\vec{\mathbf{t}}_X = \widetilde{\vec{\mathbf{t}}_A} - \mathbf{R}_X\vec{\mathbf{t}}_B}_{\text{partie classique}} + \underbrace{(\lambda - 1)\widetilde{\vec{\mathbf{t}}_A}}_{biais}$$

Il est donc nécessaire de modifier les méthodes existantes afin de tenir compte de ce facteur d'échelle inconnu.

Ainsi, notre méthode linéaire en 2 étapes peut être légèrement modifiée afin de pouvoir établir simultanément la transformation pince-caméra et le facteur d'échelle inconnu de la reconstruction.

4.2 Nouvelle formulation

Repartons du système (3.6) en remplaçant $\vec{\mathbf{t}}_A$ par $\lambda\widetilde{\vec{\mathbf{t}}_A}$ pour obtenir :

$$\begin{pmatrix} \mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B & \mathbf{0}_{9\times 3} \\ \mathbf{I}_3 \otimes (\vec{\mathbf{t}}_B^T) & \mathbf{I}_3 - \mathbf{R}_A \end{pmatrix} \begin{pmatrix} vec(\mathbf{R}_X) \\ \vec{\mathbf{t}}_X \end{pmatrix} = \begin{pmatrix} \vec{\mathbf{0}}_{9\times 1} \\ \lambda\widetilde{\vec{\mathbf{t}}_A} \end{pmatrix}$$

Comme le second membre dépend du facteur inconnu λ, nous insérons ce dernier dans le vecteur des inconnues pour obtenir :

$$\begin{pmatrix} \mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B & \mathbf{0}_{9\times 3} & \mathbf{0}_{3\times 1} \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_B^T & \mathbf{I}_3 - \mathbf{R}_A & -\widetilde{\vec{\mathbf{t}}_A} \end{pmatrix} \begin{pmatrix} vec(\mathbf{R}_X) \\ \vec{\mathbf{t}}_X \\ \lambda \end{pmatrix} = \vec{\mathbf{0}}_{12\times 1} \tag{4.2}$$

Nous pouvons donc encore résoudre ce système en 2 étapes : d'abord, la rotation (étape inchangée) puis la translation.

Deux cas se présentent néanmoins dans cette deuxième étape : lorsque nous considérons plusieurs mouvements, doit-on déterminer un ou plusieurs facteurs d'échelle inconnus ? En pratique, les méthodes de reconstruction ne fournissent qu'un seul facteur d'échelle par séquence d'images utilisée. Cependant, on peut envisager des cas "tordus" où l'on a reconstruit une partie des mouvements de caméra d'un côté et une partie de l'autre. Par exemple, une série de déplacements a été reconstruite puis un nouveau déplacement a été effectué par la caméra, ensuite, une nouvelle reconstruction a eu lieu et enfin seul le dernier mouvement a été pris en compte sans pour autant remplacer le résultat de la reconstruction initiale des premiers mouvements par le résultat de la deuxième. Pour pouvoir traiter aussi de tels cas, nous allons étudier les deux cas.

Facteur d'échelle commun — Nous ne nous intéressons ici qu'à retrouver la translation et le facteur d'échelle connaissant la rotation. Nous devons donc résoudre :

$$\begin{pmatrix} \mathbf{I}_3 - \mathbf{R}_{A_1} & -\widetilde{\vec{\mathbf{t}}_{A_1}} \\ \vdots & \\ \mathbf{I}_3 - \mathbf{R}_{A_n} & -\widetilde{\vec{\mathbf{t}}_{A_n}} \end{pmatrix} \begin{pmatrix} \vec{\mathbf{t}}_X \\ \lambda \end{pmatrix} = \begin{pmatrix} -\mathbf{R}_X \vec{\mathbf{t}}_{B_1} \\ \vdots \\ -\mathbf{R}_X \vec{\mathbf{t}}_{B_n} \end{pmatrix} \tag{4.3}$$

Ce système est de rang plein avec au moins 2 rotations et une translation indépendantes et peut alors être résolu par moindres carrés linéaires sur un espace linéaire désormais de dimension 4. Cette condition étant compatible avec les conditions d'existence d'une solution pour la rotation, nous pouvons donc effectuer pleinement l'étalonnage pince-caméra et déterminer la taille de la scène observée.

Facteurs d'échelle différents — Supposons dans ce cas que chaque mouvement est reconstruit avec un facteur d'échelle qui lui est propre. Alors, il nous faut résoudre :

$$\begin{pmatrix} \mathbf{I}_3 - \mathbf{R}_{A_1} & -\widetilde{\vec{\mathbf{t}}_{A_1}} & \vec{\mathbf{0}}_{3\times 1} & \dots & \vec{\mathbf{0}}_{3\times 1} \\ \mathbf{I}_3 - \mathbf{R}_{A_2} & \vec{\mathbf{0}}_{3\times 1} & -\widetilde{\vec{\mathbf{t}}_{A_2}} & \dots & \vec{\mathbf{0}}_{3\times 1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{I}_3 - \mathbf{R}_{A_n} & \vec{\mathbf{0}}_{3\times 1} & \vec{\mathbf{0}}_{3\times 1} & \dots & -\widetilde{\vec{\mathbf{t}}_{A_n}} \end{pmatrix} \begin{pmatrix} \vec{\mathbf{t}}_X \\ \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{pmatrix} = \begin{pmatrix} -\mathbf{R}_X \vec{\mathbf{t}}_{B_1} \\ -\mathbf{R}_X \vec{\mathbf{t}}_{B_2} \\ \vdots \\ -\mathbf{R}_X \vec{\mathbf{t}}_{B_n} \end{pmatrix} \tag{4.4}$$

De nouveau, il appert que 2 rotations et une translation indépendantes sont requises pour l'obtention d'un système de rang plein.

Cas des rotations pures du robot — Si le robot n'effectue que des rotations pures, alors le second membre des équations (4.3) et (4.4) s'annule. Quelles en sont les conséquences ? Revenons à l'équation (4.1) et remarquons qu'elle devient alors :

$$(\mathbf{I}_3 - \mathbf{R}_A)\vec{\mathbf{t}}_X - \lambda\widetilde{\vec{\mathbf{t}}_A} = 0$$

ce qui signifie que les coefficients de $\vec{\mathbf{t}}_X$ et λ sont liés. Les systèmes (4.3) et (4.4) avec des seconds membres nuls ne sont donc pas de rang plein.

Interprétation Contrairement au cas où l'on utilise une mire, il n'est plus possible ici d'étalonner entièrement la relation pince-caméra uniquement avec des rotations pures du robot. Cela s'explique par le fait que les seules informations métriques dont nous puissions disposer lorsque ni la scène, ni la transformations pince-caméra ne sont connues proviennent exclusivement des déplacements du robot. Or, parmi ceux-ci, les rotations pures sont dépourvues de la notion de longueur. Il faut donc avoir recours aux translations pour pouvoir obtenir le ou les facteurs d'échelle inconnus.

4.3 Résultats expérimentaux

Afin de comparer les résultats de la méthode d'auto-étalonnage pince-caméra avec la méthode linéaire présentée dans le chapitre précédent, nous les avons appliquées, dans un premier temps, aux mêmes images (de la mire, donc). La différence d'utilisation de ces images réside dans l'utilisation ou non du modèle géométrique de la mire pour le calcul des déplacements de la caméra. En revanche, nous avons conservé, pour la méthode d'auto-étalonnage, les points issus de la détection des cibles de la mire (utilisant le modèle de la mire) utilisée dans l'étalonnage classique.

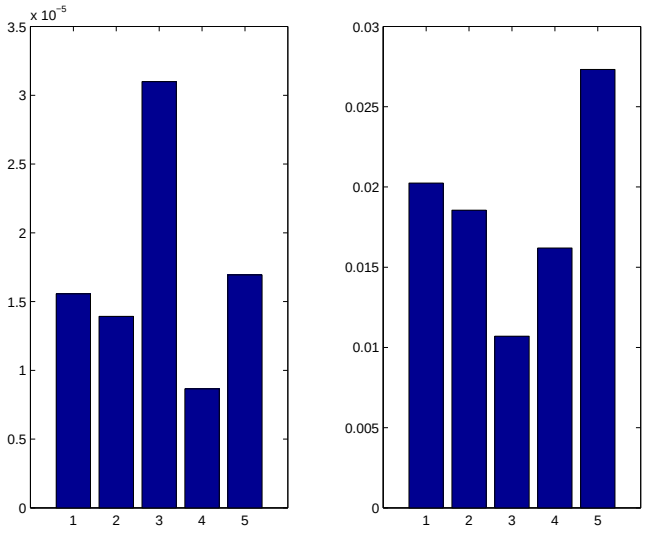


FIG. 4.1 – *Moyenne des erreurs en rotation (gauche) et en translation (droite) pour une trajectoire de 33 positions. Méthodes utilisées :* **1)** *axe/angle [TL89],* **2)** *quaternions duaux [DBC96],* **3)** *non linéaire [HD95],* **4)** *linéaire,* **5)** *auto-étalonnage.*

	4 positions		9 positions	
	rot.	trans.	rot.	trans
1	$2.7.10^{-5}$	0.18	0.62	0.99
2	$2.8.10^{-5}$	0.22	$4.7.10^{-4}$	0.41
3	1.82	1.01	0.02	0.44
4	$2.3.10^{-5}$	0.17	$2.3.10^{-3}$	0.54
5	$2.8.10^{-4}$	0.20	0.02	0.97

TAB. 4.1 – *Erreurs par rapport à la transformation moyenne.*

La Figure 4.1 reprend la Figure 3.8 et y ajoute le résultat obtenu par la méthode d'auto-étalonnage. La méthode d'auto-étalonnage est moins précise du fait de la non utilisation du modèle géométrique de la mire. Cependant, les résultats qu'elle fournit restent honorables puisque l'erreur relative à la transformation moyenne calculée est de 2.10^{-4} en rotation (correspondant à un angle résiduel d'environ un demi degré) et de 32% en translation.

Dans un second temps, nous nous sommes affranchis complètement de la mire et avons travaillé sur des images réelles. Pour cela, nous avons fait suivre au robot une trajectoire au cours de laquelle nous avons pris des images de la mire, grâce auxquelles nous avons procédé à l'étalonnage par les méthodes classiques et linéaire. Puis, nous avons fait reparcourir cette même trajectoire au robot (à sa répétabilité près) pour prendre des images d'une scène inconnue sur lesquelles nous avons utilisé la méthode d'auto-étalonnage.

Nous avons utilisé deux trajectoires, une de 4 positions éloignées et une autre de 9 positions proches (voir Figures 4.2 et 4.3). On notera ici encore que la méthode linéaire d'étalonnage avec mire est toujours aussi efficace (Figure 4.4 et Tableau 4.1). En ce qui concerne la méthode d'auto-étalonnage, les résultats sont quelque peu moins bons mais restent acceptables comparés aux méthodes avec mire.

4.4 Conclusion

Nous avons proposé dans ce chapitre une méthode d'auto-étalonnage pince-caméra, qui, à l'inverse des méthodes d'étalonnage pince-caméra existantes, fonctionne dans des scènes quelconques.

Les résultats expérimentaux montrent une faible dégradation des résultats par rapport aux méthodes existantes au profit d'une plus grande autonomie. Cette autonomie est d'autant plus grande que la méthode est robuste aux petits mouvements qui garantissent la présence d'une scène commune dans toutes les images et donc la résolution du problème.

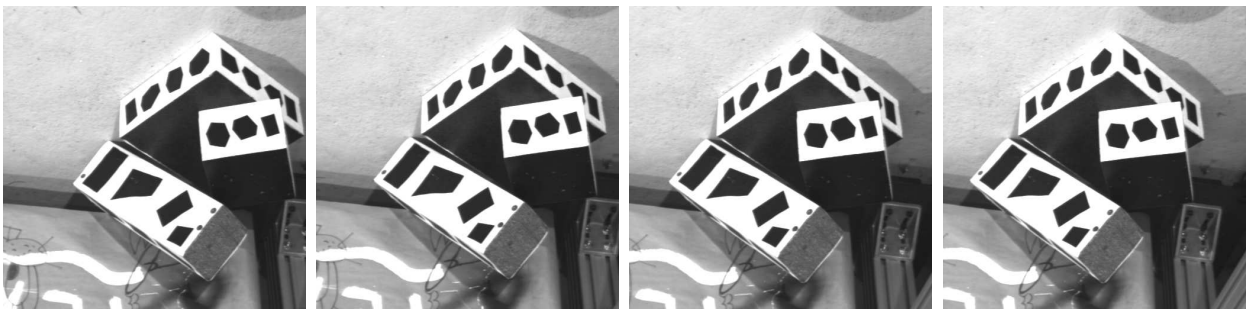


FIG. 4.2 – *La séquence de 4 images utilisées pour l'auto-étalonnage*

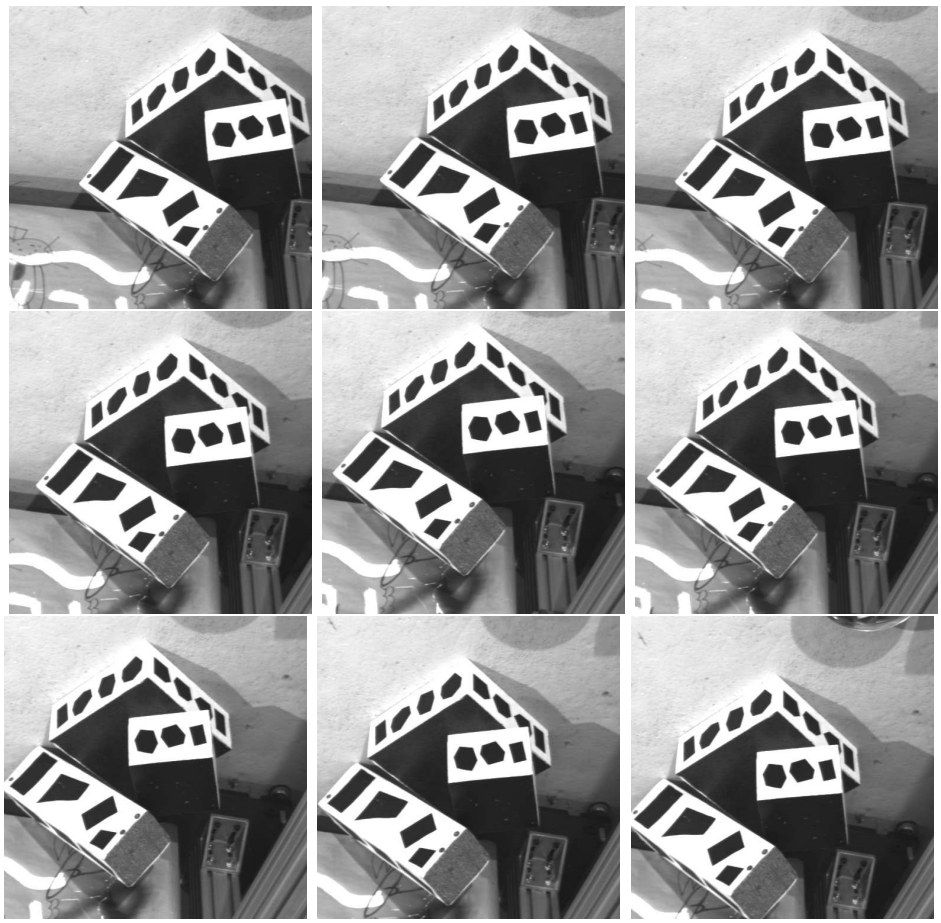


FIG. 4.3 – *La séquence de 9 images utilisées pour l'auto-étalonnage*

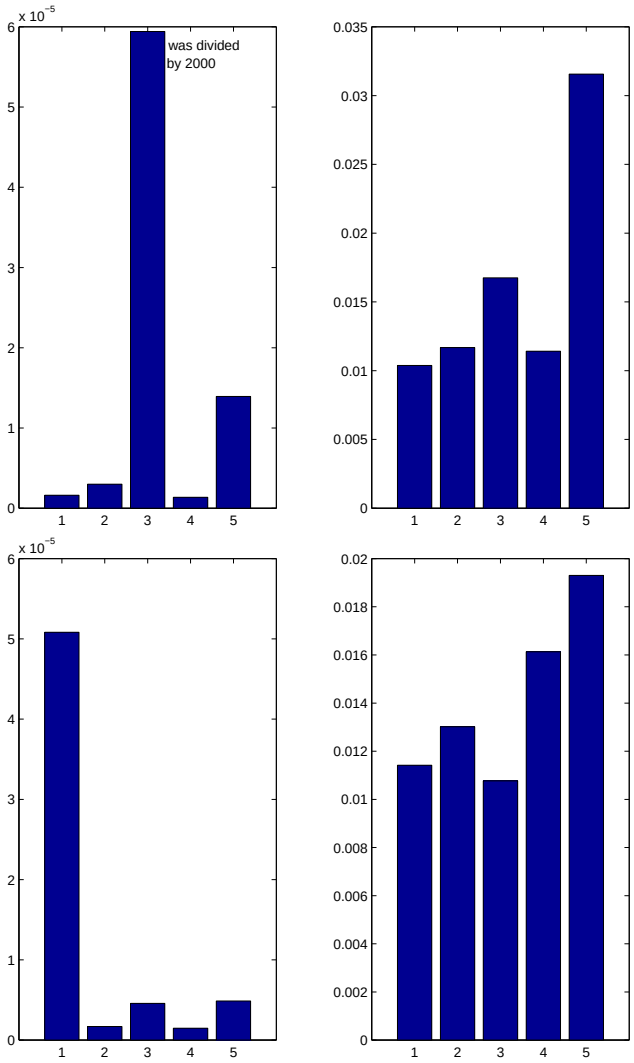


FIG. 4.4 – *Moyenne des erreurs en rotation (gauche) et en translation (droite) pour une trajectoire de 4 positions éloignées (haut) et une autre de 9 positions rapprochées (bas). Méthodes utilisées :* **1)** *axe/angle [TL89],* **2)** *quaternions duaux [DBC96],* **3)** *non linéaire [HD95],* **4)** *linéaire,* **5)** *auto-étalonnage.*

Chapitre 5

Conclusion

Ce rapport présente deux nouvelles méthodes ayant trait à l'étalonnage pince-caméra. Ces deux méthodes sont basées sur une formulation purement linéaire du problème, ce qui permet une analyse algébrique simple et élégante des mouvements nécessaires à sa résolution.

La première méthode présentée est une méthode "classique", au sens qu'elle requiert, en pratique, une mire de calibration. Sa formulation lui permet d'être plus robuste aux petits mouvements que les méthodes existantes sans pour autant perdre en efficacité pour des mouvements de plus grande amplitude.

La deuxième méthode est, en revanche, une méthode d'auto-étalonnage car elle n'a pas besoin, en pratique, d'intervention humaine (par le biais notamment de la mire). Cette méthode est elle aussi précise dans le cas de petits mouvements, permettant ainsi l'(auto-)étalonnage pince-caméra en l'absence de toute connaissance de la transformation à déterminer. Certes, ses résultats sont quelque peu moins bons que les méthodes classiques, mais l'on doit pouvoir les améliorer lors du déroulement de la mission du robot grâce à des méthodes d'auto-étalonnage pince-caméra en ligne. Une telle méthode peut être aisément envisagée à partir de la méthode itérative présentée dans [And97].

Bibliographie

[And97] N. Andreff. Towards the embedding of on-line hand-eye calibration into visual servoing. In *IEEE/RSJ/INRIA Workshop On New Trends in Image-based Robot Servoing*, pages 64–70, Grenoble, September 1997.

[AW86] K. Arbenz and A. Wohlhauser. Analyse numérique. In *Méthodes mathématiques pour l'ingénieur*. Presses polytechniques et universitaires romandes, 1986.

[Bel60] R. Bellman. *Introduction to matrix analysis*. McGraw-Hill, 1960.

[Bre78] J. W. Brewer. Kronecker products and matrix calculus in system theory. *IEEE Trans. on Circuits and Systems*, CAS-25(9):772–781, 1978.

[BS72] R. H. Bartels and G. H. Stewart. Solution of the equation AX+XB = C. *Communications of the ACM*, 15(9):820–826, 1972.

[Cha90] F. Chaumette. *La relation vision-commande : théorie et application à des tâches robotques* . Thèse, Université de Rennes I, 1990.

[Che91] H. H. Chen. A screw motion approach to uniqueness analysis of head-eye geometry. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 145–151, 1991.

[DBC96] K. Daniilidis and E. Bayro-Corrochano. The dual quaternion approach to hand-eye calibration. In *Proc. IAPR International Conference on Pattern Recognition*, pages 318–322, 1996.

[DSH95] A. S. Deif, N. P. Seif, and S. A. Hussein. Sylvester's equation : accuracy and computational stability. *Journal of Computational and Applied Mathematics*, 61:1–11, 1995.

[Dua96] G.-R. Duan. On the solution to the Sylvester matrix equation AV+BW = EVF. *IEEE Trans. on Automatic Control*, 41(4):612–314, 1996.

[ECR92] B. Espiau, F. Chaumette, and P. Rives. A New Approach To Visual Servoing in Robotics. *IEEE Trans. on Robotics and Automation*, 8(3), June 1992.

[GNL79] G. Golub, C. Nash, and C. Van Loan. A Hessenberg-Schur method for the problem AX+XB = C. *IEEE Trans. on Automatic Control*, 24:903–913, 1979.

[HD95] R. Horaud and F. Dornaika. Hand-eye calibration. *International Journal of Robotics Research*, 14(3):195–210, June 1995.

[HR92] D. Y. Hu and L. Reichel. Krylov-subspace methods for the Sylvester equation. *Linear Algebra and its Applications*, 172:283–313, 1992.

[HRRS86] F.R. Hampel, E.M. Ronchetti, P.J. Rousseeuw, and W.A. Stahel. *Robust Statistics : the Approach Based on Influence Functions*. Wiley series in probability and mathematical. John Wiley & Sons, Ltd., New York, 1986.

[Hub81] P.J. Huber. *Robust Statistics*, volume IX. John Wiley & Sons, Ltd., New York, 1981.

[Neu69] H. Neudecker. A note on Kronecker matrix product and matrix equation systems. *SIAM J. Appl. Math.*, 17(3):603–606, 1969.

[Pen96] X. Pennec. *L'Incertitude dans les problèmes de reconnaissance et de recalage. Application en imagerie médicale et biologie moléculaire.* Thèse de doctorat, École Polytechnique., October 1996.

[PTVF92] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C - The Art of Scientific Computing.* Cambridge University Press, 2nd edition, 1992.

[RB89] F. Rotella and P. Borne. Explicit solution of Sylvester and Lyapunov equations. *Mathematics and Computers in Simulation*, 31:271–281, 1989.

[RL87] P.J. Rousseeuw and A.M. Leroy. *Robust Regression and Outlier Detection*, volume XIV. John Wiley & Sons, Ltd., New York, 1987.

[Rou97] P.J. Rousseeuw. Introduction to positive-breakdown methods. *Handbook of Statistics*, 15:101–121, 1997.

[SLBE91] C. Samson, M. Le Borgne, and B. Espiau. *Robot Control: The Task Function Approach* . Clarendon Press, Oxford, 1991.

[TL89] R. Y. Tsai and R. K. Lenz. A new technique for fully autonomous and efficient 3d robotics hand/eye calibration. *IEEE Journal of Robotics and Automation*, 5(3):345–358, 1989.

Annexe A

Annexe

A.1 Contraintes orthogonales

La matrice de rotation $\mathbf{R}_X$ de la transformation caméra-pince est composée de 3 vecteurs-lignes $\vec{\mathbf{i}}, \vec{\mathbf{j}}, \vec{\mathbf{k}}$ unitaires et perpendiculaires entre eux :

$$\begin{cases} \mathbf{R}_X = \begin{pmatrix} \vec{\mathbf{i}}^T \\ \vec{\mathbf{j}}^T \\ \vec{\mathbf{k}}^T \end{pmatrix} \\ \\ \|\vec{\mathbf{i}}\|^2 = \|\vec{\mathbf{j}}\|^2 = \|\vec{\mathbf{k}}\|^2 = 1 \\ \\ \vec{\mathbf{i}}^T\vec{\mathbf{j}} = \vec{\mathbf{i}}^T\vec{\mathbf{k}} = \vec{\mathbf{j}}^T\vec{\mathbf{k}} = 0 \end{cases}$$

A.2 Produit de Kronecker et rotations

Proposition 3 *Soient 2 matrices de rotation conjuguées* $\mathbf{R}$ *et* $\mathbf{R}'$*, c'est-à-dire qu'il existe une matrice de rotation* $\mathbf{R}_X$ *telle que* $\mathbf{R}' = \mathbf{R}_X\mathbf{R}\mathbf{R}_X^T$*.*

1) Si $\vec{\mathbf{v}}$ *est un vecteur propre de* $\mathbf{R} \otimes \mathbf{R}'$*, alors* $(\mathbf{I} \otimes \mathbf{R}_X^T)\vec{\mathbf{v}}$ *est un vecteur propre de* $\mathbf{R} \otimes \mathbf{R}$ *pour la meme valeur propre.*

2) Inversement, si $\vec{\mathbf{x}}$ *est un vecteur propre de* $\mathbf{R} \otimes \mathbf{R}$*, alors* $(\mathbf{I} \otimes \mathbf{R}_X)\vec{\mathbf{x}}$ *est un vecteur propre de* $\mathbf{R} \otimes \mathbf{R}'$ *pour la meme valeur propre.*

Preuve : 1) Soit $\vec{\mathbf{v}}$ un vecteur propre de $\mathbf{R} \otimes \mathbf{R}'$ associé à la valeur propre λ : $(\mathbf{R} \otimes \mathbf{R}')\vec{\mathbf{v}} = \lambda\vec{\mathbf{v}}$. En remplaçant $\mathbf{R}'$ par $\mathbf{R}_X\mathbf{R}\mathbf{R}_X^T$ dans cette relation, nous obtenons :

$$(\mathbf{R} \otimes \mathbf{R}_X\mathbf{R}\mathbf{R}_X^T)\vec{\mathbf{v}} = \lambda\vec{\mathbf{v}}$$

qui devient, grâce à $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD})$[Bel60] :

$$(\mathbf{I} \otimes \mathbf{R}_X)(\mathbf{R} \otimes \mathbf{R})(\mathbf{I} \otimes \mathbf{R}_X^T)\vec{\mathbf{v}} = \lambda\vec{\mathbf{v}}$$

Comme $(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}$[Bel60], nous obtenons finalement:

$$(\mathbf{I} \otimes \mathbf{R}_X^T)^{-1}(\mathbf{R} \otimes \mathbf{R})(\mathbf{I} \otimes \mathbf{R}_X^T)\vec{\mathbf{v}} = \lambda\vec{\mathbf{v}}$$

D'où, $(\mathbf{R} \otimes \mathbf{R})(\mathbf{I} \otimes \mathbf{R}_X^T)\vec{\mathbf{v}} = \lambda(\mathbf{I} \otimes \mathbf{R}_X^T)\vec{\mathbf{v}}$.

2) Soit $\vec{\mathbf{x}}$ un vecteur propre de $\mathbf{R} \otimes \mathbf{R}$ associé à la valeur propre α. Alors,

$$\begin{aligned} (\mathbf{R} \otimes \mathbf{R})\vec{\mathbf{x}} &= \alpha\vec{\mathbf{x}} \\ (\mathbf{R} \otimes \mathbf{R})(\mathbf{I} \otimes \mathbf{R}_X^T)(\mathbf{I} \otimes \mathbf{R}_X)\vec{\mathbf{x}} &= \alpha(\mathbf{I} \otimes \mathbf{R}_X^T)(\mathbf{I} \otimes \mathbf{R}_X)\vec{\mathbf{x}} \\ (\mathbf{I} \otimes \mathbf{R}_X)(\mathbf{R} \otimes \mathbf{R})(\mathbf{I} \otimes \mathbf{R}_X^T)(\mathbf{I} \otimes \mathbf{R}_X)\vec{\mathbf{x}} &= \alpha(\mathbf{I} \otimes \mathbf{R}_X)\vec{\mathbf{x}} \\ (\mathbf{R} \otimes \mathbf{R}')(\mathbf{I} \otimes \mathbf{R}_X)\vec{\mathbf{x}} &= \alpha(\mathbf{I} \otimes \mathbf{R}_X)\vec{\mathbf{x}}. \end{aligned}$$

□

Proposition 4 *Soient* $\mathbf{R}_1$ *et* $\mathbf{R}_2$ *2 matrices de rotation d'axes différents. Soit* $\mathbf{R}$ *une autre matrice de rotation. Alors,*

$$\left.\begin{array}{l}\mathbf{R}_1 \otimes \mathbf{R}_1 \; vec(\mathbf{R}) = vec(\mathbf{R}) \\ \mathbf{R}_2 \otimes \mathbf{R}_2 \; vec(\mathbf{R}) = vec(\mathbf{R})\end{array}\right\} \Rightarrow \mathbf{R} = \mathbf{I}_3$$

Preuve : Le système précédent est équivalent à :

$$\begin{array}{l}\mathbf{R}_1\mathbf{R} = \mathbf{R}\mathbf{R}_1 \\ \mathbf{R}_2\mathbf{R} = \mathbf{R}\mathbf{R}_2\end{array}$$

Si $\mathbf{R}$ satisfait la première équation, alors soit $\mathbf{R}$ est l'identité soit elle a le même axe de rotation que $\mathbf{R}_1$. De même, $\mathbf{R}$ est l'identité ou bien a le même axe de rotation que $\mathbf{R}_2$. Comme $\mathbf{R}_1$ et $\mathbf{R}_2$ ont des axes de rotations différents, $\mathbf{R}$ est forcément l'identité. □

A.3 Démonstration de la Proposition 2

Le système (3.9) s'exprime en fonction de $vec(\mathbf{R}_X)$. Avec les définitions précédentes, on a donc :

$$(\mathbf{I}_9 - \mathbf{R}_A \otimes \mathbf{R}_B) \begin{pmatrix} \vec{\mathbf{i}} \\ \vec{\mathbf{j}} \\ \vec{\mathbf{k}} \end{pmatrix} = 0 \quad avec \; vec(\mathbf{R}_X) = \begin{pmatrix} \vec{\mathbf{i}} \\ \vec{\mathbf{j}} \\ \vec{\mathbf{k}} \end{pmatrix}$$

Ce système permet l'obtention d'un espace de solutions de dimension 1 (ceci est une conséquence directe des propriétés énoncées en A.2). Soit $\vec{\mathbf{v}} = (\vec{\mathbf{i}}_v^T, \vec{\mathbf{j}}_v^T, \vec{\mathbf{k}}_v^T)^T$, un vecteur directeur de cet espace. A $\vec{\mathbf{v}}$, on associe la matrice :

$$\mathbf{R}_v = vec^{-1}(\vec{\mathbf{v}}) = \begin{pmatrix} \vec{\mathbf{i}}_v^T \\ \vec{\mathbf{j}}_v^T \\ \vec{\mathbf{k}}_v^T \end{pmatrix}$$

Proposition 5 *La matrice* $\mathbf{R}_v$ *est inversible.*

Preuve : La matrice de rotation $\mathbf{R}_X$ de la transformation pince-caméra est solution du système. Comme l'espace des solutions est de dimension 1, $\mathbf{R}_v$ et $\mathbf{R}_X$ sont proportionnelles.

$\mathbf{R}_X$ étant inversible, $\mathbf{R}_v$ l'est donc aussi. □

Soit $\vec{\mathbf{v}}_0 = (\vec{\mathbf{i}}_0^T, \vec{\mathbf{j}}_0^T, \vec{\mathbf{k}}_0^T)^T$ un vecteur colinéaire à $\vec{\mathbf{v}}$ tel que la matrice $\mathbf{R}_0 = vec^{-1}(\vec{\mathbf{v}}_0)$ soit de déterminant égal à 1. On l'obtient par :

$$\mathbf{R}_0 = \frac{sgn(\mathbf{R}_v)}{|\det(\mathbf{R}_v)|^{\frac{1}{3}}} \mathbf{R}_v$$

Le vecteur $\vec{\mathbf{v}}_0$ est donc égal à :

$$\vec{\mathbf{v}}_0 = \frac{sgn(\mathbf{R}_v)}{|\det(\mathbf{R}_v)|^{\frac{1}{3}}} \vec{\mathbf{v}}$$

et par conséquent :

Proposition 6 *La matrice* $\mathbf{R}_0$ *définie précédemment est une matrice de rotation.*

Preuve : Le vecteur $\vec{\mathbf{v}}_0$ est solution du système (3.9). Il vérifie donc le système d'équations :

$$\begin{array}{rcl} \vec{\mathbf{i}}_0 & = & \mathbf{R}_{A_{11}}\mathbf{R}_B\vec{\mathbf{i}}_0 + \mathbf{R}_{A_{12}}\mathbf{R}_B\vec{\mathbf{j}}_0 + \mathbf{R}_{A_{13}}\mathbf{R}_B\vec{\mathbf{k}}_0 \\ \vec{\mathbf{j}}_0 & = & \mathbf{R}_{A_{21}}\mathbf{R}_B\vec{\mathbf{i}}_0 + \mathbf{R}_{A_{22}}\mathbf{R}_B\vec{\mathbf{j}}_0 + \mathbf{R}_{A_{23}}\mathbf{R}_B\vec{\mathbf{k}}_0 \\ \vec{\mathbf{k}}_0 & = & \mathbf{R}_{A_{31}}\mathbf{R}_B\vec{\mathbf{i}}_0 + \mathbf{R}_{A_{32}}\mathbf{R}_B\vec{\mathbf{j}}_0 + \mathbf{R}_{A_{33}}\mathbf{R}_B\vec{\mathbf{k}}_0 \end{array} \qquad (A.1)$$

Calculons à présent le produit vectoriel de $\vec{\mathbf{i}}_0$ par $\vec{\mathbf{j}}_0$:

$$\begin{aligned}
\vec{\mathbf{i}}_0 \times \vec{\mathbf{j}}_0 &= (\mathbf{R}_{A_{11}}\mathbf{R}_B\vec{\mathbf{i}}_0 + \mathbf{R}_{A_{12}}\mathbf{R}_B\vec{\mathbf{j}}_0 + \mathbf{R}_{A_{13}}\mathbf{R}_B\vec{\mathbf{k}}_0) \\
&\times (\mathbf{R}_{A_{21}}\mathbf{R}_B\vec{\mathbf{i}}_0 + \mathbf{R}_{A_{22}}\mathbf{R}_B\vec{\mathbf{j}}_0 + \mathbf{R}_{A_{23}}\mathbf{R}_B\vec{\mathbf{k}}_0) \\
&= (\mathbf{R}_{A_{11}}\mathbf{R}_B\vec{\mathbf{i}}_0) \times (\mathbf{R}_{A_{21}}\mathbf{R}_B\vec{\mathbf{i}}_0) + (\mathbf{R}_{A_{11}}\mathbf{R}_B\vec{\mathbf{i}}_0) \times (\mathbf{R}_{A_{22}}\mathbf{R}_B\vec{\mathbf{j}}_0) \\
&+ (\mathbf{R}_{A_{11}}\mathbf{R}_B\vec{\mathbf{i}}_0) \times (\mathbf{R}_{A_{23}}\mathbf{R}_B\vec{\mathbf{k}}_0) + (\mathbf{R}_{A_{12}}\mathbf{R}_B\vec{\mathbf{j}}_0) \times (\mathbf{R}_{A_{21}}\mathbf{R}_B\vec{\mathbf{i}}_0) \\
&+ (\mathbf{R}_{A_{12}}\mathbf{R}_B\vec{\mathbf{j}}_0) \times (\mathbf{R}_{A_{22}}\mathbf{R}_B\vec{\mathbf{j}}_0) + (\mathbf{R}_{A_{12}}\mathbf{R}_B\vec{\mathbf{j}}_0) \times (\mathbf{R}_{A_{23}}\mathbf{R}_B\vec{\mathbf{k}}_0) \\
&+ (\mathbf{R}_{A_{13}}\mathbf{R}_B\vec{\mathbf{k}}_0) \times (\mathbf{R}_{A_{21}}\mathbf{R}_B\vec{\mathbf{i}}_0) + (\mathbf{R}_{A_{13}}\mathbf{R}_B\vec{\mathbf{k}}_0) \times (\mathbf{R}_{A_{22}}\mathbf{R}_B\vec{\mathbf{j}}_0) \\
&+ (\mathbf{R}_{A_{13}}\mathbf{R}_B\vec{\mathbf{k}}_0) \times (\mathbf{R}_{A_{23}}\mathbf{R}_B\vec{\mathbf{k}}_0) \\
&= (\mathbf{R}_{A_{11}}\mathbf{R}_{A_{22}} - \mathbf{R}_{A_{12}}\mathbf{R}_{A_{21}})(\mathbf{R}_B\vec{\mathbf{i}}_0) \times (\mathbf{R}_B\vec{\mathbf{j}}_0) \\
&+ (\mathbf{R}_{A_{11}}\mathbf{R}_{A_{23}} - \mathbf{R}_{A_{13}}\mathbf{R}_{A_{21}})(\mathbf{R}_B\vec{\mathbf{i}}_0) \times (\mathbf{R}_B\vec{\mathbf{k}}_0) \\
&+ (\mathbf{R}_{A_{12}}\mathbf{R}_{A_{23}} - \mathbf{R}_{A_{13}}\mathbf{R}_{A_{22}})(\mathbf{R}_B\vec{\mathbf{j}}_0) \times (\mathbf{R}_B\vec{\mathbf{k}}_0)
\end{aligned}$$

Comme $\mathbf{R}_A$ est une matrice de rotation, nous avons :

$$\begin{aligned}
\mathbf{R}_{A_{11}}\mathbf{R}_{A_{22}} - \mathbf{R}_{A_{12}}\mathbf{R}_{A_{21}} &= \mathbf{R}_{A_{33}} \\
\mathbf{R}_{A_{11}}\mathbf{R}_{A_{23}} - \mathbf{R}_{A_{13}}\mathbf{R}_{A_{21}} &= -\mathbf{R}_{A_{32}} \\
\mathbf{R}_{A_{12}}\mathbf{R}_{A_{23}} - \mathbf{R}_{A_{13}}\mathbf{R}_{A_{22}} &= \mathbf{R}_{A_{31}}
\end{aligned}$$

De plus, on a, pour tous vecteurs $\vec{\mathbf{u}}_1$ et $\vec{\mathbf{u}}_2$, l'égalité:

$$(\mathbf{R}_B\vec{\mathbf{u}}_1) \times (\mathbf{R}_B\vec{\mathbf{u}}_2) = \mathbf{R}_B(\vec{\mathbf{u}}_1 \times \vec{\mathbf{u}}_2)$$

En reportant ces résultats dans le produit vectoriel $\vec{\mathbf{i}}_0 \times \vec{\mathbf{j}}_0$ et en refaisant les mêmes calculs sur $\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0$ et $\vec{\mathbf{k}}_0 \times \vec{\mathbf{i}}_0$, nous obtenons :

$$\begin{aligned}
\vec{\mathbf{i}}_0 \times \vec{\mathbf{j}}_0 &= \mathbf{R}_{A_{31}}\mathbf{R}_B(\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0) + \mathbf{R}_{A_{32}}\mathbf{R}_B(\vec{\mathbf{k}}_0 \times \vec{\mathbf{i}}_0) + \mathbf{R}_{A_{33}}\mathbf{R}_B(\vec{\mathbf{i}}_0 \times \vec{\mathbf{j}}_0) \\
\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0 &= \mathbf{R}_{A_{11}}\mathbf{R}_B(\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0) + \mathbf{R}_{A_{12}}\mathbf{R}_B(\vec{\mathbf{k}}_0 \times \vec{\mathbf{i}}_0) + \mathbf{R}_{A_{13}}\mathbf{R}_B(\vec{\mathbf{i}}_0 \times \vec{\mathbf{j}}_0) \\
\vec{\mathbf{k}}_0 \times \vec{\mathbf{i}}_0 &= \mathbf{R}_{A_{21}}\mathbf{R}_B(\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0) + \mathbf{R}_{A_{22}}\mathbf{R}_B(\vec{\mathbf{k}}_0 \times \vec{\mathbf{i}}_0) + \mathbf{R}_{A_{23}}\mathbf{R}_B(\vec{\mathbf{i}}_0 \times \vec{\mathbf{j}}_0)
\end{aligned}$$

Par conséquent, les vecteurs $(\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0, \vec{\mathbf{k}}_0 \times \vec{\mathbf{i}}_0, \vec{\mathbf{i}}_0 \times \vec{\mathbf{j}}_0)$ sont solution du même système (3.9) que les vecteurs $(\vec{\mathbf{i}}_0, \vec{\mathbf{j}}_0, \vec{\mathbf{k}}_0)$. Il existe donc un facteur λ non nul tel que :

$$\begin{pmatrix} \vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0 \\ \vec{\mathbf{k}}_0 \times \vec{\mathbf{i}}_0 \\ \vec{\mathbf{i}}_0 \times \vec{\mathbf{j}}_0 \end{pmatrix} = \lambda \begin{pmatrix} \vec{\mathbf{i}}_0 \\ \vec{\mathbf{j}}_0 \\ \vec{\mathbf{k}}_0 \end{pmatrix}$$

De $\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0 = \lambda\vec{\mathbf{i}}_0$, nous déduisons que $\vec{\mathbf{i}}_0$ est perpendiculaire à $\vec{\mathbf{j}}_0$ et $\vec{\mathbf{k}}_0$ et de $\vec{\mathbf{k}}_0 \times \vec{\mathbf{i}}_0 = \lambda\vec{\mathbf{j}}_0$, nous obtenons la pièce manquante : $\vec{\mathbf{j}}_0$ est perpendiculaire à $\vec{\mathbf{k}}_0$.

Par conséquent, **les vecteurs $(\vec{\mathbf{i}}_0, \vec{\mathbf{j}}_0, \vec{\mathbf{k}}_0)$ forment une base orthogonale directe.**

Il reste à vérifier qu'ils sont tous de norme 1. Calculons, pour ce faire, le produit scalaire $\vec{\mathbf{i}}_0 \cdot (\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0)$:

$$\vec{\mathbf{i}}_0 \cdot (\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0) = \begin{cases} \vec{\mathbf{i}}_0 \cdot (\lambda\vec{\mathbf{i}}_0) = \lambda\|\vec{\mathbf{i}}_0\|^2 \\ \\ \det(\vec{\mathbf{i}}_0\vec{\mathbf{j}}_0\vec{\mathbf{k}}_0) = 1 \end{cases}$$

On a donc :

$$\|\vec{\mathbf{i}}_0\|^2 = \|\vec{\mathbf{j}}_0\|^2 = \|\vec{\mathbf{k}}_0\|^2 = \frac{1}{\lambda}, \quad \lambda > 0 \tag{A.2}$$

L'égalité $\vec{\mathbf{j}}_0 \times \vec{\mathbf{k}}_0 = \lambda\vec{\mathbf{i}}_0$ nous donne en passant à la norme :

$$\|\vec{\mathbf{j}}_0\|\|\vec{\mathbf{k}}_0\| = \lambda\|\vec{\mathbf{i}}_0\|$$

puisque les vecteurs sont orthogonaux. D'où, en remplaçant les normes par leur valeur $\frac{1}{\lambda}$:

$$\frac{1}{\lambda^2} = \lambda \frac{1}{\lambda}$$

d'où $\lambda = 1$ découle immédiatement.

Ainsi, **les vecteurs $(\vec{\mathbf{i}}_0, \vec{\mathbf{j}}_0, \vec{\mathbf{k}}_0)$ forment une base orthonormée directe.** □

Proposition 7 *1) La matrice* $\mathbf{R}_0$ *est l'unique matrice de rotation qui soit solution du système.*
2) $\mathbf{R}_0$ *est la rotation de la transformation pince-caméra.*

Preuve :

1. Soit $\mathbf{R}$, une autre solution du système différente de $\mathbf{R}_0$.
   Alors, il existe un facteur $\lambda \neq 1$ tel que $\mathbf{R} = \lambda \mathbf{R}_0$.
   On en déduit donc que $\det(\mathbf{R}) = \lambda^3 \det(\mathbf{R}_0) = \lambda^3 \neq 1$.
   Par conséquent, **$\mathbf{R}$ n'est pas une matrice de rotation**.
2. Soit $\mathbf{R}_X$, la matrice de rotation associée à la transformation pince-caméra. $\mathbf{R}_X$ est une matrice de rotation qui est solution du système. D'après la condition d'unicité, $\mathbf{R}_X$ est donc identique à $\mathbf{R}_0$.

□



Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399