Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Cybernetics

Global Optimization Techniques in Camera-Robot Calibration

Doctoral Thesis

presented to the Faculty of the Electrical Engineering of the Czech Technical University in Prague in Partial Fulfillment of the Requirements for the Ph.D. Degree in Study Programme No. P–2612–Electrotechnics and Informatics, branch No. 3902V035–Artificial Intelligence and Biocybernetics, by

Jan Heller

Prague, August 2015

Thesis Advisor
Ing. Tomáš Pajdla, Ph.D.

Center for Machine Perception
Department of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague
Karlovo náměstí 13, 12135 Prague 2, Czech Republic
fax: +420 224 357 385, phone: +420 224 357 465
http://cmp.felk.cvut.cz

Abstract

The need to relate measurements made by a camera to a different known coordinate system arises in many engineering applications. Historically, it appeared for the first time in the connection with cameras mounted on robotic systems. This problem is commonly known as hand-eye calibration. In this thesis, we study the problem of hand-eye calibration as well as a problem closely connected to it—the problem of robot-world calibration. The first objective of this work is to apply recent results in mathematical optimization to provide globally optimal solution to these problems. The second objective is to formulate and study these problems as minimization problems under some geometrically meaningful error measures using image measurements directly. We also study global optimizers in situation where image measurements are not available using the classical problem formulations. The solutions presented in this thesis are compared to existing methods and validated by both synthetic and real world data experiments.

In the first part of the thesis, we survey the state of the art of the camera-robot calibration as well as of the main concepts of the geometrical computer vision. Further, we review several results in the globally optimization techniques and their application in the computer vision.

Next, we formulate the problem of hand-eye calibration as a minimization problem under some geometrically meaningful error measures. We provide two solutions; the first solution employs a Structure-from-Motion approach and the Second Order Cone Programming optimization and the second one uses the Brand-and-Bound optimization strategy. Both solutions provide globally optimal minimizers and work with image measurements directly, instead of using them as a pre-step for explicitly calculating camera poses. Using a similar approach, we also formulate a minimization task for the robot-world calibration problem. This time, to solve the task we use the method of Linear Matrix Inequality relaxations.

Further, we investigate the problem of hand-eye calibration in situations, where the information about the rotation of the robot is not known. This problem arises when the robot is not calibrated or the information from the robot is not available. We use the method of Gröbner basis to deal with this scenario.

Finally, we revisit the classical formulation of the hand-eye and robot-world calibration. Using the method of Linear Matrix Inequality relaxations, we provide several global optimizers in situations, where image measurements are not available and the calibration has to be estimated from robot and camera poses only.

Anotace

Potřeba porovnávat měření získaná pomocí kamery s měřeními získanými v jiných souřadných systémech vyvstává v mnoha inženýrských aplikacích. Historicky se tento požadavek poprvé objevil ve spojitosti s kamerami spojenými s robotickými systémy, proto se pro tuto úlohu vžilo označení kalibrace ruka-oko. V této práci studujeme problém kalibrace ruka-oko a problém s ním úzce spojený – kalibrace robot-svět. Hlavním cílem této práce je aplikace moderních metod matematické optimalizace k získání globálně optimálních řešení těchto dvou problémů. Dalším cílem je formulace těchto problémů jako minimalizačních úloh které pracují s fyzikálně motivovanými cílovými funkcemi a kde vstupními daty jsou přímo měření v obraze. Metody a výsledky představené v této práci byly porovnány s již existujícími metodami a ověřeny jak na syntetických datech, tak na datech získanými reálnými meření.

V první části disertace uvádíme přehled existujících metod pro kalibraci systémů kamera-robot a hlavních metod a konceptů geometrie počítačového vidění. Dále následuje přehled několika technik globální optimalizace a jejich existujících aplikací v počítačovém vidění.

V druhé části disertace předkládáme výsledky našeho originálního výzkumu problému kalibrace ruka-oko. Formulujeme zde tento problém jako optimalizační úlohu s fyzikálně motivovanou cílovou funkcí a prezentujeme dvě řešení. První řešení je založeno na technice výpočtu tvaru z pohybu (*Structure-from-Motion*) a kónickém programování druhého řádu (*Second-Order Cone Programming*), druhé pak na metodě větví a mezí (*Branch-and-Bound*). Obě formulace poskytují globálně optimální řešení a měření v obraze zde slouží přímo jako vstupní data problému místo toho, aby byly použity při předzpracování pro výpočet absolutních pozic kamery. Pomocí obdobného přístupu také formulujeme optimalizační úlohu pro problém kalibrace robot-svět. V tomto případě jsme pro výpočet řešení použili metodu relaxace lineárních maticových nerovností (*Linear Matrix Inequality relaxations*).

Dále v této práci zkoumáme problém kalibrace ruka-oko v situacích, kdy informace o rotaci robota není známa. Tento problém vyvstává v případě, kdy robot není kalibrován nebo informace o poloze robota není známa. Pro řešení tohoto problému využíváme metodu Gröbnerových bází (*Gröbner basis*).

Nakonec představujeme množinu globálně optimálních řešení, kde pomocí metody relaxace lineárních maticových nerovností řešíme problémy kalibrace ruka-oko a robot-svět pro případy, kdy měření v obraze nejsou k dispozici nebo kdy je nutno kalibraci provést pouze na základě informací o pozici kamery a robota.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my thesis advisor Dr. Tomáš Pajdla for his support during my study and research, for his immense knowledge and enthusiasm, and, above all, for his patience. Besides my advisor, I would like to thank my two closest colleagues Michal Havlena and Zuzana Kúkelová, who co-authored a great deal of the research presented in this work and without whom this work would certainly never saw the light of the day.

My sincere thanks also goes to Dr. Akihiro Sugimoto, whom I had the honor to have as my advisor during my intership at the NII in Tokyo, for his insights and scientific guidance. I am equally indebted to Dr. Didier Henrion, who guided and co-authored my research during my stay at LAAS-CNRS in Toulouse.

I would like to express my thanks to my colleagues at CMP with whom I had the pleasure of working with for their ideas and their collaborative efforts. Namely, I would like to thank Martin Meloun, Libor Wagner, and Vladimír Petrík who assisted with the practical side of the real-data experiments performed in this thesis.

I gratefully acknowledge EC projects FP7-SPA-218814 PRoVisG, FP7-SPACE-241523 PRoViScout, FP7-SPACE-2012-312377 PRoViDE, FP7-SME-2011-285839 De-Montes, and FP7-288553 CloPeMa for the financial support of my research.

Last but not least, I would like to thank my family: my parents and my wife Tereza, who supported me during my seemingly never ending work on my thesis by all means necessary. Finally, I would like to thank to my son Adam, who took pity on me and forbore one of his life pleasures and stopped screaming at times so his father could finish this thesis.

Any errors or inadequacies that may remain in this work are, of course, entirely my own.

The good life is one inspired by love and guided by knowledge.

– Bertrand Russell

Contents

Keywords: Camera-Robot Calibration, Hand-Eye Calibration, Robot-World Calibration, Branch-and-Bound, Groebner Bases, Convex Relaxation, SOCP, SDP, LMI.

1 Introduction

The shortest distance between two points is often unbearable.

– Charles Bukowski

The idea of using cameras to help navigate robots is not new and has been an area of intensive research for many decades. In such scenarios, a camera is used as a measuring tool to measure angles and—with the help of more than one vantage point—distances. It is indeed no surprise that in order to use a camera in this way, the camera needs to be calibrated, *i.e.*, one needs to be able to determine the true direction of a ray projecting into every pixel with respect to the camera pose. The problem of camera calibration is studied in the fields of photogrammetry and geometric computer vision. The next step is to determine the transformation from the camera pose to the coordinate system connected with the robot. In this work, we will be concerned with scenarios where a camera is rigidly connected with a robotic arm, the so called *eye-in-hand* scenario, and with the problem of determining this transformation—the *hand-eye transformation*. The problem of determining the hand-eye transformation is commonly known as *hand-eye calibration*. The hand-eye calibration problem arises also in seemingly unrelated fields ranging from medical applications such as ultrasound and endoscopy to automotive industry. Another problem that arises when using an eye-in-hand system is the problem of *robot-world calibration*–-the problem of determining the transformation from the coordinate system connected with the base of the robotic arm to the coordinate system connected with the world. This problem is closely connected to the hand-eye calibration problem and can be shown to be algebraically equivalent to it.

Both problems have been extensively studied in the past. The main and novel idea behind this work is to formulate the problems of hand-eye and robot-world calibration as minimization problems under geometrically meaningful error measures in the image measurements directly, without the need for explicit knowledge of the full camera poses. This approach was not investigated in the past much and it is argued in the latter that this approach can lead to superior results. The next step is to apply the recent results in global optimization to provide global optimizers to these problems. Besides providing globally optimal solutions to such formulations, we also investigate situations where the image measurements or part of the robot pose are not available.

1.1 Thesis Outline

After the introductory Chapters 1 to 3, the presented work can be divided into two distinct parts. Chapters 4 to 6 present what is commonly referred to as the state-of-the-art part of the thesis. Chapters 7 trough 11 present our original research and the main contributions of this work. The thesis is concluded in Chapter 12.

Chapter 4 deals with the fundamental concepts of the geometric computer vision to a level pertinent to the main topic—camera-robot calibration. The problem of camera-robot calibration is formulated in Chapter 5. Here, most of the solution strategies are reviewed. Chapter 6 introduces several mathematical methods that can provide globally optimal solutions for certain classes of problems. Further, computer vision problems which were successfully solved using these methods are discussed.

In Chapter 7, a method for hand-eye calibration in situation where the correct scale of the scene is not known is presented. First, Structure-from-Motion (SfM) approach is used to recover the rotational parts of the camera poses; using these rotations the rotational part of the rigid hand-eye transformation can be recovered using any classical hand-eye calibration solution. The translational part is recovered globally optimally as a solution to a Second Order Cone Programming (SOCP) problem. For this part, the known rotation is used. The need for prior knowledge of the camera poses is lifted altogether in a novel brand-and-bound method presented in Chapter 8. The algorithm can recover the rotational and translational parts simultaneously, globally optimally, and does need neither prior knowledge about the scale of the scene nor any part of the camera poses information.

In Chapter 9, hand-eye calibration is investigated in situations where information about the rotational part of the robotic arm is not available. The Gröbner bases method is used to recover the calibration in such cases.

The method of Linear Matrix Inequalities (LMI) relaxations is employed in Chapters 10 and 11. First, it is used to recover the globally optimal robot-world calibration in Chapter 10 using known hand-eye calibration and image measurements. In the case the image measurements are not available, the LMI method can still be used to recover globally optimal solutions for both the hand-eye and the robot-world calibration problems, as shown in Chapter 11.

2 Contribution

Results! Why, man, I have gotten a lot of results. I know several thousand things that won't work.

– Thomas Edison

This thesis focuses on applying global optimization techniques to the problems of robot-world calibration. Besides providing globally optimal solutions, the main goal is to improve accuracy by using geometrically meaningful error measures and to provide novel insights into these problems. Most of the methods presented in this work are designed to use image measurements directly, instead of using them as a pre-step for computing absolute or relative camera poses. In situations where the image measurements are not available, we show that globally optimal solutions, albeit based on different error measures, can still be recovered.

Primarily, the contributions are the following:

- *Globally optimal hand-eye calibration algorithm*. Chapter 8 presents a branch-and-bound algorithm that provides global optimizers for the problem of hand-eye calibration. This algorithm is based on geometrically meaningful error measure and does not require a known calibration device, *e.g.*, works with unstructured scenes with unknown scale based on image-to-image point correspondences.

- *Globally optimal solution for the classical hand-eye calibration formulation*. For situations when the image measurements are not available, Chapter 11 provides a set hand-eye calibration formulations and iterative solutions. These solutions use the method of convex linear matrix inequality (LMI) relaxations to effectively obtain globally optimal solutions.

- *Hand-eye calibration without hand orientation*. Chapter 9 investigates a hand-eye calibration scenario where the orientation of the robotic hand is not known. Such a situation arises when hand-eye calibration is to be performed on a internally uncalibrated robot. Typically, external measuring device can only observe the translational part of the motion of the robotic hand. The method presented in Chapter 9 uses the method of Gröbner bases for solving systems of polynomial equations to provide hand-eye calibration in such situations.

- *GpoSolver: A Matlab/C++ Toolbox for Global Polynomial Optimization*. As a part of this work, we developed a software for polynomial optimization based on the theory of LMI relaxations, see Section 6.2.1.1. The software can be downloaded from the project webpage at

  `http://cmp.felk.cvut.cz/gposolver`.

- The implementations of the methods presented in Chapters 8, 9, and 11 available at

  `http://cmp.felk.cvut.cz/~hellej1/bbhec`,
  `http://cmp.felk.cvut.cz/minimal/handeye.php`,
  `http://cmp.felk.cvut.cz/~hellej1/mpherwc`.

2.1 Publications

The content of this thesis is based on the material published in the following publications:

[75] Jan Heller, Michal Havlena, Akihiro Sugimoto, and Tomas Pajdla. Structure-from-motion based hand-eye calibration using L_∞ minimization. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3497–3503. IEEE, June, 2011.

[74] Jan Heller, Michal Havlena, and Tomas Pajdla. A branch-and-bound algorithm for globally optimal hand-eye calibration. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1608–1615. IEEE, June, 2012.

[97] Zuzana Kukelova, Jan Heller, and Tomas Pajdla. Hand-eye calibration without hand orientation measurement using minimal solution. *In Asian Conference on Computer Vision (ACCV)*, pages 576–589. Springer, November, 2012.

[76] Jan Heller, Didier Henrion, and Tomas Pajdla. Hand-eye and robot-world calibration by global polynomial optimization. *In IEEE International Conference on Robotics and Automation (ICRA)*, pages 3157–3164. IEEE, May, 2014.

[77] Jan Heller and Tomas Pajdla. World-base calibration by global polynomial optimization. *In 2nd International Conference on 3D Vision (3DV)*, pages 593-600. IEEE, December, 2014.

[82] Jan Heller, Michal Havlena, and Tomas Pajdla. Globally optimal hand-eye calibration using branch-and-bound. *In Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, accepted for publication, August 2015.

2.2 Authorship

I hereby certify that the results presented in this thesis were achieved during my own research in cooperation with my thesis advisor Tomáš Pajdla, Michal Havlena [75, 74, 82], work presented in Chapters 7 and 8, and Zuzana Kúkelová [97], work presented in Chapter 9. I also cooperated with Akiriho Sugimoto and Didier Henrion on work presented in Chapters 7 and 11, respectively.

Martin Meloun, Libor Wagner, and Vladimír Petrík assisted with the practical side of the real-data experiments performed in this thesis.

3 Notation and Concepts

We demand rigidly defined areas of doubt and uncertainty!

– Douglas Adams, *The Hitchhiker's Guide to the Galaxy*

3.1 Notation

The following list summarized the notation used throughout this thesis.

$a, b, \ldots, \alpha, \beta, \ldots$	scalars
$\mathcal{A}, \mathcal{B}, \ldots$	sets
$\mathbf{a}, \mathbf{b}, \ldots, \mathbf{A}, \mathbf{B}, \ldots$	column vectors
$\mathtt{A}, \mathtt{B}, \ldots$	matrices
$\mathbb{R}^n, \mathbb{P}^n, \ldots$	n-dimensional spaces
$\mathbb{R}$	real numbers
$\mathbb{Q}$	quaternions
$\mathbb{H}$	dual quaternions
$SO(3)$	special orthogonal group, group of all rotations about the origin of $\mathbb{R}^3$
$SE(3)$	special Euclidean group, set of all transformation matrices of the form $\left\{ \begin{pmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix} \middle\| \mathtt{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3 \right\}$
$\mathbb{R}[\mathbf{x}]$	ring of multivariate polynomials in the n-tuple of variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)$.
$\mathbf{0} = \begin{pmatrix} 0 \ldots 0 \end{pmatrix}^\top$	vector of zeros
$\|\cdot\|$	vector norm
$\|\cdot\|_2$	L_2-norm
$\|\cdot\|_\infty$	L_∞-norm
$\|\cdot\|_F$	Frobenius norm
$\times$	vector cross product
$\mathtt{M} \succeq \mathtt{N}$	$\mathtt{M} - \mathtt{N}$ is a positive semidefinite matrix
$\mathbf{u} \leftrightarrow \mathbf{v}$	image correspondence
$\angle(\mathbf{u}, \mathbf{v})$	angle between vectors $\mathbf{u}$ and $\mathbf{v}$

3.2 Abbreviations

The following is the list of abbreviation.

SVD	singular value decomposition
LP	linear programming
SOCP	second-order cone programming
LMI	linear matrix inequality
SDP	semidefinite programming
SfM	structure-from-motion
EXIF	exchangeable image file format used in digital cameras
RANSAC	random sample consensus

3.3 Basic Definitions

All vectors in this work are *column* vectors and are considered $n \times 1$ matrices when multiplied by a matrix. $\begin{pmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_n \end{pmatrix}$ denotes a matrix $\mathtt{A} \in \mathbb{R}^{m \times n}$, which columns are vectors $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n \in \mathbb{R}^m$.

Definition 3.1 (Skew-symmetric matrix). Let $\mathbf{a} = (a_1, a_2, a_3)^\top \in \mathbb{R}^3 \backslash \{(0,0,0)^\top\}$, then $[\mathbf{a}]_\times$ is the *skew-symmetric matrix*

$$[\mathbf{a}]_\times \stackrel{\text{def}}{=} \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & -a_1 & 0 \end{pmatrix}.$$

Definition 3.2 (Operator vec). Let $\mathtt{A} \in \mathbb{R}^{m \times n}$ be an $m \times n$ matrix and let $\mathbf{a}_1^\top, \mathbf{a}_2^\top, \dots, \mathbf{a}_m^\top$ be the rows of the matrix $\mathtt{A}$. Then $\mathrm{vec}(\mathtt{A}) \in \mathbb{R}^{mn}$ is the mn vector

$$\mathrm{vec}(\mathtt{A}) \stackrel{\text{def}}{=} \mathbf{a} = \begin{pmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_n \end{pmatrix},$$

and $\mathrm{vec}^{-1}(\mathbf{a}) \in \mathbb{R}^{m \times n}$ is the $m \times n$ matrix $\mathrm{vec}^{-1}(\mathbf{a}) = \mathtt{A}$.

Definition 3.3 (Kronecker product). Let $\mathtt{A}$ be an $m \times n$ matrix and $\mathtt{B}$ an $p \times q$ matrix. Then the *Kronecker product* $\mathtt{A} \otimes \mathtt{B}$ is the $mp \times nq$ block matrix

$$\mathtt{A} \otimes \mathtt{B} \stackrel{\text{def}}{=} \begin{pmatrix} a_{11}\mathtt{B} & \cdots & a_{1n}\mathtt{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathtt{B} & \cdots & a_{mn}\mathtt{B} \end{pmatrix}.$$

3.4 Rotation Formalisms In Three Dimensions

The concept of rotation and its mathematical formalism plays a pivotal role in the development of the theory presented in this work. A rotation about the origin is a linear transformation that preserves length of vectors and orientation of space. Since the composition of two rotations is another rotation, every rotation has a unique inverse rotation, and the identity map satisfies the definition of a rotation, the set of all rotations is a group under composition. This group is often denoted $SO(3)$.

3.4.1 Rotation Matrix

As is the case with all linear maps on finite-dimensional vector spaces, a rotation can be always expressed by a matrix $\mathtt{R}$, in this case of dimensions 3×3. Since a rotation maps orthonormal basis of $\mathbb{R}^3$ to another orthonormal basis, the columns of matrix $\mathtt{R}$

$$\mathtt{R} = \begin{pmatrix} u_1 & v_1 & w_1 \\ u_2 & v_2 & w_2 \\ u_3 & v_3 & w_3 \end{pmatrix} = \begin{pmatrix} \mathbf{u} & \mathbf{v} & \mathbf{w} \end{pmatrix}$$

must form an orthonormal basis. This fact can be express using matrix multiplication as

$$\mathtt{R}^\top \mathtt{R} = \mathtt{I}.$$

The group of all orthogonal matrices is denoted $O(3)$. Besides preserving length, rotations also preserve orientation. Whether an orthogonal matrix preserves orientation or not depends on the sign of its determinant. Since

$$\det \mathtt{R}^\top = \det \mathtt{R} \text{ and } \mathtt{R}^\top \mathtt{R} = \mathtt{I} \implies \det \mathtt{R} = \pm 1,$$

only orthogonal matrices with $\det \mathtt{R} = 1$ preserve orientation. The group of all orthogonal matrices $\mathtt{R}$ such that $\det \mathtt{R} = 1$ is called special orthogonal group, $SO(3)$.

The fact that the columns of $\mathtt{R}$ form a orthonormal basis can be also written as

$$\begin{aligned} \mathbf{v}^\top \mathbf{v} &= 1, \\ \mathbf{u}^\top \mathbf{u} &= 1, \\ \mathbf{v}^\top \mathbf{u} &= 0, \\ \mathbf{v} \times \mathbf{u} &= \mathbf{w}. \end{aligned}$$

This constitutes 6 constraints on the elements of the matrix $\mathtt{R}$, leaving it with 3 degrees of freedom.

The successive application of rotations $\mathtt{R}_1, \mathtt{R}_2 \in SO(3)$ is easily expressed by matrix multiplication $\mathtt{R}_2\mathtt{R}_1$. This operation in three dimension is generally not commutative.

3.4.2 Angle-Axis

In the angle-axis parametrization, a vector $\boldsymbol{\alpha} \in \mathbb{B}_\pi = \{\boldsymbol{\alpha} \in \mathbb{R}^3 : \|\boldsymbol{\alpha}\| \leq \pi\}$ represents the rotation about axis $\boldsymbol{\alpha} / \|\boldsymbol{\alpha}\|$ by angle $\|\boldsymbol{\alpha}\|$. The fact that all rotations can be described by a

simple rotation about a single rotation axis is called *Euler's rotation theorem* [48]. This can we expressed in the matrix form as

$$\forall \mathtt{R} \in SO(3)\, \exists \boldsymbol{\alpha} \text{ such that } \mathtt{R}\boldsymbol{\alpha} = \boldsymbol{\alpha}.$$

In other words, every rotation matrix $\mathtt{R}$ has an eigenvector $\boldsymbol{\alpha}$ with the eigenvalue $\lambda = 1$. Since

$$\begin{aligned}\det(\mathtt{R} - \mathtt{I}) = \det((\mathtt{R} - \mathtt{I})^\top) = \det(\mathtt{R}^\top - \mathtt{I}) = \det(\mathtt{R}^{-1} - \mathtt{I}) = \det -\mathtt{R}^{-1}(\mathtt{R} - \mathtt{I}) \\ = -\det(\mathtt{R}^{-1})\det(\mathtt{R} - \mathtt{I}) = -\det(\mathtt{R} - \mathtt{I}) \implies \det(\mathtt{R} - \mathtt{I}) = 0, \quad (3.1)\end{aligned}$$

$\lambda = 1$ is indeed an eigenvalue of $\mathtt{R}$, *i.e.*, $\det(\mathtt{R} - \lambda\mathtt{I}) = 0$. This shows that the matrix $(\mathtt{R} - \mathtt{I})$ is singular and that $\boldsymbol{\alpha}$ lies in its null space, *i.e.*, $\boldsymbol{\alpha} \in \ker(\mathtt{R} - \mathtt{I})$. Once the axis of rotation $\boldsymbol{\alpha}$ is known, one can determine the angle of rotation θ as the angle between a vector $\mathbf{v}$ and $\mathtt{R}\mathbf{v}$, such that $\mathbf{v}$ is perpendicular to $\boldsymbol{\alpha}$,

$$\theta = \angle(\mathbf{v}, \mathtt{R}\mathbf{v}).$$

Since the special orthogonal group $SO(3)$ is a Lie group, there is a Lie algebra associated with it. This Lie algebra is denoted $so(3)$ and consists of all skew symmetric matrices 3×3. The conversion between matrix and angle-axis parametrization can be also performed using the exponential map $\exp\colon so(3) \to SO(3)$. Let $\boldsymbol{\alpha} \in \mathbb{B}_\pi$ be a vector representing a rotation. The corresponding matrix parametrization $\mathtt{R} \in \mathrm{SO}(3)$ can be obtained as

$$\mathtt{R} = \exp[\boldsymbol{\alpha}]_\times = \mathtt{I} + \frac{[\boldsymbol{\alpha}]_\times \sin\|\boldsymbol{\alpha}\|}{\|\boldsymbol{\alpha}\|} + \frac{[\boldsymbol{\alpha}]_\times^2 (1 - \cos\|\boldsymbol{\alpha}\|)}{\|\boldsymbol{\alpha}\|}. \quad (3.2)$$

This relation is also known as *Rodrigues' rotation formula*. The inverse map is obtained using logarithmic map $\log\colon SO(3) \to so(3)$ as

$$[\boldsymbol{\alpha}]_\times = \log \mathtt{R} = \frac{1}{\sin\left(\arccos\dfrac{\mathrm{tr}(\mathtt{R}) - 1}{2}\right)}\left(\mathtt{R} - \mathtt{R}^\top\right). \quad (3.3)$$

3.4.3 Quaternions

Quaternions, $\mathbb{Q}$, form a four-dimensional associative normed division algebra over the real numbers. A quaternion $\mathbf{q} \in \mathbb{Q}$ consists of a real part and an imaginary part and is usually denoted as

$$\mathbf{q} = q_1 + q_2\mathbf{i} + q_3\mathbf{j} + q_4\mathbf{k},$$

where $\mathbf{i}, \mathbf{j}, \mathbf{k}$ are the imaginary units such that

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1.$$

As a set, quaternions are equal to $\mathbb{R}^4$ and it is sometimes useful to write them as vectors

$$\mathbf{q} \equiv (q_1, q_2, q_3, q_4)^\top = \left(q_1, \bar{\mathbf{q}}^\top\right)^\top,$$

where $\bar{\mathbf{q}} \in \mathbb{R}^3$ is the imaginary part of the quaternion. Addition of two quaternions $\mathbf{p}, \mathbf{q} \in \mathbb{Q}$

$$\begin{aligned} \mathbf{p} &= (p_1, p_2, p_3, p_4)^\top &&= p_1 + p_2\mathbf{i} + p_3\mathbf{j} + p_4\mathbf{k}, \\ \mathbf{q} &= (q_1, q_2, q_3, q_4)^\top &&= q_1 + q_2\mathbf{i} + q_3\mathbf{j} + q_4\mathbf{k}, \end{aligned}$$

is equivalent to addition in $\mathbb{R}^4$:

$$\mathbf{p} + \mathbf{q} = (p_1, p_2, p_3, p_4)^\top + (q_1, q_2, q_3, q_4)^\top = (p_1 + q_1, p_2 + q_2, p_3 + q_3, p_4 + q_4)^\top . \quad (3.4)$$

Quaternion multiplication, however, does not have a counterpart operation on vector spaces:

$$\begin{aligned} \mathbf{p} * \mathbf{q} = (p_1, p_2, p_3, p_4)^\top * (q_1, q_2, q_3, q_4)^\top = (&p_1q_1 - p_2q_2 - p_3q_3 - p_4q_4 \\ &p_1q_2 + p_2q_1 + p_3q_4 - p_4q_3, \\ &p_1q_3 - p_2q_4 + p_3q_1 + p_4q_2, \\ &p_1q_4 + p_2q_3 - p_3q_2 + p_4q_1)^\top. \end{aligned} \quad (3.5)$$

The quaternion multiplication can be also written as a matrix multiplication as

$$\mathbf{p} * \mathbf{q} = \mathtt{M}(\mathbf{p})\mathbf{q} = \begin{pmatrix} p_1 & -\bar{\mathbf{p}}^\top \\ \bar{\mathbf{p}} & p_1\mathtt{I} + [\bar{\mathbf{p}}]_\times \end{pmatrix} \begin{pmatrix} q_1 \\ \bar{\mathbf{q}} \end{pmatrix} = \begin{pmatrix} p_1 & -p_2 & -p_3 & -p_4 \\ p_2 & p_1 & -p_4 & p_3 \\ p_3 & p_4 & p_1 & -p_2 \\ p_4 & -p_3 & p_2 & p_1 \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix}. \quad (3.6)$$

We can also use the matrix form to formally switch the order of the multipliers:

$$\mathbf{p} * \mathbf{q} = \mathtt{M}(\mathbf{p})\mathbf{q} = \bar{\mathtt{M}}(\mathbf{q})\mathbf{p} = \begin{pmatrix} q_1 & -\bar{\mathbf{q}}^\top \\ \bar{\mathbf{q}} & q_1\mathtt{I} - [\bar{\mathbf{q}}]_\times \end{pmatrix} \begin{pmatrix} p_1 \\ \bar{\mathbf{p}} \end{pmatrix}. \quad (3.7)$$

Because the group of unit quaternions with multiplication, modulo the negative sign, is isomorphic to the group of rotations with composition, they can be used to represent rotations. When used in this way, the components of a unit quaternions are sometimes called *Euler–Rodrigues parameters*. Rotation about axis $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)^\top$, $||\boldsymbol{\alpha}|| = 1$, by angle θ is represented by $\mathbf{q} \in \mathbb{Q}$ of the form

$$\mathbf{q} = \cos\frac{\theta}{2} + (\alpha_1\mathbf{i} + \alpha_2\mathbf{j} + \alpha_3\mathbf{k}) \sin\frac{\theta}{2}.$$

A vector $\mathbf{v} \in \mathbb{R}^3 = (v_1, v_2, v_3)^\top$ can be rotated by identifying it with an imaginary quaternion $v_1\mathbf{i} + v_2\mathbf{j} + v_3\mathbf{k}$ and evaluating the Hamilton product

$$\mathbf{v}_{\mathrm{rot}} = \mathbf{q} * (0, v_1, v_2, v_3)^\top * \mathbf{q}^*,$$

where $\mathbf{q}^*$ is the quaternion conjugate of $\mathbf{q}$:

$$\mathbf{q}^* = \cos\frac{\theta}{2} - (\alpha_1\mathbf{i} + \alpha_2\mathbf{j} + \alpha_3\mathbf{k}) \sin\frac{\theta}{2}.$$

The composition of two rotations $\mathbf{q}_1, \mathbf{q}_2 \in \mathbb{Q}$ is done by multiplication $\mathbf{q} = \mathbf{q}_2 * \mathbf{q}_1$. Rotation matrix $\mathtt{R} \in SO(3)$ equivalent to the unit quaternion $\mathbf{q} = (q_1, q_2, q_3, q_4)$ can be constructed using the following formula:

$$\mathtt{R} = \begin{pmatrix} q_1^2 + q_2^2 - q_3^2 - q_4^2 & 2q_2q_3 - 2q_4q_1 & 2q_2q_4 + 2q_3q_1 \\ 2q_2q_3 + 2q_4q_1 & q_1^2 - q_2^2 + q_3^2 - q_4^2 & 2q_3q_4 - 2q_2q_1 \\ 2q_2q_4 - 2q_3q_1 & 2q_3q_4 + 2q_2q_1 & q_1^2 - q_2^2 - q_3^2 + q_4^2 \end{pmatrix}. \quad (3.8)$$

3.5 Screw Theory

The screw motion theory is based on the fact that a general rigid body transformation can be accomplished by means of a translation along an unique axis and a rotation about the same axis. This is known as *Chasles theorem* [32] and such a description of a rigid motion is known as a *screw*. A screw can be represented using a rotation angle θ, $0 \leq \theta \leq 2\pi$, the amount of translational displacement d and vectors $\mathbf{c}, \mathbf{n} \in \mathbb{R}^3$, such that the line $\mathbf{c} + t\mathbf{n}$, $t \in \mathbb{R}$ is the common motion axis.

3.5.1 Dual Quaternions

Dual quaternions are the algebraic counterparts of screws. They form a Clifford algebra; a dual quaternion $\hat{\mathbf{q}} \in \mathbb{H}$ can be represented in the form

$$\hat{\mathbf{q}} = \mathbf{q} + \epsilon\mathbf{q}',$$

where $\mathbf{q}, \mathbf{q}' \in \mathbb{Q}$ and ϵ is the dual unit, $\epsilon\epsilon = 0$, that commutes with every element of the algebra. It is also convenient to write dual quaternions as vectors $(\mathbf{q}^\top, \mathbf{q}'^\top)^\top$, since the set of dual quaternions is equal to $\mathbb{R}^8$. Addition of two dual quaternions $\hat{\mathbf{p}}, \hat{\mathbf{q}} \in \mathbb{H}$, $\hat{\mathbf{p}} = \left(\mathbf{p}^\top, \mathbf{p}'^\top\right)^\top$, $\hat{\mathbf{q}} = \left(\mathbf{q}^\top, \mathbf{q}'^\top\right)^\top$ is equivalent to addition in $\mathbb{R}^8$. Multiplication can be expressed using quaternion multiplication as

$$\hat{\mathbf{p}} \otimes \hat{\mathbf{q}} = ((\mathbf{p} * \mathbf{q})^\top, (\mathbf{p} * \mathbf{q}' + \mathbf{p}' * \mathbf{q})^\top)^\top.$$

Similar to the way rotations in $\mathbb{R}^3$ can be represented by the quaternions of unit length, rigid motions in $\mathbb{R}^3$ can be represented by unit dual quaternions [122]: rotation represented by a quaternion $\mathbf{p} \in \mathbb{Q}$ followed by translation $\mathbf{t} \in \mathbb{R}^3$ is represented by a dual quaternion of the form

$$\hat{\mathbf{q}}(\mathbf{p}, \mathbf{t}) = (\mathbf{p}^\top, ((0, \tfrac{1}{2}\mathbf{t}^\top)^\top * \mathbf{p})^\top)^\top. \tag{3.9}$$

Unity of a dual quaternion $\hat{\mathbf{q}}$ can be expressed using its conjugate $\hat{\mathbf{q}}^* = (\mathbf{q}^{*\top}, \mathbf{q}'^{*\top})^\top$ as $\hat{\mathbf{q}}^* \otimes \hat{\mathbf{q}} = 1$ or using the quaternion parts as

$$\mathbf{q}^\top\mathbf{q} = 1 \quad \text{and} \quad q_1q_5 + q_2q_6 + q_3q_7 + q_4q_8 = 0.$$

4 Fundamental Concepts of Geometric Computer Vision

Let no one ignorant of geometry enter here.

– Plato

In this chapter, camera models and fundamental concepts of geometric computer vision are reviewed to a level significant to our work. By a camera we understand a device that records and stores two-dimensional images of the real world. A modern review of the camera models with the emphasis on omnidirectional cameras can be found in [15]. It is presumed that the reader has a basic knowledge of analytic projective geometry. Work [127] provides an introduction to the concepts of analytic perspective geometry and its connection to the computer vision. Book [71] is the ultimate reference for the geometry of computer vision.

4.1 Camera models

A computational model of a camera tells how to project 3D entities (points, lines, ...) onto the image and how to back-project image coordinates onto directional vectors in 3D. Different taxonomies can be devised for camera models. Here, the most important division of the camera models is into

1. *central camera models*, were a single optical center through which all rays entering the camera pass exists, and

2. *non-central camera models*, which do not poses a single optical center.

In this work, we will be concerned with central cameras only. An abundance of camera types can be modeled as central cameras, ranging from the "classical" directional pinhole camera to omnidirectional catadioptric [10, 23] and fish-eye [187, 80, 126] cameras, see Figure 4.1.

4.1.1 Pinhole camera

Pinhole camera, or standard central perspective camera, model is based on projective geometry and relates the coordinates of 3D points expressed in the coordinates connected with the camera coordinate system and the respective image points by the so-called calibration matrix

$$\mathtt{K} = \begin{pmatrix} f_u & s & x_0 \\ 0 & f_v & y_0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The calibration matrix $\mathtt{K} \in \mathbb{R}^{3\times3}$ depends on up to five *intrinsic (internal) camera parameters*: f_u, f_v express the focal length measured in pixels, s is the skew term and x_0, y_0 are the coordinates of the *principal point*—the intersection point of the optical axis and the image plane. Since the pixels of modern digital cameras are almost perfectly square, the calibration matrix used nowadays has typically the following simplified form

$$\mathtt{K} = \begin{pmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The origin of the camera coordinate system coincides with the optical center of the camera and the z-axis with the optical axis. It is customary to set the camera facing the positive direction of the z-axis and the x, y-axes to be collinear with the x, y-axes of the image. Let's suppose that

$$\mathbf{X} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^3$$

is a point expressed in the coordinates connected with the world coordinate system. The relation of the world coordinate system and the coordinate system connected with the camera is captured by the *extrinsic (external) camera parameters* as

$$\mathbf{X}' = \mathtt{R}\mathbf{X} + \mathbf{t},$$

where

$$\mathtt{R} = \begin{pmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{pmatrix} \in SO(3) \quad \text{and} \quad \mathbf{t} = \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \in \mathbb{R}^3$$

are the rotation matrix and the translation vector, respectively, and $\mathbf{X}' \in \mathbb{R}^3$ is the transformed point. Standard central perspective camera model expressed in the language of projective geometry states that

$$\mathbf{x} = \mathtt{P} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} = \mathtt{K} \begin{bmatrix} \mathtt{R} & \mathbf{t} \end{bmatrix} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}, \tag{4.1}$$

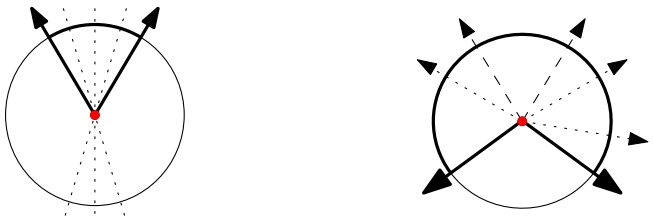
where $\mathtt{P} \in \mathbb{R}^{3\times4}$ is the so-called *camera projection matrix* and $\mathbf{x} \in \mathbb{R}^3 \backslash \{(0,0,0)^\top\}$ is the projective representation of the image point [71]. The pinhole camera model allows for easy back-projection of the image point $\mathbf{x}$ using the inverse calibration matrix as $\mathtt{K}^{-1}\mathbf{x}$.

If we wanted to express the relation using the Euclidean coordinates instead, the elegant matrix formula transforms into the following rational expression

$$\mathbf{x}' = \begin{pmatrix} f \cdot \dfrac{r_1x + r_2y + r_3z + t_1}{r_7x + r_8y + r_9z + t_3} + x_0 \\ f \cdot \dfrac{r_3x + r_4y + r_5z + t_2}{r_7x + r_8y + r_9z + t_3} + y_0 \end{pmatrix},$$

where $\mathbf{x}' \in \mathbb{R}^2$ are the pixel coordinates of the image point.

The pinhole camera model is very intuitive and easy to use, however, it does not sufficiently model real cameras with lenses. The model has been enhanced by various terms for



(a) (b)

Figure 4.1: (a) Directional camera—scene points are represented by straight lines. (b) Omnidirectional camera—scene points are represented by half-lines. (Adopted from [125])

radial, tangential, and other types of distortions [21, 152, 56]. A variant of Brown-Conrady distortion model [40, 21] is the common model that combines the radial and tangential distortions. Let $\mathbf{x}_d = (x_d, y_d)^\top \in \mathbb{R}^2$ be the distorted image point projected onto the image plane. The undistorted image point as projected by the pin-hole camera model $\mathbf{x}_u = (x_u, y_u)^\top \in \mathbb{R}^2$ is computed as

$$\begin{aligned}
\bar{x}_d &= x_d - x_0, \\
\bar{y}_d &= y_d - y_0, \\
r_d^2 &= \bar{x}_d^2 + \bar{y}_d^2, \\
x_u &= x_d + p_1(r_d^2 + 2\bar{x}_d^2) + 2p_2\bar{x}_d\bar{y}_d + \bar{x}_d \sum_{i=1}^{n} k_i r_d^{2i}, \\
y_u &= y_d + 2p_1\bar{x}_d\bar{y}_d + p_2(r_d^2 + 2\bar{y}_d^2) + \bar{y}_d \sum_{i=1}^{n} k_i r_d^{2i},
\end{aligned}$$

where p_1, p_2 are the *tangential distortion coefficients* and $k_i, i = 1, \ldots, n$ the *radial distortion coefficients*.

4.1.2 Omnidirectional camera

Central omnidirectional camera is any panoramic camera having a single effective viewpoint. Standard central perspective camera model is expressed in Equation 4.1. In this model all scene points lying on the same line passing through the optical center of the camera—in front as well as behind the camera—are represented by one image point, see 4.1a. This representation may be sufficient for directional cameras with field of view smaller than 180°, however, it is unsuitable for modeling omnidirectional cameras, where points behind the camera and points in front of the camera are projected onto different image points. This issue is addressed by the *spherical model*, where lines are split into half-lines, see Figure 4.1b. In this model, a vector $\mathbf{x} \in \mathbb{R}^3$ represents one half-line, so that one image point represents all scene points lying on a half of a line passing through the center of the camera and another image point represents all scene points lying on the opposite half of the same line. This fact formulates as

$$\exists \alpha > 0 : \ \alpha\mathbf{x} = \mathtt{P}\mathbf{X}, \tag{4.2}$$

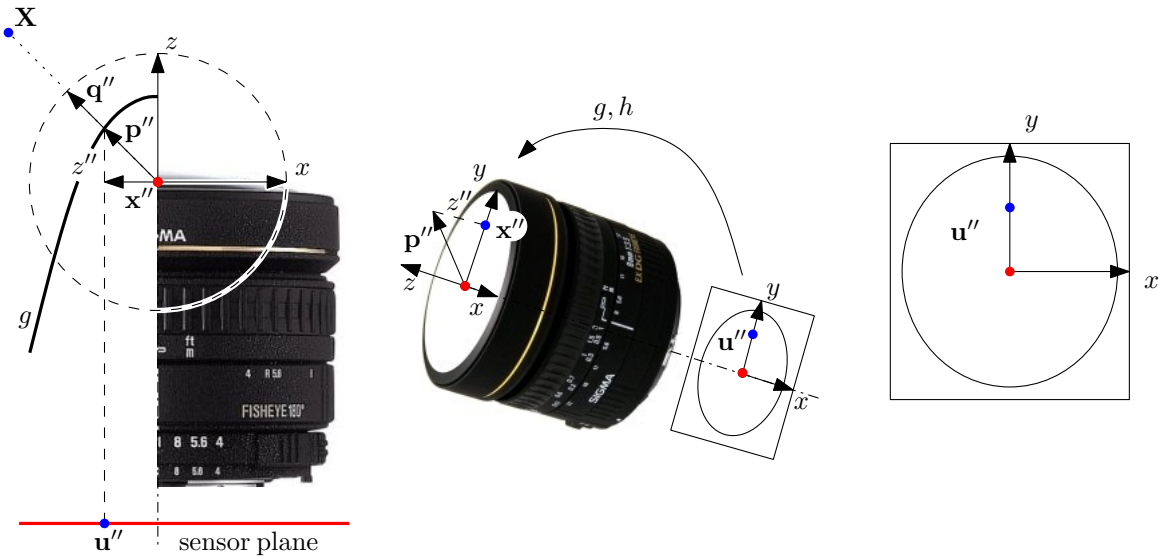


Figure 4.2: *Omnidirectional image formation*. (a–b) Mapping of a scene point $\mathbf{X}$ into a sensor plane point $\mathbf{u}''$ for a fish-eye lens. (c) The sensor plane with field of view circle. (Adopted from [125])

where $\mathbf{X}$, $\mathbf{x}$ and $\mathtt{P}$ are the same as in equation 4.1.

There are many camera models for omnidirectional cameras [59, 60, 124, 125]. Let us here briefly summarize a model for central omnidirectional cameras as described in [125]. In the next, it is assumed that the lenses and mirrors are

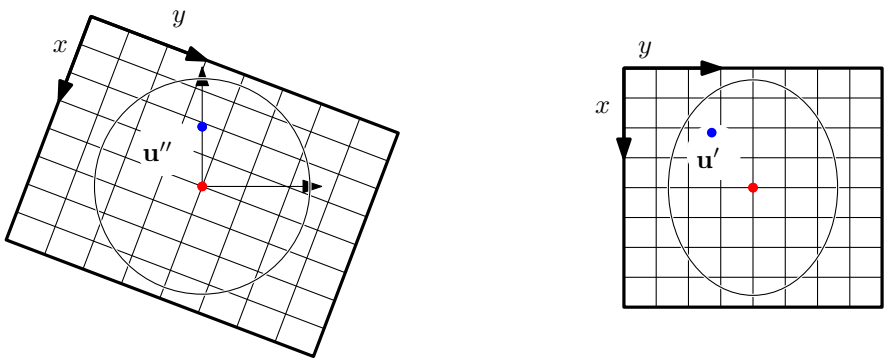
1. symmetric w.r.t. an axis and

2. the axis of the lens, or the mirror, is perpendicular to the sensor plane.

Figure 4.2 shows the process of image formation. Using the spherical model, the projection of a scene point $\mathbf{X}$ is represented by a unit vector $\mathbf{q}'' \in \mathbb{S}^3 = \{\mathbf{v} \in \mathbb{R}^3 \colon \|\mathbf{v}\| = 1\}$. From assumptions 1 and 2 one infers that there always exists a vector $\mathbf{p}'' = \left(\mathbf{x}''^\top, z''\right)^\top \in \mathbb{R}^3$ and a vector $\mathbf{u}'' \in \mathbb{R}^2$ in the sensor plane, for which the following holds:

$$\begin{aligned} \exists \alpha \in \mathbb{R}^+ : \mathbf{p}'' &= \alpha \mathbf{q}'', \\ \exists \beta \in \mathbb{R}^+ : \mathbf{x}'' &= \beta \mathbf{u}'', \qquad (4.3) \\ \mathbf{p}'' &= \begin{pmatrix} h\left(\|\mathbf{u}''\|, \mathbf{a}''\right) \mathbf{u}'' \\ g\left(\|\mathbf{u}''\|, \mathbf{a}''\right) \end{pmatrix}. \qquad (4.4) \end{aligned}$$

Functions $h, g \colon \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ are rotationally symmetric and depend on $\|\mathbf{u}''\|$, that is on the distance between the optical axis and $\mathbf{u}''$, and on a vector of parameters $\mathbf{a}'' \in \mathbb{R}^n$, where n is the number of parameters. The functions capture the type of a omnidirectional camera. The function g typically depend on the shape of the mirror for catadioptric omnidirectional cameras, the function h captures the projection of the camera. Note that β from Equation (4.3),



(a) (b)

Figure 4.3: *Digitization process*. (a–b) Affine transformation of the field of view circle. (Adopted from [125])

explicitly stating the collinearity of $\mathbf{u}''$ and $\mathbf{x}''$, equals $h\left(\left\|\mathbf{u}''\right\|, \mathbf{a}''\right)$ from Equation (4.4). Figures 4.2a–b show general relation between an image point $\mathbf{u}''$ and corresponding vector $\mathbf{p}''$.

The next step in the image formation process is digitization. The process of transforming sensor plane point $\mathbf{u}''$ into digital image point $\mathbf{u}'$ can be modeled by an affine transformation

$$\mathbf{u}'' = \mathtt{A}'\mathbf{u}' + \mathbf{t}', \tag{4.5}$$

where $\mathtt{A}' \in \mathbb{R}^{2\times 2}$ is a regular matrix, $\mathbf{t}' \in \mathbb{R}^2$ is a translation vector and $\mathbf{u}'$ is a point in a digital image. The digitization process is depicted in Figure 4.3a–b. By plugging the image formation process into Equation (4.2) the complete projection equation for omnidirectional cameras can be written as

$$\exists \alpha > 0 : \ \alpha \mathbf{p}'' = \alpha \begin{pmatrix} h\left(\left\|\mathbf{u}''\right\|, \mathbf{a}''\right) \mathbf{u}'' \\ g\left(\left\|\mathbf{u}''\right\|, \mathbf{a}''\right) \end{pmatrix} = \alpha \begin{pmatrix} h\left(\left\|\mathtt{A}'\mathbf{u}' + \mathbf{t}'\right\|, \mathbf{a}''\right) \left(\mathtt{A}'\mathbf{u}' + \mathbf{t}'\right) \\ g\left(\left\|\mathtt{A}'\mathbf{u}' + \mathbf{t}'\right\|, \mathbf{a}''\right) \end{pmatrix} = \mathtt{P}\mathbf{X}. \tag{4.6}$$

Since a camera model is always a compromise between correctness and computability, simple definitions of g, h are preferred, with back-projection in mind.

4.2 Camera Calibration

By camera calibration we understand the process of recovering intrinsic camera parameters of an appropriate camera model for a given real camera. An abundance of literature covering this topic exists. The historical results on camera calibration come from the photogrammetric community [20, 49]. An overview of the topic can be found in [123] and a modern treatment from the computer vision standpoint in [71]. Practically, we can divide the camera calibration techniques into two categories:

1. *Photogrammetric calibration*, when a camera with fixed internal parameters observes a known rigid 3D object and

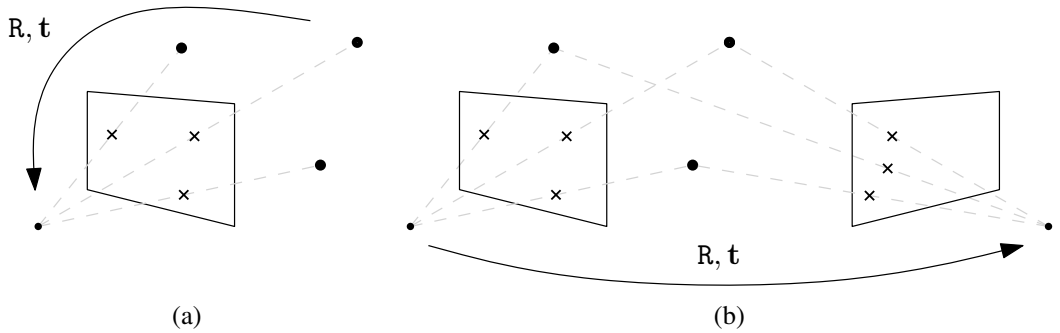


Figure 4.4: *Absolute and relative camera poses*.

2. *Self-calibration*, when camera is to be calibrated from images of an unknown rigid scene.

The input data to the problem of photogrammetric calibration is a collection of 3D-2D correspondences. The classical calibration algorithm is due to Tsai [181]. Assuming that the known 3D calibration object is non-planar, the DLT algorithm [71] can be used to compute projection matrix $\mathtt{P}$ and matrix $\mathtt{K}$ can be recovered by its subsequent factorization. The DLT algorithm should be always followed by nonlinear optimization to refine the result with respect to some geometrically meaningful objective function. For omnidirectional camera calibration, many researchers used hollow calibration devices with walls covered by a known structure [19, 11]. A method similar to the DLT algorithm was developed in [13] also for central catadioptric cameras.

Many researchers were concerned with planar camera calibration—planar calibration targets are easier to manufacture and handle. For pinhole camera model, methods [171, 189] provide closed form solutions. Again, they should be followed by a nonlinear optimization step.

Of course, other calibration methods exist, *e.g.*, using one-dimensional objects [190], collimators [175, 38], or images of individual geometric primitives, *e.g.*, [136, 83, 6, 34, 138].

Camera calibration can be also performed without a known calibration target—so called camera self-calibration. A set of 2D-2D correspondences in images of a unknown rigid scene sufficiently constraints the calibration problem [121, 114]. Of course, self-calibration methods for omnidirectional cameras also exist [124, 12, 39, 158].

4.3 Absolute Pose Estimation

The problem of absolute pose estimation is to estimate the extrinsic camera parameters $\mathtt{R} \in SO(3)$, $\mathtt{t} \in \mathbb{R}^3$ given a known object in the world coordinate frame, see Figure 4.4a. It is also referred to as camera resectioning problem. Usually, the problem is solved from point or line correspondences. In case of point correspondences, at least 3 must be given. In this case the problem has up to four solutions [53]. Approaches combining intrinsic parameters also exist, for example given four 3D-2D correspondences, focal length can be recovered [179, 24] as

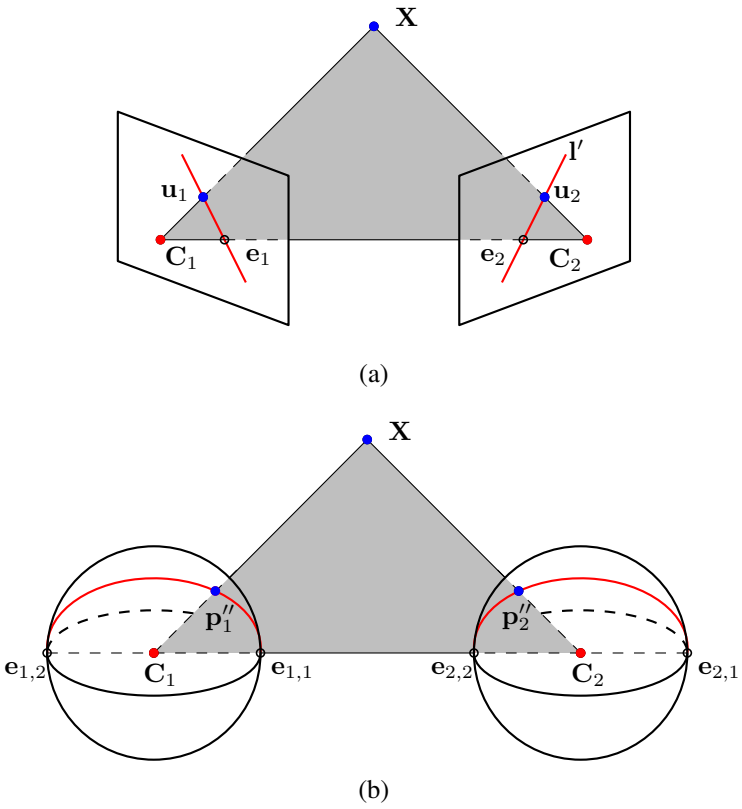


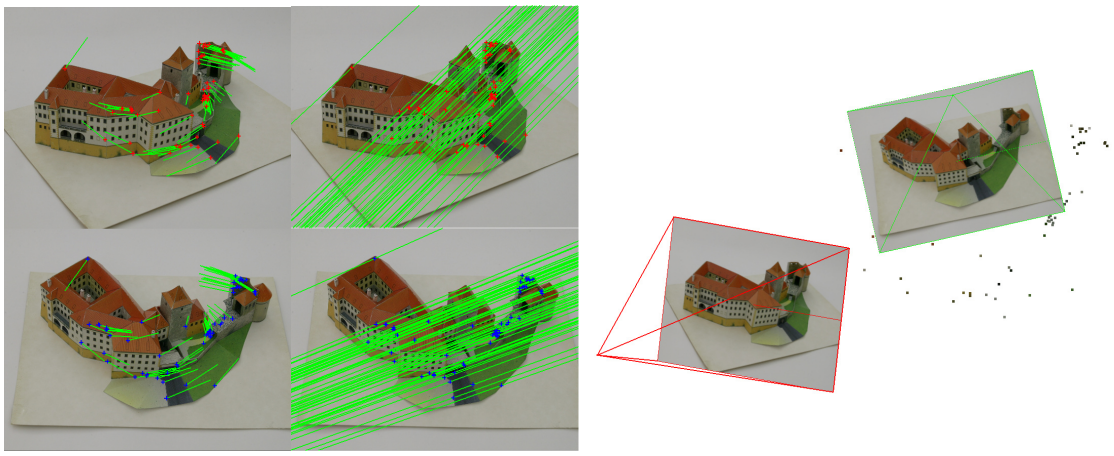
Figure 4.5: *Epipolar geometry*. (a) Epipolar geometry of standard perspective cameras. (b) Epipolar geometry of omnidirectional central cameras. (Adopted from [125]).

well as a radial distortion parameter [84, 25]. These minimal solutions are combined with RANSAC loop [53] to estimate the maximal inlier set. This step is usually followed by a non-linear optimization refinement of the solution found on the inlier set. The non-linear optimization step is in the context of geometric computer vision known as bundle adjustment.

Methods minimizing over all available data at once also exist. For example, methods [112, 101] deliver fast and accurate results, albeit suboptimal. Methods by Zhang [188] and Hartley and Kahl [68] solved the problem globally optimally with respect to L_∞-norm, Agarwal *et al.* [87] solved it for the uncalibrated case with respect to L_2-norm. [145], the authors used polynomial optimization methods to derive globally optimal L_2-norm solution with respect to the object space error.

4.4 Epipolar geometry

The epipolar geometry is a central notion of the geometry of two views. It is motivated by stereo matching, *i.e.*, by searching for the projections of a scene point $\mathbf{X} \in \mathbb{R}^3$ taken by a camera in two different poses as $\mathbf{u}_1 \in \mathbb{R}^3$ in the first view and as $\mathbf{u}_2 \in \mathbb{R}^3$ in the second, see Figure 4.5. Let us start with the calibrated case, *i.e.*, with the camera calibration matrix $\mathtt{K} = \mathtt{I}$. Now, let us suppose that the first pose is the canonic pose. The scene point $\mathbf{X}$ is



(a) (b) (c)

Figure 4.6: *Example of the epipolar geometry of two images.* (a) Several image correspondences found by an automatic method and their apparent movement between the two views. (b) Image correspondences and their respective epipolar lines. Notice that the epipoles are not visible in the images. (c) 3D points as triangulated from the image correspondences and the relative poses of the images recovered from the fundamental matrix $\mathtt{F}$. (Created using VisualSFM [3])

projected by the pinhole camera model as

$$\mathbf{u}_1 = \begin{bmatrix} \mathtt{I} & \mathbf{0} \end{bmatrix} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}.$$

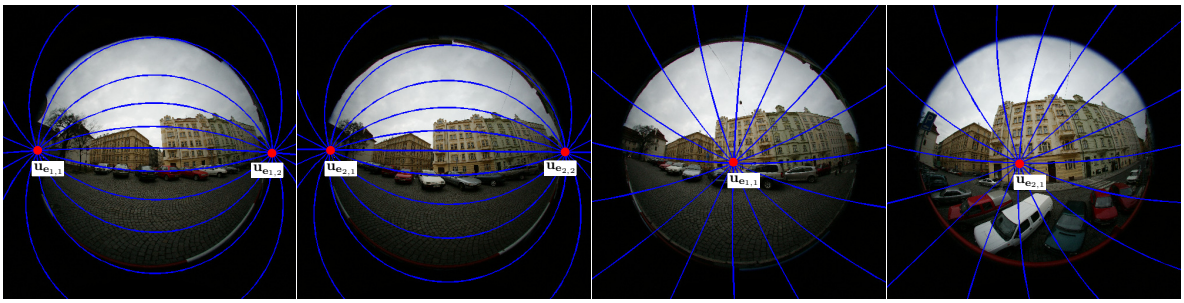
Further, suppose that $\mathtt{R} \in SO(3)$, $\mathbf{t} \in \mathbb{R}^3$ describe the second camera pose. The scene point is now projected as

$$\mathbf{u}_2 = \begin{bmatrix} \mathtt{R} & \mathbf{t} \end{bmatrix} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}.$$

Vectors $\mathbf{u}_1$ and $\mathbf{u}_2$ in this configuration form an *image correspondence*, $\mathbf{u}_1 \leftrightarrow \mathbf{u}_2$. From the fundamental properties of central projection follows that the centers of the cameras $\mathbf{C}_1$, $\mathbf{C}_2$ and the vectors $\mathtt{R}\mathbf{u}_1$, $\mathbf{u}_2$, and $\mathbf{X}$ are coplanar. This fact is referred to as the *epipolar constraint*. Scene points together with baseline $\overline{\mathbf{C}_1\mathbf{C}_2}$ create a pencil of planes called *epipolar planes*. All epipolar planes intersect the projective planes of the two views in straight lines—*epipolar lines*. Epipolar lines again form pencils of lines in their respective projective planes, that intersect in two respective points $\mathbf{e}_1$, $\mathbf{e}_2$ called *epipoles*. Epipoles can be equivalently described as projections of the camera centers into the image planes of the opposite view. An example of the epipolar geometry of two perspective cameras is given in Figure 4.5a. The fact that the vectors $\mathtt{R}\mathbf{u}_1$, $\mathbf{u}_2$, and $\mathbf{X}$ are coplanar can be expressed using cross product as

$$0 = \mathbf{u}_2^\top (\mathbf{t} \times \mathtt{R}\mathbf{u}_1) = \mathbf{u}_2^\top ([\mathbf{t}]_\times \mathtt{R})\mathbf{u}_1 = \mathbf{u}_2^\top \mathtt{E}\mathbf{u}_1 = 0.$$

The matrix $\mathtt{E} = [\mathbf{t}]_\times \mathtt{R}$ is called the *essential matrix*.



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

Figure 4.7: *Example of the epipolar geometry of two omnidirectional images*. Two image pairs acquired by a fish-eye lens with field of view of 180° with respective epipolar geometries. Images were transformed so it would appear as if they had been acquired by s para-catadioptric camera in order to transform the epipolar curves into circles. (a–b) An image pair resulting from a lateral move of the camera. Both epipoles are visible. (c–d) An image pair resulting from a forward move of the camera. Only one epipole is visible.

In the case the calibration matrix $\mathtt{K}$ is not an identity, the projections of the point $\mathbf{X}$ must be modified as

$$\begin{aligned} \mathbf{u}_1 &= \mathtt{K}\begin{bmatrix} \mathtt{I} & \mathbf{0} \end{bmatrix}\begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}, \\ \mathbf{u}_2 &= \mathtt{K}\begin{bmatrix} \mathtt{R} & \mathbf{t} \end{bmatrix}\begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}, \end{aligned}$$

and the epipolar constraint now looks as follows

$$(\mathtt{K}^{-1}\mathbf{u}_2)^\top \left(\mathbf{t} \times \mathtt{R}(\mathtt{K}^{-1}\mathbf{u}_1)\right) = \mathbf{u}_2^\top (\mathtt{K}^{-\top}[\mathbf{t}]_\times \mathtt{R}\mathtt{K}^{-1})\mathbf{u}_1 = \mathbf{u}_2^\top \mathtt{F}\mathbf{u}_1 = 0.$$

The matrix $\mathtt{F} = \mathtt{K}^{-\top}[\mathbf{t}]_\times \mathtt{R}\mathtt{K}^{-1}$ is called the *fundamental matrix*. The fundamental matrix realizes the mapping $\mathbf{u}_1 \mapsto \mathbf{l}'$, *i.e.*, it maps the image points from the first image to the epipolar lines in the second image as

$$\mathbf{l}' = \mathtt{F}\mathbf{u}_1,$$

from which follows that $\mathrm{rank}(\mathtt{F}) = 2$. Figure 4.6 shows an example of the epipolar geometry for two views of a paper model of the Daliborka Tower. Historically, the essential matrix was introduced by Longuet-Higgins [107] before the fundamental matrix. The above form of fundamental matrix was introduced by Faugeras [50] and Hartley [72] in 1992.

An analogy to the epipolar geometry of central perspective cameras can be formulated for central omnidirectional cameras. The difference between directional and omnidirectional cameras is the shape of the retinas as well as the distinguishability of the rays orientations. The pencil of planes intersect the spherical retinas of the spherical model in great circles, which are projected into sensor plane as *epipolar curves*, intersecting the $\overline{\mathbf{C}_1\mathbf{C}_2}$ baseline in *two epipoles*, $\mathbf{e}_{1,1}, \mathbf{e}_{1,2}$ in the first view, $\mathbf{e}_{2,1}, \mathbf{e}_{2,2}$ in the second view, see Figure 4.5b. The epipolar curves are conics for quadric catadioptric cameras [172] and more general curves

for fish-eye lenses [125]. In the case of omnidirectional cameras, the epipolar constraint can be formulated with the back-projected vectors $\mathbf{p}_1''$ and $\mathbf{p}_2''$ only, see Figure 4.5b, as

$$\mathbf{p}_2''^{\top} \mathtt{E} \mathbf{p}_1'' = 0, \tag{4.7}$$

where $\mathtt{E}$ is again called the essential matrix. Figures 4.7a–d show examples of two image pairs with denoted epipolar geometries.

4.5 Relative Pose Estimation

The problem of relative pose estimation, sometimes also known as motion or egomotion estimation is the problem of estimating the relative transformation $\mathtt{R} \in SO(3)$, $\mathbf{t} \in \mathbb{R}^3$ between the coordinate frames connected with cameras in two different absolute poses, see Figure 4.4b, using only image matches between the images. It is closely connected to the problem of estimation of the epipolar geometry, *i.e.*, matrices $\mathtt{F}$ and $\mathtt{E}$, since both encode the relative transformation and both can be factorized to extract it.

Again, relative pose estimation problem can be formulated as a minimal problem that is used inside a RANSAC style loop. It has been known for a long time that 5 image point correspondences are enough to estimate the relative pose [92] and that it leads up to 10 theoretical solutions. The first practical 5-point minimal solution is due to Nistér [130]; other solution followed [103, 93, 95]. In [102, 28, 95, 93, 165], it has been shown that by adding one more point correspondence focal length can also be estimated. The RANSAC loop should always be followed by the bundle adjustment procedure.

Global solution in L_∞-norm was proposed by Hartley and Kahl [69]. It is based on the branch-and-bound search over the space of all rotations.

4.6 Bundle Adjustment

The non-linear optimization of all unknowns is in the context of geometry of vision known as bundle adjustment. The unknowns may include intrinsic and extrinsic camera parameters as well as scene point positions. It is also implied that a geometrically meaningful objective function is minimized, *e.g.*, reprojection or object space error. The minimization method of choice is often the Levenberg–Marquardt algorithm [119]. The work by Triggs *et al.* [180] provides an excellent reference on bundle adjustment.

The underlying geometrical structure of the bundle adjustment problem—relatively less cameras compared to the number of points with the point typically visible only in a subset of cameras—reflect in the relative sparsity of the problem. This sparsity can be exploited via the Shur complement trick [180], as it is in the case of SBA, sparse bundle adjustment library by Lourakis and Argyros [109]. The sparsity structure can also be exploited via Cholesky factorization [35, 108] or conjugate gradient methods [29, 30, 5].

4.7 Structure-from-Motion

Given a sequence or a set of images with pairwise image correspondences, one may want to recover not only the pairwise relative pose information, but also the relative poses of all cameras (as well as the intrinsic parameters). This problem is known as structure-from-motion (SfM) [174, 71]. Despite its easy formulation, this optimization problem is inherently highly non-linear and no closed-form solution is known to the date. The work [176] by Tomasi and Kanade can be cited as one of the SfM methods. It applies the factorization technique on affine cameras. This method was extended by Sturm *et al.* [170] for perspective cameras.

Even though global approaches to SfM exist [139, 120], most of the nowadays SfM methods are based on incremental reconstruction. These methods start with a small seed reconstruction—two or three images—and add cameras and points to grow the initial reconstruction [14, 15, 153]. The incremental adding of cameras is based on the techniques of absolute and relative camera poses recovery. The fundamental role in the incremental SfM is that of bundle adjustment, which has to be performed periodically after adding a certain number of cameras and points. Without this step, the chances of recovering a realistic results are slim to none [154]. Even when using the state-of-the-art approaches to bundle adjustment, this step can become the bottle neck of the whole SfM procedure. Several methods were proposed to decrease the size of the bundle adjustment problems handled. In [155], Snavely *et al.* proposed the concept of skeletal graph covering the set of cameras. Similarly, Li *et al.* [104] proposed to use iconic scene graphs. In parallel, divide-and-conquer approaches that divide the SfM problem into many smaller subproblems were developed [54, 131, 157, 55].

4.8 Conclusion

This chapter reviewed the state of the art of the fundamental concepts of geometric computer vision pertinent to development of the presented robot-camera calibration methods. In the latter chapters, we will assume all cameras to be calibrated and that we are able to back-project the image points, *i.e.*, to recover the 3D directions of the rays that projected into the respective image points.

5 Camera-Robot Calibration

I'm completely operational and all my circuits are functioning normally.

– HAL 9000

The need to relate measurements made by a camera to a different known coordinate system arises in many engineering applications. Historically, it appeared for the first time in the connection with cameras mounted on robotic systems. This problem is commonly known as *hand-eye calibration*. The hand-eye calibration problem arises also in seemingly unrelated fields ranging from medical applications such as ultrasound [98, 17, 184, 16] and endoscopy [144, 185, 143, 117] to automotive industry [140]. However, in this work we will be concerned with the robotic hand-eye—sometimes also known as eye-in-hand—systems. The presented results can be easily extended to other applications as well.

The classical robotic hand-eye setup is shown in Figure 5.1. It shows a Mitsubishi MELFA-RV 6S serial manipulator equipped with a Canon 350D DSLR camera mounted on the robotic end-effector (gripper); Figure 5.2 depicts the situation schematically. The problem of hand-eye calibration is the problem of recovering the rigid transformation that connects a camera and robotic gripper coordinate systems. It is customary to express this transformation as a 4×4 matrix $\mathtt{X} \in SE(3)$,

$$\mathtt{X} = \begin{pmatrix} \mathtt{R}_\mathtt{X} & \mathbf{t}_\mathtt{X} \\ \mathbf{0}^\top & 1 \end{pmatrix},$$

see Figure 5.2, where $\mathtt{R}_\mathtt{X} \in SO(3)$ is a 3×3 orthonormal matrix describing the rotational part of the transformation and $\mathbf{t}_\mathtt{X} \in \mathbb{R}^3$ is a 3-dimensional vector describing the translation.

5.1 Hand-Eye Calibration

Let us suppose that a hand-eye robotic system has been manipulated into two distinct poses, see Figure 5.2. Let us denote the transformation from the world coordinate system to the camera coordinate system in the first pose of the rig as $\mathtt{A}'_1$ and the transformation in the second pose as $\mathtt{A}'_2$. Now, we can express the camera's relative motion (pose) from the first pose the second one as

$$\mathtt{A} = \begin{pmatrix} \mathtt{R}_\mathtt{A} & \mathbf{t}_\mathtt{A} \\ \mathbf{0}^\top & 1 \end{pmatrix} = \begin{pmatrix} \mathtt{R}_{\mathtt{A}'_2} & \mathbf{t}_{\mathtt{A}'_2} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} \mathtt{R}_{\mathtt{A}'_1} & \mathbf{t}_{\mathtt{A}'_1} \\ \mathbf{0}^\top & 1 \end{pmatrix}^{-1} = \mathtt{A}'_2 \mathtt{A}'^{-1}_1,$$

where $\mathtt{R}_\mathtt{A} \in SO(3)$ is a 3×3 rotation matrix and $\mathbf{t}_\mathtt{A} \in \mathbb{R}^3$ is a 3-dimensional translational vector. Analogically, relative motion (pose) of the robotic end-effector can be described as

$$\mathtt{B} = \begin{pmatrix} \mathtt{R}_\mathtt{B} & \mathbf{t}_\mathtt{B} \\ \mathbf{0}^\top & 1 \end{pmatrix} = \begin{pmatrix} \mathtt{R}_{\mathtt{B}'_2} & \mathbf{t}_{\mathtt{B}'_2} \\ \mathbf{0}^\top & 1 \end{pmatrix}^{-1} \begin{pmatrix} \mathtt{R}_{\mathtt{B}'_1} & \mathbf{t}_{\mathtt{B}'_1} \\ \mathbf{0}^\top & 1 \end{pmatrix} = \mathtt{B}'^{-1}_2 \mathtt{B}'_1,$$

(a) (b)

Figure 5.1: *Hand-eye calibration robotic setup.* (a) A Mitsubishi MELFA-RV 6S serial manipulator equipped with a Canon 350D DSLR camera. (b) Detail of the hand-eye setup.

with $\mathtt{B}'_1$, $\mathtt{B}'_2$ being the respective transformations from the end-effector's coordinate system to the robot base coordinate system. In the rest of this work, we will call $\mathtt{A}'_i$ and $\mathtt{B}'_i$ absolute camera and robot poses, respectively. Assuming we have measured two sets of absolute poses $\mathtt{A}'_i$ and $\mathtt{B}'_i$, we can compute the relative poses$\mathtt{A}$ and $\mathtt{B}$ and the problem of hand-eye calibration can be expressed analytically as the solution to the following matrix equation:

$$\mathtt{AX} = \mathtt{XB}. \tag{5.1}$$

The earliest solution strategies can by found in [182, 183, 149, 37, 186]. All of these authors realized that Equation 5.1 forms an underdetermined system and that two absolute robot poses are not enough to uniquely determine the transformation $\mathtt{X}$. In [149], Shiu and Ahmad showed that at least two relative poses with non-parallel rotational axes are needed. In practice, the robot is manipulated into m absolute poses, giving rise to the set of matrices $\mathtt{A}'_i, \mathtt{B}'_i$, $i = 1, \ldots, m$. These absolute poses are then combined into n relative poses $\mathtt{A}_i, \mathtt{B}_i$, $i = 1, \ldots, n$. Which pairs of the n absolute poses are combined to produce the relative poses depends on the application and ultimately on the user's decision. For example, all possible combinations of the absolute poses lead to $n = \binom{m}{2}$ relative poses. Unless the absolute poses can be measured with perfect accuracy—we get a noisy system of matrix equations

$$\begin{aligned} \mathtt{A}_1\mathtt{X} &= \mathtt{XB}_1, \\ \mathtt{A}_2\mathtt{X} &= \mathtt{XB}_2, \\ &\vdots \\ \mathtt{A}_n\mathtt{X} &= \mathtt{XB}_n. \end{aligned} \tag{5.2}$$

Since System 5.2 can be expressed in more detail as

$$\begin{pmatrix} \mathtt{R}_{\mathtt{A}_i} & \mathbf{t}_{\mathtt{A}_i} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} \mathtt{R}_{\mathtt{X}} & \mathbf{t}_{\mathtt{X}} \\ \mathbf{0}^\top & 1 \end{pmatrix} = \begin{pmatrix} \mathtt{R}_{\mathtt{X}} & \mathbf{t}_{\mathtt{X}} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} \mathtt{R}_{\mathtt{B}_i} & \mathbf{t}_{\mathtt{B}_i} \\ \mathbf{0}^\top & 1 \end{pmatrix}, \; i = 1, \ldots, n \tag{5.3}$$

it can be further decomposed into rotational matrix equations and translational vector equa-

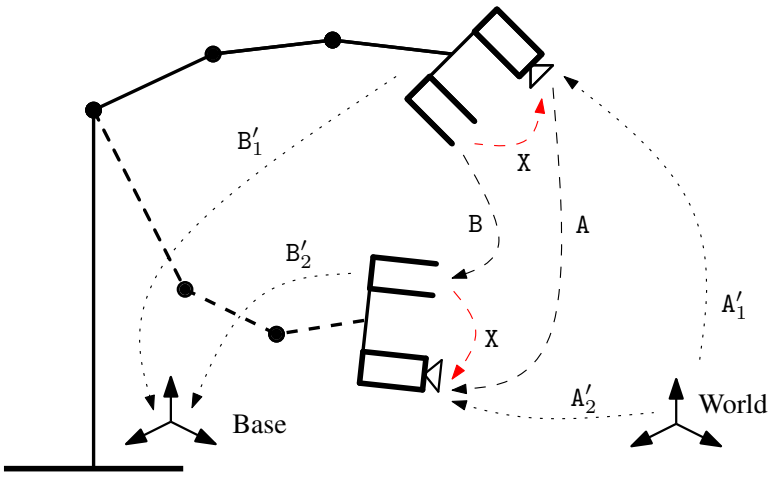


Figure 5.2: *Schematics of hand-eye calibration.* A Robotic arm in two different poses.

tions

$$\mathtt{R}_{\mathtt{A}_i}\mathtt{R}_{\mathtt{X}} = \mathtt{R}_{\mathtt{X}}\mathtt{R}_{\mathtt{B}_i},\ i = 1, \ldots, n \tag{5.4}$$

$$\mathtt{R}_{\mathtt{A}_i}\mathbf{t}_{\mathtt{X}} + \mathbf{t}_{\mathtt{A}_i} = \mathtt{R}_{\mathtt{X}}\mathbf{t}_{\mathtt{B}_i} + \mathbf{t}_{\mathtt{X}},\ i = 1, \ldots, n. \tag{5.5}$$

Notice that Equations 5.4 do not depend on the unknown translation $\mathbf{t}_{\mathtt{X}}$. Once rotation $\mathtt{R}_{\mathtt{X}}$ is known, Equations 5.5 are simply a system of linear equations in $\mathbf{t}_{\mathtt{X}}$ and the translation can be easily determined using the tools of linear algebra.

Work [148] contains a concise review of the classical solutions to hand-eye calibration. The classical solution strategies of the hand-eye calibration problem can be divided into tree categories:

1. *decomposed closed-form solutions* that explicitly use the decomposition of Equation 5.2 into Equations 5.4 and 5.5,
2. *simultaneous closed-form solutions,* and
3. *simultaneous iterative solutions* that employ techniques of mathematical optimization.

5.1.1 Decomposed Solutions

In [149], Shiu and Ahmad proposed the first solution to the hand-eye calibration problem formulated as Equation 5.1. They used the angle-axis parametrization of the group of rotations $SO(3)$, see Section 3.4.2. The authors observed that if $\exp\left[\boldsymbol{\alpha}\right]_\times = \mathtt{R}_{\mathtt{A}}$ and $\exp\left[\boldsymbol{\beta}\right]_\times = \mathtt{R}_{\mathtt{B}}$, then

$$\boldsymbol{\alpha} = \mathtt{R}_{\mathtt{X}}\boldsymbol{\beta}. \tag{5.6}$$

A general solution to Equation 5.6 can be found as a rotation about axis perpendicular to both $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as

$$\mathtt{R}_{\mathtt{X}_i} = \exp\left[a_i\boldsymbol{\alpha}\right]_\times \exp\left[r\boldsymbol{\omega}\right]_\times,$$

where $\boldsymbol{\omega} = \boldsymbol{\alpha} \times \boldsymbol{\beta}$ and $r = \text{atan2}(|\boldsymbol{\alpha} \times \boldsymbol{\beta}|, \boldsymbol{\alpha}^\top \boldsymbol{\beta})$. Considering two relative movement, we can write

$$\begin{aligned} \mathtt{R}_{\mathtt{X}_1} &= \mathtt{R}_{\mathtt{X}_2}, \\ \exp\left[a_1\boldsymbol{\alpha}\right]_\times \exp\left[r_1\boldsymbol{\omega}_1\right]_\times &= \exp\left[a_2\boldsymbol{\alpha}\right]_\times \exp\left[r_2\boldsymbol{\omega}_2\right]_\times . \end{aligned}$$

This leads to a system of 9 linear equations in four unknowns $\sin a_1$, $\cos a_1$, $\sin a_2$, $\cos a_2$. Once the rotation $\mathtt{R}_\mathtt{X}$ is known, the translational part is recovered from the linear system 5.5.

Tsai and Lenz [183] again used the angle-axis representation. To recover $\mathtt{R}_\mathtt{X}$, they suggested the following linear least square problem

$$\left[\boldsymbol{\alpha}'_i + \boldsymbol{\beta}'_i\right]_\times \boldsymbol{\omega}' = \boldsymbol{\alpha}'_i - \boldsymbol{\beta}'_i,\ i = 1, \ldots, n,$$

where

$$\begin{aligned} \boldsymbol{\alpha}'_i &= \frac{2\sin\left(\left\|\frac{\boldsymbol{\alpha}_i}{2}\right\|\right)}{\|\boldsymbol{\alpha}_i\|}\boldsymbol{\alpha}_i, \\ \boldsymbol{\beta}'_i &= \frac{2\sin\left(\left\|\frac{\boldsymbol{\beta}_i}{2}\right\|\right)}{\|\boldsymbol{\beta}_i\|}\boldsymbol{\beta}_i, \end{aligned}$$

and n is the number of relative poses. Once $\boldsymbol{\omega}'$ is recovered, $\mathtt{R}_\mathtt{X}$ is computed as

$$\mathtt{R}_\mathtt{X} = \exp\left[\frac{2\tan^{-1}\|\boldsymbol{\omega}'\|}{\|\boldsymbol{\omega}'\|}\boldsymbol{\omega}'\right]_\times .$$

Again, the translation $\mathtt{t}_\mathtt{X}$ is recovered from the linear system 5.5.

Wang [186] suggested three methods that basically correspond to the solution [183]. He compared his best method to the methods [149, 183] and concluded that method by Tsai and Lenz [183] performed best.

In [134], based on the fact that $\boldsymbol{\alpha} = \mathtt{R}_\mathtt{X}\boldsymbol{\beta}$, Park and Martin suggested a different minimization problem to recover $\mathtt{R}_\mathtt{X}$:

$$\min_{\mathtt{R}_\mathtt{X}} \sum_{i=1}^{n} \left\|\mathtt{R}_\mathtt{X}\boldsymbol{\beta}_i - \boldsymbol{\alpha}_i\right\|^2 . \tag{5.7}$$

Based on a matrix

$$\mathtt{M} = \sum_{i=1}^{n} \boldsymbol{\beta}_i \boldsymbol{\alpha}_i^\top,$$

they also provided a closed form solution

$$\mathtt{R}_\mathtt{X} = \sqrt{\mathtt{M}^\top\mathtt{M}}\,\mathtt{M}^\top.$$

Chou and Kamel [37] proposed to solve the hand-eye calibration problem using quaternion representation of rotation, see Section 3.4.3. They noticed that Equation 5.4 can be written in the quaternion notation as

$$\mathbf{q}_\mathtt{A} * \mathbf{q}_\mathtt{X} = \mathbf{q}_\mathtt{X} * \mathbf{q}_\mathtt{B},$$

and used the matrix form of quaternion multiplication, see Equations 3.6 and 3.7, to construct the following linear system:

$$\mathtt{M}(\mathbf{q}_\mathtt{A})\mathbf{q}_\mathtt{X} - \bar{\mathtt{M}}(\mathbf{q}_\mathtt{B})\mathbf{q}_\mathtt{X} = \mathbf{0}. \tag{5.8}$$

The authors used singular value decomposition (SVD) [63] to solve System 5.8 with additional adjustments to obtain solution satisfying the unity condition $||\mathbf{q}_\mathtt{X}|| = 1$.

In [81], Houraud and Dornaika proposed to recover $\mathtt{R}_\mathtt{X}$ as a minimizer of the same objective function as [134]—Equation 5.7—this time, however, parametrized using quaternions:

$$\min_{\mathbf{q}_\mathtt{X}} \sum_{i=1}^{n} \left\| \mathbf{q}_\mathtt{X} * (0, \boldsymbol{\beta}_i)^\top * \mathbf{q}_\mathtt{X}^* - (0, \boldsymbol{\alpha}_i)^\top \right\|^2.$$

They provided a closed form solution as the eigenvector associated with the smallest positive eigenvalue of the matrix

$$\mathtt{A} = \sum_{i=1}^{n} \left(\bar{\mathtt{M}}((0, \boldsymbol{\beta}_i)^\top) - \mathtt{M}((0, \boldsymbol{\alpha}_i)^\top\right)^\top \left((\bar{\mathtt{M}}((0, \boldsymbol{\beta}_i)^\top) - \mathtt{M}((0, \boldsymbol{\alpha}_i)^\top\right).$$

A different approach to the solution of Equation 5.4 was presented by Liang and Mao in [105]. They proposed to use the Kronecker product together with vec operator, see Definitions 3.3 and 3.2, respectively, to rewrite Equation 5.4 as a linear system

$$(\mathtt{R}_{\mathtt{A}_i} \otimes \mathtt{I} - \mathtt{I} \otimes \mathtt{R}_{\mathtt{B}_i}^\top)\mathrm{vec}(\mathtt{R}_\mathtt{X}) = \mathtt{A}\mathbf{x} = \mathbf{0}.$$

Once this system is solved using SVD, the vector $\mathbf{x}$ is reorganized back into a 3×3 matrix as $\mathtt{R}' = \mathrm{vec}^{-1}(\mathbf{x})$. Since there are no additional constraints on the vector $\mathbf{x}$, matrix $\mathtt{R}'$ will generally not be a rotation matrix. The authors also showed how to obtain $\mathtt{R}_\mathtt{X} \in SO(3)$, such that $||\mathtt{R}' - \mathtt{R}_\mathtt{X}||_F$ is minimal using SVD and a determinant sign test.

5.1.2 Simultaneous closed-form solutions

In [33], Chen employed the screw motion theory, see Section 3.5, to investigate the necessary and sufficient conditions for the solutions of Equation 5.1. The author showed that for well-defined screws, the necessary and sufficient condition for a unique solution is that the screw axes of two robot motions are either skew or intersecting. Moreover, that even for undefined and ambiguous screws a partial solution or even a complete solution may be recovered. The author concluded that the rotational and translational parts of the transformation $\mathtt{X}$ should not be decoupled, because otherwise the generality and efficacy of the resulting algorithm would be negatively affected.

In [43, 44], Daniilidis and Bayro-Corrochano used dual quaternions to represent screws and formulated Equation 5.1 as a linear system

$$\mathtt{T} \begin{pmatrix} \mathbf{q} \\ \mathbf{q}' \end{pmatrix} = \mathbf{0}, \tag{5.9}$$

where $(\mathbf{q}, \mathbf{q}')^\top \in \mathbb{H}$ is the dual quaternion representing the hand-eye transformation $\mathtt{X}$ and matrix $\mathtt{T}$ is a $6n \times 8$ matrix

$$\mathtt{T} = \begin{pmatrix} \mathtt{S}_1^\top & \mathtt{S}_2^\top & \cdots & \mathtt{S}_n \end{pmatrix}^\top. \tag{5.10}$$

Let $\hat{\mathbf{a}}_i = \mathbf{a}_i + \epsilon\mathbf{a}'_i$, $\hat{\mathbf{b}}_i = \mathbf{b}_i + \epsilon\mathbf{b}'_i$ be the unit dual quaternions representing the motions $\mathtt{A}_i$ and $\mathtt{B}_i$, respectively. Matrices $\mathtt{S}_i$ are constructed using the quaternion parts as

$$\mathtt{S}_i = \begin{pmatrix} \bar{\mathbf{a}}_i - \bar{\mathbf{b}}_i & \left[\bar{\mathbf{a}}_i + \bar{\mathbf{b}}_i\right]_\times & \mathbf{0}_3 & \mathtt{0}_{3\times 3} \\ \bar{\mathbf{a}}'_i - \bar{\mathbf{b}}'_i & \left[\bar{\mathbf{a}}'_i + \bar{\mathbf{b}}'_i\right]_\times & \bar{\mathbf{a}}_i - \bar{\mathbf{b}}_i & \left[\bar{\mathbf{a}}_i + \bar{\mathbf{b}}\right]_\times \end{pmatrix}, \tag{5.11}$$

where the barred vectors represent the imaginary parts of the respective quaternions. Finally, $(\mathbf{q}, \mathbf{q}')^\top$ is found as an intersection of the null space of System 5.9 and the set of unit dual quaternions. The authors compared the method to the method of Tsai and Lenz [183] and to the nonlinear method of Houraud and Dornaika [81] and concluded that the dual quaternion formulation outperforms the former methods. In [192], Zhao and Liu presented a similar method derived directly from the geometry of the screw motion.

In [8], Andreff *et al*. proposed to linearize Equation 5.1 using Kronecker product as

$$\mathtt{T}\begin{pmatrix} \mathrm{vec}(\mathtt{R}_\mathtt{X}) \\ \mathbf{t}_\mathtt{X} \end{pmatrix} = \mathtt{T}\begin{pmatrix} \mathbf{r}_\mathtt{X} \\ \mathbf{t}_\mathtt{X} \end{pmatrix} = \begin{pmatrix} \mathbf{0}_9 \\ \mathbf{t}_{\mathtt{A}_i} \end{pmatrix},$$

where the matrix $\mathtt{T}$ has the same block structure as in Equation 5.10. Here, the matrices $\mathtt{S}_i$ are constructed as

$$\mathtt{S}_i = \begin{pmatrix} \mathtt{I} - \mathtt{R}_{\mathtt{A}_i} \otimes \mathtt{R}_{\mathtt{B}_i} & \mathtt{0}_{9\times 3} \\ \mathtt{I} \otimes \mathbf{t}_{\mathtt{B}_i}^\top & \mathtt{I} - \mathtt{R}_{\mathtt{A}_i} \end{pmatrix}. \tag{5.12}$$

This system is solved using SVD. As is the case of method [105], such a solution will generally not yield a rotation matrix and it needs to be further orthogonalized as

$$\mathtt{R}_\mathtt{X} = \frac{\mathrm{sign}(\det(\mathrm{vec}^{-1}(\mathbf{r}_\mathtt{X})))}{|\det(\mathrm{vec}^{-1}(\mathbf{r}_\mathtt{X}))|^{\frac{1}{3}}}\mathrm{vec}^{-1}(\mathbf{r}_\mathtt{X}). \tag{5.13}$$

However, the authors observe that the proposed solution is not independent of the physical units used for the translation and that due to the orthogonalization step it is improbable to find the corresponding correction on the translation. They conclude that a decomposed solution, similar to [105], may provide more accurate results.

5.1.3 Simultaneous Iterative Solutions

Zhuang and Shiu [194] proposed an iterative non-linear method to minimize function

$$\sum_{i=1}^{n} \left\|\mathtt{A}_i\mathtt{X} - \mathtt{X}\mathtt{B}_i\right\|^2$$

to simultaneously estimate the rotational and translational parts of $\mathtt{X}$. As a part of [81], Horaud and Dornaika also proposed a simultaneous iterative method based of quaternions and Levenberg-Marquardt non-linear optimization [119]. They observed that the method performed well only after introducing two *ad hoc* selected scaling factors. Both methods need to be provided with an initial solution estimate and depending on its accuracy may not converge to the global optimum.

In [191], Zhao suggested two iterative methods based on second order cone programming (SOCP). These methods are guaranteed to converge to a global optimum and do not require an initial solution estimate. The first formulation is based on the method by Andreff *et al.* [8]. Using $\mathtt{S}_i$ from Equation 5.12 and linearization of the rotation matrix $\mathbf{r}_\mathtt{X} = \mathrm{vec}(\mathtt{R}_\mathtt{X})$, the residual error of the i-th relative motion is defined as

$$e_i = \left\| \mathtt{S}_i \begin{pmatrix} \mathbf{r}_\mathtt{X} \\ \mathbf{t}_\mathtt{X} \end{pmatrix} - \begin{pmatrix} \mathbf{0}_9 \\ \mathbf{t}_{\mathtt{A}_i} \end{pmatrix} \right\|.$$

The unknown hand-eye transformation is recovered as

$$(\mathbf{r}_\mathtt{X}^*, \mathbf{t}_\mathtt{X}^*) = \min_{\mathbf{r}_\mathtt{X}, \mathbf{t}_\mathtt{X}} \max_{i=1,\ldots,n} e_i. \tag{5.14}$$

This min-max formulation can be also viewed as minimization of L_∞-norm of the vector of residuals $\mathbf{e} = (e_1, e_2, \ldots, e_n)^\top$, *i.e.*,

$$(\mathbf{r}_\mathtt{X}^*, \mathbf{t}_\mathtt{X}^*) = \min_{\mathbf{r}_\mathtt{X}, \mathbf{t}_\mathtt{X}} \|\mathbf{e}\|_\infty .$$

Even though it is possible to recover the global optimum of Problem 5.14 using SOCP, the resulting $\mathbf{r}_\mathtt{X}$ still needs to be orthogonalized, *e.g.*, using Equation 5.13. The second method suggested by Zhao is based on the dual quaternion method of Daniilidis and Bayro-Corrochano [43, 44]. This time, the residual error of the i-th relative motion is defined using $\mathtt{S}_i$ from Equation 5.11 as

$$e_i = \left\| \mathtt{S}_i \begin{pmatrix} \mathbf{q} \\ \mathbf{q}' \end{pmatrix} \right\|,$$

where $(\mathbf{q}, \mathbf{q}')^\top$ is the dual quaternion representation of $\mathtt{X}$. The minimum is again recovered by solving Problem 5.14. However, an additional constraint is needed to avoid the trivial solution.

Strobl and Hirzinger [167] suggested an iterative method based on a parametrization of a stochastic model using a novel metric on $SE(3)$. However, for their method to perform optimally, some prior information on data noise characteristics is needed.

5.1.4 Image Measurement Error Minimization

Recently, several researchers proposed hand-eye calibration methods that use image measurements directly to eliminate the errors resulting from explicit computation of matrices $\mathtt{A}_i'$.

In [160], Stewenius proposed a hand-eye calibration method based on image point tracking. By formulating the problem using multilinear constraints and by executing robot movement not involving robot rotation, *i.e.*, $\mathtt{R}_{\mathtt{B}_i} = \mathtt{I}$ for all relative movements, the rotation $\mathtt{R}_\mathtt{X}$ and the internal camera calibration matrix $\mathtt{K}$ can be recovered using SVD. Once $\mathtt{R}_\mathtt{X}$ is recovered, the multilinear constraints can be now formulated with $\mathbf{t}_\mathtt{X}$ as the unknown and again solved for using SVD. Using this approach, computation of matrices $\mathtt{A}_i'$ can be avoided, however, the solution is not based on minimization of a geometrically meaningful function.

Kim *et al.* [91] proposed to minimize the variance of known 3D points, *i.e.*, of the calibration target points measured in different robot poses. The authors proposed to use nonlinear

minimization method by Powell and Brent to minimize the objective function. This method can recover a local minimum and requires initial solution estimate. The author compared their method to the methods of Tsai [183] and Strobl [167] and concluded that their method outperforms both competitors.

The recent progress in globally optimal methods for computer vision problems [69] allowed Seo *et al.* [146] to formulate a branch-and-bound algorithm to recover a globally optimal solution for class of hand-eye calibration problems where only rotations are involved, *i.e.*, System 5.1 is reduced to System 5.4. The unknown rotation $\mathtt{R}_\mathtt{X}$ is recovered by minimizing a cost function based on the image reprojection error in L_∞-norm. To avoid working in the pixel space, the authors consider a calibrated camera scenario and work with measurements represented by unit vectors pointing from the camera center to the respective 3D points. In a relative displacement of a rig that only involves rotation, two image measurements—$\mathbf{v} \in \mathbb{R}^3$ in the first camera position and $\mathbf{u} \in \mathbb{R}^3$ in the second camera pose—of the same world point are related as

$$\mathbf{u} = \mathtt{R}_\mathtt{A}\mathbf{v}.$$

Considering n of such rotational displacements and m correspondences in each image pair, we get

$$\mathbf{u}_{ij} = \mathtt{R}_{\mathtt{A}_i}\mathbf{v}_{ij},\ i = 1, \ldots, n,\ j = 1, ..., m.$$

Since $\mathtt{R}_\mathtt{A}$ and $\mathtt{R}_\mathtt{B}$ are related by Equation 5.4 as

$$\mathtt{R}_\mathtt{A} = \mathtt{R}_\mathtt{X}\mathtt{R}_\mathtt{B}\mathtt{R}_\mathtt{X}^\top,$$

the authors suggest to use measure the error between $\mathbf{u}$ and $\mathtt{R}_\mathtt{A}\mathbf{v}$ as the angle between the two vectors

$$e = \angle(\mathbf{u}, \mathtt{R}_\mathtt{X}\mathtt{R}_\mathtt{B}\mathtt{R}_\mathtt{X}^\top\mathbf{v}),$$

and to minimized the L_∞ norm of the residual error vector $\mathbf{e} = (e_{11}, e_{12}, \ldots, e_{nm})$, *i.e.*,

$$f(\mathtt{R}_\mathtt{X}) = \min_{\mathtt{R}_\mathtt{X}} \max_{i,j} \angle(\mathbf{u}_{ij}, \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top\mathbf{v}_{ij}). \tag{5.15}$$

The cost function f is minimized using a branch-and-bound strategy proposed in Hartley [69]. This strategy is based on recursive subdivision of the angle-axis parametrization of the rotational space represented by a cube $C = [-\pi, \pi]^3$.

5.2 Robot-World Calibration

In [193], Zhuang *et al.* extended the hand-eye calibration problem to also include calibration of the transformation from the coordinate system connected with the base of the robot to the world coordinate frame, see Figure 5.3. The method of Zhuang *et al.* uses quaternion rotation representation to solve an equation analogical to the Equation 5.1,

$$\mathtt{A}_i'^{-1}\mathtt{X} = \mathtt{Z}\mathtt{B}_i',\ i = 1, \ldots, m, \tag{5.16}$$

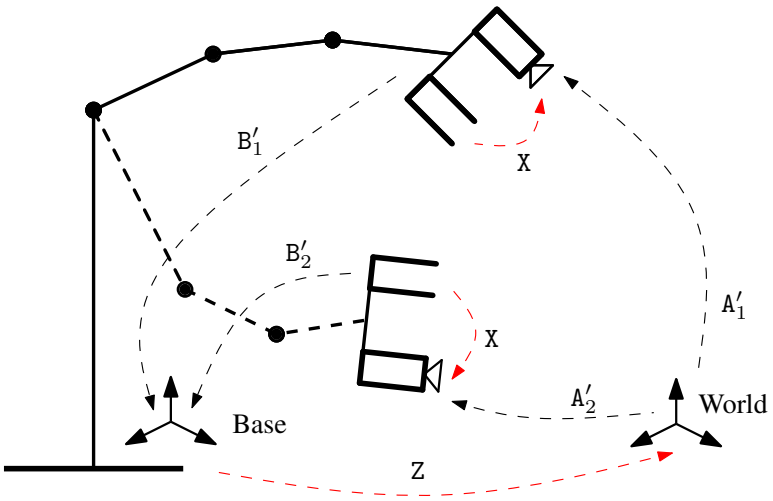


Figure 5.3: *Schematics of hand-eye and robot-world calibration.* A Robotic arm in two different poses.

where $\mathtt{Z} \in SE(3)$ represents the robot-world transformation. In this case, the kinematic loop is closed using the absolute camera and robot poses $\mathtt{A}'_i$ and $\mathtt{B}'_i$. Dornaika and Horaud [46] suggested a different solution, also based on quaternions. In [7], Li *et al.* proposed two different solutions based on Kronecker product and dual quaternions, analogous to the solutions to the hand-eye calibration problem in [8] and [44], respectively.

In the dual quaternion solution [7], special care has to be taken of the orientation ambiguity of the rotational quaternions $\mathbf{a}_i, \mathbf{b}_i$ when converting matrices $\mathtt{A}'_i$, $\mathtt{B}'_i$ to dual quaternions $\hat{\mathbf{a}}_i$, $\hat{\mathbf{b}}_i$. This is due to the fact that even though quaternions $\mathbf{a}_i, -\mathbf{a}_i$ and $\mathbf{b}_i, -\mathbf{b}_i$ represent the same rotation, the sign matters when Equation 5.1 is expressed using dual quaternions, *i.e.*,

$$\hat{\mathbf{a}}_i(\mathbf{a}_i, \mathbf{t}_{\mathtt{A}_i}) \otimes \hat{\mathbf{q}}_{\mathtt{X}} - \hat{\mathbf{q}}_{\mathtt{Z}} \otimes \hat{\mathbf{b}}_i(\mathbf{b}_i, \mathbf{t}_{\mathtt{B}_i}) \neq \hat{\mathbf{a}}_i(-\mathbf{a}_i, \mathbf{t}_{\mathtt{A}_i}) \otimes \hat{\mathbf{q}}_{\mathtt{X}} - \hat{\mathbf{q}}_{\mathtt{Z}} \otimes \hat{\mathbf{b}}_i(\mathbf{b}_i, \mathbf{t}_{\mathtt{B}_i}),$$

where the notation $\hat{\mathbf{q}}(\mathbf{q}, \mathbf{t})$ stands for the dual quaternion representing the rotation $\mathbf{q} \in \mathbb{Q}$ followed by the translation $\mathbf{t} \in \mathbb{R}^3$, see Equation 3.9, $\hat{\mathbf{a}}_i$, $\hat{\mathbf{b}}_i$, $\hat{\mathbf{q}}_{\mathtt{X}}$ and $\hat{\mathbf{q}}_{\mathtt{Z}}$ are the dual quaternion representations of $\mathtt{A}_i$, $\mathtt{B}_i$, $\mathtt{X}$, and $\mathtt{Z}$ respectively, and $\otimes$ is the dual quaternion multiplication. The obvious solution it to try all of the 2^m sign combinations and keep the combination with the smallest value of the criteria function. The sign problem is also inherent to the quaternion method of Dornaika [46].

5.3 Obtaining matrices $\mathtt{A}'_i$ and $\mathtt{B}'_i$

Most of the above discussed methods for hand-eye calibration considered the computation of absolute pose matrices $\mathtt{A}'_i$ and $\mathtt{B}'_i$ to be a separate step, carried out prior to hand-eye calibration itself. Once the matrices were known, an optimization step of computing $\mathtt{X}$ follows.

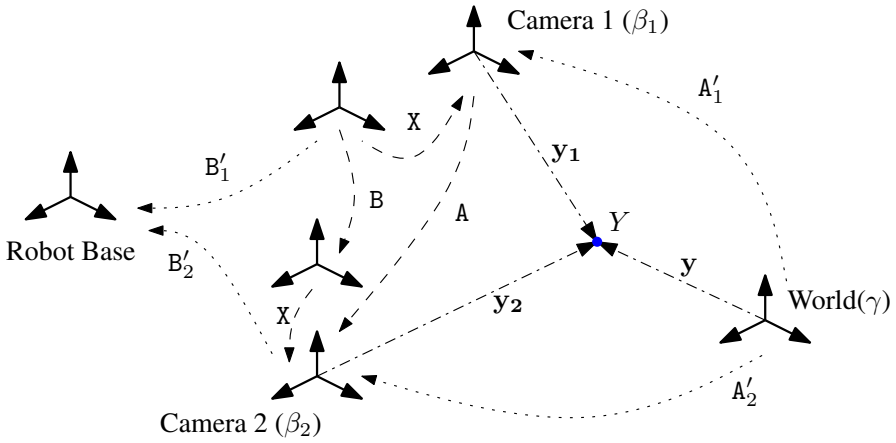


Figure 5.4: The transformations between different coordinate systems at the first and the second pose of the camera-gripper rig.

5.3.1 Matrix $\mathtt{A}'$ —External Camera Calibration

Let us consider a point $\mathbf{Y} \in \mathbb{R}^3$ that can be seen in both camera positions, see Figure 5.4. The point $\mathbf{Y}$ is expressed in the world coordinate frame γ as $\mathbf{Y}_\gamma \in \mathbb{R}^3$ and as $\mathbf{Y}_{\beta_1} \in \mathbb{R}^3$ in the coordinate system β_1 connected with the camera in it's first absolute pose. Matrix $\mathtt{A}'_1$ connects these two coordinates as

$$\begin{pmatrix} \mathbf{Y}_{\beta_1} \\ 1 \end{pmatrix} = \mathtt{A}'_1 \begin{pmatrix} \mathbf{Y}_\gamma \\ 1 \end{pmatrix}.$$

If the same point $\mathbf{Y}$ is expressed as $\mathbf{Y}_{\beta_2} \in \mathbb{R}^3$ in the coordinate system β_2 connected with the camera in it's second absolute pose, then the matrix $\mathtt{A}'_2$ connects these two measurements as

$$\begin{pmatrix} \mathbf{Y}_{\beta_2} \\ 1 \end{pmatrix} = \mathtt{A}'_2 \begin{pmatrix} \mathbf{Y}_\gamma \\ 1 \end{pmatrix}.$$

Matrix $\mathtt{A}$, see Figure 5.4, represents the transformation from the coordinated system β_1 to the coordinate system β_2 as

$$\begin{pmatrix} \mathbf{Y}_{\beta_2} \\ 1 \end{pmatrix} = \mathtt{A} \begin{pmatrix} \mathbf{Y}_{\beta_1} \\ 1 \end{pmatrix}.$$

and can be expressed as

$$\mathtt{A} = \mathtt{A'}_2 \mathtt{A}'^{-1}_1.$$

In this sense, matrices $\mathtt{A}'_1$ and $\mathtt{A}'_2$ represent the external camera parameters for the respective camera poses, see Section 4.3, whereas matrix $\mathtt{A}$ represents the relative pose, see Section 4.5.

5.3.2 Matrix $\mathtt{B}'$ —Forward kinematics

The transformation from the robotic end-effector to the robot base $\mathtt{B}'$ can computed from the values of the robot joint parameters by the process called *forward kinematics*. Robotic kinematics [150] is a vast research and engineering field and reviewing it in its totality is

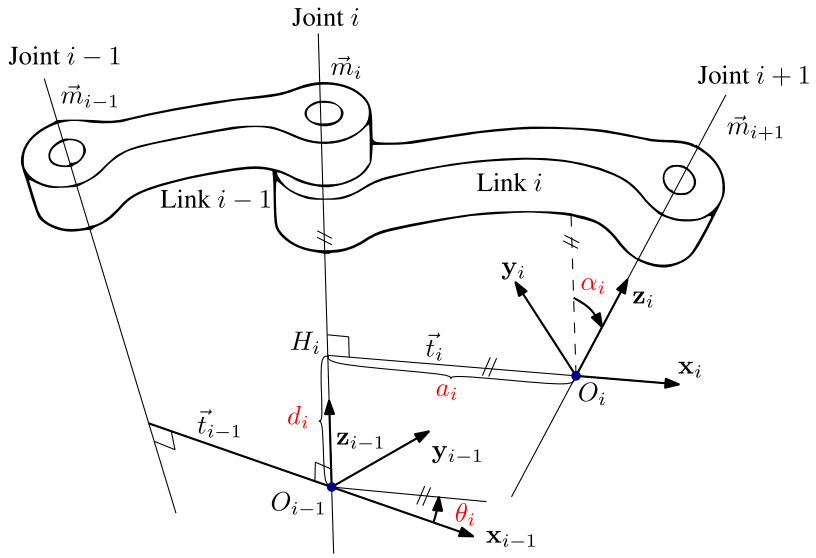


Figure 5.5: The positions of the $(i-1)$-th and the i-th coordinate frames according to the Denavit-Hartenberg convention. The four parameters describing the transformation between these frames—θ_i, d_i, a_i, α_i—are displayed in red.

out of the scope of this work. Here, we will review only the most widely used kinematic convention for the serial-link manipulators considered in our work—*Denavit-Hartenberg convention* [45].

Let us suppose that we seek to describe a serial-link manipulator with k links. In order to define transformations between its joints and ultimately to describe the location of the end-effector relative to the base of the robot, a coordinate frame has to be set up for each joint, see Figure 5.5. First, we need to identify all motion axes $\vec{m}_1, \ldots, \vec{m}_{i-1}, \vec{m}_i, \vec{m}_{i+1}, \ldots, \vec{m}_{k+1}$. Next, we need to identify the shortest transversals $\vec{t}_i$ between the motion axes $\vec{m}_i$ and $\vec{m}_{i+1}$. If $\vec{m}_i$ is parallel to $\vec{m}_{i+1}$, then an arbitrary $\vec{t}_i$ can be taken, however the simplest choice is $\vec{t}_i$ that intersects $\vec{t}_{i-1}$. The center of the first coordinate frame, O_0, can be placed anywhere on $\vec{m}_1$, but placing it such that $O_0 = H_1$ is the simplest choice. The positions of the centers of the coordinate frames O_i and the points H_i, $i = 1, \ldots, k-1$, are fixed as

$$\begin{aligned} O_i &= \vec{t}_i \wedge \vec{m}_{i+1}, \\ H_i &= \vec{m}_i \wedge \vec{t}_i, \end{aligned}$$

where symbol $\wedge$ stands for line intersection. The center of the last coordinate frame O_k can be placed anywhere, but the simplest choice is to place it such that $O_k = H_k = O_{k-1}$. Now we can set up the z-axis of the first coordinate frame $\mathbf{z}_0$ to be parallel with $\vec{m}_1$; we are free to choose one of the two possible orientations. Next, we set up the x-axis $\mathbf{x}_0$ to be parallel with $\vec{t}_1$ and to point in the direction of O_1. Finally, we set up the y-axis $\mathbf{y}_0$ to complete the right-handed coordinate frame. We place the rest of the z-axes $\mathbf{z}_i$ along the $\vec{m}_{i+1}$ axis, preferably to form a sharp angle with the previous axis $\mathbf{z}_{i-1}$and the rest of the x-axes $\mathbf{x}_i$ along $\vec{t}_i$ in the direction $O_i - H_i$. In case $\vec{m}_i$ and $\vec{m}_{i+1}$ intersect, then we select $\mathbf{x}_i$ to be perpendicular to both $\vec{m}_i$ and $\vec{m}_{i+1}$ and, again, preferably to form a sharp angle with the $\mathbf{x}_{i-1}$ axis. We choose the $\mathbf{y}_i$ axes to complete right-handed coordinate frames.

Now that we have set up a coordinate frame in each link, we need to derive a transformation relating each two consecutive links, ${}^{i-1}\mathtt{T}_i \in SE(3)$. Before we will do that, let us define four transformation functions. Let's $\delta \in \mathbb{R}$ and $\mathbf{t} = (t_1, t_2, t_3)^\top \in \mathbb{R}^3$. Now we can define three pure rotation transformations performing rotations about x, y and z axis, respectively,

$$\begin{aligned}
\mathrm{Rot}_x(\delta) &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\delta & -\sin\delta & 0 \\ 0 & \sin\delta & \cos\delta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \\
\mathrm{Rot}_y(\delta) &= \begin{pmatrix} \cos\delta & 0 & \sin\delta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\delta & 0 & \cos\delta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \\
\mathrm{Rot}_z(\delta) &= \begin{pmatrix} \cos\delta & -\sin\delta & 0 & 0 \\ \sin\delta & \cos\delta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},
\end{aligned}$$

and the pure translation transformation

$$\mathrm{Trans}(\mathbf{t}) = \begin{pmatrix} 1 & 0 & 0 & t_1 \\ 0 & 1 & 0 & t_2 \\ 0 & 0 & 1 & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

It can be shown that for rotary and prismatic joints typically used in robotics exactly 4 independent variables are needed to describe the transformation between two consecutive joint coordinate frames. In the Denavit-Hartenberg convention, these four parameters are typically denoted $\theta_i, d_i, a_i, \alpha_i \in \mathbb{R}$. The meaning of these parameters is as follows:

- θ_i, the angle from $\mathbf{x}_{i-1}$ to $\mathbf{x}_i$, measured about $\mathbf{z}_{i-1}$,
- d_i, the distance from O_{i-1} to H_i, measured along $\mathbf{z}_{i-1}$,
- a_i, the distance from H_i to O_i, measured along $\mathbf{x}_i$, and
- α_i, the angle from $\mathbf{z}_{i-1}$ to $\mathbf{z}_i$, measured about $\mathbf{x}_i$.

For a rotary joint, θ_i varies, for a prismatic joint, d_i varies. The transformation ${}^{i-1}\mathtt{T}_i$ transforming points in homogeneous coordinates from the coordinate frame connected with the i-th link to the coordinate frame connected with the $(i-1)$-th link can be expressed using these parameters as

$${}^{i-1}\mathtt{T}_i(\theta_i, d_i, a_i, \alpha_i) = \mathrm{Rot}_z(\theta_i)\mathrm{Trans}((0,0,d_i)^\top)\mathrm{Trans}((a_i,0,0)^\top)\mathrm{Rot}_x(\alpha_i),$$

and after expansion,

$${}^{i-1}\mathtt{T}_i(\theta_i, d_i, a_i, \alpha_i) = \begin{pmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Matrix ${}^{i-1}\mathtt{T}_i$ is sometimes called the *Denavit-Hartenberg matrix*.

The geometry of a k-link serial manipulator is completely described by a set of $4 \times k$ parameters

$$\begin{aligned} \boldsymbol{\theta} &= (\theta_1, \ldots, \theta_k)^\top, \\ \mathbf{d} &= (d_1, \ldots, d_k)^\top, \\ \mathbf{a} &= (a_1, \ldots, a_k)^\top, \\ \boldsymbol{\alpha} &= (\alpha_1, \ldots, \alpha_k)^\top, \end{aligned}$$

where $\boldsymbol{\theta}, \mathbf{d}, \mathbf{a}, \boldsymbol{\alpha} \in \mathbb{R}^k$. To solve the forward kinematics problem, that is to find the pose of the coordinate frame connected with the last link of our k-link serial-link manipulator with respect to its base frame, we chain the homogeneous transformations and get

$${}^0\mathtt{T}_k(\boldsymbol{\theta}, \mathbf{d}, \mathbf{a}, \boldsymbol{\alpha}) = {}^0\mathtt{T}_1(\theta_1, d_1, a_1, \alpha_1){}^1\mathtt{T}_2(\theta_2, d_2, a_2, \alpha_2) \cdots {}^{k-1}\mathtt{T}_k(\theta_k, d_k, a_k, \alpha_k).$$

Now let's suppose that all of the k joints are rotational and that the robot was manipulated into a position such that $\boldsymbol{\delta} = (\delta_1, \delta_2, \ldots, \delta_k)^\top \in \mathbb{R}^k$ are the values of the respective joint offsets in radians. Then the matrix $\mathtt{B}'$ associated with this pose can be computed as

$$\mathtt{B}' = {}^0\mathtt{T}_k(\boldsymbol{\theta} + \boldsymbol{\delta}, \mathbf{d}, \mathbf{a}, \boldsymbol{\alpha}).$$

The classical Denavit-Hartenberg convention is quite satisfactory when used to solve the forward kinematics task, however, several problems arise when the convention is used in a robot calibration procedure. The main problem is that the constants in ${}^{i-1}\mathtt{T}_i$ vary by large amounts for revolute joints with parallel or nearly parallel axes, in other words, small variations in the position and orientation of two consecutive links are modeled by large variations of the link parameters. To tackle this issue, several modifications to the convention have been proposed. Although it is slightly out of the scope of this work—since here we concerned with robot-camera calibration and assume that the robot is already calibrated—we will briefly review the modification proposed by Hayati and Mirmirani in [73].

The modified convention is identical to the classical Denavit-Hartenberg convention in cases where the position of the transversal $\vec{t}$ is "stable". For situations where this is not the case, *i.e.*, in cases where the links have parallel or nearly parallel motion axes, the authors proposed a new rule to determine the link parameters, see Figure 5.6. First, we pass a plane π that is perpendicular to the motion axis $\vec{m}_i$ and contains O_{i-1}. The intersection of the plane π and the motion axis $\vec{m}_{i+1}$ determines the position of O_i. To transform the coordinate frame connected with the i-th joint to the $(i+1)$-th joint we rotate the frame about $\mathbf{z}_{i-1}$ to align $\mathbf{x}_{i-1}$ with the line connecting frame centers O_{i-1} and O_i. This gives us the first link parameter θ_i. Next, we translate the frame in the direction $\mathbf{x}_{i-1}$ to align it with O_i and obtain the second link parameter d_i. Finally, we rotate the frame about its intermediate x and y-axes to align its z-axis with the $\mathbf{z}_i$ axis of the $(i+1)$-th joint, obtaining parameters α_i and β_i. The modified Denavit-Hartenberg matrix has now the following form:

$${}^{i-1}\mathtt{T}'_i(\theta_i, d_i, \alpha_i, \beta_i) = \mathrm{Rot}_z(\theta_i)\mathrm{Trans}((d_i, 0, 0)^\top)\mathrm{Rot}_x(\alpha_i)\mathrm{Rot}_y(\beta_i),$$

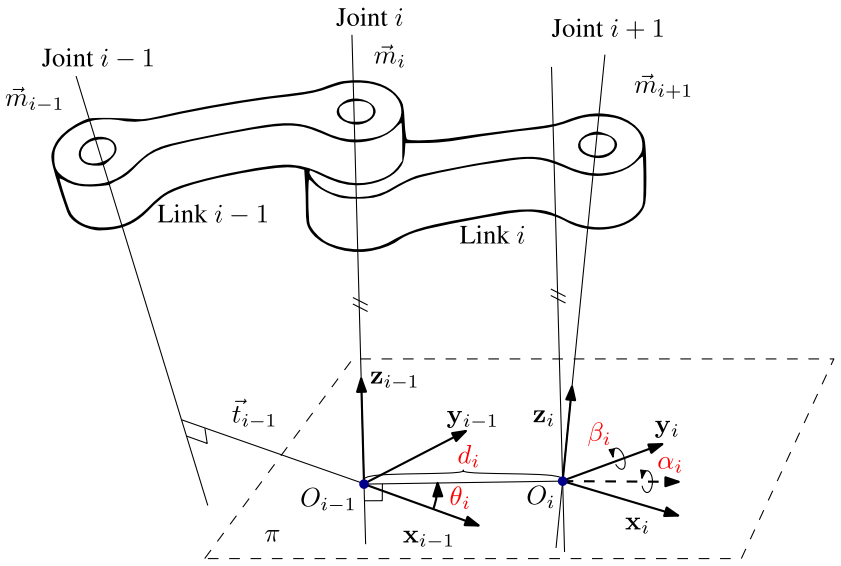


Figure 5.6: The positions of the $(i-1)$-th and the i-th coordinate frames according to the modified Denavit-Hartenberg convention. The four parameters describing the transformation between these frames—θ_i, d_i, α_i, β_i—are displayed in red.

and after expansion,

$$
{}^{i-1}\mathtt{T}'_i = \begin{pmatrix}
-\mathrm{s}\,\alpha_i\mathrm{s}\,\beta_i\mathrm{s}\,\theta_i + \mathrm{c}\,\beta_i\mathrm{c}\,\theta_i & -\mathrm{c}\,\alpha_i\mathrm{s}\,\theta_i & \mathrm{s}\,\alpha_i\mathrm{c}\,\beta_i\mathrm{s}\,\theta_i + \mathrm{s}\,\beta_i\mathrm{c}\,\theta_i & d_i\mathrm{c}\,\theta_i \\
\mathrm{s}\,\alpha_i\mathrm{s}\,\beta_i\mathrm{c}\,\theta_i + \mathrm{c}\,\beta_i\mathrm{s}\,\theta_i & \mathrm{c}\,\alpha_i\mathrm{c}\,\theta_i & -\mathrm{s}\,\alpha_i\mathrm{c}\,\beta_i\mathrm{c}\,\theta_i + \mathrm{s}\,\beta_i\mathrm{s}\,\theta_i & d_i\mathrm{s}\,\theta_i \\
-\mathrm{c}\,\alpha_i\mathrm{s}\,\beta_i & \mathrm{s}\,\alpha_i & \mathrm{c}\,\alpha_i\mathrm{c}\,\beta_i & 0 \\
0 & 0 & 0 & 1
\end{pmatrix},
$$

where s and c are shorthands for $\sin$ and $\cos$ functions, respectively.

6 Optimal Algorithms in Computer Vision

If you're teaching today what you were teaching five years ago, either the field is dead or you are.

– Noam Chomsky

Optimization problems in computer vision and robotics arise when fitting a parametrized models to some kind of measurements, in this case usually image data. To solve a problem, one has to minimize a defined *objective function*, sometimes also called the *cost function*, over a set of parameters. Let us consider a set of data measurements $\mathbf{x}_i$. A parametrized data model will provide a set of predicted model values $\mathbf{x}'_i$. In real life situations—that is in the case of noisy data and imperfect model—the measured and predicted data will differ. We can express this difference as a set of residuals

$$\delta_i = \|\mathbf{x}_i - \mathbf{x}'_i\| ,$$

where $\|\cdot\|$ is a norm that is relevant to the problem at hand and to the space of the measurements. Arranging the residuals into a vector, we get the *vector of residuals* $\boldsymbol{\delta} = (\delta_1, \delta_2, \dots)$. By solving for the model parameters we search to minimize an objective function that is a norm of the vector of residuals.

Previous two decades of computer vision research led to the establishment of today's number one choice optimization technique—bundle adjustment. Bundle adjustment is the problem of refining a visual reconstruction to produce a jointly optimal 3D structure and viewing parameter estimates based on nonlinear least squares fit [180]. It is a fairly universal tool that is easy to apply and, when used on a sparse problem, runs reasonably fast. However, to its main drawbacks one can place the fact that, as any other optimization method based on gradient descent, it is susceptible to running into a local minima of the objective function in case of poor initial parameter estimates. It has been observed in [70] that many problems arising in computer vision have multiple minima and for such problems bundle adjustment does not guarantee finding the optimal solution.

A lot of research in previous decades has also gone into the field of algebraic methods. These are often used to search for starting point for bundle adjustment, *e.g.*, the 8-point algorithm [107, 67, 178]. These methods are based on algebraic formulation of conditions that hold true in the idealized mathematical model of the problem. Substituting the real measurements in the problems formulations one searches to minimize a *algebraic cost function* to satisfy the conditions. The algebraic cost function is typically minimized using the apparatus of linear algebra. Even though the optimal solution in this sense can be found, it does not tell us much about a real quality of such a solution since the algebraic cost function is often in no simple and meaningful relation to the geometric structure of the problem.

As a result of these shortcomings a number of new methods with guaranteed and provable optimal solutions have been investigated in the recent years. These methods use results from the fields of convex and polynomial optimization and bring new ways of understanding the problems in computer vision and robotics. A number of these problems has already been successfully solved, yet others are still open. The work [68] provides an excellent review of optimal methods in computer vision as well as of the successfully solved problems.

6.1 Optimal Solutions

Before talking about "globally optimal algorithms" we are bound to consider whether solutions that such algorithms provide, albeit optimal in the mathematical sense, qualify as meaningful optimal solutions to the original problems.

6.1.1 L_2-norm

A popular cost function is based on error measurements minimization in L_2-norm. An argument can be put forth that the least square cost function formulation,

$$\|\boldsymbol{\delta}\|_2 = \sum_i \|\mathbf{x}_i - \mathbf{x}'_i\|^2 ,$$

leads to the optimal solution assuming that the measured data is corrupted with Gaussian noise. For Gaussian noise with variance σ^2, we can express the probability of the *i*-th measurement as

$$P(\mathbf{x}_i|\mathbf{x}'_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}'_i\|^2}{2\sigma^2}\right).$$

Further, assuming the independence of the measured data and denoting the vector of parameters θ, we can write the probability of the whole set of measurements as

$$P(\{\mathbf{x}_i\}|\{\mathbf{x}'_i(\theta)\}) = K \prod_i \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}'_i(\theta)\|^2}{2\sigma^2}\right),$$

where K is a normalization constant. While maximizing the probability of a set of measurements,

$$\arg\max_\theta K \prod_i \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}'_i(\theta)\|^2}{2\sigma^2}\right) = \arg\max_\theta \left[\log K + \sum_i \left(-\frac{\|\mathbf{x}_i - \mathbf{x}'_i(\theta)\|^2}{2\sigma^2}\right)\right],$$

we can see that the maximum likelihood estimate θ^* for the set of measurements also minimizes $\|\boldsymbol{\delta}\|_2$.

Although the minimization of a cost function based on L_2-norm has proven a successful tool for geometrical problems, such a cost function is often plagued with multiple local minima [173, 156] and solving for L_2-norm is a hard non-convex problem. Getting a certificate of optimality for L_2-norm based cost function is often next to impossible. Besides, the assumption of Gaussian noise is unlikely to hold for measurements in a digital image.

6.1.2 L_∞-norm

By replacing the assumption of Gaussian noise with the assumption of uniform bounded noise, one can make an analogous argument in favor of L_∞-norm.

Assuming the measurement error probability model

$$P(\{\mathbf{x}_i\}|\{\mathbf{x}'_i(\theta)\}) = K \prod_i \exp\left(-\left(\frac{\|\mathbf{x}_i - \mathbf{x}'_i(\theta)\|}{\sigma}\right)^p\right),$$

then the maximum likelihood estimate θ^* for a set of measurements also minimizes

$$\sum_i \|\mathbf{x}_i - \mathbf{x}'_i\|^p .$$

As p increases to infinity, the probability distribution converges to a uniform distribution for $\|\mathbf{x}_i - \mathbf{x}'_i\| \leq \sigma$ and $\sum_i \|\mathbf{x}_i - \mathbf{x}'_i\|^p$ converges to the L_∞-norm $\|\boldsymbol{\delta}\|_\infty$.

Since the L_∞-norm takes the largest component of a vector in absolute value and disregards the rest,

$$\min \|\boldsymbol{\delta}\|_\infty = \min \max_i \|\mathbf{x}_i - \mathbf{x}'_i\| ,$$

a potential disadvantage is that L_∞-norm is not robust to outliers. By minimizing L_∞-norm we are fitting the outliers, not good data. It is therefore imperative to remove outliers before starting the optimization. Several methods have been proposed to deal with the problem of outliers in L_∞-norm minimization problems [151, 147].

6.2 Optimal Algorithms

There are several classes of methods that can provide globally optimal solutions to problems formulated as either L_2 or L_∞-norm minimization tasks. In the case the minimization task is a problem of polynomial optimization, the method of linear matrix inequalities (LMI) relaxations or its dual, the method of polynomial sums of squares (SOS), can be used. Both methods relax the original problem to a problem of semidefinite programming. Quasi-convex optimization and branch-and-bound methods can be employed in case of L_∞-norm minimization.

However, not all problems lead to minimization tasks. Many problems in computer vision can be modeled by systems of polynomial equations. Such systems can be solved numerically or—as is more and more the case in recent years—using symbolic, *i.e.*, algebraic, methods.

6.2.1 Polynomial Optimization

From a theoretical point of view, every optimization problem—as long as it can be formulated using polynomial functions—can solved globally optimally using tools of elementary calculus by enumerating all stationary points of the cost function and checking for global minima. The requirement of a differentiable polynomial cost function will hold for many vision problems. The exception are those based on the L_∞-norm, which are generally not

differentiable. Although this approach is bound to find the optimum, because of the computationally prohibitive demands, only small problems are tractable. It can be shown that general problem of minimizing polynomial function is NP-hard [128], if the degree of the polynomial is at least four.

Before formally introducing the problem of polynomial optimization, let us propose a few definitions first.

Definition 6.1 (**Monomial**). Let $\mathbf{x} \in \mathbb{R}^n$ be a real vector and $\boldsymbol{\alpha}^n \in \mathbb{N}$ a integer vector. A *monomial* is defined as

$$\mathbf{x}^{\boldsymbol{\alpha}} = \prod_{i=1}^{n} x_i^{\alpha_i}.$$

Definition 6.2 (**Multivariate polynomial**). A scalar multivariate polynomial $p \in \mathbb{R}[\mathbf{x}]$ of degree $d \in \mathbb{N}$ is a linear combination of monomials

$$p(\mathbf{x}) = \sum_{|\boldsymbol{\alpha}| \leq d} p_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}} = \sum_{|\boldsymbol{\alpha}| \leq d} p_{\boldsymbol{\alpha}} x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n} = (p_{\boldsymbol{\alpha}})_{|\boldsymbol{\alpha}| \leq d}^{\top} (\mathbf{x}_{\boldsymbol{\alpha}})_{|\boldsymbol{\alpha}| \leq d},$$

where $\boldsymbol{\alpha} \in \mathbb{N}^n$ is the vector of indices, $|\boldsymbol{\alpha}| = \|\boldsymbol{\alpha}\|_1 = |\alpha_1| + |\alpha_2| + \cdots + |\alpha_n|$, $(p_{\boldsymbol{\alpha}})_{|\boldsymbol{\alpha}| \leq d}$ is the vector of the polynomial coefficients and $(\mathbf{x}_{\boldsymbol{\alpha}})_{|\boldsymbol{\alpha}| \leq d}$ is the polynomial basis.

Let $p_i(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, $i = 0, 1, \ldots, \ell$ be scalar multivariate polynomials in $\mathbf{x} \in \mathbb{R}^m$. Formally, the problem of multivariate polynomial optimization can be stated as follows:

Problem 6.3 (**Polynomial optimization**).

$$\begin{array}{rl} \text{minimize} & p_0(\mathbf{x}) \\ \text{subject to} & p_i(\mathbf{x}) \geq 0,\ i = 1, \ldots, \ell. \\ \text{where} & \mathbf{x} = (x_1, x_2, \ldots, x_m)^{\top} \in \mathbb{R}^m, \\ & p_0(\mathbf{x}), p_i(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]. \end{array}$$

Problem 6.3 can be also viewed as minimization of $p_0(\mathbf{x})$ over the basic semialgebraic set S,

$$S = \{\mathbf{x} \in \mathbb{R}^m \,|\, p_i(\mathbf{x}) \geq 0, i = 1, \ldots, \ell\}. \tag{6.1}$$

It is a non-convex problem with many local minima. Several techniques for relaxing this problem were developed, namely the Lasserre's LMI relaxation hierarchy and its dual theory of the polynomial sums of squares. It can be shown that under some circumstances these techniques can recover the global optimum of the original problem as well as certify its optionality. The work [100] provides an excellent survey of polynomial relaxation techniques.

6.2.1.1 Lasserre's LMI Relaxations

In [99], Lasserre cast Problem 6.3 as a linear optimization problem over the infinite-dimensional set of probability measures supported on S. A nice introduction to the Lasserre's hierarchy can be found in [78].

Definition 6.4 (Riesz functional). Given a sequence $\mathbf{y} = (y_{\boldsymbol{\alpha}})_{\boldsymbol{\alpha}\in\mathbb{N}^n}$, the *Riesz functional* $L_{\mathbf{y}}\colon \mathbb{R}[\mathbf{x}] \to \mathbb{R}$ is a linear functional that given a polynomial $p(\mathbf{x}) = \sum_{\boldsymbol{\alpha}} p_{\boldsymbol{\alpha}}\mathbf{x}^{\boldsymbol{\alpha}}$ returns

$$L_{\mathbf{y}}(p(\mathbf{x})) = \sum_{\boldsymbol{\alpha}} p_{\boldsymbol{\alpha}} y_{\boldsymbol{\alpha}}.$$

Definition 6.5 (Moment matrix). The *moment matrix* $M_d(\mathbf{y})$ of order d is the Gram matrix of the quadratic form $p(\mathbf{x}) \mapsto L_{\mathbf{y}}(p^2(\mathbf{x}))$, where $p(\mathbf{x}) = \sum_{\boldsymbol{\alpha}} p_{\boldsymbol{\alpha}}\mathbf{x}^{\boldsymbol{\alpha}}$ is a polynomial of degree d, i.e.,

$$L_{\mathbf{y}}(p^2(\mathbf{x})) = (p_{\boldsymbol{\alpha}})^\top M_d(\mathbf{y})(p_{\boldsymbol{\alpha}}).$$

Note, that from Definition 6.5 follows that $M_d(\mathbf{y})$ is a symmetrical matrix of dimensions $n \times n$, where

$$n = \binom{m+d}{m} = \frac{(m+d)!}{m!d!}$$

is the number of monomials of m variables of degree at most d. For example, if $n = 2$, then

$$M_0(\mathbf{y}) = y_{00}, \quad M_1(\mathbf{y}) = \begin{pmatrix} y_{00} & y_{10} & y_{01} \\ y_{10} & y_{20} & y_{11} \\ y_{01} & y_{11} & y_{02} \end{pmatrix}.$$

Definition 6.6 (Localizing matrix). The *localizing matrix* $M_d(q, \mathbf{y})$ of order d of a polynomial $q(\mathbf{x})$ is the Gram matrix of the quadratic form $p(\mathbf{x}) \mapsto L_{\mathbf{y}}(q(\mathbf{x})p^2(\mathbf{x}))$, i.e.,

$$L_{\mathbf{y}}(q(\mathbf{x})p^2(\mathbf{x})) = (p_{\boldsymbol{\alpha}})^\top M_d(q, \mathbf{y})(p_{\boldsymbol{\alpha}}).$$

In [137], Putinar proved that probability measures can be represented via sequences $\mathbf{y} = (y_\alpha)_{\alpha\in\mathbb{N}}$ of its moments. Using this result, Lasserre showed that by truncating these sequences one can construct a hierarchy of convex relaxations $\mathcal{P}_1, \mathcal{P}_2, \ldots$ that produces a monotonically non-decreasing sequence of lower bounds on Problem 6.3 converging to the global minimum. He also showed, that the series of the respective global optimizers $\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots$ of problems $\mathcal{P}_1, \mathcal{P}_2, \ldots$ asymptotically converges to $\mathbf{x}^*$, $\lim_{i\to\infty} \mathbf{x}_i^* = \mathbf{x}^*$, and that under some mild conditions global optimality of a relaxation can be detected and the global minimizers can be extracted by the tools of linear algebra from the solution of the relaxation. Practically, $(\mathbf{x}_i^*)_{i\in\mathbb{N}}$ converges to $\mathbf{x}^*$ in finitely many steps, *i.e.*, there exists $j \in \mathbb{N}$, such that $\mathbf{x}_j^* = \mathbf{x}^*$. Problems for which the finite convergence does not occur are in some sense degenerate and exceptional [129].

The main point of the Lasserre's hierarchy is the fact that the relaxations $\mathcal{P}_i$ can be formulated as semidefinite programs (SDP) and solved by any convenient SDP solver. The LMI relaxation $\mathcal{P}_\delta$ of order δ is built by linearizing all monomials $\mathbf{x}^{\boldsymbol{\alpha}}$ of the objective function p_0 up to degree 2δ, *i.e.*, $|\boldsymbol{\alpha}| \leq 2\delta$ using the Riesz functional $L_{\mathbf{y}}(p_0(x))$. If the objective function contains monomials of a higher degree, one has to start with a relaxation of a higher order. Next, the semialgebraic set S is relaxed using the localizing matrices $M_{\delta-1}(p_i(\mathbf{x}), \mathbf{y})$ for the polynomials p_i, $i = 1, \ldots, \ell$ by introducing ℓ LMI constraints $M_{\delta-1}(p_i(\mathbf{x}), \mathbf{y}) \succeq 0$. Finally, we add the moment matrix constraint $M_\delta(\mathbf{y}) \succeq 0$. Formally, the LMI relaxation $\mathcal{P}_\delta$ of order δ can be written as

Problem 6.7 (The LMI relaxation $\mathcal{P}_\delta$ of order δ).

$$\begin{array}{ll} \text{minimize} & L_{\mathbf{y}}(p_0(\mathbf{x})) = \sum_{\boldsymbol{\alpha}} p_{\boldsymbol{\alpha}} y_{\boldsymbol{\alpha}} \\ \text{subject to} & M_{\delta-1}(p_i(\mathbf{x}), \mathbf{y}) \succeq 0,\ i = 1, \ldots, \ell, \\ & M_\delta(\mathbf{y}) \succeq 0. \end{array}$$

Since there are exactly $n = \binom{m+2\delta}{m}$ monomials in m variables up to degree 2δ, SDP Problem 6.7 will have $\mathbf{y} \in \mathbb{R}^n$ linear variables. See [99] for the technical justification of this procedure.

In [86], Kahl and Henrion first applied the LMI relaxation technique in the context of computer vision to a number of classical vision problems: triangulation, pose estimation, homography estimation and epipolar geometry estimation.

6.2.1.2 Sums of Squares Relaxations

The theory of polynomial sums of squares is the dual theory to the Lasserre's LMI hierarchy. First, let us consider a problem of unconstrained polynomial minimization problem $\min p_0(\mathbf{x})$. It is an trivial observation, that this problem can be reformulated as

Problem 6.8 (Unconstrained polynomial minimization).

$$\begin{array}{ll} \text{maximize} & \gamma \\ \text{subject to} & p_0(\mathbf{x}) - \gamma \geq 0, \\ & \gamma \in \mathbb{R}. \end{array}$$

A natural idea how to tackle the hard problem of the non-negativity condition is to replace it by some simpler condition. Computationally more tractable than showing the non-negativity of $p_0(\mathbf{x}) - \gamma$ is to solve a relaxed problem of showing that the polynomial $p_0(\mathbf{x}) - \gamma$ is a sum of squares:

Definition 6.9 (Sum of squares). A multivariate polynomial $p(\mathbf{x})$ is a sum of squares (SOS) if there exist polynomials $f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})$ such that

$$p(\mathbf{x}) = \sum_{i=1}^{k} f^2(\mathbf{x}).$$

If a polynomial $p(\mathbf{x})$ is an SOS, it follows that $p(\mathbf{x}) \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$. It can be shown, see [135] for the proof of this claim, that a polynomial $p(\mathbf{x})$ of degree $2d$ is an SOS if and only if there exists a positive semidefinite matrix $\mathtt{Q}$, such that

$$p(\mathbf{x}) = \mathbf{z}(\mathbf{x})^\top \mathtt{Q} \mathbf{z}(\mathbf{x}),$$

where $\mathbf{z}(\mathbf{x})$ is a vector containing monomials in $\mathbf{x}$ of degree less or equal to d. By comparing the coefficient of the polynomials $p_0(\mathbf{x}) - \gamma$ and $\mathbf{z}(\mathbf{x})^\top \mathtt{Q} \mathbf{z}(\mathbf{x})$, we get a system of linear equations

$$\mathtt{A} \begin{pmatrix} \gamma \\ \mathrm{vec}(\mathtt{Q}) \end{pmatrix} = \mathbf{b}$$

in the unknown elements of matrix $\mathtt{Q}$ and γ. This leads to the relaxation of Problem 6.8, where the non-negativity of $p_0(\mathbf{x}) - \gamma$ is replaced by the constraint that $p_0(\mathbf{x}) - \gamma$ is an SOS:

Problem 6.10 (Unconstrained sum of squares relaxation).

$$\begin{aligned} \text{maximize} \quad & \gamma \\ \text{subject to} \quad & \mathtt{A}\begin{pmatrix} \gamma \\ \operatorname{vec}(\mathtt{Q}) \end{pmatrix} = \mathbf{b}, \\ & \mathtt{Q} \succcurlyeq 0, \\ & \gamma \in \mathbb{R}. \end{aligned}$$

Notice that Problem 6.8 is in the form of a semidefinite program (SDP).

In the case of Problem 6.3, we have to minimize the polynomial $p_0(\mathbf{x}) - \gamma$ over a basic semialgebraic set S, see Equation 6.1. One way how to relax this problem is to consider a sum of square decomposition $p_0(\mathbf{x}) - \gamma = s_0(\mathbf{x}) + \sum_{i=1}^{\ell} s_i(\mathbf{x})p_i(\mathbf{x})$, where $s_0(\mathbf{x})$, $s_i(\mathbf{x})$ are SOS. Such a decomposition does not directly lead to a semidefinite program, since cancellation of terms may occur in the polynomial $s_0(\mathbf{x}) + \sum_{i=1}^{\ell} s_i(\mathbf{x})p_i(\mathbf{x})$ and it is not clear how to bound the degrees of polynomials $s_0(\mathbf{x})$, $s_i(\mathbf{x})$. However, once the degree is fixed, the problem becomes a semidefinite program:

Problem 6.11 (Constrained sum of squares relaxation).

$$\begin{aligned} \text{maximize} \quad & \gamma \\ \text{subject to} \quad & p_0(\mathbf{x}) - \gamma = s_0(\mathbf{x}) + \textstyle\sum_{i=1}^{\ell} s_i(\mathbf{x})p_i(\mathbf{x}), \\ & s_0(\mathbf{x}), s_i(\mathbf{x}), i = 1, \dots, \ell \text{ are SOS}, \\ & \deg(s_0(\mathbf{x})), \deg(s_i(\mathbf{x})g_i(\mathbf{x})) < 2t. \end{aligned}$$

Using Problem 6.11, a hierarchy of SOS relaxations $\mathcal{P}'_1, \mathcal{P}'_2, \dots$ of the Problem 6.3 be constructed by increasing the parameter t. This hierarchy starts with

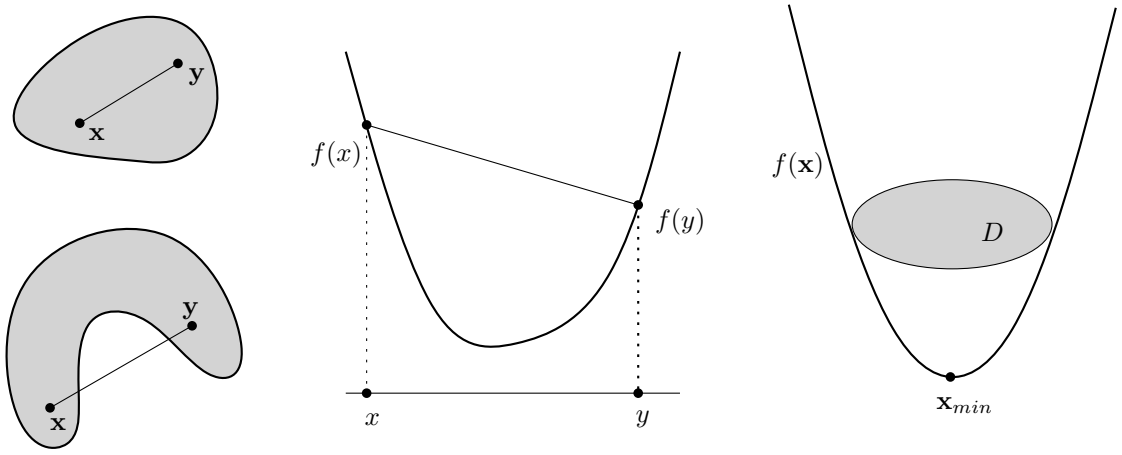
$$2t \geq \max(\deg(p(\mathbf{x})), \deg(s_0(\mathbf{x})), \dots, \deg(s_\ell(\mathbf{x})))$$

and as was the case of the Lasserre's LMI hierarchy, the respective global optimizers $\mathbf{x}'^*_1 \leq \mathbf{x}'^*_2 \leq \mathbf{x}'^*_3 \dots$ of problems $\mathcal{P}'_1, \mathcal{P}'_2, \mathcal{P}'_3 \dots$ asymptotically converge to $\mathbf{x}^*$, $\lim_{i\to\infty} \mathbf{x}'^*_i = \mathbf{x}^*$.

In [145], Schweighofer and Pinz applied the SOS relaxation method to the absolute camera pose problem to get the globally optimal solution in L_2-norm. They assumed a calibrated camera model and based the cost function on the object space residuals

$$e_i(\mathbf{x}_i, \mathbf{c}_i, \mathbf{v}_i) = \left\| \left(\mathtt{I} - \frac{\mathbf{v}_i \mathbf{v}_i^\top}{\mathbf{v}_i^\top \mathbf{v}_i} \right) (\mathtt{R}\mathbf{X}_i + \mathbf{t} - \mathbf{c}_i) \right\|_2,$$

where $\mathbf{X}_i \in \mathbb{R}^3$ are known 3D scene points, $\mathbf{c}_i, \mathbf{v}_i \in \mathbb{R}^3$ represent the measurements of $\mathbf{X}_i$ in the calibrated camera by the line of sight with the origins in $\mathbf{c}_i$ and directions $\mathbf{v}_i$, and $\mathtt{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3$ represent the unknown absolute camera pose. In other words, the error function e_i measures the distance of a scene point $\mathbf{X}_i$ from the line of sight back-projected from the camera. Notice that the objective function $\|\mathbf{e}\|_2^2 = \|(e_1, e_2, \dots)\|_2^2$ is a polynomial function. The translation can be factored out of the residuals e_i by differentiating with respect to $\mathbf{t}$ and equaling the result to zero. By further parametrizing the rotation $\mathtt{R}$ by a unit quaternion $\mathbf{q}$, see Equation 3.8, the cost function $\|\mathbf{e}\|_2^2$ can be written as a quadratic polynomial f in four unknowns $\mathbf{q} \in \mathbb{Q}$. Using the SOS decomposition and the Positivstellensatz, the authors relaxed the absolute pose problem as a solution to the following convex problem:



(a) (b) (c)

Figure 6.1: (a) An example of a convex (top) and a non-convex (bottom) set. (b) An example of a convex function. (c) Optimization of a convex function over a convex domain.

Problem 6.12 (SOS relaxation of the absolute camera pose problem [145]).

$$\begin{array}{ll} \text{maximize} & \gamma \\ \text{subject to} & f(\mathbf{q}) - \gamma - \lambda\left(\|\mathbf{q}\|_2^2 - 1\right) - \sigma q_1 \text{ is SOS}, \\ & \sigma \text{ is SOS}. \end{array}$$

6.2.2 Quasi-convex optimization

Quasi-convex optimization is closely related to convex optimization. First, let us define some basic notions.

Definition 6.13 (Convex set). A set $S \subset \mathbb{R}^n$ is *convex* if for any two points $\mathbf{x}, \mathbf{y} \in S$ and any α such that $0 \leq \alpha \leq 1$, we have

$$\alpha\mathbf{x} + (1-\alpha)\mathbf{y} \in S.$$

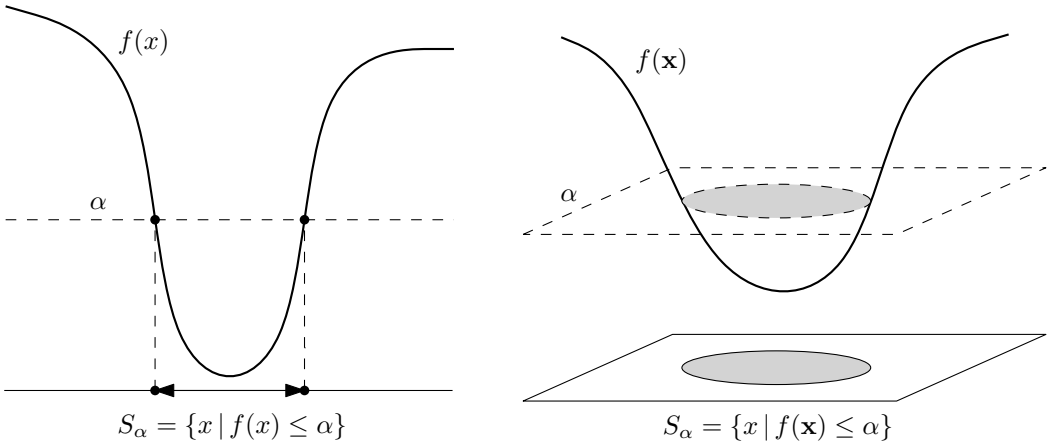
Definition 6.14 (Convex function). A function $f\colon \mathbb{R}^n \to \mathbb{R}$ is *convex* if $\operatorname{dom} f$ is a convex set and if for all $\mathbf{x}, \mathbf{y} \in \operatorname{dom} f$ and α such that $0 \leq \alpha \leq 1$, we have

$$f(\alpha\mathbf{x} + (1-\alpha)\mathbf{y}) \leq \alpha f(\mathbf{x}) + (1-\alpha) f(\mathbf{y}).$$

Definition 6.15 (α-sublevel set). An *α-sublevel set* of a function $f : \mathbb{R}^n \to \mathbb{R}$ is defined as

$$S_\alpha = \{\mathbf{x} \in \operatorname{dom} f \mid f(\mathbf{x}) \leq \alpha\}.$$

Definition 6.16 (Quasi-convex function). A function $f\colon \mathbb{R}^n \to \mathbb{R}$ is *quasi-convex* if its domain and all its sublevels sets $S_\alpha, \alpha \in \mathbb{R}$, are convex.



(a) (b)

Figure 6.2: Example of quasi-convex functions.

Book [18] is a popular reference work on convex optimization. See Figures 6.1 and 6.2 for examples of convex and quasi-convex functions.

A convex optimization problem is the one of the following form:

Problem 6.17 (Convex optimization problem).

$$\begin{aligned} \text{minimize} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_i(\mathbf{x}) \leq 0,\ i = 1, \ldots, m, \\ & \mathbf{a}_j^\top \mathbf{x} = b_j,\ j = 1, \ldots, \ell, \\ \text{where} \quad & f_0, \ldots, f_m \text{ are convex functions.} \end{aligned}$$

That is, we minimize a convex objective function over a convex set. A quasi-convex optimization problem is defined in the same way, except that the objective function $f_0(\mathbf{x})$ is quasi-convex. An agreeable fact about a convex function is that the global minimum value is attained at a single point, and there are no local minima apart from the global minimum. That is also true for quasi-convex functions. Another useful property of both convex and quasi-convex functions is that the point-wise maximum of a set of (quasi-)convex functions is (quasi-)convex, see Figure 6.3.

In [85] Hartley and Kahl proposed a framework for optimally solving structure-from-motion problems in L_∞-norm. The authors considered problems for which the residuals e_i to be minimized can be written as

$$e_i(\mathbf{x}) = \frac{\|\mathtt{F}_i\mathbf{x} + \mathbf{b}_i\|_2}{\mathbf{c}_i^\top \mathbf{x} + d_i},$$

and proposed to minimize the L_∞-norm of the vector of residuals $\mathbf{e} = (e_1, e_2, \ldots)^\top$:

Problem 6.18.

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{e}\|_\infty = \max_i \frac{\|\mathtt{F}_i\mathbf{x} + \mathbf{b}_i\|_2}{\mathbf{c}_i^\top \mathbf{x} + d_i} \\ \text{subject to} \quad & \mathbf{c}_i^\top \mathbf{x} + d_i \geq 0. \end{aligned}$$

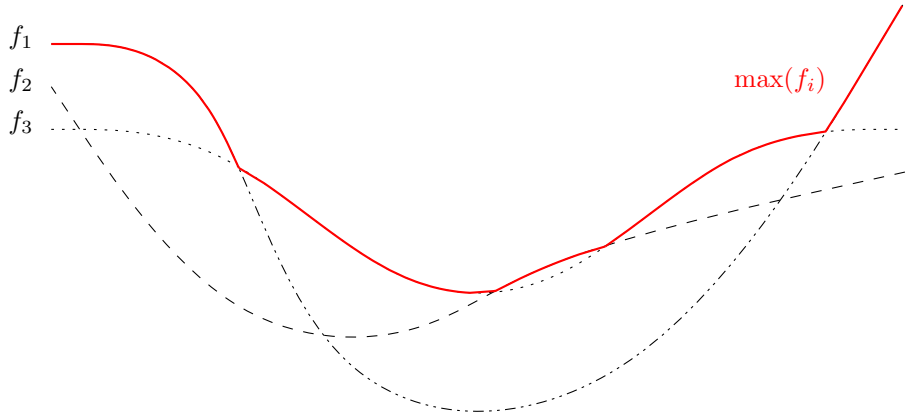


Figure 6.3: The point-wise maximum of a set of quasi-convex functions is quasi-convex.

Since the individual residuals $e_i(\mathbf{x})$ are not convex functions in $\mathbf{x}$ on the feasible region, Problem 6.18 is not a convex optimization problem, however, it is a quasi-convex problem. By introducing an additional variable γ, the problem can be transformed into the following equivalent form:

Problem 6.19.

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & \|\mathbf{F}_i\mathbf{x} + \mathbf{b}_i\|_2 - \gamma(\mathbf{c}_i^\top \mathbf{x} + d_i) \leq 0, \text{ for } \forall i \ , \\ & \gamma \geq 0. \end{array}$$

Problem 6.19 still isn't a convex problem, however once γ is fixed to a non-negative value, it reduces to a second-order cone problem, which is a type of convex optimization problem [18]. The authors of [85] employed a bisection scheme to solve Problem 6.19: for a fixed value $\gamma_{\text{test}} \geq 0$, the SOCP feasibility problem is solved. If the SOCP problem is feasible for γ_{test}, then the minimum value of γ must be less than γ_{test}, otherwise, it must be greater. Using this framework, the authors solved several problems such as triangulation, 2D homography estimation, camera resectioning, and the multiview structure-from-motion problem, assuming known rotations. Independently, a similar framework for quasi-convex problems was presented in [90].

6.2.3 Branch-and-Bound

The method [85] is not applicable to problems involving rotations since such problems are no longer quasi-convex. Branch-and-bound algorithms have been recently used with success to overcome this drawback.

Branch-and-bound algorithms are methods for global optimization of non-convex problems. They maintain a provable lower and upper bounds on the globally optimal objective value and terminate with a certificate proving that the solution is within $\epsilon > 0$ of the global optimum, for arbitrarily small ϵ. The practical applicability of the algorithm depends on whether one is able to cheaply compute such bounding functions and to reasonably bound the parameter space.

Let us consider $f\colon \mathbb{R}^m \to \mathbb{R}$ over a rectangle $Q_{\text{init}} \in \mathbb{R}^m$ with $f^\star$ being the minimum,

$$f^\star = \min_{\mathbf{x} \in Q_{\text{init}}} f(\mathbf{x}).$$

For a rectangle $Q \subseteq Q_{\text{init}}$ we define

$$\Phi_{\text{min}}(Q) = \min_{\mathbf{x} \in Q} f(\mathbf{x}).$$

Now, let $\Phi_{\text{lb}}(Q), \Phi_{\text{ub}}(Q)$ be functions that compute lower and upper bounds for $\Phi_{\text{min}}(Q)$,

$$\Phi_{\text{lb}}(Q) \leq \Phi_{\text{min}}(Q) \leq \Phi_{\text{ub}}(Q).$$

For the algorithm to converge, the functions $\Phi_{\text{lb}}(Q), \Phi_{\text{ub}}(Q)$ must become tighter as the rectangle Q shrinks to a point,

$$\forall \epsilon > 0\, \exists \delta > 0\, \forall Q \subseteq Q_{\text{init}} \colon \text{size}(Q) \leq \delta \implies \Phi_{\text{ub}}(Q) - \Phi_{\text{lb}}(Q) \leq \epsilon.$$

The algorithm recursively divides Q_{init} into smaller rectangles until it finds $Q^\star$, such that $\Phi_{\text{ub}}(Q^\star) - \Phi_{\text{lb}}(Q^\star) \leq \epsilon$.

This general method was used in [113, 87] to solve the L_2-norm multiview triangulation problem and the uncalibrated camera resection problem.

In [69] Hartley and Kahl presented a method for global optimization through rotation space search. They focused on the absolute and relative camera pose problems and gave optimal solutions under the L_∞-norm. Both of these problems require optimization over translation as well as rotation. The authors used the branch-and-bound approach and formulated the problems so that the the translation is not included in the branch-and-bound parameter search. They chose the angle-axis representation of rotation for which the whole space can be contained by a cube $C = [-\pi, \pi]^3$. Parameter space C is then subdivided in the branch-and-bound algorithm. A lower bound function $\Phi_{\text{lb}}(C)$ is given such that it is the lower bound for the reprojection errors that can be attained for every rotation in C given the optimal translation. It is formulated in the form of SOCP feasibility problem based on the similar argumentation as in [85]. Resulting rotation and translation are thus optimal in L_∞-norm.

A recent work by Choi *et al.* [36] further extended this approach by explicitly formulating the cost function based on pixel distance instead of the angular distance and by including the focal length as a variable in addition to the pose.

6.2.4 Algebraic Methods

Solving systems of polynomial equations is a classical problem with abundance of solution methods. One of the possible taxonomies of the solution strategies is into numerical and algebraic methods and it is the latter that are currently receiving the most attention in the computer vision community.

The idea behind the algebraic methods is to eliminate variables from the system and reduce the original problem to a problem of finding roots of univariate polynomials. The main

representatives of the algebraic methods are resultant methods and Gröbner basis methods. An overview of classical symbolic methods can be found in [88, 41].

The resultant methods are based on the theory of determinants and form the basis of the classical elimination theory. They were originally used to test whether a system of polynomial equations has a common factor, however, they can be used to find its solution as well. The idea behind the resultant methods is to linearize the original problem by construction a system of n independent polynomial equations, where n is the number of monomials in the original problem. An example of a resultant is the Sylvester determinant defined for two univariate polynomials. In [115], Macaulay showed the construction of multivariate resultants. Another examples of resultants are Bezout's resultants [47], Dixon's resultants [89], and hidden variable resultants [42]. In the method of hidden variable resultants one or more variables are considered as constants and to eliminate other variables from the system. This method was used by Li to solve the five-point problem [103] and the six-point two-view focal-length problem [102].

Whereas the resultant method is based on the linearization of the original problem, the second class of algebraic methods—the Gröbner basis methods—is based on polynomial ideal theory and multivariate polynomial division. The Gröbner basis is a base of the same ideal as is the set of original polynomial equations, *i.e.*, the solutions of the set of polynomial equations forming the Gröbner basis are the same as the solutions of the original polynomial system. The main difference is the agreeable fact that the Gröbner basis has the nice property of being easily solvable. This reduces the problem of solving a system of polynomial equations to the problem of transforming the system into a Gröbner basis. The theory of Gröbner bases was introduced by Bruno Buchberger [26]. He was also the first to provide an algorithm for computing these bases based on S-polynomials and polynomial division [41, 42]. However, in many situations the standard Buchberger's algorithm is prohibitively time and space consuming. This led to the development of several improvements [57, 62, 58, 31]. A well known algorithm that improves not only on the S-pair selection, but also on the monomial reduction, is the F4 algorithm by Faugère [51]. Another algorithm by the same author called F5 [52] applies a reduction strategy that removes S-polynomials that would reduce to zero.

The algebraic methods of solving systems of polynomial equations were successfully applied in computer vision, namely in the field of camera calibration. In [179], the method of Macaulay's resultant was used to solve the problem of estimating the absolute pose of a camera with unknown focal length from four 2D-to-3D correspondences and the problem of estimating the absolute pose of a camera with unknown focal length and unknown principal point from five 2D-to-3D correspondences. The method of hidden variables resultants was used by Li to solve the problems of estimating relative camera pose from five 2D-to-2D correspondences [103] and from six 2D-to-2D correspondences and unknown focal length [102].

A specific property of many polynomial problems in geometrical computer vision is the fact that these problem need to be solved many times, for example in a RANSAC loop, and that the overall structure of the polynomials does not change, only their coefficients do. This led to creation of specific polynomial solvers that are able to solve only problems with a predetermined polynomial structure, however, they are able to do it very fast. In [162], Stewénius proposed a method for constructing solvers for polynomial problems

based on Gröbner bases. Based on this method, Stewénius *et al.* solved the five 2D-to-2D correspondences problem [164], the six 2D-to-2D correspondences and unknown focal length problem [166] as well as a few other geometrical problems [61, 163]. Similar approach has been successfully applied to the absolute pose problems [24, 84], the relative pose problems [95, 96] and the panorama stitching problems [22, 27]. Also, issues of numerical accuracy [28] and theoretical solvability [132] were addressed.

7 Hand Eye Calibration Using Structure-from-Motion

Research! A mere excuse for idleness; it has never achieved, and will never achieve any results of the slightest value.

– Benjamin Jowett

This chapter presents a novel method for solving the hand-eye calibration problem. Using a calibration target is not possible for many applications of hand-eye calibration. In such situations structure-from-motion (SfM) approach of hand-eye calibration is commonly used to recover the camera poses up to scaling. The presented method takes the advantage of recent results in the quasi-convex optimization to recover the correct scale. Further, the correctly scaled displacement of the hand-eye transformation is recovered solely from the image correspondences and robot measurements, and it is guaranteed to be globally optimal with respect to the L_∞-norm. The method is experimentally validated using both synthetic and real world datasets.

7.1 Introduction

The common aspect of most of the hand-eye calibration methods is that they do not work with the camera measurements directly, but rather with the camera poses derived from them by other methods. The camera poses are usually estimated by observing a known calibration target. Since the calibration target has known dimensions, camera poses with correct scale can be obtained. However, there are many situations where using an accurately manufactured calibration target is not convenient or is not possible at all. Indeed, using a calibration target in applications such as mobile robotics or endoscopy may be unacceptable due to the restrictions in limited on-board weight or respectively to the strict sanitary conditions.

In [9], Andreff *et al.* proposed a method for "target-less" hand-eye calibration based on the structure-from-motion approach. The authors employed SfM to recover the unknown camera poses. Since SfM can recover camera poses up to scale only, the work introduced an explicit scaling factor to the hand-eye calibration equation. A similar approach was presented in [143], where the scaling factor was included into methods [81] and [44].

In this chapter, we present a modification to the SfM approach to hand-eye calibration. First, we estimate the rotational part of the hand-eye calibration separately using any convenient method. Next, we use second order cone programming to estimate the translational part from the original image correspondences and robot measurements. This formulation does not require the scaling factor to be estimated explicitly, but it can be recovered easily if needed. Furthermore, the estimated translation is globally optimal with respect to the reprojection error and the L_∞-norm.

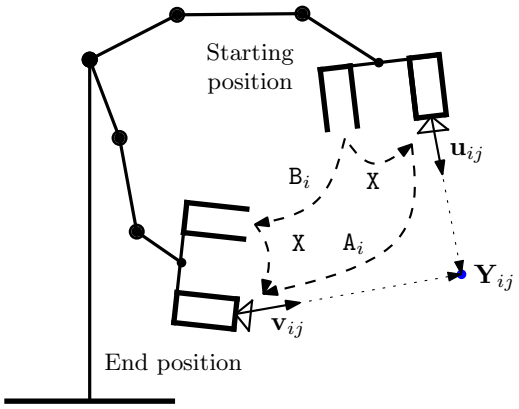


Figure 7.1: A relative movement of the camera–gripper rig.

7.2 Problem Formulation

Let us present the notation used in this chapter. The objective of hand-eye calibration is to derive the transformation

$$\mathtt{X} = \begin{pmatrix} \mathtt{R}_\mathtt{X} & \mathbf{t}_\mathtt{X} \\ \mathbf{0}^\top & 1 \end{pmatrix}, \tag{7.1}$$

where rotation $\mathtt{R}_\mathtt{X} \in SO(3)$ and $\mathbf{t}_\mathtt{X} \in \mathbb{R}^3$ are relating the coordinate frames of the gripper and the camera, see Section 5.1. This can be done by manipulating the gripper into two or more general positions and observing a scene from different camera viewpoints. In the rest of this chapter we will assume that the internal calibration of the camera is known and that the camera measurements are unit vectors representing the directions from the centers of the cameras to the respective 3D points.

Now, let's suppose that the gripper has been manipulated into n relative movements with the camera measuring m correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$, $j = 1, \ldots, m$ for every movement $i = 1, \ldots, n$. The restriction for the equal number of correspondences m for every movement is used here just to simplify the notation and can be easily removed by adding another level of indexing. Let $\mathtt{B}_i$ denote the transformation from the coordinate frame of the gripper in its starting position to the coordinate system of the gripper's end position. Transformations $\mathtt{B}_i$ can be obtained from the robot's positioning software and are thus considered to be known (Section 5.3.2). Let $\mathtt{A}_i$ denote the relative camera pose transformations. Camera poses with correct scale can be determined by camera calibration with a known calibration target (Section 4.3), or—up to scaling—using SfM approach (Section 4.7). As we known from Section 5.1, transformations $\mathtt{A}_i$,$\mathtt{B}_i$ and $\mathtt{X}$ are connected by the following relation, see Figure 7.1:

$$\mathtt{A}_i\mathtt{X} = \mathtt{X}\mathtt{B}_i. \tag{7.2}$$

This equation can be easily decomposed into rotational and translational parts

$$\mathtt{R}_{\mathtt{A}_i}\mathtt{R}_\mathtt{X} = \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}, \tag{7.3}$$

$$\mathtt{R}_{\mathtt{A}_i}\mathbf{t}_\mathtt{X} + s\mathbf{t}_{\mathtt{A}_i} = \mathtt{R}_\mathtt{X}\mathbf{t}_{\mathtt{B}_i} + \mathbf{t}_\mathtt{X}, \tag{7.4}$$

where $\mathtt{R}_{\mathtt{A}_i}, \mathtt{R}_{\mathtt{B}_i} \in SO(3)$, $\mathbf{t}_{\mathtt{A}_i}, \mathbf{t}_{\mathtt{B}_i} \in \mathbb{R}^3$ capture the respective rotations and translations. If both $\mathbf{t}_{\mathtt{A}_i}$ and $\mathbf{t}_{\mathtt{B}_i}$ were measured using the same unit, Equation 7.4 would hold for scaling factor $s = 1$. Note that Equation 7.3 can be solved for $\mathtt{R}_{\mathtt{X}}$ regardless of the value of s.

7.3 Second order Cone Programming

It was observed in [85] that various problems from multiview geometry can be written in the min-max form of Problem 6.18:

$$\min_{\mathbf{x}} \max_{i} \frac{\|\mathtt{F}_i\mathbf{x} + \mathbf{b}_i\|_2}{\mathbf{c}_i^\top \mathbf{x} + d_i} \text{ subject to } \mathbf{c}_i^\top \mathbf{x} + d_i \geq 0,$$

where $\mathbf{x}$ is the vector of unknowns to minimize over, i is the number of measurements, $\mathtt{F}_i$ matrices, and $\mathbf{b}_i, \mathbf{c}_i^\top$ vectors, all of compatible dimensions. If we consider the individual functions $\|\mathtt{F}_i\mathbf{x} + \mathbf{b}_i\|_2 / \left(\mathbf{c}_i^\top \mathbf{x} + d_i\right)$ as the components of a vector, Problem 6.18 may be thought of as L_∞-norm minimization of this vector. Problem 6.18 can also be formulated in the following, more convenient, form (Problem 6.19):

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & \|\mathtt{F}_i\mathbf{x} + \mathbf{b}_i\|_2 - \gamma\left(\mathbf{c}_i^\top \mathbf{x} + d_i\right) \leq 0 \end{array}$$

Note that because each of the constraints is convex and the objective function is linear, Problem 6.19 has a unique solution.

The key observation in [85] is that since for a fixed $\gamma \geq 0$ the constraint

$$\|\mathtt{F}_i\mathbf{x} + \mathbf{b}_i\|_2 - \gamma\left(\mathbf{c}_i^\top \mathbf{x} + d_i\right) \leq 0$$

is a second-order cone constraint, we can formulate the following second-order cone programming (SOCP) feasibility problem

Problem 7.1 (SOCP feasibility test).

$$\begin{array}{rl} \text{Given} & \gamma \\ \text{does there exist} & \mathbf{x} \\ \text{subject to} & \|\mathtt{F}_i\mathbf{x} + \mathbf{b}_i\|_2 - \gamma\left(\mathbf{c}_i^\top \mathbf{x} + d_i\right) \leq 0 \\ \text{for} & \forall i \,? \end{array}$$

and to solve the original Problem 6.18 we can employ a bisection scheme for $\gamma \geq 0$ and evaluate the feasibility test 7.1 repeatedly for fixed values of γ.

Further, it was also observed in [85] that angular reprojection error can be minimized using this approach. With known internal camera calibration, image measurements may be taken to represent unit direction vectors in space. Given a correspondence $\mathbf{u} \leftrightarrow \mathbf{v}$, $\mathbf{u}, \mathbf{v} \in \mathbb{R}^3$ and assuming the angle $\angle(\mathbf{u}, \mathbf{v})$ is positive and smaller than $\pi/2$, the reprojection error can be represented as

$$\tan\left(\angle(\mathbf{u}, \mathbf{v})\right) = \frac{\sin\angle(\mathbf{u}, \mathbf{v})}{\cos\angle(\mathbf{u}, \mathbf{v})} = \frac{\left\|[\mathbf{u}]_\times \mathbf{v}\right\|_2}{\mathbf{u}^\top \mathbf{v}}, \tag{7.5}$$

where matrix notation $[\mathbf{u}]_\times \mathbf{v}$ represents the cross-product $\mathbf{u} \times \mathbf{v}$.

7.4 Feasibility Test

As we can see from Equation 7.2, the relative camera pose for the i-th relative rig movement can be expressed in robot measurements, rotation $\mathtt{R}_\mathtt{X}$, and translation $\mathbf{t}_\mathtt{X}$ as

$$\begin{aligned} \mathtt{A}_i &= \mathtt{X}\mathtt{B}_i\mathtt{X}^{-1} = \begin{pmatrix} \mathtt{R}_{\mathtt{A}_i} & s\mathbf{t}_{\mathtt{A}_i} \\ \mathbf{0}^\top & 1 \end{pmatrix}, \\ \mathtt{R}_{\mathtt{A}_i} &= \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top, \\ s\mathbf{t}_{\mathtt{A}_i} &= \left(\mathtt{I} - \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top\right)\mathbf{t}_\mathtt{X} + \mathtt{R}_\mathtt{X}\mathbf{t}_{\mathtt{B}_i}. \end{aligned} \tag{7.6}$$

In was observed in [85, 69] that knowing a relative camera rotation $\mathtt{R}$, correspondences $\mathbf{u}_j \leftrightarrow \mathbf{v}_j, j = 1, \ldots, m$ and error bound γ, the bisection scheme can be used to solve for relative camera translation $\mathbf{t}$ using the following feasibility problem formulation:

Problem 7.2 (Relative camera pose feasibility test).

$$\begin{aligned} \text{Given} \quad & \mathtt{R}, \gamma \\ \text{do there exist} \quad & \mathbf{t}, \mathbf{Y}_j \\ \text{subject to} \quad & \angle\left(\mathbf{u}_j, \mathbf{Y}_j\right) \leq \gamma \\ & \angle\left(\mathbf{v}_j, \mathtt{R}\mathbf{Y}_j + \mathbf{t}\right) \leq \gamma \\ \text{for} \quad & j = 1, \ldots, m\,? \end{aligned}$$

Notice that since the pose transformation cannot be applied directly onto the correspondences, scene points $\mathbf{Y}_j \in \mathbb{R}^3$ also need to be recovered.

In order to apply the bisection framework [85] to hand-eye calibration, we will use Problem 7.2 to formulate a related feasibility test with $\mathbf{t}_\mathtt{X}$ as the unknown. By substituting $\mathtt{R} = \mathtt{R}_{\mathtt{A}_i}$ and $\mathbf{t} = s\mathbf{t}_{\mathtt{A}_i}$ from Equation 7.6 and repeating for all relative rig movements $i = 1, \ldots, n$ we get

Problem 7.3 (Hand-eye calibration feasibility test).

$$\begin{aligned} \text{Given} \quad & \mathtt{R}_\mathtt{X}, \gamma \\ \text{do there exist} \quad & \mathbf{t}_\mathtt{X}, \mathbf{Y}_{ij} \\ \text{subject to} \quad & \angle\left(\mathbf{u}_{ij}, \mathbf{Y}_{ij}\right) \leq \gamma \\ & \angle\left(\mathbf{v}_{ij}, \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top\mathbf{Y}_{ij} + \right. \\ & \quad \left.\left(\mathtt{I} - \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top\right)\mathbf{t}_\mathtt{X} + \mathtt{R}_\mathtt{X}\mathbf{t}_{\mathtt{B}_i}\right) \leq \gamma \\ \text{for} \quad & i = 1, \ldots, n,\ j = 1, \ldots, m\,? \end{aligned}$$

Again, as a "by-product" of the problem formulation, scene points $\mathbf{Y}_{ij} \in \mathbb{R}^3$ are recovered. Using the angular error formulation from Equation 7.5 we can formulate equivalent constraints so that they are linear in the optimized variables $\mathbf{t}_\mathtt{X}$ and $\mathbf{Y}_{ij}$, making Problem 7.3 an SOCP feasibility problem solvable by any SOCP solver. Indeed, we can write the constraints of the first type as

$$\begin{aligned} & \angle(\mathbf{u}_{ij}, \mathbf{Y}_{ij}) \leq \gamma \\ \Leftrightarrow \quad & \frac{\left\|\mathbf{u}_{ij} \times \mathbf{Y}_{ij}\right\|_2}{\mathbf{u}_{ij}^\top\mathbf{Y}_{ij}} \leq \tan\left(\gamma\right) \\ \Leftrightarrow \quad & \left\|\left[\mathbf{u}_{ij}\right]_\times \mathbf{Y}_{ij}\right\|_2 - \tan\left(\gamma\right)\mathbf{u}_{ij}^\top\mathbf{Y}_{ij} \leq 0. \end{aligned} \tag{7.7}$$

Equation 7.7 can be written in the formalism of Section 7.3 as

$$\left\| \overbrace{\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & [\mathbf{u}_{11}]_\times & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & [\mathbf{u}_{ij}]_\times & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & [\mathbf{u}_{nm}]_\times \end{pmatrix}}^{\mathtt{F}_{ij}} \overbrace{\begin{pmatrix} \mathbf{t}_\mathtt{X} \\ \mathbf{Y}_{11} \\ \vdots \\ \mathbf{Y}_{ij} \\ \vdots \\ \mathbf{Y}_{nm} \end{pmatrix}}^{\mathbf{x}} + \overbrace{\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix}}^{\mathbf{b}_{ij}} \right\|_2 - \tan(\gamma) \left(\underbrace{\begin{pmatrix} \mathbf{0} & \mathbf{0} & \cdots & \mathbf{u}_{ij}^\top & \cdots & \mathbf{0} \end{pmatrix}}_{\mathbf{c}_{ij}^\top} \underbrace{\begin{pmatrix} \mathbf{t}_\mathtt{X} \\ \mathbf{Y}_{11} \\ \vdots \\ \mathbf{Y}_{ij} \\ \vdots \\ \mathbf{Y}_{nm} \end{pmatrix}}_{\mathbf{x}} + \underbrace{0}_{d_{ij}} \right) \leq 0.$$

Analogously, for the second type of constraints we get

$$\begin{aligned} & \angle(\mathbf{v}_{ij}, \mathtt{R}_{\mathtt{A}_i}\mathbf{Y}_{ij} + s\mathbf{t}_{\mathtt{A}_i}) \leq \gamma \\ \Leftrightarrow \quad & \frac{\|\mathbf{v}_{ij} \times (\mathtt{R}_{\mathtt{A}_i}\mathbf{Y}_{ij} + s\mathbf{t}_{\mathtt{A}_i})\|_2}{\mathbf{v}_{ij}^\top (\mathtt{R}_{\mathtt{A}_i}\mathbf{Y}_{ij} + s\mathbf{t}_{\mathtt{A}_i})} \leq \tan(\gamma) \\ \Leftrightarrow \quad & \left\| \left([\mathbf{v}_{ij}]_\times \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top \mathbf{Y}_{ij} + [\mathbf{v}_{ij}]_\times \left(\mathtt{I} - \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top \right) \mathbf{t}_\mathtt{X} \right) + [\mathbf{v}_{ij}]_\times \mathtt{R}_\mathtt{X}\mathbf{t}_{\mathtt{B}_i} \right\|_2 - \\ & \tan(\gamma) \left(\left(\mathbf{v}_{ij}^\top \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top \mathbf{Y}_{ij} + \mathbf{v}_{ij}^\top \left(\mathtt{I} - \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top \right) \mathbf{t}_\mathtt{X} \right) + \mathbf{v}_{ij}^\top \mathtt{R}_\mathtt{X}\mathbf{t}_{\mathtt{B}_i} \right) \leq 0. \end{aligned} \tag{7.8}$$

Again, Equation 7.8 can we written in the formalism of Section 7.3 as

$$\left\| \overbrace{\begin{pmatrix} [\mathbf{v}_{ij}]_\times \left(\mathtt{I} - \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top \right) & 0 & 0 & 0 & 0 & 0 \\ 0 & \ddots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & [\mathbf{v}_{ij}]_\times \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \ddots \end{pmatrix}}^{\mathtt{F}_{ij}} \overbrace{\begin{pmatrix} \mathbf{t}_\mathtt{X} \\ \mathbf{Y}_{11} \\ \vdots \\ \mathbf{Y}_{ij} \\ \vdots \\ \mathbf{Y}_{nm} \end{pmatrix}}^{\mathbf{x}} + \overbrace{\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ [\mathbf{v}_{ij}]_\times \mathtt{R}_\mathtt{X}\mathbf{t}_{\mathtt{B}_i} \\ \vdots \\ \mathbf{0} \end{pmatrix}}^{\mathbf{b}_{ij}} \right\|_2 - \tan(\gamma) \left(\underbrace{\begin{pmatrix} \left(\mathbf{v}_{ij}^\top \left(\mathtt{I} - \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top \right) & \mathbf{0} & \cdots & \mathbf{v}_{ij}^\top \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_\mathtt{X}^\top & \cdots & \mathbf{0} \end{pmatrix}}_{\mathbf{c}_{ij}^\top} \underbrace{\begin{pmatrix} \mathbf{t}_\mathtt{X} \\ \mathbf{Y}_{11} \\ \vdots \\ \mathbf{Y}_{ij} \\ \vdots \\ \mathbf{Y}_{nm} \end{pmatrix}}_{\mathbf{x}} + \underbrace{\mathbf{v}_{ij}^\top \mathtt{R}_\mathtt{X}\mathbf{t}_{\mathtt{B}_i}}_{d_{ij}} \right) \leq 0.$$

As can be observed from the formulation of Problem 7.3, since the actual values of translations $\mathtt{t}_{\mathtt{A}_i}$ are not used, the scaling factor s does not need to be known. The correct scale of $\mathtt{t}_{\mathtt{X}}$ is derived solely from $\mathtt{t}_{\mathtt{B}_i}$. However, the value of s can be computed using Equation 7.6 if needed.

7.5 Bisection

In order to use the feasibility test 7.3 for the estimation of the translation $\mathtt{t}_{\mathtt{X}}$, we employ a simple bisection scheme. In every iteration, the value of γ is fixed and the feasibility test 7.3 is solved. The algorithm starts with $\gamma = \tan(\pi/4)$ and the iteration loop ends once the difference of the lower and upper bounds $\gamma_{\text{low}}, \gamma_{\text{high}}$ reaches a prescribed accuracy ϵ.

Algorithm 1 SfM Hand-Eye Bisection

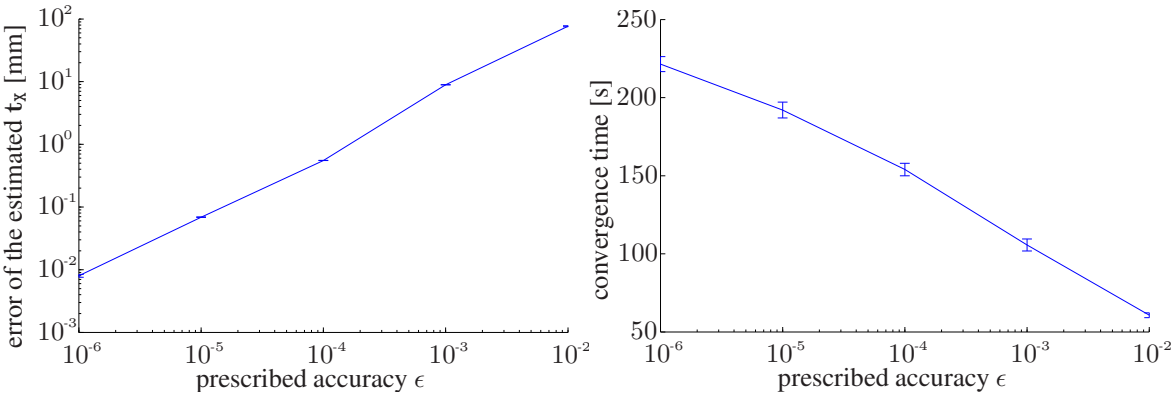
Require: $\mathtt{R}_{\mathtt{X}}, \epsilon > 0$
  $\gamma_{\text{low}} \leftarrow 0$
  $\gamma_{\text{high}} \leftarrow 2$
  while $(\gamma_{\text{high}} - \gamma_{\text{low}}) \geq \epsilon$ **do**
    $\gamma \leftarrow (\gamma_{\text{high}} + \gamma_{\text{low}})/2$
    $(\mathtt{t}_{\mathtt{X}}, \textit{feasible}) \leftarrow$ Feasibility test 7.3
    if *feasible* **then**
      $\gamma_{\text{high}} \leftarrow \gamma$
    else
      $\gamma_{\text{low}} \leftarrow \gamma$
    end if
  end while
  return $\mathtt{t}_{\mathtt{X}}$

7.6 SfM Algorithm for Hand-Eye Calibration

Finally, we can formulate the complete algorithm for the hand-eye calibration using our SfM approach. Since both the SfM method and the method for $\mathtt{R}_{\mathtt{X}}$ estimation can be changed at the user's convenience, this formulation can be seen as a *meta*-algorithm.

Algorithm 2 SfM Hand-Eye Calibration

1. Estimate the relative camera rotations $\mathtt{R}_{\mathtt{A}_i}$ using a convenient SfM method, *e.g.*, method [153].
2. Estimate $\mathtt{R}_{\mathtt{X}}$ using $\mathtt{R}_{\mathtt{A}_i}$ and $\mathtt{R}_{\mathtt{B}_i}$, *e.g.*, using method [134].
3. Use Algorithm 1 to find the optimal $\mathtt{t}_{\mathtt{X}}$ using $\mathtt{R}_{\mathtt{X}}$ and required precision ϵ.



(a) (b)

Figure 7.2: ACCURACY experiment. (a) The mean error of the estimated $\mathbf{t}_X$ for the various values of ϵ plotted in loglog scale together with the variance over the twenty constructed tasks. (b) The mean convergence time of Algorithm 1 for the various values of ϵ plotted in semilogx scale together with the indicated variance.

7.7 Experimental Results

In this section, the proposed Algorithm 2 is validated both by synthetic and real data experiments. We used method [134] to obtain $\mathtt{R}_X$ from $\mathtt{R}_{A_i}$ and $\mathtt{R}_{B_i}$ and SeDuMi [168] as the SOCP solver. All the reported times were achieved using a standard Intel Core 2 based consumer PC running 64-bit Linux and MATLAB 7.6.

7.7.1 Synthetic-data Experiment

A synthetic scene consisting of 100 3D points randomly generated into a ball of radius 1000 mm and 10 absolute camera poses set such that the cameras faced approximately the center of the ball were created. The generated 3D points were measured in the respective cameras giving raise to correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$. Two experiments were conducted with the generated scene. First, the accuracy and efficiency of Algorithm 1 was studied for different values of the prescribed accuracy ϵ. Second, the performance of Algorithm 2 was tested on noised correspondences.

ACCURACY experiment Twenty random transformations $\mathtt{X}$ were generated and the tasks composed of the known $\mathtt{R}_X$, the known correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$, and 9 relative movements $\mathtt{B}_i$, computed from the known absolute camera poses and the generated transformations, were constructed for each of them. These tasks were solved for 5 different values of ϵ ranging from 10^{-6} to 10^{-2} with equal steps on a logarithmic scale and the output was plotted to Figure 7.2.

The results show that not only the optimized maximum angular reprojection error but also the error of the estimated $\mathbf{t}_X$, measured as the distance between the known and the estimated value of $\mathbf{t}_X$, decreases rapidly with decreasing ϵ. On the other hand, the convergence time increases because more iterations are needed for the bisection.

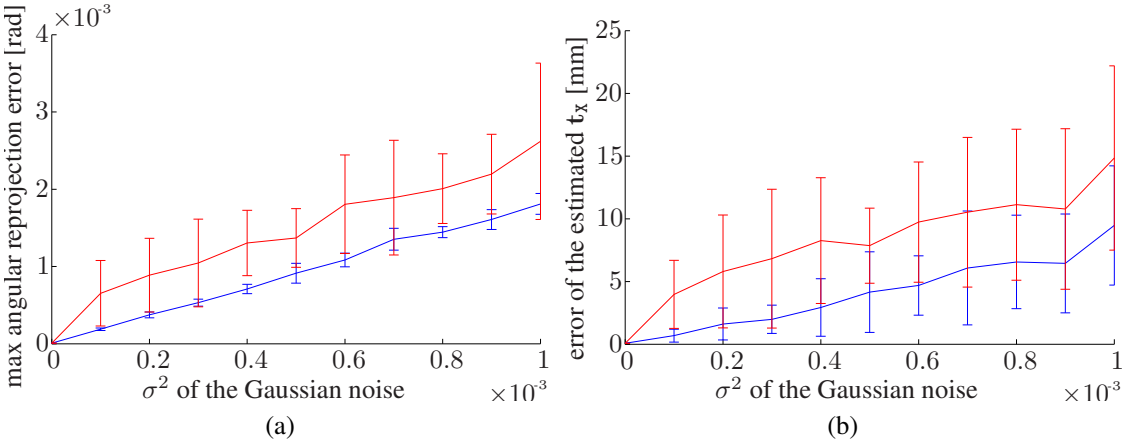


Figure 7.3: NOISE experiment. (a) The maximum angular reprojection error for the various values of σ^2 recovered by Algorithm 2 with $\epsilon = 10^{-5}$. Blue: Mean maximum error for the known $\mathtt{R}_\mathtt{X}$ together with the variance over the twenty tasks. Red: Mean maximum error for $\mathtt{R}_\mathtt{X}$ computed by [134] from $\mathtt{R}_{\mathtt{A}_i}$ from the noised correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$ together with the indicated variance. (b) The error of the estimated $\mathbf{t}_\mathtt{X}$ for the various values of σ^2, see (a) for the description of the colors.

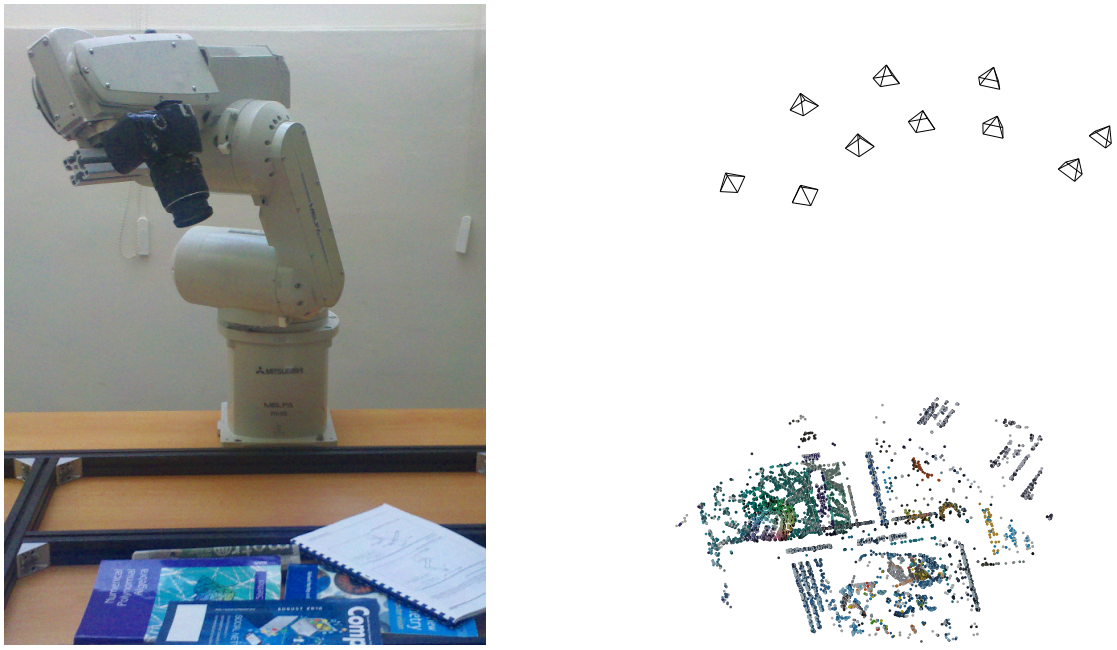
NOISE experiment The same twenty random transformations $\mathtt{X}$ and the corresponding relative movements $\mathtt{B}_i$ were used in the second experiment, where ϵ was fixed to 10^{-5} but the measurements $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$ were corrupted with Gaussian noise in the angular domain. 11 noise levels were used, $\sigma^2 \in \langle 0, 10^{-3} \rangle$ in 10^{-4} steps.

The experiment consisted of two parts, the first one testing the stability of the computation of $\mathbf{t}_\mathtt{X}$ for known $\mathtt{R}_\mathtt{X}$ by assigning relative camera rotations $\mathtt{R}_{\mathtt{A}_i}$ to the known values while the latter one testing the stability of the whole proposed algorithm. Relative camera rotations were computed by decomposing the essential matrices $\mathtt{E}_{\mathtt{A}_i}$, describing relative camera poses up to scaling, robustly computed from the noised correspondences by RANSAC [53] using the 5-point minimal relative pose problems for calibrated cameras [130] in the latter part.

Figure 7.3a shows the relation of the maximum angular reprojection error achieved by Algorithm 2 for the various values of σ^2 with $\epsilon = 10^{-5}$ both for the known and for the computed $\mathtt{R}_\mathtt{X}$. The relation of the error of the estimated $\mathbf{t}_\mathtt{X}$ for the various σ^2 values for both parts of the test can be seen in Figure 7.3b. As the mean length of the generated $\mathbf{t}_\mathtt{X}$ was 259.1 mm in our experiment, the error of the estimation stayed under 5% even for high noise levels. The convergence times of the algorithm were around 3 minutes for the noise-free task and increasing towards 4 minutes for the tasks using noised correspondences.

7.7.2 Real-data Experiment

A Mitsubishi MELFA-RV-6S serial manipulator with a Nikon D3000 digital SLR camera and an AF-S DX NIKKOR 18–55 mm VR lens (set to 55 mm) was used to acquire data for the experiment, see Figure 7.4a. The robot was instructed to move the gripper along the surface of a sphere of radius approximately 700 mm centered in the middle of the scene objects. The position of the gripper was adjusted using the yaw and pitch angles measured from the center



(a) (b)

Figure 7.4: Real-data experiment. (a) A Mitsubishi MELFA-RV-6S serial manipulator used to acquire the data for the experiment. A Nikon D3000 digital SLR camera mounted on the gripper using a self-made mechanical reduction. (b) The 3D model output from Bundler containing 10,680 triangulated 3D points and the poses of all the ten cameras.

of the sphere to reach ten different locations with five different yaw angles for each of the two possible pitch angles. The gripper was set to face the center of the sphere up to a small additive noise. The camera was set to manual mode and images of 3,872×2,592 pixels were taken using a remote trigger.

Two image sets for two different scenes were acquired—a scene with a calibration target used for obtaining internal camera calibration and a scene with general objects to show the contribution of the proposed method over the hand-eye calibration approaches that rely on a known calibration target. The calibration matrix together with two parameters of radial distortion were computed using [118] and images were radially undistorted prior being further used in order to improve SfM results. Knowing the focal length of the camera, the angular resolution of the acquired images could be computed as 1 pixel $\approx 1.15 \times 10^{-4}$ rad.

CALIBRATION scene Scene CALIBRATION, see Figure 7.5a–c, was primarily used for internal camera calibration but since the calibration procedure outputs the $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$ correspondences as its by-product, we used the scene for a hand-eye calibration experiment as well. The approach used for the NOISE experiment with the synthetic data was also used to obtain relative camera rotations $\mathtt{R}_{\mathtt{A}_i}$ from the correspondences. Relative robot rotations $\mathtt{R}_{\mathtt{B}_i}$ and translations $\mathbf{t}_{\mathtt{B}_i}$ were obtained from the known gripper-to-base transformations $\mathtt{B}'_i$.

A task composed of 9 motions, which were the relative motions between gripper positions 1–2, 2–3, ..., 9–10, and the respective correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$, totaling 1,583 en-

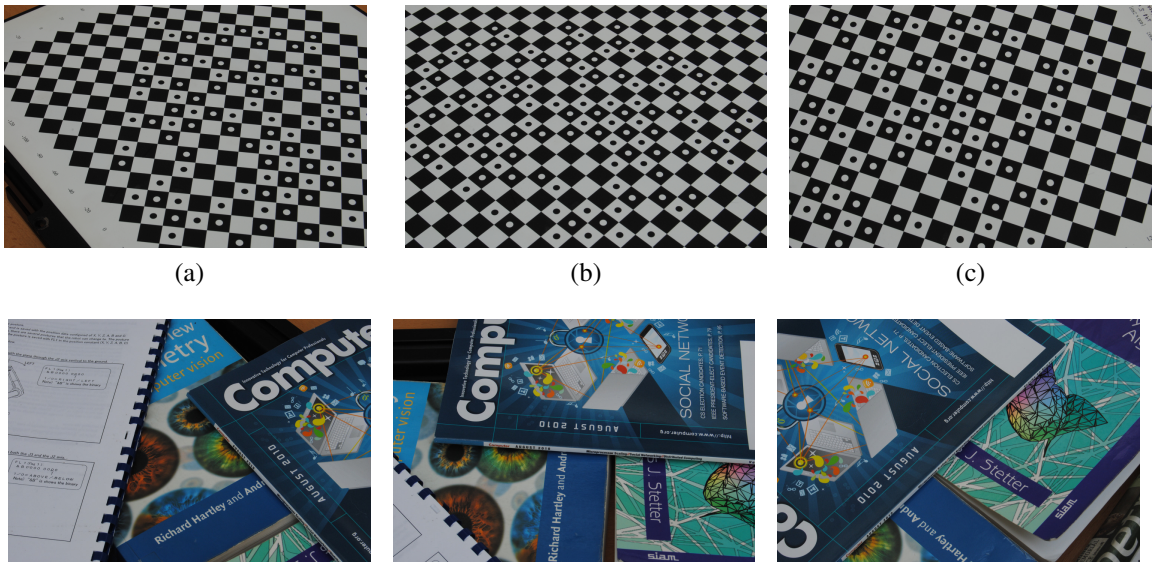


Figure 7.5: Sample images of our scenes taken by the camera mounted on the gripper of the robot. (a–c) Scene CALIBRATION. (d–f) Scene GENERAL.

tries was constructed. Algorithm 1 converged in 596 seconds for $\epsilon = 10^{-5}$ giving a solution with the maximum angular error 7.3×10^{-4} rad. The computed rotation $\mathtt{R}_\mathtt{X}$ was close to the expected, rotation along the z-axis by $-\pi/2$, and the obtained translation from the gripper to the camera center, $-\mathtt{R}_\mathtt{X}^\top \mathbf{t}_\mathtt{X} = (83.5, -19.3, 130.6)^\top$, was close to the result of the method of Tsai [183], $(83.9, -17.8, 133.8)^\top$, and corresponded with a rough physical measurement on the mechanical reduction, $(90, -20, 130)^\top$, showing the validity of the obtained results.

GENERAL scene Scene GENERAL, see Figure 7.5d–f, was acquired in order to show the performance of the method in the real-world conditions. SIFT [111] image features and Bundler [153]—a state-of-the-art open source SfM implementation—were used to obtain the camera poses. Camera focal length from the internal camera calibration was stored as EXIF information in the individual images and Bundler was instructed to preserve the focal lengths read from EXIF.

The resulting 3D model output from Bundler contained 10,680 triangulated points and the poses of all the ten cameras, see Figure 7.4b. By examining the 3D point cloud,28 3D points were manually labeled as erroneous and the projections of these points were excluded from the correspondences. This procedure could have been skipped if an SfM method triangulating 3D points from camera triplets instead of pairs was used, since the error rate of such methods is close to zero. Relative camera rotations $\mathtt{R}_{\mathtt{A}_i}$ were computed from the camera projection matrices $\mathtt{P}_{\mathtt{A}_i}$ output from Bundler and relative robot rotations $\mathtt{R}_{\mathtt{B}_i}$ and translations $\mathbf{t}_{\mathtt{B}_i}$ were again obtained from the known gripper-to-base transformations $\mathtt{T}_{\mathtt{B}_i}$. Due to a high number of correspondences, only every tenth member of the set of correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$ was used for computation giving 1,929 entries in total for the task composed of the same 9 motions as in the previous experiment.

Algorithm 1 converged in 1,067 seconds for $\epsilon = 10^{-5}$ giving a solution with the maximum angular error 6.1×10^{-4} rad. Again, the computed rotation $\mathtt{R}_\mathtt{X}$ was close to the expected one and the obtained translation from the gripper to the camera center, $(97.4, -14.0, 129.9)^\top$, corresponded with the rough physical measurement. We suspect that the difference in the x and y coordinates was caused by the fact that Bundler does not optimize for the position of the principal point and therefore a simplified camera calibration was used, which led to a slightly biased estimation of $\mathtt{R}_{\mathtt{A}_i}$.

8 Hand-Eye Calibration Using Brand-and-Bound

The solution of every problem is another problem.

– Johann Wolfgang von Goethe

In the previous chapter, we proposed a method for optimal estimation of the translational part from camera measurement. However, this method still requires prior knowledge of the relative camera rotations and solves for rotation separately. In work by Seo *et al.* [146], the rotational part is solved optimally but all the translations are assumed to be zero.

In this chapter, we solve for rotation and translation simultaneously by minimizing an objective function based on the epipolar constraint without any prior knowledge of the external camera calibration. Our method is based on branch-and-bound search over the space of rotations presented in [69] and is guaranteed to converge to the optimum with respect to L_∞-norm.

8.1 Problem Formulation

Again, Let us assume a camera that has been rigidly mounted on a robot's gripper and we want to determine the homogeneous transformation

$$\mathtt{X} = \begin{pmatrix} \mathtt{R}_\mathtt{X} & \mathbf{t}_\mathtt{X} \\ \mathbf{0}^\top & 1 \end{pmatrix},$$

such that rotation $\mathtt{R}_\mathtt{X} \in \mathrm{SO}(3) \subset \mathbb{R}^{3\times 3}$ and translation $\mathbf{t}_\mathtt{X} \in \mathbb{R}^3$ transform point coordinates from the coordinate system of the gripper to the coordinate system of the camera.

Now, let us suppose that the gripper has been manipulated into $n+1$ positions resulting into n relative robot and camera motions $\mathtt{B}_i$ and $\mathtt{A}_i$, $i = 1, \ldots, n$. These motions lead to the well known hand-eye system

$$\mathtt{A}_i\mathtt{X} = \mathtt{X}\,\mathtt{B}_i,$$

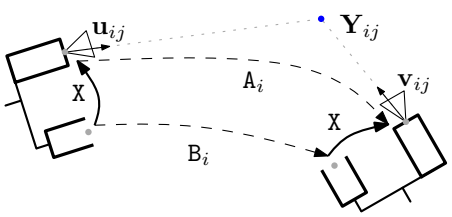


Figure 8.1: A gripper-camera rig motion.

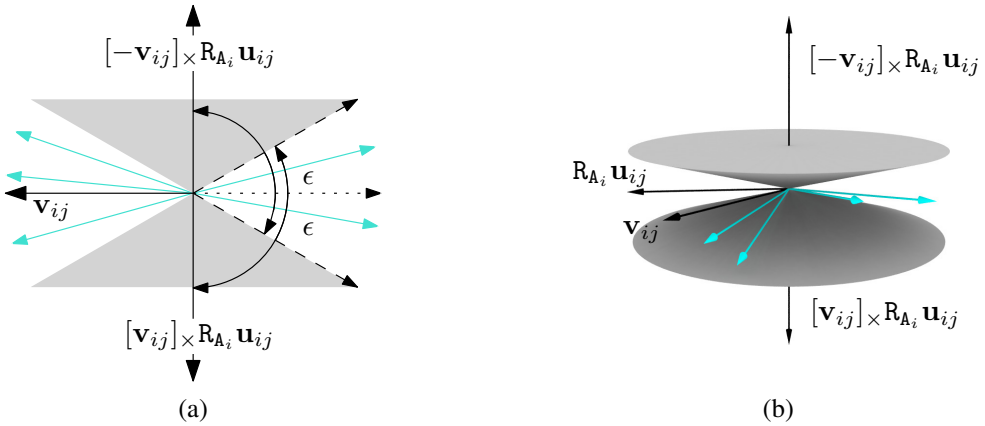


Figure 8.2: (a) Geometric interpretation of the conic constraint with parameter ϵ imposed by the correspondence $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$ on the position of $\mathbf{t}_{\mathtt{A}_i}$. The cyan vectors show examples of the admissible configurations of $\mathbf{t}_{\mathtt{A}_i}$. (b) 3D view of the same.

see Figure 8.1. This system can be further decomposed to

$$\begin{aligned} \mathtt{R}_{\mathtt{A}_i}\mathtt{R}_{\mathtt{X}} &= \mathtt{R}_{\mathtt{X}}\mathtt{R}_{\mathtt{B}_i}, \\ \mathtt{R}_{\mathtt{A}_i}\mathbf{t}_{\mathtt{X}} + \mathbf{t}_{\mathtt{A}_i} &= \mathtt{R}_{\mathtt{X}}\mathbf{t}_{\mathtt{B}_i} + \mathbf{t}_{\mathtt{X}}, \end{aligned}$$

where $\mathtt{R}_{\mathtt{A}_i}, \mathtt{R}_{\mathtt{B}_i} \in \mathrm{SO}(3)$ and $\mathbf{t}_{\mathtt{A}_i}, \mathbf{t}_{\mathtt{B}_i} \in \mathbb{R}^3$. By substituting $\mathbf{t}'_{\mathtt{X}} = -\mathtt{R}_{\mathtt{X}}^\top \mathbf{t}_{\mathtt{X}}$ and isolating $\mathtt{R}_{\mathtt{A}_i}$ and $\mathbf{t}_{\mathtt{A}_i}$, respectively, we get

$$\begin{aligned} \mathtt{R}_{\mathtt{A}_i} &= \mathtt{R}_{\mathtt{X}}\mathtt{R}_{\mathtt{B}_i}\mathtt{R}_{\mathtt{X}}^\top, \\ \mathbf{t}_{\mathtt{A}_i} &= \mathtt{R}_{\mathtt{X}}\left(\left(\mathtt{R}_{\mathtt{B}_i} - \mathtt{I}\right)\mathbf{t}'_{\mathtt{X}} + \mathbf{t}_{\mathtt{B}_i}\right). \end{aligned}$$

Further, suppose that the camera measured m correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$, $j = 1, \ldots, m$ in the i-th motion. Again, we will assume that the camera's internal calibration is known [66] and that $\mathbf{u}_{ij}, \mathbf{v}_{ij} \in \mathbb{R}^3$ are unit vectors representing the directions to scene points from the respective camera positions. We will also assume that the correspondences satisfy the *cheirality condition* [66], *i.e.*, that $\mathbf{u}_{ij}, \mathbf{v}_{ij}$ correspond to scene points $\mathbf{Y}_{ij} \in \mathbb{R}^3$ that lie in front of the cameras.

Let us consider an elementary fact from the geometry of stereo vision known as the *epipolar constraint* [66]. It states that vectors $\mathbf{u}_{ij}$ and $\mathbf{v}_{ij}$ form a correspondence for camera motion $\mathtt{A}_i$, if $\mathbf{t}_{\mathtt{A}_i}$ lies in the plane containing the two vectors. This fact is commonly expressed in the form

$$\mathbf{t}_{\mathtt{A}_i}^\top \left(\mathbf{v}_{ij} \times \left(\mathtt{R}_{\mathtt{A}_i}\mathbf{u}_{ij}\right)\right) = 0.$$

Since we will work here with angular measurements, it is convenient to equivalently rephrase the constraint as

$$e_{ij} = \angle\left(\left[\mathbf{v}_{ij}\right]_\times \mathtt{R}_{\mathtt{A}_i}\mathbf{u}_{ij}, \mathbf{t}_{\mathtt{A}_i}\right) - \frac{\pi}{2} = 0.$$

In case of noisy measurements, however, the epipolar constraint will not hold, *i.e.*, e_{ij} won't generally be zeros, and the values of $|e_{ij}|$ will encode the angular deviations of the estimated camera translation $\mathbf{t}_{\mathtt{A}_i}$ from the epipolar planes defined by correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$.

This leads us to defining an ϵ-*epipolar constraint*. The correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$ and camera translation $\mathbf{t}_{\mathtt{A}_i}$ satisfy the ϵ-epipolar constraint with parameter $\epsilon > 0$, iff $|e_{ij}| \leq \epsilon$. This constraint can be equivalently expressed as

$$\frac{\pi}{2} - \epsilon \leq \angle([\mathbf{v}_{ij}]_\times \mathtt{R}_{\mathtt{A}_i} \mathbf{u}_{ij}, \mathbf{t}_{\mathtt{A}_i}) \leq \frac{\pi}{2} + \epsilon,$$

i.e., $\mathbf{t}_{\mathtt{A}_i}$ has to lie outside of the cone determined by axis $[\mathbf{v}_{ij}]_\times \mathtt{R}_{\mathtt{A}_i} \mathbf{u}_{ij}$ and aperture $\pi - 2\epsilon$, see Figure 8.2. Note here that as a complement of a double cone ϵ-epipolar constraint is not a convex constraint. Let us rewrite the left inequality as

$$\frac{\pi}{2} - \epsilon \leq \angle([\mathbf{v}_{ij}]_\times \mathtt{R}_{\mathtt{A}_i} \mathbf{u}_{ij}, \mathbf{t}_{\mathtt{A}_i}) \Leftrightarrow \frac{\pi}{2} + \epsilon \geq \angle(-[\mathbf{v}_{ij}]_\times \mathtt{R}_{\mathtt{A}_i} \mathbf{u}_{ij}, \mathbf{t}_{\mathtt{A}_i}).$$

Now, we can formulate hand-eye calibration as the minimization of the ϵ-epipolar constraint, *i.e.*, as L_∞-norm minimization of the vector of residuals $\mathbf{e} = (|e_{11}|, \ldots, |e_{nm}|)$:

Problem 8.1.

$$(\hat{\mathtt{R}}_\mathtt{X}, \hat{\mathbf{t}}'_\mathtt{X}) = \underset{\mathtt{R}_\mathtt{X}, \mathbf{t}'_\mathtt{X}}{\arg\min} \max_{i,j} |e_{ij}| = \underset{\mathtt{R}_\mathtt{X}, \mathbf{t}'_\mathtt{X}}{\arg\min} \|\mathbf{e}\|_\infty .$$

The optimal residual error can be expressed as $\epsilon_{\min} = \left\|\mathbf{e}(\hat{\mathtt{R}}_\mathtt{X}, \hat{\mathbf{t}}'_\mathtt{X})\right\|_\infty$. After solving Problem 8.1, the optimal translation is determined as $\hat{\mathbf{t}}_\mathtt{X} = -\hat{\mathtt{R}}_\mathtt{X}\hat{\mathbf{t}}'_\mathtt{X}$. This substitution may seem superfluous, but it will allow us to prove Lemma 8.5.

In order to solve Problem 8.1 we employ branch-and-bound optimization to search over the space of all rotations presented in [69]. The algorithm is based on the angle-axis parameterization of rotations, see Section 3.4.2. Let $\boldsymbol{\alpha} \in \mathbb{B}_\pi = \{\boldsymbol{\beta} : \boldsymbol{\beta} \in \mathbb{R}^3 \wedge \|\boldsymbol{\beta}\| \leq \pi\}$, then $\boldsymbol{\alpha}$ represents the rotation about axis $\boldsymbol{\alpha}/\|\boldsymbol{\alpha}\|$ by angle $\|\boldsymbol{\alpha}\|$. The corresponding matrix parametrization $\mathtt{R} \in \mathrm{SO}(3)$ can be obtained as $\mathtt{R} = \exp[\boldsymbol{\alpha}]_\times$. The inverse map is given by $[\boldsymbol{\alpha}]_\times = \log \mathtt{R}$.

For $\mathtt{R}_1, \mathtt{R}_2 \in \mathrm{SO}(3)$ we define the distance $d_\angle(\mathtt{R}_1, \mathtt{R}_2)$ as the angle θ of the rotation $\mathtt{R}_1^\top \mathtt{R}_2$, *i.e.*, $[\boldsymbol{\alpha}]_\times = \log(\mathtt{R}_1^\top \mathtt{R}_2)$, such that $0 \leq \theta = \|\boldsymbol{\alpha}\| \leq \pi$. In the following, we will use the notation "$\mathtt{R} \in D$, $D \subset \mathbb{B}_\pi$" to mean

$$\mathtt{R} \in \left\{\mathtt{R}' \in \mathrm{SO}(3) \colon \exists \boldsymbol{\alpha} \in D \text{ such that } \mathtt{R}' = \exp[\boldsymbol{\alpha}]_\times\right\}.$$

8.2 Branch and Bound

Let us consider Problem 8.1 restricted to D_σ, where $D_\sigma \subset \mathbb{B}_\pi$ is a cubic block in the rotation space with side length 2σ (Figure 8.3a) :

Problem 8.2.

$$(\hat{\mathtt{R}}_\mathtt{X}, \hat{\mathbf{t}}'_\mathtt{X}) = \underset{\mathtt{R}_\mathtt{X} \in D_\sigma, \mathbf{t}'_\mathtt{X}}{\arg\min} \max_{i,j} |e_{ij}| .$$

The schematic branch-and-bound algorithm for solving Problem 8.1 is now as follows:

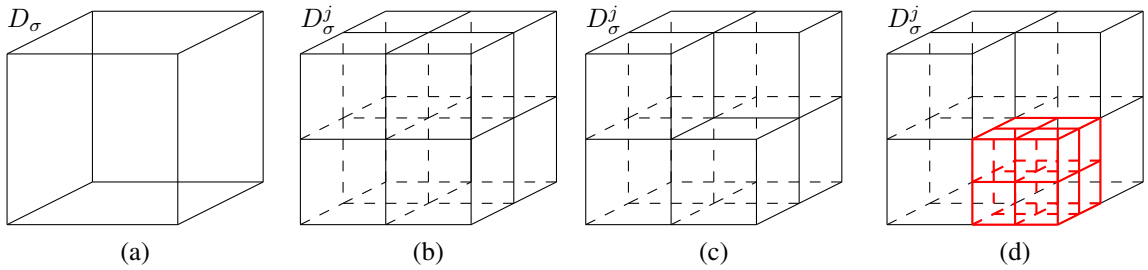


Figure 8.3: Branch-and-bound over the space of rotations.

1. Obtain an initial estimate of $\epsilon_{\min}$ for the optimal residual error of Problem 8.1.

2. Divide up the space of rotations into cubic subblocks D_σ^s (Figure 8.3b), $s = 1, \ldots, 8$ and repeat the following steps:

   (a) For each block D_σ^s test whether there exists a solution to Problem 8.2 restricted on D_σ^s having the residual error smaller than $\epsilon_{\min}$. This test can be formulated as a *feasibility test*, see Section 8.3.

   (b) If the answer to the test is no, throw the block away (Figure 8.3c).

   (c) Otherwise, evaluate the residual error ϵ for some rotation from block D_σ^s. If $\epsilon < \epsilon_{\min}$ then update the value $\epsilon_{\min} \leftarrow \epsilon$. Subdivide D_σ^s into eight cubic subblocks and continue to (a) (Figure 8.3d).

The iteration loop is terminated when the half-size of the blocks σ reaches a sufficiently small size $\sigma_{\min}$.

Note that although Problem 8.1 has 6 degrees of freedom, we search only over the three dimensional space of rotations. By limiting rotations in angle-axis parametrization to D_σ we are able to decide the feasibility test for Problem 8.2 effectively and optimally using LP. The LP solution also provides $\mathbf{t}'_{\mathtt{X}}$ needed to compute the residual error in step (c) and thus there is no need to search over the space of translations.

8.3 Feasibility Test

In this section the feasibility test based on Problem 8.2 is formulated. First, let us introduce a few more technical shorthands:

$$\begin{aligned}
\bar{\mathtt{R}}_{\mathtt{A}_i} &= \bar{\mathtt{R}}_{\mathtt{X}} \mathtt{R}_{\mathtt{B}_i} \bar{\mathtt{R}}_{\mathtt{X}}^{\top}, \\
\bar{\mathbf{t}}_{\mathtt{A}_i} &= \bar{\mathtt{R}}_{\mathtt{X}} \left((\mathtt{R}_{\mathtt{B}_i} - \mathtt{I})\, \mathbf{t}'_{\mathtt{X}} + \mathbf{t}_{\mathtt{B}_i} \right), \\
\hat{\mathtt{R}}_{\mathtt{A}_i} &= \hat{\mathtt{R}}_{\mathtt{X}} \mathtt{R}_{\mathtt{B}_i} \hat{\mathtt{R}}_{\mathtt{X}}^{\top}, \\
\hat{\mathbf{t}}_{\mathtt{A}_i} &= \hat{\mathtt{R}}_{\mathtt{X}} \left((\mathtt{R}_{\mathtt{B}_i} - \mathtt{I})\, \mathbf{t}'_{\mathtt{X}} + \mathbf{t}_{\mathtt{B}_i} \right).
\end{aligned}$$

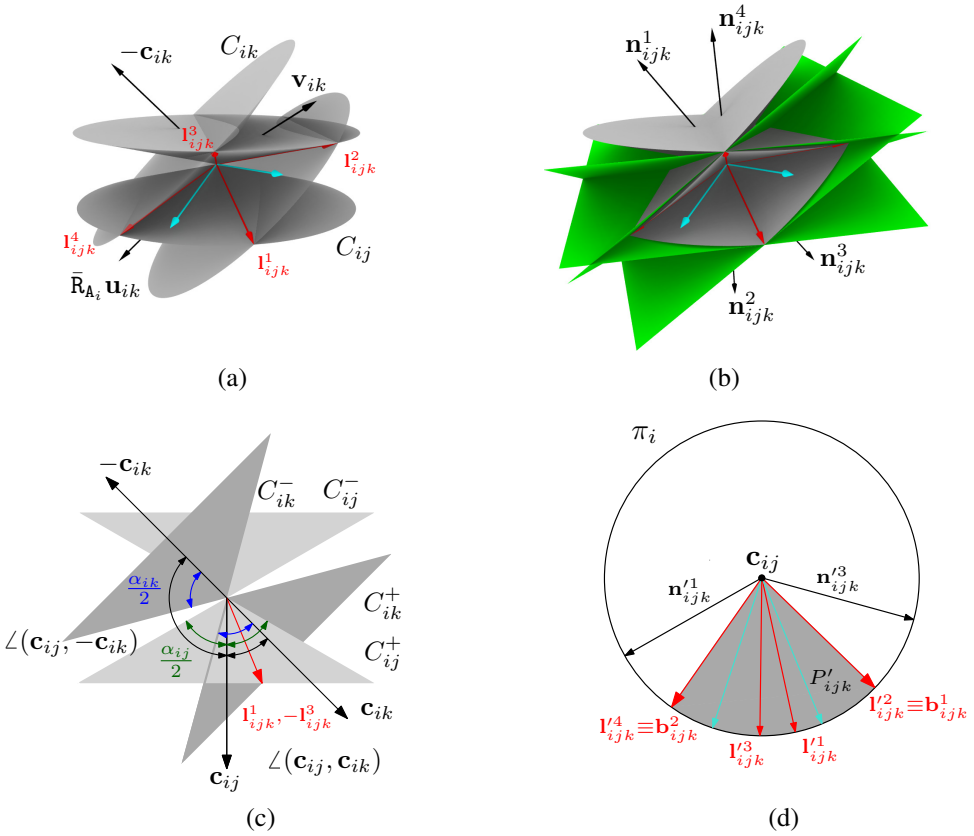


Figure 8.4: (a) If the apertures α_j, α_k are sufficiently large, cones C_{ij} and C_{ik} intersect in up to four lines $\mathbf{l}^1_{ijk}, \mathbf{l}^2_{ijk}, \mathbf{l}^3_{ijk}, \mathbf{l}^4_{ijk}$. (b) The linear constraints imposed by correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$, $\mathbf{u}_{ik} \leftrightarrow \mathbf{v}_{ik}$ are determined by the normals $\mathbf{n}^1_{ijk}, \mathbf{n}^2_{ijk}, \mathbf{n}^3_{ijk}, \mathbf{n}^4_{ijk}$. (c) Since Inequality 8.2 holds, the nappes C^+_{ij}, C^+_{ik} intersect in $\mathbf{l}^1_{ijk}$ and $-\mathbf{l}^3_{ijk}$. However, Inequality 8.3 does not hold and the cones C_{ij}, C_{ik} do not form the pyramid P_{ijk}. (d) The projection of the pyramid P_{ijk} into the plane π_i determined the boundary vectors $\mathbf{b}^1_{ijk}, \mathbf{b}^2_{ijk}$ (2D projection of the situation in (b)).

8.3.1 Feasibility Test Formulation

The following is Problem 8.2 rephrased as a feasibility test:

Problem 8.3.

$$\begin{aligned} \text{Given} \quad & D_\sigma, \epsilon_{\min} \\ \text{do there exist} \quad & \mathtt{R}_\mathtt{X} \in D_\sigma, \mathbf{t}'_\mathtt{X} \\ \text{such that} \quad & \angle(\pm [\mathbf{v}_{ij}]_\times \mathtt{R}_{\mathtt{A}_i} \mathbf{u}_{ij}, \mathbf{t}_{\mathtt{A}_i}) \leq \tfrac{\pi}{2} + \epsilon_{\min} \\ \text{for} \quad & i = 1, \ldots, n, \; j = 1, \ldots, m \,? \end{aligned}$$

Problem 8.3 is a non-convex problem—its feasible set is an intersection of non-convex ϵ-epipolar constraints—and as such is hard to solve. In order to do so, we start by bringing down the number of variables. We formulate a relaxation of Problem 8.3 where the rotation is fixed. Let $\bar{\mathtt{R}}_\mathtt{X}$ be the rotation represented by the center of cube D_σ and let $\boldsymbol{\beta}_i \in \mathbb{B}_\pi$ such that $\exp [\boldsymbol{\beta}_i]_\times = \mathtt{R}_{\mathtt{B}_i}$.

Problem 8.4.

$$\begin{aligned}
\text{Given} \quad & D_\sigma, \epsilon_{\min}, \bar{\mathtt{R}}_{\mathtt{X}} \\
\text{does there exist} \quad & \mathbf{t}'_{\mathtt{X}} \\
\text{such that} \quad & \angle(\pm\left[\mathbf{v}_{ij}\right]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, \bar{\mathbf{t}}_{\mathtt{A}_i}) \leq \tfrac{\pi}{2} + \epsilon_{\min} + \gamma_{ij} \\
\text{for} \quad & i = 1, \ldots, n,\ j = 1, \ldots, m \\
\text{such that} \quad & \angle(\pm\mathbf{v}_{ij}, \bar{\mathtt{R}}_{\mathtt{A}}\mathbf{u}_{ij}) > 2\left\|\boldsymbol{\beta}_i\right\| \sin(\sqrt{3}\sigma/2)\ ?
\end{aligned}$$

Lemma 8.5 describes the relation between Problems 8.3 and 8.4 and its proof provides the exact formulations of bounds γ_{ij}.

Lemma 8.5 (Relation between Problems 8.3 and 8.4). *If Problem 8.3 is feasible, so is Problem 8.4. If Problem 8.3 is infeasible, then D_σ may be split into subdomains $D^s_{\sigma'}$ of sufficiently small half-side length σ' such that Problem 8.4 is infeasible in every $D^s_{\sigma'}$.*

Note that Problem 8.4 contains one more set of constraints. These are prerequisites of Lemma 8.15, see Section 8.6, which in turn is needed for the proof Lemma 8.5. Since this proof is rather technical, it is deferred to Section 8.6.

Lemma 8.5 justifies the substitution of the hard 6-degrees-of-freedom Problem 8.3 by 3-degrees-of-freedom Problem 8.4 as the feasibility test for branch-and-bound algorithm. Albeit easier, Problem 8.4 is still a non-convex problem. In the following section we propose yet another relaxation of Problem 8.3. This time however, the relaxed problem will be convex and easily decidable by Linear Programming.

8.3.2 Intersecting ϵ-epipolar Constraints

Let once again $D_\sigma \subset \mathbb{B}_\pi$ be a cubic block and $\bar{\mathtt{R}}_{\mathtt{X}}$ the rotation represented by the center of the block. In Problem 8.4, a correspondence $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$ imposes a ϵ-epipolar constraint on $\bar{\mathbf{t}}_{\mathtt{A}_i}$ determined by the cone C_{ij} with axis $\mathbf{c}_{ij} = \left[\mathbf{v}_{ij}\right]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}$ and aperture $\alpha_{ij} = \pi - 2(\epsilon_{\min} + \gamma_{ij})$. Another correspondence $\mathbf{u}_{ik} \leftrightarrow \mathbf{v}_{ik}$ from the same i-th motion imposes a different ϵ-epipolar constraint, this time determined by the cone C_{ik} with axis $\mathbf{c}_{ik} = \left[\mathbf{v}_{ik}\right]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ik}$ and aperture $\alpha_{ik} = \pi - 2(\epsilon_{\min} + \gamma_{ik})$.

Now let us consider the mutual configuration of cones C_{ij} and C_{ik}. If the apertures α_{ij}, α_{ik} are sufficiently large, the two cones intersect, see Figure 9.5b. Since the cones share the same apex, they intersect in up to four lines—generatrices of the both cones—$\mathbf{l}^1_{ijk}$, $\mathbf{l}^2_{ijk}$, $\mathbf{l}^3_{ijk}$, and $\mathbf{l}^4_{ijk}$. These lines form the edges of a pyramid P_{ijk} in which every vector satisfying *both* ϵ-epipolar constraints has to lie. The pyramid P_{ijk} has four faces lying in four planes, see Figure 8.4b. These planes can be determined by their normals $\mathbf{n}^1_{ijk}$, $\mathbf{n}^2_{ijk}$, $\mathbf{n}^3_{ijk}$, and $\mathbf{n}^4_{ijk}$ as

$$\begin{aligned}
\mathbf{n}^1_{ijk} &= \left[\mathbf{l}^1_{ijk}\right]_\times \mathbf{l}^2_{ijk}, \\
\mathbf{n}^2_{ijk} &= \left[\mathbf{l}^2_{ijk}\right]_\times \mathbf{l}^3_{ijk}, \\
\mathbf{n}^3_{ijk} &= \left[\mathbf{l}^3_{ijk}\right]_\times \mathbf{l}^4_{ijk}, \\
\mathbf{n}^4_{ijk} &= \left[\mathbf{l}^4_{ijk}\right]_\times \mathbf{l}^1_{ijk}.
\end{aligned}$$

It is easy to see that every vector $\bar{\mathbf{t}}_{\mathtt{A}_i}$ that satisfies both ϵ-epipolar constrains satisfies all of the following linear constraints as well

$$\begin{aligned} \bar{\mathbf{t}}_{\mathtt{A}_i}^\top \mathbf{n}_{ijk}^1 &\geq 0, \\ \bar{\mathbf{t}}_{\mathtt{A}_i}^\top \mathbf{n}_{ijk}^2 &\geq 0, \\ \bar{\mathbf{t}}_{\mathtt{A}_i}^\top \mathbf{n}_{ijk}^3 &\geq 0, \\ \bar{\mathbf{t}}_{\mathtt{A}_i}^\top \mathbf{n}_{ijk}^4 &\geq 0. \end{aligned} \tag{8.1}$$

Note that these constraints are also linear in $\mathbf{t}'_{\mathtt{X}}$.

On the other hand, if the apertures α_{ij}, α_{ik} are too small, cones C_{ij} and C_{ik} don't necessarily have to intersect and the ϵ-epipolar constraints cannot be replaced by the linear ones. Figure 8.4c will help us to determine when exactly such a situation arises. Let $C_{ij}^+, C_{ij}^-, C_{ik}^+, C_{ik}^-$ be the nappes of the cones C_{ij}, C_{ik} determined by the vectors $\mathbf{c}_{ij}$, $-\mathbf{c}_{ij}$, $\mathbf{c}_{ik}$, $-\mathbf{c}_{ik}$ respectively. The nappes C_{ij}^+, C_{ik}^+ intersect in two line segments $\mathbf{l}_{ijk}^1$ and $-\mathbf{l}_{ijk}^3$ iff

$$\angle(\mathbf{c}_{ij}, \mathbf{c}_{ik}) \;<\; \frac{\alpha_{ij} + \alpha_{ik}}{2}. \tag{8.2}$$

In case of equality of the terms in 8.2 the nappes are tangential and share only one common generatrix. Analogously, the line segments $\mathbf{l}_{ijk}^2$ and $-\mathbf{l}_{ijk}^4$ are the intersections of the nappes C_{ij}^+, C_{ik}^- iff

$$\angle(\mathbf{c}_{ij}, -\mathbf{c}_{ik}) \;<\; \frac{\alpha_{ij} + \alpha_{ik}}{2}. \tag{8.3}$$

Inequalities 8.2 and 8.3 are thus necessary and sufficient conditions for C_{ij}, C_{ik} to form the pyramid P_{ijk}.

Note that the intersection of cones C_{ij} and C_{ik} determines not only the pyramid P_{ijk}, but also a pyramid P'_{ijk} symmetrical to P_{ijk}, with the origin as the point of symmetry. However, since we assumed that the correspondences satisfy the cheirality condition, only constraints relevant to one of the pyramids are applicable. In the following we will assume, without loss of generality, that P_{ijk} forms the applicable constraints. Formulas for lines $\mathbf{l}_{ijk}^1$, $\mathbf{l}_{ijk}^2$, $\mathbf{l}_{ijk}^3$, $\mathbf{l}_{ijk}^4$ can be obtained by using elementary algebra.

Now we can formulate a linear relaxation of Problem 8.4, which in turn is a non-convex relaxation of Problem 8.3.

Problem 8.6.

$$\begin{aligned} \text{Given} \quad & D_\sigma, \epsilon_{\min}, \bar{\mathtt{R}}_{\mathtt{X}} \\ \text{does there exist} \quad & \mathbf{t}'_{\mathtt{X}} \\ \text{such that} \quad & \bar{\mathbf{t}}_{\mathtt{A}_i}^\top \mathbf{n}_{ijk}^1 \geq 0,\ \bar{\mathbf{t}}_{\mathtt{A}_i}^\top \mathbf{n}_{ijk}^2 \geq 0 \\ & \bar{\mathbf{t}}_{\mathtt{A}_i}^\top \mathbf{n}_{ijk}^3 \geq 0,\ \bar{\mathbf{t}}_{\mathtt{A}_i}^\top \mathbf{n}_{ijk}^4 \geq 0 \\ \text{for} \quad & i = 1, \ldots, n,\ j, k = 1, \ldots, m \\ \text{such that} \quad & \angle(\pm \mathbf{v}_{ij}, \bar{\mathtt{R}}_{\mathtt{A}} \mathbf{u}_{ij}) > 2 \left\| \boldsymbol{\beta}_i \right\| \sin(\sqrt{3}\sigma/2) \\ \text{and} \quad & \text{Inequalities 8.2 and 8.3 hold?} \end{aligned}$$

An analogy to the formulation of Lemma 8.5 holds for Problems 8.4 and 8.6. Its proof follows from the geometrical construction of the linear constraints as the convex hull of the ϵ-epipolar constraints.

8.3.3 Selecting ϵ-epipolar Constraints

It is easy to see that a naive implementation of Problem 8.6 would lead to prohibitively large linear programs for problems with many correspondences. In this section we show how to reduce the number of linear constraints by selecting only few of the available correspondences based on their relative positions.

Let us consider the i-th motion for a cubic block $D_\sigma \subset \mathbb{B}_\pi$. First, let $\mathbf{C}_i$ be the list of indexes of correspondences satisfying the assumptions of Lemma 8.15. Now, let us assume, without loss of generality, that $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$, $j \in \mathbf{C}_i$ is the correspondence with the widest ϵ-epipolar constraint aperture α_{ij}. In the rest of this section, we will call the j-th constraint the *base constraint*. Next, let's throw away from $\mathbf{C}_i$ the indexes k standing for correspondences $\mathbf{u}_{ik} \leftrightarrow \mathbf{v}_{ik}$ which do not intersect with the base constraint, *i.e.*, indexes for which Inequalities 8.2 and 8.3 do not hold. Further, let us sort $\mathbf{C}_i$ according to the distance of the cones axes $|\mathbf{c}_{ij}^\top \mathbf{c}_{ik}|$,$k \in \mathbf{C}_i$ in the ascending order. The idea is to test the correspondences in $\mathbf{C}_i$ that are "more perpendicular" to the base constraint first, since they are more likely to form a tighter pyramid P_{ijk} and thus tighter linear bounds. Also, since the smaller the value $|\mathbf{c}_{ij}^\top \mathbf{c}_{ik}|$ gets, the more loose and skewed pyramid P_{ijk} becomes—it is therefore reasonable to test only first $s \ll |\mathbf{C}_i|$ correspondences. In our experiments we set $s = 20$.

Algorithm 3 SelectEpipolarConstraints (Section 8.3.3)

Require: $i, \mathbf{C}_i$
  $\mathit{feasible} \leftarrow 1, s \leftarrow 0, \mathbf{L}_i, \mathbf{b}_i^1, \mathbf{b}_i^2 \leftarrow \emptyset, j \leftarrow \arg\max_k \alpha_{ik}$
  $\mathbf{C}_i \leftarrow \mathbf{C}_i \backslash \{j\}$ sorted s.t. $\forall k < |\mathbf{C}_i| : |\mathbf{c}_{ij}^\top \mathbf{c}_{ik}| \leq |\mathbf{c}_{ij}^\top \mathbf{c}_{ik+1}|$
  while $(|\mathbf{C}_i| > 0) \wedge (s <$ `MAX_CONSTRAINTS`$)$ **do**
    $k \leftarrow \text{PopFront}(\mathbf{C}_i)$
    if $\angle(\mathbf{c}_{ij}, \pm\mathbf{c}_{ik}) < (\alpha_{ij} + \alpha_{ik})/2$ **then**
      $(\mathbf{b}_{ijk}^1, \mathbf{b}_{ijk}^2) \leftarrow \text{GetBounds}(\mathbf{l}_{ijk}^1, \mathbf{l}_{ijk}^2, \mathbf{l}_{ijk}^3, \mathbf{l}_{ijk}^4, \mathbf{n}_{ijk}^1, \mathbf{n}_{ijk}^3)$
      if $\mathbf{b}_i^1, \mathbf{b}_i^2 \equiv \emptyset$ **then**
        $(\mathbf{b}_i^1, \mathbf{b}_i^2) \leftarrow (\mathbf{b}_{ijk}^1, \mathbf{b}_{ijk}^2)$
        $\mathbf{L}_i \leftarrow \mathbf{L}_i \cup \{\langle i, \mathbf{n}_{ijk}^1 \rangle, \langle i, \mathbf{n}_{ijk}^2 \rangle, \langle i, \mathbf{n}_{ijk}^3 \rangle, \langle i, \mathbf{n}_{ijk}^4 \rangle\}$
      else
        $(\mathbf{b}_i^1, \mathbf{b}_i^2) \leftarrow \text{IntersectBounds}(\mathbf{b}_i^1, \mathbf{b}_i^2, \mathbf{b}_{ijk}^1, \mathbf{b}_{ijk}^2)$
        if $\mathbf{b}_i^1, \mathbf{b}_i^2 \equiv \emptyset$ **then**
          $\mathit{feasible} \leftarrow 0$
          return $\{\mathit{feasible}, \mathbf{L}_i\}$
        else
          $s \leftarrow s + 1$
          $\mathbf{L}_i \leftarrow \mathbf{L}_i \cup \{\langle i, \mathbf{n}_{ijk}^1 \rangle, \langle i, \mathbf{n}_{ijk}^3 \rangle\}$
        end if
      end if
    end if
  end while
  return $\{\mathit{feasible}, \mathbf{L}_i\}$

The second reason for intersecting all ϵ-epipolar constraints with the base constraints is that by projecting pyramids P_{ijk} into the plane π_i defined by vectors $\bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}$ and $\mathbf{v}_{ij}$, a simple test based on 2D feasibility of $\mathbf{t}_{\mathtt{A}_i}$ can be constructed. Let $\mathbf{l}'^1_{ijk}$, $\mathbf{l}'^2_{ijk}$, $\mathbf{l}'^3_{ijk}$, $\mathbf{l}'^4_{ijk} \in \mathbb{R}^2$ be normalized 2D projections of the edges of the pyramid P_{ijk} into the plane π_i, see Figure 8.4d. It is a trivial observation that the projection of the pyramid P'_{ijk} will be bounded by two vectors $\mathbf{b}^1_{ijk}, \mathbf{b}^2_{ijk}$ that will coincide with two of the projected edges. To decide which edges will form the boundaries, projections $\mathbf{n}'^1_{ijk}, \mathbf{n}'^3_{ijk} \in \mathbb{R}^2$ of the pyramid's faces need to considered as well. Since the face $\mathbf{n}^1_{ijk}$ is defined by edges $\mathbf{l}^1_{ijk}$, $\mathbf{l}^2_{ijk}$, one of the projections $\mathbf{l}'^1_{ijk}$, $\mathbf{l}'^2_{ijk}$ forming larger angle with the projection $\mathbf{n}'^1_{ijk}$ will form the boundary. In the case of the situation in Figure 8.4d, $\mathbf{l}'^{1\top}_{ijk}\mathbf{n}'^1_{ijk} > \mathbf{l}'^{2\top}_{ijk}\mathbf{n}'^1_{ijk}$ so the $\mathbf{b}^1_{ijk}$ coincides with $\mathbf{l}'^2_{ijk}$. Analogously, we can use $\mathbf{n}'^3_{ijk}$ to decide that $\mathbf{b}^2_{ijk}$ coincides with $\mathbf{l}'^4_{ijk}$. Faces $\mathbf{n}^2_{ijk}$ and $\mathbf{n}^4_{ijk}$ cannot be used in this manner, since their projections to π_i are equally distant from the projection of the respective edges forming them. Since P_{ijk} forms constraints on the position of $\mathbf{t}_{\mathtt{A}_i}$, it is easy to see that P'_{ijk} forms constraints on the projection $\mathbf{t}'_{\mathtt{A}_i}$. From this follows that the intersection of all P'_{ijk}, $k \in \mathbf{C}_i$ must not be empty for block D_σ to be feasible. The implication in the opposite direction does not hold (two projections P'_{ijk}, $P'_{ij\ell}$, $k, \ell \in \mathbf{C}_i$ can intersect, even though P_{ijk} and $P_{ij\ell}$ do not), so the block can still be infeasible even if the intersection is not empty. However, this simple pre-test can decide infeasibility for *ca.* 30% of infeasible cubes. Practically, the correspondences are processed sequentially in the order given by $\mathbf{C}_i$ and a "running intersection" of projections is kept as $\mathbf{b}^1_i, \mathbf{b}^2_i$, compared and updated with every upcoming projection, see Algorithm 3.

8.4 The Hand-Eye Calibration Algorithm

This section sums up the branch-and-bound algorithm for hand-eye calibration into a more comprehensible pseudo-code form.

First, let us review the hand-eye feasibility test, see Algorithm 4. For every cubic block D_σ the algorithm solves Problem 8.6. Due to the quite strict assumptions—Lemma 8.15 and Inequalities 8.2, 8.3—not all correspondences produce linear constraints. Indeed, for a large block there might be no feasible correspondences at all. Feasibility of such a block cannot be decided by Problem 8.6 and it has to be declared feasible by default. However, the smaller the blocks get, the more correspondences can produce linear constraints and Problem 8.6 is more likely to be decidable.

Further, because Problem 8.6 is a feasibility problem, an LP solver will generally provide a basic feasible solution that does not minimize the nonlinear objective function $\|\mathbf{e}\|_\infty$. In order to speed up the convergence of the algorithm, we use a feasible solution to Problem 8.6 as an initial estimate for the following non-linear optimization problem:

Problem 8.7.

$$\begin{aligned} \text{Given} \quad & \bar{\mathtt{R}}_{\mathtt{X}} \\ \text{minimize} \quad & \|\mathbf{e}\|_2^2 = \textstyle\sum_{i,j}(\angle([\mathbf{v}_{ij}]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, \bar{\mathbf{t}}_{\mathtt{A}_i}) - \frac{\pi}{2})^2 \\ \text{for} \quad & i = 1, \ldots, n,\ j, k = 1, \ldots, m \\ \text{subject to} \quad & \text{the initial estimate } \mathbf{t}'_{\mathtt{X}}. \end{aligned}$$

Algorithm 4 Feasibility Test

Require: $D, \epsilon_{\min} > 0$
  $\bar{\mathtt{R}}_{\mathtt{X}} \leftarrow$ rotation represented by the center of cube D
  $\sigma \leftarrow$ half-side length of D
  $\mathbf{L} \leftarrow \emptyset$

  for $i = 1$ **to** number of motions **do**
    // Collect feasible correspondences s.t. Lemma 8.15
    $\mathbf{C}_i \leftarrow \emptyset$
    for $j = 1$ **to** number of correspondences **do**
      if $\angle(\pm\mathbf{v}_{ij}, \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}) > 2\,\|\boldsymbol{\beta}_i\|\sin(\sqrt{3}\sigma/2)$ **then**
        $\mathbf{C}_i \leftarrow \mathbf{C}_i \cup \{j\}$
      end if
    end for

    // Collect linear constraints
    $(\mathit{feasible}, \mathbf{L}_i) \leftarrow$ SelectEpipolarConstraints$(i, \mathbf{C}_i)$
    if $\mathit{feasible} \equiv 0$ **then**
      return *feasible*
    else
      $\mathbf{L} \leftarrow \mathbf{L} \cup \mathbf{L}_i$
    end if
  end for

  if $\mathbf{L} \equiv \emptyset$ **then**
    $\mathit{feasible} \leftarrow 1$, **return** *feasible*
  end if

  // Solve Problem 8.6
  $\mathbf{t}'_{\mathtt{X}} \leftarrow \mathbf{t}'_{\mathtt{X}}$ such that $\forall\,\langle i, \mathbf{n}\rangle \in \mathbf{L}\colon \bar{\mathbf{t}}_{\mathtt{A}_i}^{\top}\mathbf{n} \geq 0$
  if such a $\mathbf{t}'_{\mathtt{X}}$ does not exist **then**
    $\mathit{feasible} \leftarrow 0$, **return** *feasible*
  else
    $\mathit{feasible} \leftarrow 1$
    $\epsilon \leftarrow \max_{i,j} |\angle([\mathbf{v}_{ij}]_{\times}\,\bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, \bar{\mathbf{t}}_{\mathtt{A}_i}) - \frac{\pi}{2}|$
  end if

  // Solve Problem 8.7
  $\tilde{\mathbf{t}}'_{\mathtt{X}} \leftarrow \min_{\mathbf{t}'_{\mathtt{X}}} \sum_{i,j} \left(\angle([\mathbf{v}_{ij}]_{\times}\,\bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, \bar{\mathbf{t}}_{\mathtt{A}_i}) - \frac{\pi}{2}\right)^2$
  $\tilde{\epsilon} \leftarrow \max_{i,j} |\angle([\mathbf{v}_{ij}]_{\times}\,\bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, \bar{\mathbf{t}}_{\mathtt{A}_i}(\tilde{\mathbf{t}}'_{\mathtt{X}})) - \frac{\pi}{2}|$
  if $\epsilon \leftarrow \tilde{\epsilon}$ **then** $\epsilon \leftarrow \tilde{\epsilon}$, $\mathbf{t}'_{\mathtt{X}} \leftarrow \tilde{\mathbf{t}}'_{\mathtt{X}}$ **end if**
  return $\{\mathit{feasible}, \epsilon, \bar{\mathtt{R}}_{\mathtt{X}}, \mathbf{t}'_{\mathtt{X}}\}$

Since Problem 8.7 minimizes $\|\mathbf{e}\|_2$, it does not necessarily have to provide a solution with better $\|\mathbf{e}\|_\infty$. However, we observe that it helps to accelerate the overall convergence.

Finally, let us conclude with the branch-and-bound part of the algorithm, see Algorithm 5. The algorithm is initialized by the cubic block $D_\pi = \langle -\pi, \pi \rangle^3$. Such a block is redundant since $\mathbb{B}_\pi \subsetneq \langle -\pi, \pi \rangle^3$, but this fact does not impair the algorithm, since blocks $D_\sigma \cap \mathbb{B}_\pi = \emptyset$ can be easily detected and thrown away. Feasible blocks are divided into 8 cubic subblocks and stored in a queue $\mathbf{Q}$, making the algorithm breadth-first search. The search is terminated when the size of the blocks σ reaches a sufficiently small size $\sigma_{\min}$.

The proposed algorithm is easily parallelizable by running the feasibility test in multiple threads consuming a mutual queue $\mathbf{Q}$. Note that the parallel processing of the cubes can theoretically result in accepting some cubes that would be rejected in single thread mode, making $\mathbf{Q}$ larger. However, the experiments show that this is not a practical issue and that the computation time is linear in the number of threads.

Algorithm 5 Branch and Bound

Require: initial estimate of $\epsilon_{\min}$, stopping crit. $\sigma_{\min}$
  $D_\pi \leftarrow \langle -\pi, \pi \rangle^3, \sigma \leftarrow 2\pi$
  PushBack($\mathbf{Q}, D_\pi$)
  while $\sigma > \sigma_{\min}$ **do**
    $D \leftarrow$ PopFront($\mathbf{Q}$), $\sigma \leftarrow$ half-side length of D
    $\{\mathit{feasible}, \epsilon, \bar{\mathtt{R}}_\mathtt{X}, \mathbf{t}'_\mathtt{X}\} \leftarrow$ FeasibilityTest($D, \epsilon_{\min}$)
    if *feasible* $\equiv$ **true then**
      if $\epsilon < \epsilon_{\min}$ **then**
        $\hat{\mathtt{R}}_\mathtt{X} \leftarrow \bar{\mathtt{R}}_\mathtt{X}$
        $\hat{\mathbf{t}}'_\mathtt{X} \leftarrow \mathbf{t}'_\mathtt{X}$
        $\epsilon_{\min} \leftarrow \epsilon$
      end if
      PushBack($\mathbf{Q}$, SubdivideBlock(D))
    end if
  end while
  $\hat{\mathbf{t}}_\mathtt{X} \leftarrow -\hat{\mathtt{R}}_\mathtt{X}\hat{\mathbf{t}}'_\mathtt{X}$
  return $\{\hat{\mathtt{R}}_\mathtt{X}, \hat{\mathbf{t}}_\mathtt{X}, \epsilon_{\min}\}$

8.5 Experimental Results

Next, the performance of the proposed algorithm is evaluated using both synthetically generated and real world data measurements. The values of the initial estimate $\epsilon_{\min}$ and the stopping criterion $2\sigma_{\min}$ were set to 0.02 and 0.001 respectively. We use GLPK [4] to solve Problem 8.6 and levmar [110] to solve nonlinear Problem 8.7. All the reported times were achieved using a C++ implementation on a 32×2.6 GHz AMD Opteron based computer running 64-bit Linux. The source code is available at `http://cmp.felk.cvut.cz/~hellej1/bbhec/`.

8.5.1 Experiment with Synthetic Data

Ball Experiment. A synthetic scene consisting of 100 3D points was generated into a ball of radius 1,000 mm. 10 absolute camera poses were set up so that (i) the centers of the cameras were outside the ball but close to its "surface", (ii) the centers were positioned so that the offsets of the camera motions would be ∼500 mm and (iii) the cameras faced approximately the center of the ball. In order to simulate the effect of decreasing field of view (FOV), 7 progressively smaller balls were generated inside the initial ball so that the balls shared the centers and the volume of the newly created ball was half the volume of the previous ball. This defined 8 FOV levels, namely 119°, 85°, 66°, 52°, 40°, 31°, 24°, and 19°. Additional 3D points were generated inside each of the smaller balls in order to have exactly 100 3D points at each FOV level measured in the respective cameras giving raise to correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$.

Further, fifty random transformations $\mathtt{X}$ were generated and the optimization tasks composed of the known correspondences $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$ and 9 motions $\mathtt{B}_i$—computed from the known absolute camera poses and the generated $\mathtt{X}$—were constructed for each of the 8 FOV levels. Finally, the correspondences were corrupted with Gaussian noise in the angular domain using 11 levels, $\sigma \in \langle 0, 10^{-3} \rangle$ in 10^{-4} steps resulting to 88 tasks per transformation.

The results of the experiment are shown in Figures 8.5a–e. Note that the median of the measured errors over all correspondences—denoted by red horizontal lines in Figure 8.5a—is approximately one order of magnitude smaller than the maximum residual error. The actual errors of the calibration, *i.e.*, the Euclidean distance between the 3D points transformed to the gripper's coordinate systems using ground truth $\mathtt{X}$ and the 3D points transformed to the gripper's coordinate systems using the estimated $\mathtt{X}$, are higher for narrow FOV. The reason is that the wide FOV situations contain correspondences that produce tighter linear bounds and thus result in higher accuracy. Considering the computational times, the solutions are found faster for wide FOV as the correspondences of narrow FOV camera pairs do not generate enough linear constraints for large blocks and more blocks need to be subdivided.

Planar Experiment. In order to compare our method to the previously proposed ones, we generated a planar calibration device consisting of a rectangular 11×11 grid with known 3D position.The camera poses were set up so that camera centers were ∼1,000 mm away from the calibration device and the correspondences were corrupted by the same amount of Gaussian noise as in Ball experiment. This time we generated twenty random transformations $\mathtt{X}$ for 3 standard FOV levels 64°, 47° and 35° (by varying calibration device size). In order to simulate the manufacturing and structural errors introduced while using a calibration device we also corrupted the 3D positions of the grid points with systematic errors. We multiplied the grid by 1.005 along the y-axis and deformed it using the function $\sin x \cos y$ along the z-axis, again, so as not to differ by more that 5‰ of the grid's side length from the original, see Figure 8.6b. Since most of the pre-existing methods require also the additional knowledge of the external camera poses, these were recovered using EPnP algorithm [101]. Figure 8.6a shows the results of the comparison using the same Euclidean distance measure as in Ball experiment. The labels "Tsai89", "Shiu89", "Park94", "Dan98", "Heller11", and "Zhao11" stand for methods [183, 149, 134, 44, 75], and [191], respectively. Label "Heller12" stands for the proposed method.

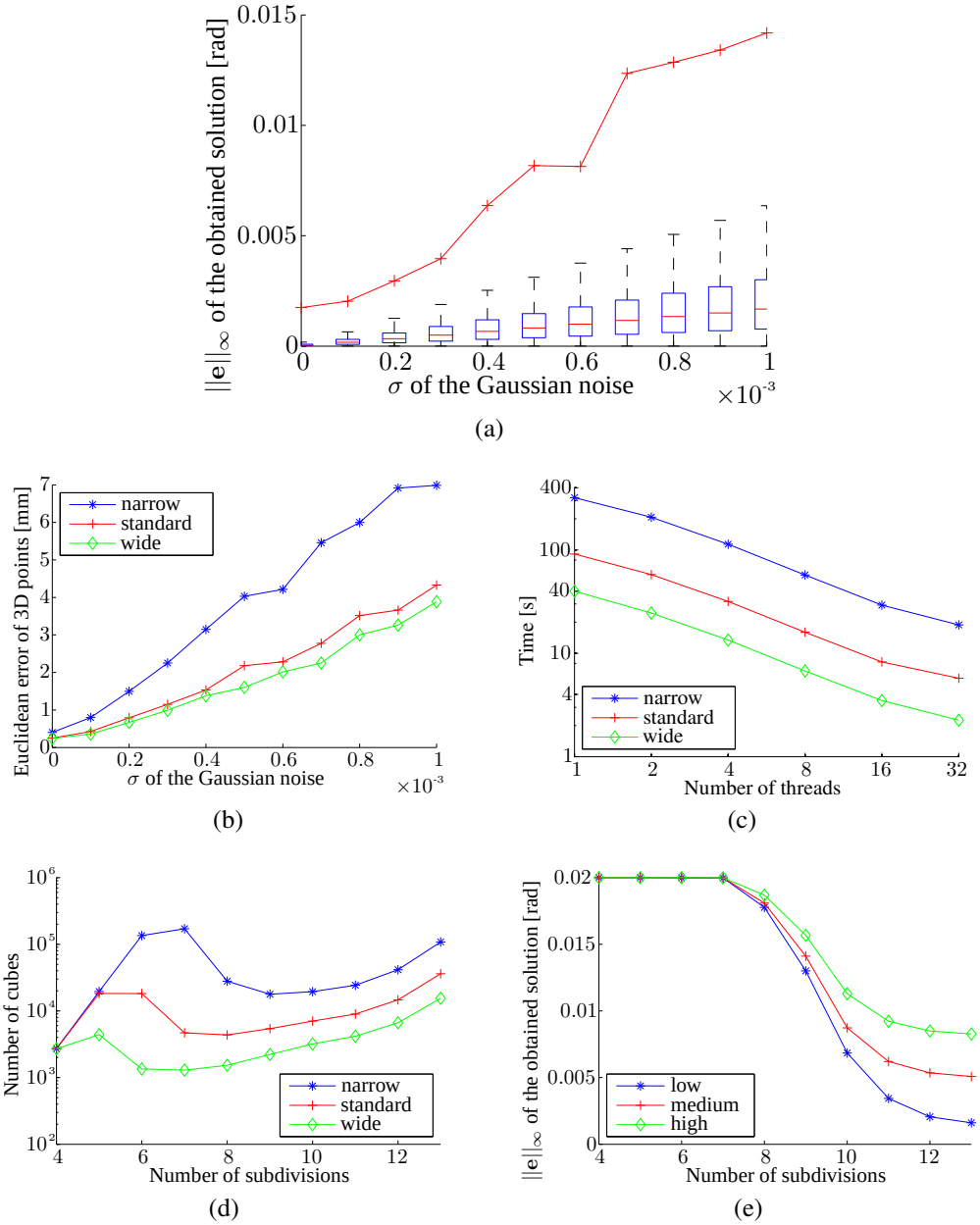


Figure 8.5: *Ball experiment.* (a) The maximum residual error of the obtained solutions for the various values of σ (red line) and the distribution of the measured errors over all correspondences (boxes). (b) The mean Euclidean distance between the 3D points transformed to the gripper's coordinate systems using ground truth `X` and the 3D points transformed to the gripper's coordinate systems using the estimated `X`. Different FOV levels were clustered into three groups. (c) Loglog plot of the computational time as a function of the number of threads. (d) The mean number of remaining cubes plotted against the number of subdivision phases. Note that the computation starts after the fourth subdivision. (e) The mean residual error at the beginning of the respective subdivision phase. Different noise levels were clustered into three groups.

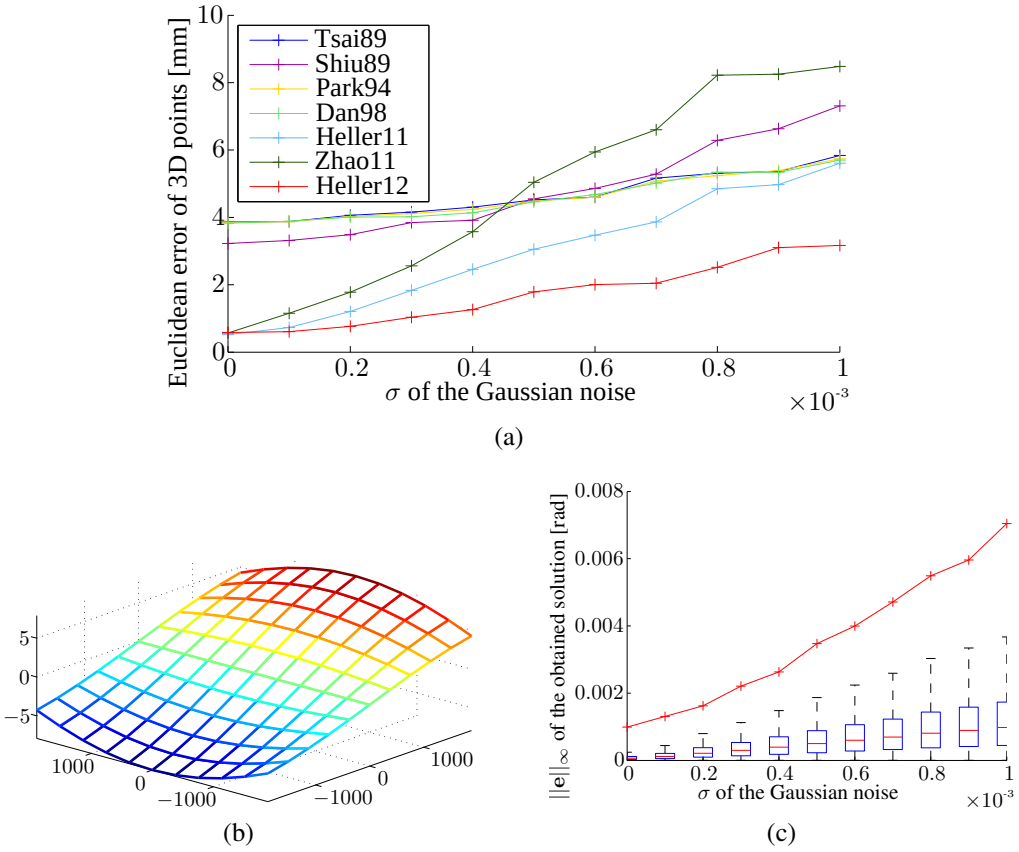


Figure 8.6: *Planar experiment.* (a) Comparison of the proposed method with pre-existing methods using Euclidean distance measure. (b) Grid deformation (shown out of proportions). (c) The maximum (red line) and median (boxes) residual error of the solutions for the various values of σ.

8.5.2 Experiment with Real Data

Motoman MA1400 Experiment. In the first real data experiment, an Asus Xtion Pro sensor was rigidly attached to the 5th link of a Motoman MA1400 serial 6-DOF manipulator, see Figure 8.7a. The manipulator was instructed into 18 poses $\mathtt{B}'_i$ and in each position a 640×480 picture of a calibration sheet consisting of 315 distinguishable dots was taken. A optimization task consisting of 9 relative movements and totaling 2,835 correspondences was constructed. The algorithm converged to a solution with residual error $\|\mathbf{e}\|_\infty = 0.003$ rad in 85 seconds (running in 8 threads).

In order to compare the result to the pre-existing methods, absolute camera poses $\mathtt{A}'_i$ were recovered using EPnP algorithm [101], see Figure 8.7b. Next, hand-eye transformations $\mathtt{X}_k$, $k = 1, \ldots, 6$ were computed by methods "Tsai89", "Shiu89", "Park94", "Dan98", "Heller11", and "Zhao11" using the same relative movements. Figure 8.8a shows a distribution of the Euclidean distances between the 3D points of the calibration device C and the 3D points of the calibration device transformed by the relative movements $\mathtt{A}'^{-1}_j \mathtt{X}_k \mathtt{B}'^{-1}_j \mathtt{B}'_i \mathtt{X}^{-1}_k \mathtt{A}'_i$, for $i, j = 1, \ldots, 18$. Note that this error measure is different from the one used in the synthetic experiment, since here the ground truth transformation $\mathtt{X}$ is not known. Also note that lower error values do not necessarily mean "better" $\mathtt{X}$, since both camera and robot calibra-

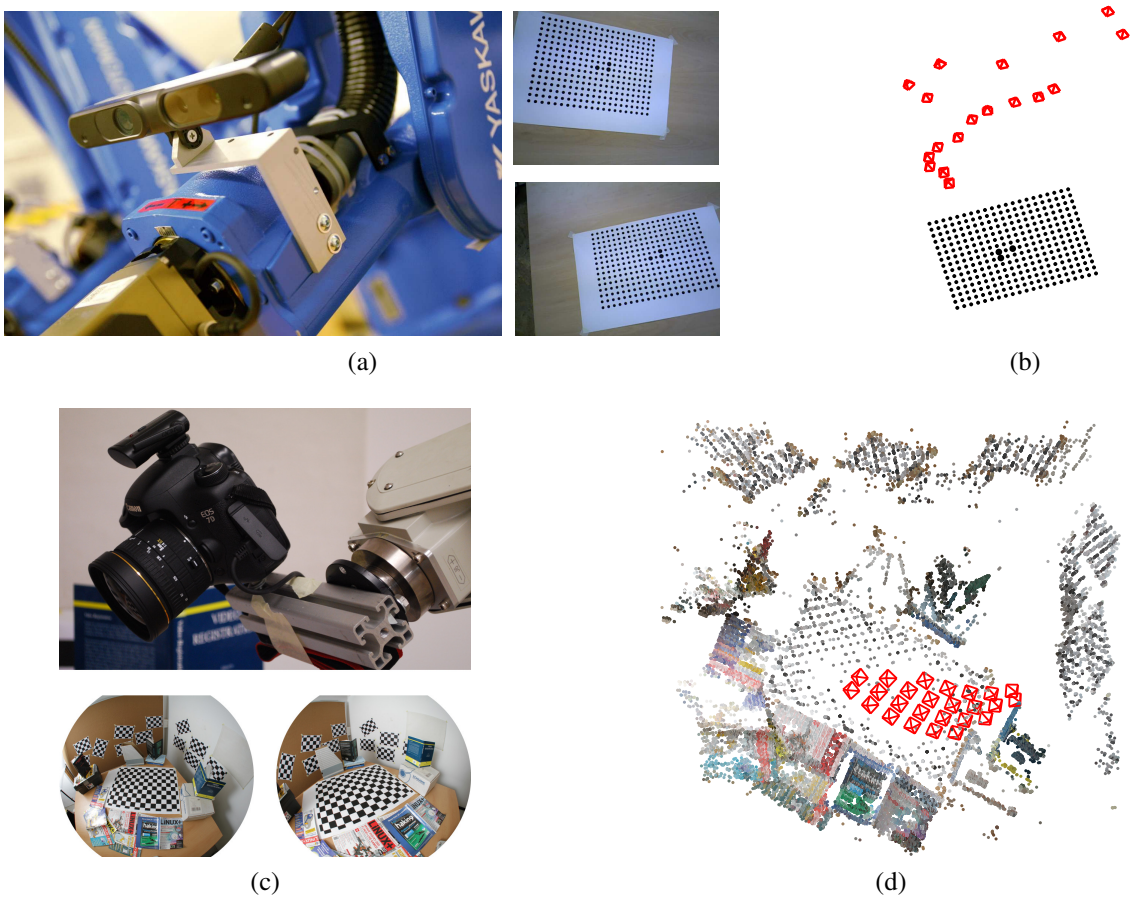


Figure 8.7: *Real data experiment*. (a–b) Motoman MA1400. Close up of the camera-gripper rig, sample images from the sequence; Camera poses reconstruction (c–d) Mitsubishi MELFA-RV-6S. Close up of the camera-gripper rig, sample images from the sequence; Model resulting from SfM, cameras are denoted by red pyramids.

tions can be slightly noisy. In this case, however, medians for all the methods—except for "Shiu89"—are well below 0.5 mm, validating the result by the proposed method labeled as "H12".

Mitsubishi MELFA-RV-6S Experiment. A Mitsubishi MELFA-RV-6S serial manipulator with a Canon 7D digital SLR camera and a Sigma 8 mm lens (pixel size $\sim$0.0011 rad, FOV $\sim$130°) were used to acquire data for the second real experiment. The robot was instructed to move the gripper along the surface of a sphere of radius $\sim$700 mm centered in the middle of the scene objects. The position of the gripper was adjusted to reach 25 different locations at four different pitch angles and the gripper was set to face the center of the sphere, see Figure 8.7c. The internal calibration of the camera was obtained from several images of a checkerboard using OCamCalib [142]

First, SfM software [177] was used to automatically generate correspondences, see Fig-

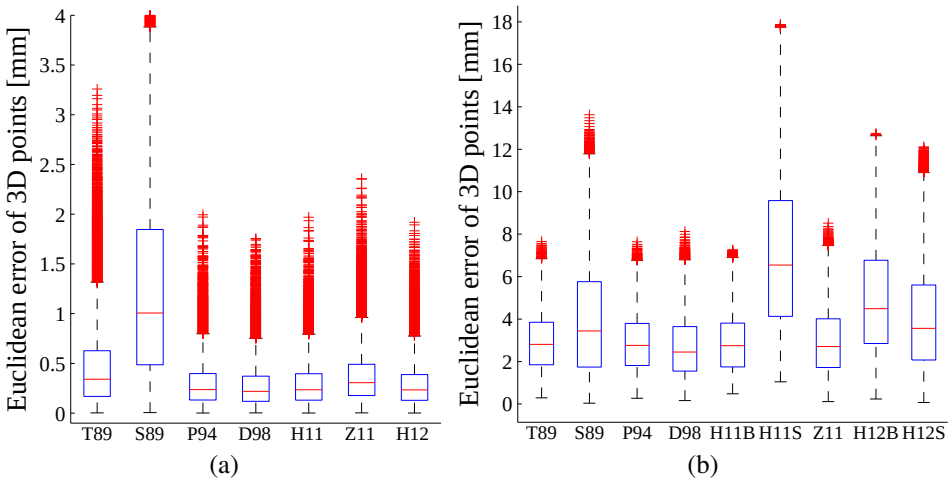


Figure 8.8: *Real data experiment error*. (a) Motoman MA1400 (b) Mitsubishi MELFA-RV-6S.

ure 8.7d. We used 7 motions $\mathtt{B}_i$ and with 447 randomly selected correspondences to construct the optimization task. The algorithm converged to a solution with residual error $\|\mathbf{e}\|_\infty = 0.006$ rad in 8 seconds (8 threads). The result is labeled as "H12S" in Figure 8.8b. We used the same optimization task to recover the calibration using the method [75], labeled as "H11S".

Next, we detected the grid pattern in every image. Using the same 7 relative movements, a calibration task consisting of 1,155 correspondences was constructed. The algorithm converged to a solution with resudual error $\|\mathbf{e}\|_\infty = 0.005$ rad in 56 seconds (8 threads). The result is labeled as "H12B" in Figure 8.8b. Again, method [75] labeled "H11B" was used with the same task. Finally, since the grid's dimensions were known, the camera poses were recovered in scale using EPnP algorithm and calibration was performed using methods "Tsai89", "Shiu89", "Park94", "Dan98", and "Zhao11".

We can see in Figure 8.8b that the resulting calibrations are relatively worse than the results in the MA1400 experiment. We explain it by the fact that the MELFA-RV-6S robot was slightly miscalibrated, showing the fact that the proposed method is more sensitive to the robot calibration. This is probably since the error cannot be compensated by the known camera positions as in the previous approaches. However, in case the robot is properly calibrated, it can deliver comparable or better results than its competitors.

8.6 Theory

The following four lemmas are from [69].

Lemma 8.8. *Let* $\mathtt{R}_1, \mathtt{R}_2 \in \mathrm{SO}(3)$. *Then for* $\forall\, \mathbf{v} \in \mathbb{R}^3$

$$\angle(\mathtt{R}_1\mathbf{v}, \mathtt{R}_2\mathbf{v}) \leq d_\angle(\mathtt{R}_1, \mathtt{R}_2).$$

Lemma 8.9. *Let* $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 \in \mathbb{B}_\pi$ *and* $\mathtt{R}_1, \mathtt{R}_2 \in \mathrm{SO}(3)$ *such that* $\mathtt{R}_1 = \exp[\boldsymbol{\alpha}_1]_\times$ *and* $\mathtt{R}_2 = \exp[\boldsymbol{\alpha}_2]_\times$. *The*

$$d_\angle(\mathtt{R}_1, \mathtt{R}_2) \leq \|\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2\|.$$

Lemma 8.10. *Let* $\bar{\mathtt{R}}_\mathtt{X}$ *be the rotation represented by the center of a cube* $D_\sigma \subset \mathbb{B}_\pi$ *and* $\mathtt{R}_\mathtt{X} \in D_\sigma$. *Then for* $\forall\, \mathbf{v} \in \mathbb{R}^3$

$$\angle(\bar{\mathtt{R}}_\mathtt{X}\mathbf{v}, \mathtt{R}_\mathtt{X}\mathbf{v}) \leq \sqrt{3}\sigma.$$

Lemma 8.11. *Let* $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$ *be unit vectors determining a spherical triangle on a unit sphere and the edges be arcs of lengths* u, v, *and* w *respectively. Let* α, β *be the angles at* $\mathbf{v}$ *and* $\mathbf{w}$ *respectively. It follows that if* β *is a right angle then* $\sin\alpha = \sin v / \sin w$.

The following two lemmas are from [146].

Lemma 8.12. *Let* $\mathtt{R}_\mathtt{X} \in \mathrm{SO}(3)$ *and* $\boldsymbol{\beta} \in \mathbb{B}_\pi$. *Then*

$$\log\left(\mathtt{R}_\mathtt{X} \exp[\boldsymbol{\beta}]_\times \mathtt{R}_\mathtt{X}^\top\right) = [\mathtt{R}_\mathtt{X}\boldsymbol{\beta}]_\times.$$

Lemma 8.13. *Let* $\bar{\mathtt{R}}_\mathtt{X}$ *be the rotation represented by the center of a cube* $D_\sigma \subset \mathbb{B}_\pi$, $\boldsymbol{\beta} \in \mathbb{B}_\pi$. *Then for* $\forall\, \mathtt{R}_\mathtt{X} \in D_\sigma$

$$\|\bar{\mathtt{R}}_\mathtt{X}\boldsymbol{\beta} - \mathtt{R}_\mathtt{X}\boldsymbol{\beta}\| \leq 2\|\boldsymbol{\beta}\| \sin(\sqrt{3}\sigma/2).$$

Let us prove two more lemmas here.

Lemma 8.14. *Let* $\bar{\mathtt{R}}_\mathtt{X}$ *be the rotation represented by the center of a cube* $D_\sigma \subset \mathbb{B}_\pi$, $\boldsymbol{\beta} \in \mathbb{B}_\pi$. *Let* $\mathtt{R}_\mathtt{X} \in D_\sigma$ *and* $\mathtt{R}_\mathtt{A} = \mathtt{R}_\mathtt{X} \exp[\boldsymbol{\beta}]_\times \mathtt{R}_\mathtt{X}^\top$, $\bar{\mathtt{R}}_\mathtt{A} = \bar{\mathtt{R}}_\mathtt{X} \exp[\boldsymbol{\beta}]_\times \bar{\mathtt{R}}_\mathtt{X}^\top$. *Then for* $\forall\, \mathbf{u} \in \mathbb{R}^3$

$$\angle(\bar{\mathtt{R}}_\mathtt{A}\mathbf{u}, \mathtt{R}_\mathtt{A}\mathbf{u}) \leq 2\|\boldsymbol{\beta}\| \sin\left(\sqrt{3}\sigma/2\right).$$

Proof. Note where Lemmas 8.8, 8.12, 8.9 and 8.6 were used, respectively.

$$\begin{aligned}
\angle(\bar{\mathtt{R}}_\mathtt{A}\mathbf{u}, \mathtt{R}_\mathtt{A}\mathbf{u}) &\leq d_\angle(\bar{\mathtt{R}}_\mathtt{A}, \mathtt{R}_\mathtt{A}) \\
&= d_\angle(\exp[\bar{\mathtt{R}}_\mathtt{X}\boldsymbol{\beta}]_\times, \exp[\mathtt{R}_\mathtt{X}\boldsymbol{\beta}]_\times) \\
&\leq \|\bar{\mathtt{R}}_\mathtt{X}\boldsymbol{\beta} - \mathtt{R}_\mathtt{X}\boldsymbol{\beta}\| \\
&\leq 2\|\boldsymbol{\beta}\| \sin(\sqrt{3}\sigma/2).
\end{aligned}$$

□

Lemma 8.15. *Let* $\bar{\mathtt{R}}_\mathtt{X}$ *be the rotation represented by the center of a cube* $D_\sigma \subset \mathbb{B}_\pi$, $\boldsymbol{\beta} \in \mathbb{B}_\pi$ *and* $\mathbf{u}, \mathbf{v} \in \mathbb{R}^3$. *Let* $\mathtt{R}_\mathtt{X} \in D_\sigma$ *and* $\mathtt{R}_\mathtt{A} = \mathtt{R}_\mathtt{X} \exp[\boldsymbol{\beta}]_\times \mathtt{R}_\mathtt{X}^\top$, $\bar{\mathtt{R}}_\mathtt{A} = \bar{\mathtt{R}}_\mathtt{X} \exp[\boldsymbol{\beta}]_\times \bar{\mathtt{R}}_\mathtt{X}^\top$. *Then if* $\angle(\pm\mathbf{v}, \bar{\mathtt{R}}_\mathtt{A}\mathbf{u}) > 2\|\boldsymbol{\beta}\| \sin(\sqrt{3}\sigma/2)$, *the following inequality holds*

$$\angle([\mathbf{v}]_\times \bar{\mathtt{R}}_\mathtt{A}\mathbf{u}, [\mathbf{v}]_\times \mathtt{R}_\mathtt{A}\mathbf{u}) \leq \arcsin\left(\frac{\sin(2\|\boldsymbol{\beta}\| \sin(\sqrt{3}\sigma/2))}{\sqrt{1 - (\mathbf{v}^\top \bar{\mathtt{R}}_\mathtt{A}\mathbf{u})^2}}\right).$$

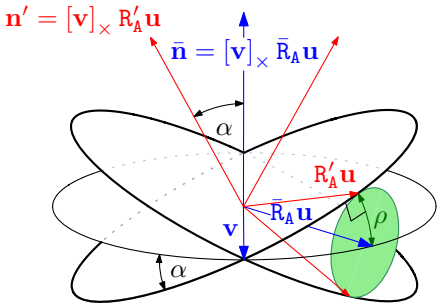


Figure 8.9: Illustration of the proof of Lemma 8.15.

Proof. We know from Lemma 8.14 that for every $\mathtt{R}_\mathtt{X} \in D_\sigma$ and $\mathbf{u} \in \mathbb{R}^3$ the angle $\angle(\bar{\mathtt{R}}_\mathtt{A}\mathbf{u}, \mathtt{R}_\mathtt{A}\mathbf{u})$ is limited by $\rho = 2\,\|\boldsymbol{\beta}\| \sin(\sqrt{3}\sigma/2)$, see Figure 8.9. Let $\mathbf{v} \in \mathbb{R}^3$ such that $\angle(\pm\mathbf{v}, \bar{\mathtt{R}}_\mathtt{A}\mathbf{u}) > \rho$, *i.e.*, vectors $\pm\mathbf{v}$ do not lie in the cone C determined by vector $\bar{\mathtt{R}}_\mathtt{A}\mathbf{u}$ and radius ρ. Now let's consider the geometrical relation between vectors $\bar{\mathbf{n}} = [\mathbf{v}]_\times \bar{\mathtt{R}}_\mathtt{A}\mathbf{u}$ and $\mathbf{n} = [\mathbf{v}]_\times \mathtt{R}_\mathtt{A}\mathbf{u}$. It is an elementary geometrical fact, that if $\mathbf{v}$, $\mathtt{R}_\mathtt{A}\mathbf{u}$ and $\bar{\mathtt{R}}_\mathtt{A}\mathbf{u}$ are coplanar vectors, then $\angle(\bar{\mathbf{n}}, \mathbf{n}) = 0$. Let $\mathtt{R}'_\mathtt{X} \in D_\sigma$ be a rotation such that the plane determined by vector $\mathbf{n}' = [\mathbf{v}]_\times \mathtt{R}'_\mathtt{A}\mathbf{u}$ is tangential to the cone C. Trivially, $\forall \mathtt{R}_\mathtt{X} \in D_\sigma \colon \angle(\bar{\mathbf{n}}, \mathbf{n}) \leq \angle(\bar{\mathbf{n}}, \mathbf{n}')$. Now the angle $\alpha = \angle(\bar{\mathbf{n}}, \mathbf{n}')$ can be determined. By using Lemma 8.11 on vectors $\mathbf{v}$, $\mathtt{R}'_\mathtt{A}\mathbf{u}$ and $\bar{\mathtt{R}}_\mathtt{A}\mathbf{u}$ we get

$$\sin\alpha = \frac{\sin\angle([\mathbf{v}]_\times \bar{\mathtt{R}}_\mathtt{A}\mathbf{u}, [\mathbf{v}]_\times \mathtt{R}_\mathtt{A}\mathbf{u})}{\sin\angle(\mathbf{v}, \bar{\mathtt{R}}_\mathtt{A}\mathbf{u})} = \frac{\sin\rho}{\sin\arccos \mathbf{v}^\top \bar{\mathtt{R}}_\mathtt{A}\mathbf{u}}.$$

From this follows that for $\forall \mathtt{R}_\mathtt{X} \in D_\sigma$

$$\angle(\bar{\mathbf{n}}, \mathbf{n}) \leq \alpha = \arcsin \frac{\sin\rho}{\sqrt{1 - (\mathbf{v}^\top \bar{\mathtt{R}}_\mathtt{A}\mathbf{u})^2}}.$$

□

Finally, we can proceed with the proof of Lemma 8.5.

Proof. First we prove the first part of the lemma. Let's suppose that $\hat{\mathtt{R}}_\mathtt{X} \in D_\sigma$ and $\mathbf{t}'_\mathtt{X} \in \mathbb{R}^3$ constitute a feasible solution to Problem 8.3. We show that $\hat{\mathtt{R}}_\mathtt{X}$, $\mathbf{t}'_\mathtt{X}$ constitute a solution to Problem 8.4 as well. Let us consider a correspondence $\mathbf{u}_{ij} \leftrightarrow \mathbf{v}_{ij}$. If $\angle(\pm\mathbf{v}_{ij}, \bar{\mathtt{R}}_\mathtt{A}\mathbf{u}_{ij}) \leq 2\,\|\boldsymbol{\beta}_i\| \sin(\sqrt{3}\sigma/2)$—one of the preconditions of Lemma 8.15 is not fulfilled—we cannot easily decide about the constraint imposed by the correspondence. If, on the other hand, this

condition is met, we can write the following inequality

$$\begin{aligned}
&\angle([\mathbf{v}_{ij}]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, \bar{\mathbf{t}}_{\mathtt{A}_i}) && (8.4)\\
&\leq \angle([\mathbf{v}_{ij}]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, [\mathbf{v}_{ij}]_\times \hat{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}) + \angle([\mathbf{v}_{ij}]_\times \hat{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, \hat{\mathbf{t}}_{\mathtt{A}_i}) + \angle(\hat{\mathbf{t}}_{\mathtt{A}_i}, \bar{\mathbf{t}}_{\mathtt{A}_i}) && (8.5)\\
&\leq \angle([\mathbf{v}_{ij}]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, [\mathbf{v}_{ij}]_\times \hat{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}) + \frac{\pi}{2} + \epsilon_{\min} + && (8.6)\\
&\qquad\qquad \angle\left(\hat{\mathtt{R}}_{\mathtt{X}}((\mathtt{R}_{\mathtt{B}_i} - \mathtt{I})\mathbf{t}'_{\mathtt{X}} + \mathbf{t}_{\mathtt{B}_i}), \bar{\mathtt{R}}_{\mathtt{X}}((\mathtt{R}_{\mathtt{B}_i} - \mathtt{I})\mathbf{t}'_{\mathtt{X}} + \mathbf{t}_{\mathtt{B}_i})\right)\\
&\leq \angle([\mathbf{v}_{ij}]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, [\mathbf{v}_{ij}]_\times \hat{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}) + \frac{\pi}{2} + \epsilon_{\min} + \sqrt{3}\sigma && (8.7)\\
&\leq \arcsin\left(\frac{\sin\left(2\left\|\boldsymbol{\beta}_i\right\|\sin\left(\sqrt{3}\sigma/2\right)\right)}{\sqrt{1-\left(\mathbf{v}_{ij}^\top \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}\right)^2}}\right) + \frac{\pi}{2} + \epsilon_{\min} + \sqrt{3}\sigma && (8.8)
\end{aligned}$$

To elaborate, we get from line 8.4 to line 8.5 by twice applying the triangle inequality, to line 8.6 using the presumption that $\hat{\mathtt{R}}_{\mathtt{X}}$, $\mathbf{t}'_{\mathtt{X}}$ constitute a solution to Problem 8.3, to line 8.7 using Lemma 8.10, and finally to line 8.8 using Lemma 8.15. Note that because of the substitution $\mathbf{t}'_{\mathtt{X}} = -\mathtt{R}_{\mathtt{X}}^\top \mathbf{t}_{\mathtt{X}}$, translation vectors $\hat{\mathbf{t}}_{\mathtt{A}_i}, \bar{\mathbf{t}}_{\mathtt{A}_i}$ can be written as products of rotation matrices $\hat{\mathtt{R}}_{\mathtt{X}}, \bar{\mathtt{R}}_{\mathtt{X}}$ and vector $((\mathtt{R}_{\mathtt{B}_i} - \mathtt{I})\mathbf{t}'_{\mathtt{X}} + \mathbf{t}_{\mathtt{B}_i})$. This makes it possible to apply Lemma 8.10 to term $\angle(\hat{\mathbf{t}}_{\mathtt{A}_i}, \bar{\mathbf{t}}_{\mathtt{A}_i})$ and justify the inequality of lines 8.6 and 8.7.

Now, line 8.8 gives us value of the bound γ_{ij} as

$$\gamma_{ij} = \arcsin\left(\frac{\sin\left(2\left\|\boldsymbol{\beta}_i\right\|\sin\left(\sqrt{3}\sigma/2\right)\right)}{\sqrt{1-\left(\mathbf{v}_{ij}^\top \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}\right)^2}}\right) + \sqrt{3}\sigma.$$

The proof of the second inequality,

$$\angle(-[\mathbf{v}_{ij}]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, \bar{\mathbf{t}}_{\mathtt{A}_i}) \leq \frac{\pi}{2} + \epsilon_{\min} + \gamma_{ij},$$

is almost identical. We just use the fact that

$$\angle(-[\mathbf{v}_{ij}]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, -[\mathbf{v}_{ij}]_\times \hat{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}) = \angle([\mathbf{v}_{ij}]_\times \bar{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, [\mathbf{v}_{ij}]_\times \hat{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij})$$

and the presumption

$$\angle(-[\mathbf{v}_{ij}]_\times \hat{\mathtt{R}}_{\mathtt{A}_i}\mathbf{u}_{ij}, \hat{\mathbf{t}}_{\mathtt{A}_i}) \leq \frac{\pi}{2} + \epsilon_{\min}.$$

The proof of the second part of the lemma is analogous to the proof of the second part of Lemma 3 in [69]. □

9 Hand-Eye Calibration without Hand Orientation

Arithmetic! Algebra! Geometry! Grandiose trinity! Luminous triangle! Whoever has not known you is without sense!

– Comte de Lautréamont

In this chapter we will be concerned with a variation of hand-eye calibration problem that has scarcely been addressed in the literature so far—hand-eye calibration with unknown hand rotation. This problem arises when the robot is not calibrated or the information from the robot is not available. In these situations, one has to measure the robot's pose by an external measurement device. In many cases such a measurement device is not able to measure the whole pose, but only the translational part of it, since translation is much easier to measure than rotation. Without the hand rotation measurements none of the previously discussed methods can be used. A method presented in [194] addresses this problem by nonlinear optimization and estimates simultaneously both rotational and translational parts. However, it requires a good initial estimate of $\mathtt{X}$.

In case of two relative motions, we formulate this problem as a system of seven equations in seven unknowns and solve it using the Gröbner basis method for solving systems of polynomial equations. This provides an exact algebraic solution and has none of the problems of the former numerical minimization methods, *i.e.*, problems with convergence or the necessity of having a good initial estimate. In case of three of more motions, we use a residual function to select an initial solution among the candidates provided by the Gröbner basis method to initialize the method of [194]. By evaluating our solution on both synthetic and real scene data, we demonstrate that it is efficient, fast, and numerically stable. Further, we show that in case of more than two motions it provides a good estimate for nonlinear optimization.

9.1 Problem Formulation

First, let us consider the classical hand-eye calibration problem. The goal is to estimate the relative pose, *i.e.*, the rotation and the translation of the camera w.r.t. the gripper given n absolute camera poses $\mathtt{A}'_i$ and robot poses $\mathtt{B}'_i$, see Section 5.1. We describe the hand-eye transformation by a homogeneous transformation matrix

$$\mathtt{X} = \begin{pmatrix} \mathtt{R}_\mathtt{X} & \mathbf{t}_\mathtt{X} \\ \mathbf{0}^\top & 1 \end{pmatrix}. \tag{9.1}$$

The hand-eye tranformation is constrained by the relative camera and robot poses $\mathtt{A}_i = \mathtt{A'}_{i+1}\mathtt{A}'^{-1}_i$ and $\mathtt{B}_i = \mathtt{B'}^{-1}{}_{i+1}\mathtt{B}'_i$ as

$$\mathtt{A}_i\mathtt{X} = \mathtt{X}\mathtt{B}_i, \tag{9.2}$$

which can be decomposed into a matrix and a vector equations

$$\mathtt{R}_{\mathtt{A}_i}\mathtt{R}_{\mathtt{X}} = \mathtt{R}_{\mathtt{X}}\mathtt{R}_{\mathtt{B}_i}, \tag{9.3}$$

$$\mathtt{R}_{\mathtt{A}_i}\mathbf{t}_{\mathtt{X}} + \mathbf{t}_{\mathtt{A}} = \mathtt{R}_{\mathtt{X}}\mathbf{t}_{\mathtt{B}_i} + \mathbf{t}_{\mathtt{X}}. \tag{9.4}$$

At least two relative poses with non-parallel rotation axes are required to solve this system of equations. With two or more relative poses known, we obtain an overconstrained system of polynomial equations.

In situations where one does not have the information from the robot's positioning software or the robot is not precisely calibrated transformations $\mathtt{B}'_i$ are not readily known. To recover them, one has to use some external measurement equipment. In this chapter, we are interested in situations where such a measurement device does not allow to recover the whole pose of the robotic gripper, but only its translational part.

Typically, the external measurement devices are able to recover absolute gripper's positions $\mathbf{t}'_{\mathtt{B}}$ w.r.t. robot's base. However, in Equation 9.4 relative translations $\mathbf{t}_{\mathtt{B}_i}$ appear. In order to compute the relative translations $\mathbf{t}_{\mathtt{B}_i}$ there has to be at least one position where the full pose of the robot can be recovered, *i.e.*, where the rotation $\mathtt{R}'_{\mathtt{B}_i}$ is known as well. Even for an uncalibrated robot, the robot's home position can be used as such *a priori* known pose. By constructing the relative movements in such a way as to always end in a position with a known rotation $\mathtt{R}'_{\mathtt{B}_i}$, relative translations $\mathbf{t}_{\mathtt{B}_i}$ can be recovered. Since the positions with *a priori* known poses are usually hard to come by, it is advantageous for a method to be able to calibrate from a minimal number of movements possible.

9.2 Minimal Hand-Eye Calibration

First, let us suppose that we can measure two gripper's relative translations $\mathbf{t}_{\mathtt{B}_i}$ and $\mathbf{t}_{\mathtt{B}_j}$ and two respective relative camera poses $\mathtt{A}_i$ and $\mathtt{A}_j$. Now, notice that the vector Equation 9.4 does not contain the unknown gripper's rotations $\mathtt{R}_{\mathtt{B}_i}$. By parametrizing the rotation $\mathtt{R}_{\mathtt{X}}$ by the unit quaternion $q = a + bi + cj + dk$ as

$$\mathtt{R}_{\mathtt{X}} \mapsto \mathtt{R}^q_{\mathtt{X}} = \begin{pmatrix} a^2+b^2-c^2-d^2 & 2bc-2ad & 2ac+2bd \\ 2ad+2bc & a^2-b^2+c^2-d^2 & 2cd-2ab \\ 2bd-2ac & 2ab+2cd & a^2-b^2-c^2+d^2 \end{pmatrix} \tag{9.5}$$

and substituting it into the vector Equation 9.4 we get three polynomial equations in seven unknowns, *i.e.*, three translation parameters for $\mathbf{t}_{\mathtt{X}}$ and four rotation parameters a, b, c, and d. Now we can apply this substitution to the two motions i and j and by adding the equation defining the unit quaternion q we obtain the following system of equations:

Problem 9.1 (Minimal hand-eye calibration).

$$\begin{aligned} \text{Given} \quad & \mathtt{R}_{\mathtt{A}_i}, \mathtt{R}_{\mathtt{A}_j}, \mathbf{t}_{\mathtt{A}_i}, \mathbf{t}_{\mathtt{A}_j}, \mathbf{t}_{\mathtt{B}_i}, \mathbf{t}_{\mathtt{B}_j} \\ \text{find} \quad & \mathtt{R}_{\mathtt{X}} \in SO(3), \mathbf{t}_{\mathtt{X}} \in \mathbb{R}^3 \\ \text{subject to} \quad & \mathtt{R}_{\mathtt{A}_i}\mathbf{t}_{\mathtt{X}} + \mathbf{t}_{\mathtt{A}_i} = \mathtt{R}^q_{\mathtt{X}}\mathbf{t}_{\mathtt{B}_i} + \mathbf{t}_{\mathtt{X}}, \\ & \mathtt{R}_{\mathtt{A}_j}\mathbf{t}_{\mathtt{X}} + \mathbf{t}_{\mathtt{A}_j} = \mathtt{R}^q_{\mathtt{X}}\mathbf{t}_{\mathtt{B}_j} + \mathbf{t}_{\mathtt{X}}, \\ & a^2+b^2+c^2+d^2 = 1. \end{aligned}$$

Problem 9.1 is a well-constrained system of seven equations in seven unknowns. To solve it for the unknown hand-eye calibration $\mathtt{X}$, the Gröbner basis method can be readily used. This leads to a fast and non-iterative solution with no need for an initial solution estimate. Note that the minimal number of two relative movements without rotations $\mathtt{R}_{\mathtt{B}_i}$ and $\mathtt{R}_{\mathtt{B}_j}$ is needed to construct the system.

In the case rotations $\mathtt{R}_{\mathtt{B}_i}$ and $\mathtt{R}_{\mathtt{B}_j}$ need to be recovered as well, by substituting the solutions for the rotation $\mathtt{R}_{\mathtt{X}}$ into the Equation 9.3 we get the rotations as

$$\mathtt{R}_{\mathtt{B}_i} = \mathtt{R}_{\mathtt{X}}^{-1}\mathtt{R}_{\mathtt{A}_i}\mathtt{R}_{\mathtt{X}}, \tag{9.6}$$

$$\mathtt{R}_{\mathtt{B}_j} = \mathtt{R}_{\mathtt{X}}^{-1}\mathtt{R}_{\mathtt{A}_j}\mathtt{R}_{\mathtt{X}}. \tag{9.7}$$

9.2.1 Gröbner Basis Method

The Gröbner basis method for solving systems of polynomial equations has recently became popular in computer vision and it has been used to create very fast, efficient and numerically stable solvers to many difficult problems. The method is based on polynomial ideal theory and is concerned with special bases of these ideals called Gröbner bases [41]. Gröbner bases have the same solutions as the initial system of polynomial equations defining the ideal but are often easier to solve. Gröbner bases are usually used to construct special multiplication (action) matrices [159], which can be viewed as a generalization of the companion matrix used in solving one polynomial equation in one unknown. The solutions to the system of polynomial equations is then obtained from the eigenvalues and eigenvectors of such action matrices. See [41, 42] for more on Gröbner basis methods and [161, 94, 24] for their applications in computer vision.

Since general algorithms [41] for computing Gröbner basis are not very efficient for solving problems which appear for example in computer vision, an automatic generator of specific polynomial equations solvers based on the Gröbner basis method has been proposed in [95]. These specific solvers often provide very efficient solutions to a class of systems of polynomial equations consisting of the same monomials and differing only in the coefficients.

Computer vision problems, like the hand-eye calibration problem presented in this chapter, share the convenient property that the monomials appearing in the set of initial polynomials are always the same irrespective of the concrete coefficients arising from non-degenerate measurements. Therefore it is possible to use efficient specific solvers instead of less efficient general algorithms [41] for constructing the Gröbner bases.

The process of creating the specific solvers consists of two phases. In the first "offline" phase, the so-called "elimination templates" are found. These templates decide the elimination sequence in order to obtain all polynomials from the Gröbner basis or at least all polynomials necessary for the construction of the action matrix. This phase is performed only once for a given problem. In the second "online" phase, the elimination templates are used with coefficients arising from the specific measurements to construct the action matrix. Then, eigenvalues and eigenvectors of the action matrix provide solutions to the original polynomial equations. The automatic generator presented in [95] performs the offline phase automatically and for an input system of polynomial equations outputs an efficient online solver.

9.2.2 Gröbner Basis Solver

To create an efficient solver for Problem 9.1 we used the automatic generator proposed in [95]. The Gröbner basis solver of the proposed hand-eye calibration problem starts with seven equations in seven unknowns, *i.e.*, three translation parameters for $\mathbf{t}_\mathtt{X}$ and four rotation parameters a, b, c, and d.

From the generator we obtained an elimination template which encodes how to multiply the seven input polynomials by the monomials and then how to eliminate the polynomias using the Gauss-Jordan (G-J) elimination process to obtain all polynomials necessary for the construction of the action matrix. In our case the automatic generator created the action matrix $\mathtt{M}_a$ for multiplication by a.

To get the elimination template the generator first generated all monomial multiples of the initial seven polynomial equations up to the total degree of four. This resulted in 252 polynomials in 330 monomials. Then the generator removed all unnecessary polynomials and monomials, *i.e.*, polynomials and monomials that do not influence the resulting action matrix. This resulted in matrix a 182×203 $\mathtt{Q}$ representing the polynomials for the construction of the action matrix $\mathtt{M}_a$, *i.e.*, the elimination template.

The online solver then only performs one G-J elimination of matrix $\mathtt{Q}$ from the elimination template identified in the offline stage. This matrix contains coefficients which arise from specific measurements, *i.e.*, rotations $\mathtt{R}_{\mathtt{A}_i}$ and $\mathtt{R}_{\mathtt{A}_j}$ and translations $\mathbf{t}_{\mathtt{A}_i}$,$\mathbf{t}_{\mathtt{A}_j}$,$\mathbf{t}_{\mathtt{B}_i}$, and $\mathbf{t}_{\mathtt{B}_j}$. After G-J elimination of matrix $\mathtt{Q}$, action matrix $\mathtt{M}_a$ can be created from its rows. The solutions to all seven unknowns can be found from the eigenvectors of the action matrix $\mathtt{M}_a$. The online stage takes about 1 ms to finish in case of Problem 9.1.

This gives us a set $\mathcal{X}_{ij}$ of up to 16 real solutions of $\mathtt{X}$. However each of these solutions appears twice, *i.e.*, there are double roots. Therefore we have only up to 8 different real solutions. Usually only one to four of them are *geometrically feasible*, *i.e.*, are real and of a reasonable length of the translation. The correct one can be chosen from the feasible solutions manually using some prior knowledge about the transformation $\mathtt{X}$ or automatically using an additional set of solutions for different relative movements. The next section describes an automatic procedure for selecting the correct transformation.

9.3 Automatic Solution Selection

In order to automatically select the geometrically correct solution among the algebraically correct ones in $\mathcal{X}_{ij}$, at least one more set of solutions to Problem 9.1 for a different combination of relative poses is needed. Let $\mathcal{X}_{k\ell}$ be such a set for two additional relative poses k and ℓ. Supposing that the relative poses i, j and k, ℓ form a geometrically non-degenerate configuration, we will find the geometrically correct solution as $\mathcal{X}_{ij} \cap \mathcal{X}_{k\ell}$. In the presence of noise however, the intersection $\mathcal{X}_{ij} \cap \mathcal{X}_{k\ell}$ will most likely be an empty set. In this case we have to select a solution from the union $\mathcal{X}_{ij} \cup \mathcal{X}_{k\ell}$ that best fits the equations of Problem 9.1 for different motions. We will measure the fitness of a solution $\mathtt{X}$ by the residual error of Equation 9.4

$$\mathbf{e}_i(\mathtt{X}) = \mathtt{R}_{\mathtt{A}_i}\mathbf{t}_\mathtt{X} + \mathbf{t}_{\mathtt{A}_i} - \mathtt{R}_\mathtt{X}\mathbf{t}_{\mathtt{B}_i} - \mathbf{t}_\mathtt{X}. \tag{9.8}$$

Now let us formalize the idea of selecting the best solution and to extend it to the case of

more that two solution sets. Let n be the number of available relative movements and let I be a set of pairs of indexes of the relative poses

$$I \subset \{\{i,j\} \colon i,j \leq n\}, \quad |I| \geq 2. \tag{9.9}$$

Let $\mathcal{X}$ be a set of solutions to Problem 9.1 for the pairs from the index set I,

$$\mathcal{X} = \bigcup_{\{i,j\} \in I} \mathcal{X}_{ij}. \tag{9.10}$$

We select the geometrically correct solution among the solutions in $\mathcal{X}$ by solving the following problem:

Problem 9.2 (Minimal hand-eye calibration for n relative poses).

$$\begin{aligned} \text{Given} \quad & \mathtt{R}_{\mathtt{A}_i}, \mathbf{t}_{\mathtt{A}_i}, \mathbf{t}_{\mathtt{B}_i}, I, i = 1, \ldots, n \\ \text{and} \quad & \text{a set of solutions } \mathcal{X} = \textstyle\bigcup_{\{i,j\} \in I} \mathcal{X}_{ij} \\ \text{find} \quad & \mathtt{X}^\star = \arg\min_{\mathtt{X} \in \mathcal{X}} \sum_{i=1}^{n} \mathbf{e}_i(\mathtt{X})^\top \mathbf{e}_i(\mathtt{X}) \end{aligned}$$

As we can see from the above formulation, solving Problem 9.2 amounts to selecting a minimum from a set of $|\mathcal{X}|$ real numbers.

In the presence of noise and in case $n > 2$, we can further refine the solution by an optimization method. For our experiments, we chose the method of Zhuang and Shiu [194] which requires a good initial estimate $\mathtt{X}^0$. By setting $\mathtt{X}^0 \equiv \mathtt{X}^\star$, we can refine the solution by solving the following minimization problem:

Problem 9.3 (Zhuang [194]).

$$\begin{aligned} \text{Given} \quad & \mathtt{R}_{\mathtt{A}_i}, \mathbf{t}_{\mathtt{A}_i}, \mathbf{t}_{\mathtt{B}_i}, i = 1, \ldots, n \\ \text{and} \quad & \text{an initial solution estimate } \mathtt{X}^0 \\ \text{find} \quad & \mathtt{X}^\star_{\text{opt}} = \arg\min \sum_{i=1}^{n} \mathbf{e}_i(\mathtt{X})^\top \mathbf{e}_i(\mathtt{X}) \\ \text{subject to} \quad & \mathtt{R}_{\mathtt{X}} \in SO(3) \end{aligned}$$

9.4 Experiments

To experimentally validate the proposed solutions, we use both synthetically generated and real word calibration scenarios. First, we use synthetically generated ground truth scenes to study the numerical stability of the proposed solution to Problem 9.1. Next, we study the behavior of the solutions to Problem 9.2 and Problem 9.3 on synthetic scenes consisting of 4 non-degenerate poses. Finally, we show the viability of the minimal solution in a real life experiment with a Mitsubishi MELFA-RV-6S serial manipulator with four draw-wire encoders attached to its end effector to recover the translations $\mathbf{t}_{\mathtt{B}_i}$.

In all of the experiments we scaled the lengths of the input translation vectors $\mathbf{t}_{\mathtt{B}_i}$ and $\mathbf{t}_{\mathtt{A}_i}$ by the length of the largest one of them prior to running the Gröbner basis solver. We observe that this scaling improves the numerical stability of the solution.

The experiments were run on a 3GHz Intel Core i7 based desktop computer running 64-bit Linux. The Matlab implementation of the proposed method used in the experiments is available at `http://cmp.felk.cvut.cz/minimal/handeye.php`.

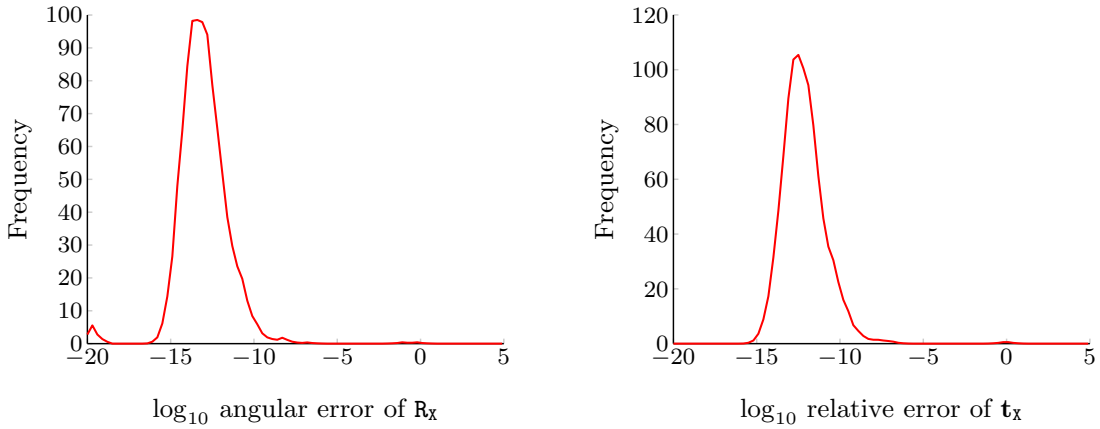


Figure 9.1: $\log_{10}$ angular error of the estimated rotation $\mathtt{R}_\mathtt{X}$ (Left) and $\log_{10}$ translation error of $\mathbf{t}_\mathtt{X}$ (Right) for noise free data.

9.4.1 Experiments with Synthetic Data

Numerical Stability Experiment First, we studied the behavior of the proposed Gröbner basis solver of Problem 9.1 to check its numerical stability. We generated 1000 random scenes with 100 points $\mathbf{P}^k$, $k = 1, \ldots, 100$, evenly distributed in the unit ball. Each scene consisted of 3 random absolute camera poses $\mathtt{A}'_i$. The cameras were positioned to (i) be facing the center of the scene, (ii) see the scene points from the field of view (FOV) ranging from 40° to 80°. For every scene ground truth transformation $\mathtt{X}_{\text{gt}}$ was generated so that the angle and the axis of $\mathtt{R}_{\mathtt{X}_{\text{gt}}}$ were random and uniformly distributed and that $\left\|\mathbf{t}_{\mathtt{X}_{\text{gt}}}\right\| \approx 0.1$. Absolute robot poses $\mathtt{B}'_i$ were determined by chaining $\mathtt{X}_{\text{gt}}^{-1}$ and the generated absolute camera positions. For every combination of ground truth $\mathtt{R}_{\mathtt{X}_{\text{gt}}}$, $\mathbf{t}_{\mathtt{X}_{\text{gt}}}$ and the recovered $\mathtt{R}_\mathtt{X}$, $\mathbf{t}_\mathtt{X}$ we measured the error of the rotation as the angle θ of the rotation $\mathtt{R}_\mathtt{X}^\top \mathtt{R}_{\mathtt{X}_{\text{gt}}}$, such that $0 \leq \theta \leq \pi$ and the error of translation as the relative error $\left\|\mathbf{t}_\mathtt{X} - \mathbf{t}_{\mathtt{X}_{\text{gt}}}\right\| / \left\|\mathbf{t}_{\mathtt{X}_{\text{gt}}}\right\|$. Figure 9.1 shows the histograms of the respective errors, certifying the numerical stability of the solver.

Calibration Experiment In this experiment we analyzed the performance with respect to image noise. We used the same scheme to generate random scenes as in Numerical Stability Experiment. This time, we generated four absolute robot poses in each scene and recovered the absolute camera positions by P3P algorithm [133].

We started by computing $\mathbf{P}_i^k$—the positions of the 100 random points $\mathbf{P}^k$ with respect to the coordinate systems of the cameras $\mathtt{A}'_i$, $i = 1, \ldots, 4$. Further, we normalized $\mathbf{P}_i^k$ to get only the directional vectors $\mathbf{p}_i^k$ that were progressively corrupted with angular Gaussian noise. Finally, we used P3P in RANSAC loop to obtain noise corrupted absolute camera poses $\mathtt{A}'_i$, $i = 1, \ldots 4$.

We experimented with 11 levels of angular Gaussian noise with the standard deviation σ ranging from 0 to 0.5 degrees, with the highest noise level translating to σ of ca. 20–40 pixels for a 8MP camera with 40°–80° field of view. We generated and recovered camera poses for 1000 random scenes for every noise level.

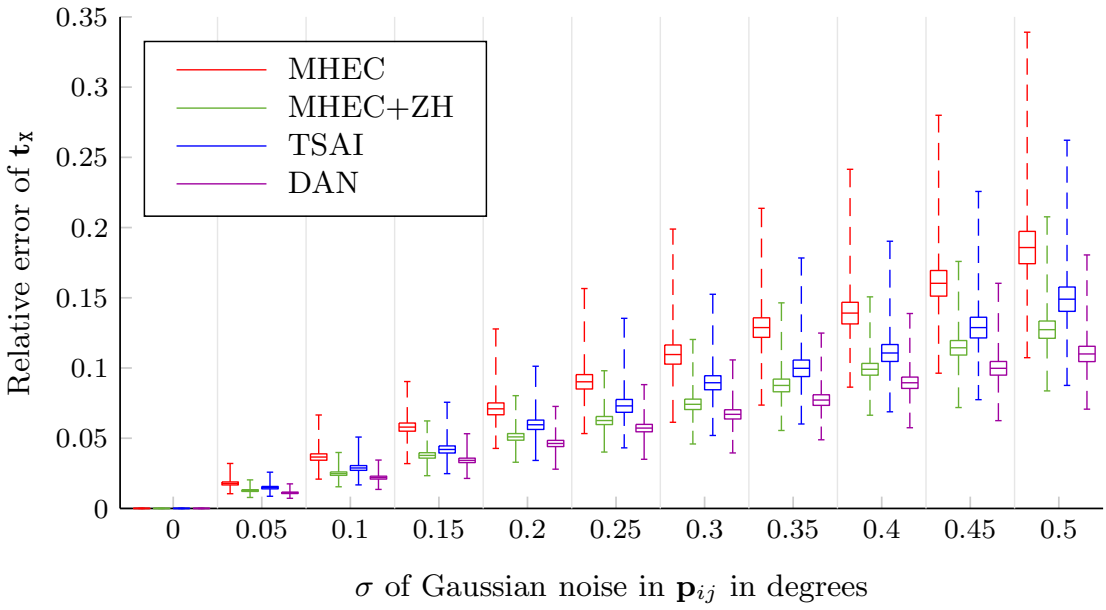


Figure 9.2: Relative error of recovered translation $\mathbf{t}_\mathtt{X}$ for different levels of Gaussian noise.

We recovered hand-eye calibrations $\mathtt{X}$ by four different methods. The first method MHEC identifies the results obtained by the Gröbner basis solver with the solution selected according to Problem 9.2. The second method MHEC+ZH stands for the results obtained by the method [194] (Problem 9.3) when initialized by the results of MHEC. For completeness sake, we include results obtained by the methods [183] labeled as TSAI and [44] labeled in the figures as DAN. These methods are not the direct competitors, since they require known robot rotations $\mathtt{R}_\mathtt{B}$. However, they can be used to gauge the accuracy of the results obtained by MHEC and MHEC+ZH.

Figures 9.2, 9.3, and 9.4 show the statistics of the obtained solutions using the Matlab `boxplot` function depicting values 25% to 75% quantile as a box with horizontal line at median. Figures 9.2 and 9.3 show the respective errors of $\mathbf{t}_\mathtt{X}$ and $\mathtt{R}_\mathtt{X}$ using the same measures as described in Numerical Stability Experiment. Figure 9.4 shows the mean distance between the points $\mathbf{P}_i^k$ transformed into the coordinate system of the gripper using the ground truth hand-eye transformation and the same points transformed into the coordinate system of the gripper using the estimated $\mathtt{X}$. Note that the points were generated into the unit ball, *i.e.*, considering the diameter of this ball to be one meter means that the errors in Figure 9.4 are in meters.

9.4.2 Real Scene Data Experiment

In order to acquire a real scene calibration data, four draw-wire encoders were connected to the gripper of a Mitsubishi MELFA-RV-6S serial manipulator. A Canon 350D digital SLR camera with a Sigma 8 mm lens (cca. 130° field of view) was also attached to the gripper to

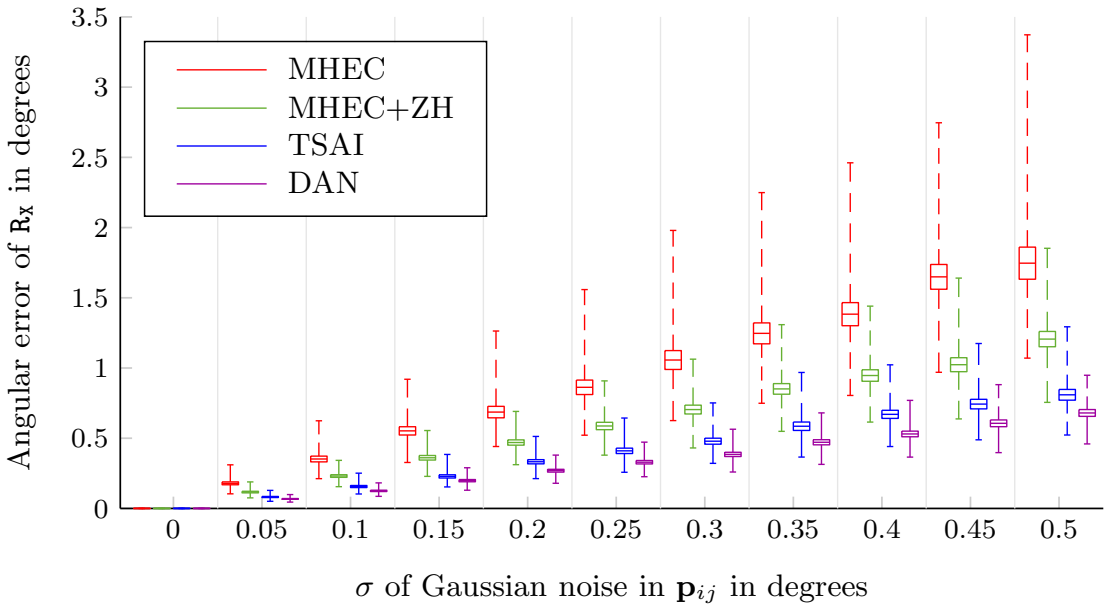


Figure 9.3: Angular error of recovered rotation $\mathtt{R}_\mathtt{X}$ for different levels of Gaussian noise.

form a hand-eye system.

The robot was instructed to move the gripper to (i) the home position with the known rotation w.r.t. the robot base, (ii) the four positions (backward, forward, left, right) distant approximately 400 mm at 10 degree pitch, (iii) the same four positions at 20 degree pitch, (iv) the position approximately 250 mm under the home position, and (v) the four positions at this height at 10 and 20 degree pitch again. While the robot was moving, the camera was remotely triggered to acquire $2{,}592 \times 1{,}728$ pixels large images of a circular view field with 1,040 pixels radius.

The internal calibration of the camera in the form of a two-parameter equiangular model [125] was obtained using an image of a checkerboard with manually labeled corners. Then, a state-of-the-art sequential structure-from-motion pipeline [177] was used to automatically generate MSER, SIFT, and SURF feature points, perform approximate nearest neighbor matching in the descriptor space, verify the matches by pairwise epipolar geometries estimated by the 5-point algorithm [130] in a RANSAC loop, and create tracks and triangulated 3D points from verified matches spanning several images. The reconstructed 3D model was scaled to millimeter units by knowing the real dimensions of the checkerboard and measuring the distance of the corresponding 3D points in the model.

We used the system of four draw-wire encoders to determine the absolute positions of the gripper w.r.t. the robot base. For the experiment we chose 2 motions ending in the robots home position. Since the rotation of the robot in the home position is known, it is possible to transform the positions provided by the draw-wire encoders into the home position coordinate system and obtain translations $\mathbf{t}_{\mathtt{B}_1}$ and $\mathbf{t}_{\mathtt{B}_2}$. We used $\mathbf{t}_{\mathtt{B}_1}$ and $\mathbf{t}_{\mathtt{B}_2}$ in combination with $\mathtt{A}_1$, $\mathtt{A}_2$ obtained from structure-from-motion to compute the hand-eye transformation $\mathtt{X}$ and

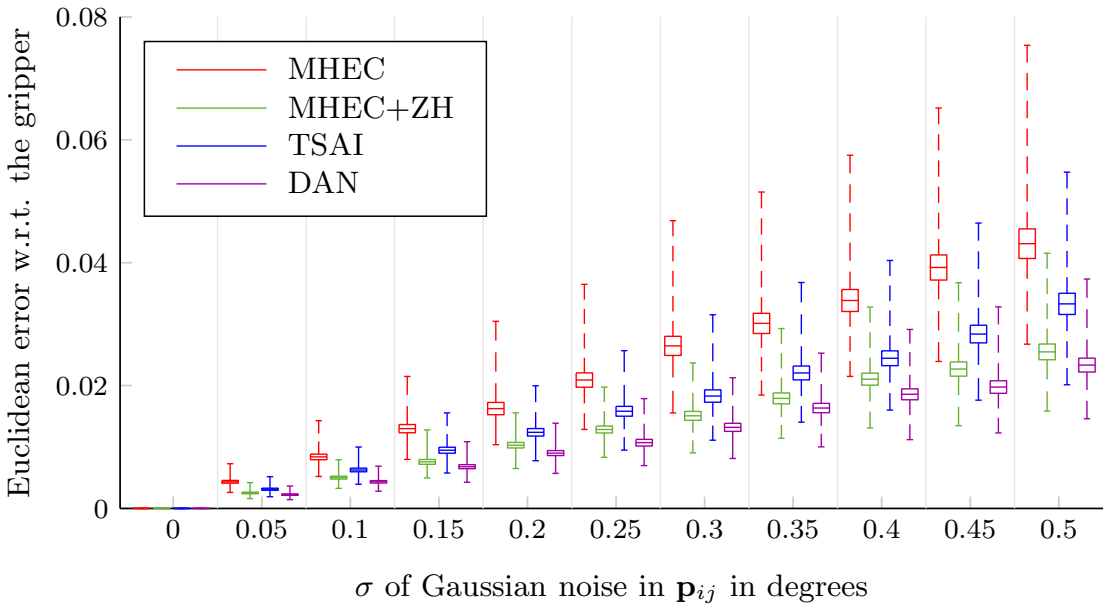


Figure 9.4: Euclidean error of recovered calibration $\mathtt{X}$ for different levels of Gaussian noise.

Table 9.1: Angular rotation errors of estimated gripper rotations in degrees.

	$\mathtt{R}_{\mathtt{B}_1}$	$\mathtt{R}_{\mathtt{B}_2}$	$\bar{\mathtt{R}}_{\mathtt{B}_1}$	$\bar{\mathtt{R}}_{\mathtt{B}_2}$
$\mathtt{R}_{\mathtt{B}_{gt1}}$	0.84	—	0.89	—
$\mathtt{R}_{\mathtt{B}_{gt2}}$	—	0.61	—	1.09

the relative gripper rotations $\mathtt{R}_{\mathtt{B}_1}$ and $\mathtt{R}_{\mathtt{B}_2}$.

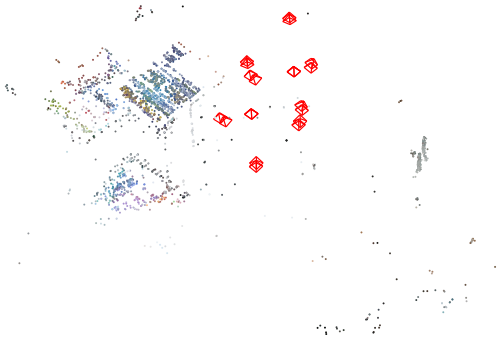
For comparison, we also used $\mathbf{t}_{\mathtt{B}_{gt2}}$ and $\mathbf{t}_{\mathtt{B}_{gt2}}$ from robots positioning software with the same camera motions $\mathtt{A}_1$ and $\mathtt{A}_2$ to compute hand-eye transformation $\bar{\mathtt{X}}$, $\bar{\mathtt{R}}_{\mathtt{B}_1}$, and $\bar{\mathtt{R}}_{\mathtt{B}_2}$.

Since the robot was calibrated, we can also compare the computed gripper rotations $\mathtt{R}_{\mathtt{B}_1}$, $\mathtt{R}_{\mathtt{B}_2}$, $\bar{\mathtt{R}}_{\mathtt{B}_1}$, and $\bar{\mathtt{R}}_{\mathtt{B}_2}$ with the rotations $\mathtt{R}_{\mathtt{B}_{gt1}}$ and $\mathtt{R}_{\mathtt{B}_{gt2}}$ from the robots positioning software, see Table 9.1.

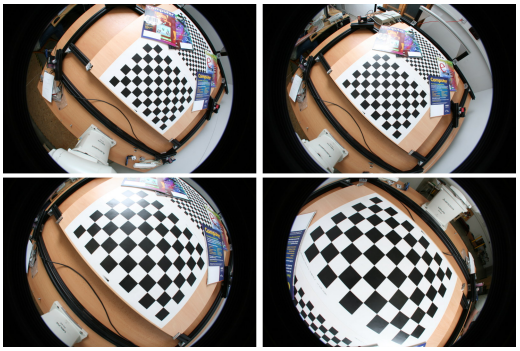
Finally, let us express the obtained translations from the gripper to the camera center using the translation from the draw-wire encoders $-\mathtt{R}_{\mathtt{X}}^{\top}\mathbf{t}_{\mathtt{X}} = (110.2, 26.2, 47.9)^{\top}$, and using the translation from the robot, $-\mathtt{R}_{\bar{\mathtt{X}}}^{\top}\mathbf{t}_{\bar{\mathtt{X}}} = (126.5, 28.7, 51.1)^{\top}$.

These result are consistent with each other as well as with the rough physical measurement of the mechanical reduction and show the validity of the obtained results.

(a)



(b)



(c)

(d)

Figure 9.5: Real data experiment. (a) A Mitsubishi MELFA-RV 6S serial manupulator used to acquire the data for the experiment. (b) The 3D model obtained from SfM. (c) Sample images of our scene taken by the camera mounted on the gripper of the robot. (d) Close up of the camera-gripper rig with draw-wire encoders.

10 Robot-World Calibration by LMI Relaxations

Work. Don't Think. Relax.

– Ray Bradbury

This chapter presents a novel solution to the robot-world calibration problem. It is applicable in situations where a known calibration target is observed by a camera attached to the end effector of a robotic manipulator. The presented method works by minimizing geometrically meaningful error function based on image projections. Our formulation leads to a non-convex multivariate polynomial optimization problem of a constant size. However, we show how such a problem can be relaxed using linear matrix inequality (LMI) relaxations and effectively solved using Semidefinite Programming. Although the technique of LMI relaxations guaranties only a lower bound on the global minimum of the original problem, it can provide a certificate of optimality in cases when the global minimum is reached. Indeed, we reached the global minimum for all calibration tasks in our experiments with both synthetic and real data. The experiments also show that the presented method is fast and noise resistant.

10.1 Introduction

In this chapter, we study situations where a robotic manipulator is self-calibrating by observing known calibration target by a camera attached to the end effector of a the manipulator. Specifically, we are interested in situation where the transformation relating the world coordinate system with the base coordinate system of the robot is to be recovered—the robot-world calibration problem [193, 46]—and can be considered as a subproblem of the complete robot self-calibration.

We formulate the robot-world calibration problem as a special form of camera resection problem under the so-called algebraic error minimization [64]. Historically, the algebraic error has been sometimes unjustly dismissed in favor of the geometric error. This preconception has been challenged in literature [65] since. Because the algebraic error function does not contain division by the unknown coefficients, it can—as in the case of the formulation presented in this chapter—lead to polynomial optimization problems.

In theory, polynomial optimization problems can be handled using tools of elementary calculus. In real life however, this is not always tractable, as it can be shown that general problem of minimizing polynomial function is NP-hard [128], if the degree of the polynomial is at least four. The field of polynomial optimization over semialgebraic sets received significant research effort during the last decade. To tackle our problem, we leverage on the

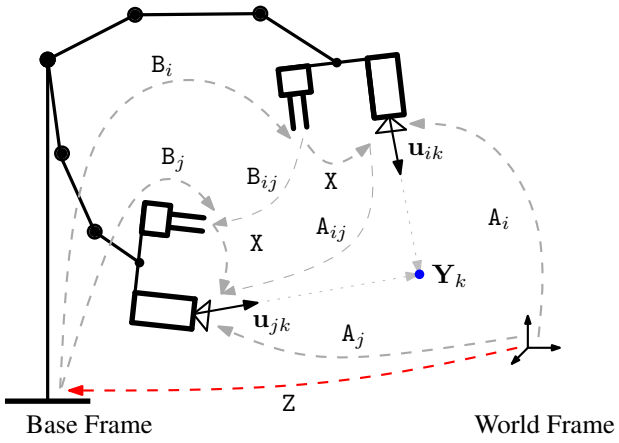


Figure 10.1: *The world-robot transformation*.

results presented by Lasserre in [99], see Section 6.2.1.1. Lasserre formulated a procedure to construct a hierarchy of convex relaxations, so-called linear matrix inequality (LMI) relaxations, that form a monotonous series of lower bounds on the original non-convex problem. These hierarchies lead to global optimal solutions for many practical problems, even for relaxations of low orders. The method of LMI relaxations was first introduced to computer vision community by Kahl and Henrion in [86]. The problem of camera resection was solved in [145] using an approach similar to that presented in this chapter. However, this method is not directly applicable to the robot-world calibration problem, since in this case the camera transformation has further inner structure that changes with the robot position.

The main contribution of this chapter is in formulating the robot-world calibration as a multivariate polynomial minimization problem of a constant size. Further, we show how to build an LMI relaxation of the problem of order two. The order two relaxation had quickly led to certifiably globally optimal solutions for all calibrations task in our experiment with both synthetic an real data.

10.2 Problem Formulation

The objective of robot-world calibration is to derive a transformation

$$\mathtt{Z} = \begin{pmatrix} \mathtt{R}_\mathtt{Z} & \mathbf{t}_\mathtt{Z} \\ \mathbf{0} & 1 \end{pmatrix}, \tag{10.1}$$

$\mathtt{R}_\mathtt{Z} \in SO(3)$, $\mathbf{t}_\mathtt{Z} \in \mathbb{R}^3$, relating the coordinate system connected with a robot's base to the world coordinate system, see Section 5.2. In this chapter, however, we will mean $\mathtt{Z}$ to denote the inverse transformation, from the world coordinate system to the coordinate system connected with the robot: the *world-robot transformation*, see Figure 10.1. This change may seem superfluous, however, it will help us to simplify the notation, since the problem formulation inherently requires the inverse direction of the transformation. To obtain the robot-world transformation, one can simply inverse the resulting matrix *ex post*.

Let us suppose that a camera has been rigidly attached to the end effector of a robotic manipulator. The camera's coordinate system is connected to the coordinate system of the

end effector by a transformation $\mathtt{X}$. This transformation is commonly referred to as hand-eye transformation and here is assumed to be known, *i.e.*, estimated previously using any convenient method. Let us manipulate the robot into n poses so that in each of them the camera observes a calibration target consisting of known points $\mathbf{Y}_k \in \mathbb{R}^3$ in the world coordinate frame, $k = 1, \ldots, m$. The points $\mathbf{Y}_k$ are measured by the camera in the i-th pose at pixel positions $\mathbf{u}'_{ik} \in \mathbb{R}^2$. Since we will assume through the rest of the chapter that the camera is internally calibrated [64], we can easily convert the pixel positions into directional vectors $\mathbf{u}_{ik} \in \mathbb{R}^3$. The algorithm does not require the same set of points to be visible in every camera pose, but we write formulas here assuming that it is so to simplify indexing. Finally, the respective positions of the end effector in each pose can be obtained from the forward kinematics of the robot in the form of transformations $\mathtt{B}'_i$, $i = 1, \ldots, n$, see Section 5.3.2.

The problem of world-robot calibration is closely connected to the problem of hand-eye calibration. The classical approach to solving hand-eye calibration [183, 134, 44] is to recover absolute camera poses $\mathtt{A}'_i$ using known points $\mathbf{Y}_k$ and projections $\mathbf{u}_{ik}$ to construct a set of relative camera movements $\mathtt{A}_{ij} = \mathtt{A}'_j \mathtt{A}'^{-1}_i$, see Section 5.3.1. Using $\mathtt{A}_{ij}$ and relative robot movements $\mathtt{B}_{ij} = \mathtt{B}'_j \mathtt{B}'^{-1}_i$, $i, j = 1, \ldots, n$, an overconstrained system of equations

$$\mathtt{A}_{ij}\mathtt{X} = \mathtt{X}\mathtt{B}_{ij},\ i, j = 1, \ldots, n \tag{10.2}$$

is solved. As noted in [193], a similar transformation loop can be constructed using absolute poses and world-robot transformation as

$$\mathtt{A}'^{-1}_i \mathtt{X} = \mathtt{Z}^{-1} \mathtt{B}'^{-1}_i. \tag{10.3}$$

Since $\mathtt{X} = \mathtt{A}'_i \mathtt{Z}^{-1} \mathtt{B}'^{-1}_i$, $\mathtt{X}$ can be eliminated from Equation 10.3 using an another pair of transformations $\mathtt{A}'_j$, $\mathtt{B}'_j$, and Equation 10.2 can be converted into an instance of hand-eye calibration problem

$$\mathtt{A}'^{-1}_i \mathtt{A}'_j \mathtt{Z}^{-1} = \mathtt{Z}^{-1} \mathtt{B}'^{-1}_i \mathtt{B}'_j. \tag{10.4}$$

Using this approach, world-robot calibration can be cast as hand-eye calibration and solved by any method that solves Equation 10.2, inheriting any strengths and weaknesses of the method at hand. In [193, 46] authors proposed to solve Equation 10.3 directly, obtaining solution for $\mathtt{X}$ and $\mathtt{Z}$ simultaneously.

All of the above methods work with camera poses $\mathtt{A}'_i$. In this chapter, we will show that working with camera projections directly leads to superior results. To do this, we will formulate the world-robot calibration problem as a special form of camera resection problem [64]. The aim of camera resection is to find external camera pose given known 3D points and the calibrated projections. First of all, let us parametrize the camera pose using the known transformations $\mathtt{B}'_i$, $\mathtt{X}$ and unknown transformation $\mathtt{Z}$ as

$$\mathtt{A}'_i = \mathtt{X}\mathtt{B}'_i\mathtt{Z} = \begin{pmatrix} \mathtt{R}_{\mathtt{A}'_i} & \mathbf{t}_{\mathtt{A}'_i} \\ \mathbf{0} & 1 \end{pmatrix} = \begin{pmatrix} a_1 & a_2 & a_3 & a_{10} \\ a_4 & a_5 & a_6 & a_{11} \\ a_7 & a_8 & a_9 & a_{12} \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{10.5}$$

Now, let us denote the coefficients of calibration target points $\mathbf{Y}_{ij} = \begin{pmatrix} x_{ij} & y_{ij} & z_{ij} \end{pmatrix}^\top$ and their projections $\mathbf{u}_{ij} = \begin{pmatrix} u_{ij} & v_{ij} & w_{ij} \end{pmatrix}^\top$. Points $\mathbf{Y}_{ij}$ are transformed into the coordinate

system of the camera as

$$\mathbf{Y}'_{ij} = \mathtt{R}_{\mathtt{A}'_i}\mathbf{Y}_{ij} + \mathbf{t}_{\mathtt{A}'_i}. \tag{10.6}$$

We can relate the projections $\mathbf{u}_{ij}$ and the transformed points $\mathbf{Y}'_{ij}$ by the following equation

$$\begin{pmatrix} \frac{u_{ij}}{w_{ij}} \\ \frac{v_{ij}}{w_{ij}} \end{pmatrix} = \begin{pmatrix} \frac{x'_{ij}}{z'_{ij}} \\ \frac{y'_{ij}}{z'_{ij}} \end{pmatrix} = \begin{pmatrix} \frac{x_{ij}a_1+y_{ij}a_2+z_{ij}a_3+a_{10}}{x_{ij}a_7+y_{ij}a_8+z_{ij}a_9+a_{12}} \\ \frac{x_{ij}a_4+y_{ij}a_5+z_{ij}a_6+a_{11}}{x_{ij}a_7+y_{ij}a_8+z_{ij}a_9+a_{12}} \end{pmatrix}. \tag{10.7}$$

Let's multiply by z'_{ij} and subtract the left side to get

$$\begin{pmatrix} x_{ij}a_1 + y_{ij}a_2 + z_{ij}a_3 + a_{10} - \frac{u_{ij}}{w_{ij}}(x_{ij}a_7 + y_{ij}a_8 + z_{ij}a_9 + a_{12}) \\ x_{ij}a_4 + y_{ij}a_5 + z_{ij}a_6 + a_{11} - \frac{v_{ij}}{w_{ij}}(x_{ij}a_7 + y_{ij}a_8 + z_{ij}a_9 + a_{12}) \end{pmatrix} = \mathbf{0}. \tag{10.8}$$

By reorganizing the constant coefficients in Equation 10.8 into a 2×12 matrix

$$\mathtt{C}_{ij} = \begin{pmatrix} x_{ij} & y_{ij} & z_{ij} & 0 & 0 & 0 & -\frac{u_{ij}}{w_{ij}}x_{ij} & -\frac{u_{ij}}{w_{ij}}y_{ij} & -\frac{u_{ij}}{w_{ij}}z_{ij} & 1 & 0 & \frac{u_{ij}}{w_{ij}} \\ 0 & 0 & 0 & x_{ij} & y_{ij} & z_{ij} & -\frac{v_{ij}}{w_{ij}}x_{ij} & -\frac{v_{ij}}{w_{ij}}y_{ij} & -\frac{v_{ij}}{w_{ij}}z_{ij} & 0 & 1 & \frac{v_{ij}}{w_{ij}} \end{pmatrix} \tag{10.9}$$

and by vectorizing the coefficients of matrix $\mathtt{A}'_i$ as

$$\mathbf{a}_i = \begin{pmatrix} a_1 & a_2 & \dots & a_{12} \end{pmatrix}^\top, \tag{10.10}$$

we can write Equation 10.8 in a more concise form $\mathtt{C}_{ij}\mathbf{a}_i = \mathbf{0}$. In the presence of noise, the residuals $\mathtt{C}_{ij}\mathbf{a}_i$ won't be zero vectors and we can formulate an objective function.

Problem 10.1 (World-robot Calibration).

$$\begin{array}{ll} \text{minimize} & F(\mathtt{Z}) = \sum_{i=1}^{n}\sum_{j=1}^{m} \left\|\mathtt{C}_{ij}\mathbf{a}_i\right\|^2 \\ \text{subject to} & \mathtt{R}_{\mathtt{Z}} \in SO(3). \end{array} \tag{10.11}$$

In [64], the length of the residual vector $\mathtt{C}_{ij}\mathbf{a}_i$ is called algebraic error. It can be given the following geometrical interpretation. Let π be a plane parallel to the projection plane such that $\mathbf{Y}'_{ij} \in \pi$. Let ρ be a ray in direction $\mathbf{u}_{ij}$ originating from the camera center. The algebraic error is the distance between point $\mathbf{Y}'_{ij}$ and the intersection of the ray ρ and the plane π. Because the error model doesn't contain any division by the unknown parameters, we use it to manipulate Problem 10.1 into a polynomial minimization problem. The choice of the error model is also validated by the presented experiments.

Note that the unknown coefficients of the matrix $\mathtt{Z}$ are hidden in the vector $\mathbf{a}_i$. Due to the fact that we need $\mathtt{R}_{\mathtt{Z}}$ to be a rotation matrix, Problem 10.1 is not a convex problem.

10.3 Polynomial World-robot Calibration

In this section we will show how to convert Problem 10.1 to a problem of polynomial minimization of a constant size. We will do it by detaching the unknown coefficient of $\mathtt{Z}$ from $\mathbf{a}_i$.

First, let us parametrize the unknown transformation $\mathtt{Z}$ by the unit quaternion $\mathbf{q}_\mathtt{Z} = \begin{pmatrix} q_1 & q_2 & q_3 & q_4 \end{pmatrix}^\top$ as

$$\mathtt{R}_\mathtt{Z}(\mathbf{q}_\mathtt{Z}) = \begin{pmatrix} q_1^2+q_2^2-q_3^2-2q_4^2 & 2q_2q_3-2q_4q_1 & 2q_2q_4+2q_3q_1 \\ 2q_2q_3+2q_4q_1 & q_1^2-q_2^2+q_3^2-q_4^2 & 2q_3q_4-2q_2q_1 \\ 2q_2q_4-2q_3q_1 & 2q_3q_4+2q_2q_1 & q_1^2-q_2^2-q_3^2+2q_4^2 \end{pmatrix}. \quad (10.12)$$

Further, let's rearrange the columns of the re-parametrized matrix $\mathtt{Z}$ on top of each other into vector $\mathbf{z} = \begin{pmatrix} \mathbf{z}_1^\top & \mathbf{z}_2^\top & \mathbf{z}_3^\top & \mathbf{z}_4^\top \end{pmatrix}^\top$ and the first three rows $\mathbf{d}_{i1}^\top$, $\mathbf{d}_{i2}^\top$, and $\mathbf{d}_{i3}^\top$ of the matrix $\mathtt{D}_i = \mathtt{X}\mathtt{B}_i$ into a 12×16 matrix

$$\mathtt{D}_i'^\top = \begin{pmatrix} \mathbf{d}_{i1} & 0 & 0 & \mathbf{d}_{i2} & 0 & 0 & \mathbf{d}_{i3} & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{d}_{i1} & 0 & 0 & \mathbf{d}_{i2} & 0 & 0 & \mathbf{d}_{i3} & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{d}_{i1} & 0 & 0 & \mathbf{d}_{i2} & 0 & 0 & \mathbf{d}_{i3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{d}_{i1} & \mathbf{d}_{i2} & \mathbf{d}_{i3} \end{pmatrix}. \quad (10.13)$$

Thanks to this rearrangement we can write the residual $\mathtt{C}_{ij}\mathbf{a}_i$ as $\mathtt{C}_{ij}\mathtt{D}'_{ij}\mathbf{z}$. Using the distributive property of sums we can write the objective function $F(\mathtt{Z})$ of Problem 10.1 as

$$F(\mathtt{Z}) = \sum_{i=1}^{n}\sum_{j=1}^{m} \left\|\mathtt{C}_{ij}\mathbf{a}_i\right\|^2 = \sum_{i=1}^{n}\sum_{j=1}^{m} \mathbf{a}_i^\top \mathtt{C}_{ij}^\top \mathtt{C}_{ij}\mathbf{a}_i = \\ = \mathbf{z}^\top \sum_{i=1}^{n}\sum_{j=1}^{m} \left(\mathtt{D}'^\top_{ij}\mathtt{C}_{ij}^\top\mathtt{C}_{ij}\mathtt{D}'_{ij}\right)\mathbf{z} = \mathbf{z}^\top \mathtt{E}\,\mathbf{z}, \quad (10.14)$$

where $\mathtt{E} = (e_{\ell,k})$ is a 16×16 matrix constructed from the known data. The above derivation transformed the objective function F into a polynomial function of degree 4 in 7 unknown parameters $\mathbf{q}_\mathtt{Z}$ and $\mathbf{t}_\mathtt{Z}$. The constraint $\mathtt{R}_\mathtt{Z} \in SO(3)$ is now replaced by the constraint $\left\|\mathbf{q}_\mathtt{Z}\right\|^2 = 1$.

Next, we notice that F is a quadratic function in $\mathbf{t}_\mathtt{Z}$. That means it has one global minimum in $\mathbf{t}_\mathtt{Z}$ that can be easily found by solving $\frac{\partial F}{\partial \mathbf{t}_\mathtt{Z}} = \mathbf{0}$. After a few tedious derivation steps we find that

$$\frac{\partial F}{\partial \mathbf{t}_\mathtt{Z}} = \begin{pmatrix} p^{13} & p^{14} & p^{15} \end{pmatrix}^\top - \mathtt{N}\,\mathbf{t}_\mathtt{Z}, \quad (10.15)$$

where p^k are polynomials in the unknown parameters $\mathbf{q}_\mathtt{Z}$ and $\mathtt{N}$ is a 3×3 matrix

$$\mathtt{N} = -2\begin{pmatrix} e_{13,13} & e_{13,14} & e_{13,15} \\ e_{13,14} & e_{14,14} & e_{14,15} \\ e_{13,15} & e_{14,15} & e_{15,15} \end{pmatrix}. \quad (10.16)$$

Using 5×5 matrices $\mathtt{P}^k$, see Equation 10.24, polynomials p^k are constructed as

$$p^k = \begin{pmatrix} 1 & \mathbf{q}_\mathtt{Z}^\top \end{pmatrix} \mathtt{P}^k \begin{pmatrix} 1 & \mathbf{q}_\mathtt{Z}^\top \end{pmatrix}^\top.$$

Finally, by solving $\frac{\partial F}{\partial \mathbf{t}_\mathtt{Z}} = \mathbf{0}$ we get the optimal $\mathbf{t}_\mathtt{Z}$ as

$$\mathbf{t}_\mathtt{Z}^* = \begin{pmatrix} t_1^* & t_2^* & t_3^* \end{pmatrix}^\top = \mathtt{N}^{-1}\begin{pmatrix} p^{13} & p^{14} & p^{15} \end{pmatrix}^\top. \quad (10.17)$$

$$\mathtt{Q}^\top = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h^1_{1,1} & h^2_{1,1} & h^3_{1,1} & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & h^1_{2,2} & h^2_{2,2} & h^3_{2,2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & -2 & 0 & 0 & h^1_{3,2} & h^2_{2,3} & h^3_{2,3} & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & h^1_{3,3} & h^2_{3,3} & h^3_{3,3} & 0 \\
0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & h^1_{4,2} & h^2_{4,2} & h^3_{4,2} & 0 \\
0 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h^1_{4,3} & h^2_{4,3} & h^3_{4,3} & 0 \\
-1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & h^1_{4,4} & h^2_{4,4} & h^3_{4,4} & 0 \\
0 & 2 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h^1_{5,2} & h^2_{5,2} & h^3_{5,2} & 0 \\
0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & h^1_{5,3} & h^2_{5,3} & h^3_{5,3} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & h^1_{5,4} & h^2_{5,4} & h^3_{5,4} & 0 \\
-1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & h^1_{5,5} & h^2_{5,5} & h^3_{5,5} & 0
\end{pmatrix} \tag{10.22}$$

To get the explicit formula for t^*_i, $i = 1, ..., 3$, let's us denote the elements of the matrix $\mathtt{N}^{-1}$ as $(n'_{\ell,k})$. Now, by constructing matrices

$$\mathtt{H}^k = n'_{k,1}\mathtt{P}^{13} + n'_{k,2}\mathtt{P}^{14} + n'_{k,3}\mathtt{P}^{15} = (h_{\ell,k}), \tag{10.18}$$

we can write the t^*_i explicitly as

$$t^*_i = \begin{pmatrix} 1 & \mathbf{q}_\mathtt{Z}^\top \end{pmatrix} \mathtt{H}^i \begin{pmatrix} 1 & \mathbf{q}_\mathtt{Z}^\top \end{pmatrix}^\top, \; i = 1, \ldots, 3. \tag{10.19}$$

The Substitution of $\mathbf{t}^*_\mathtt{Z}$ into $F(\mathtt{Z})$ yields a new objective function $F'(\mathbf{q}_\mathtt{Z})$. This substitution takes the number of unknown parameters down to four.

Let us simplify the notation of $F'(\mathbf{q}_\mathtt{Z})$ a bit more. First, we notice from the formula for the matrices $\mathtt{H}^k$ that they have the same amount of non-zero elements as the matrices $\mathtt{P}^k$. Let us rearrange these non-zero elements into a vector

$$\mathbf{h}^k = \begin{pmatrix} h^k_{1,1} & h^k_{2,2} & h^k_{3,2} & h^k_{3,3} & h^k_{4,2} & h^k_{4,3} & h^k_{4,4} & h^k_{5,2} & h^k_{5,3} & h^k_{5,4} & h^k_{5,5} \end{pmatrix}^\top. \tag{10.20}$$

Now, using a vector of 11 monomials

$$\mathbf{m} = \begin{pmatrix} 1 & q_1^2 & q_1q_2 & q_2^2 & q_1q_3 & q_2q_3 & q_3^2 & q_1q_4 & q_2q_4 & q_3q_4 & q_4^2 \end{pmatrix}, \tag{10.21}$$

we can write $t^*_i = \mathbf{h}^{i\top}\mathbf{m}$. In other words, $\mathbf{h}^i$ are the coordinates of t^*_i in the monomial basis $\mathbf{m}$. Since the monomial basis $\mathbf{m}$ can be used for all of the polynomials in the vector $\mathbf{z}$, we can stack their coordinates into a 16×11 matrix $\mathtt{Q}$, see Equation 10.22, and write $\mathbf{z}$ as $\mathtt{Q}\mathbf{m}$. Using this notation we can state an equivalent formulation of Problem 10.1:

Problem 10.2.

$$\begin{array}{ll} \text{minimize} & F'(\mathbf{q}_\mathtt{Z}) = \mathbf{m}^\top(\mathtt{Q}^\top \mathtt{E}\, \mathtt{Q})\, \mathbf{m} = \mathbf{m}^\top \mathtt{E}'^\top \mathbf{m} \\ \text{subject to} & q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1, \\ & q_1 \geq 0. \end{array} \tag{10.23}$$

$$\mathtt{P}^k = 2\begin{pmatrix} e_{k,16} & 0 & 0 & 0 & 0 \\ 0 & e_{1,k}+e_{6,k}+e_{11,k} & 0 & 0 & 0 \\ 0 & 2(e_{7,k}-e_{10,k}) & e_{1,k}-e_{6,k}-e_{11,k} & 0 & 0 \\ 0 & 2(e_{9,k}-e_{3,k}) & 2(e_{2,k}+e_{5,k}) & -e_{1,k}+e_{6,k}-e_{11,k} & 0 \\ 0 & 2(e_{2,k}-e_{5,k}) & 2(e_{3,k}+e_{9,k}) & 2(e_{7,k}+e_{10,k}) & -e_{1,k}-e_{6,k}+e_{11,k} \end{pmatrix} \tag{10.24}$$

Problem 10.2 is a polynomial optimization problem of degree 4 in 4 unknowns and it is a problem of a constant size, *i.e.*, the dimensions of the matrix $\mathtt{E}'$ are 11×11 regardless of the number of measurements $\mathbf{u}_{ij}$. Notice that we have replaced the constraint $\mathtt{R}_\mathtt{Z} \in SO(3)$ with the constraint $\mathbf{q}_\mathtt{Z}$ has to be a unit quaternion. Since for every rotation there are two equivalent quaternions $\mathbf{q}$ and $-\mathbf{q}$ representing it, we have added another constraint $q_1 \geq 0$ to eliminate most of the double solutions. Suppose $\mathbf{q}_\mathtt{Z}^*$ is the minimizer of Problem 10.2, the optimal translation $\mathbf{t}_\mathtt{Z}^*$ can be then computed using Equation 10.19.

10.4 Convex LMI Relaxations

Here, we will show the LMI relaxation of Problem 10.2 of order two. To do that, we need monomial bases $\mathbf{v}_1$ and $\mathbf{v}_2$ in variables $\mathbf{q}_\mathtt{Z}$,

$$\begin{aligned} \mathbf{v}_1(\mathbf{q}_\mathtt{Z}) &= (\ 1 \quad q_1 \quad q_2 \quad q_3 \quad q_4\ )^\top, \\ \mathbf{v}_2(\mathbf{q}_\mathtt{Z}) &= (\ 1 \quad q_1 \quad q_2 \quad q_3 \quad q_4 \quad q_1q_2 \quad q_1q_3 \quad q_1q_4 \quad q_2q_3 \\ &\qquad\qquad\qquad\qquad\quad q_2q_4 \quad q_3q_4 \quad q_1^2 \quad q_2^2 \quad q_3^2 \quad q_4^2\ )^\top. \end{aligned} \tag{10.25}$$

Further, we need to lift the variables in the objective function F', the constraints and the two moment matrices $\mathtt{M}_1 = \mathbf{v}_1(\mathbf{q}_\mathtt{Z})\mathbf{v}_1(\mathbf{q}_\mathtt{Z})^\top$ and $\mathtt{M}_2 = \mathbf{v}_2(\mathbf{q}_\mathtt{Z})\mathbf{v}_2(\mathbf{q}_\mathtt{Z})^\top$. For the lack of space we will show the results of the lifting for $\mathtt{M}_1$ and the constraints only:

$$\mathtt{M}_1 \Rightarrow \begin{pmatrix} 1 & y_{1000} & y_{0100} & y_{0010} & y_{0001} \\ y_{1000} & y_{2000} & y_{1100} & y_{1010} & y_{1001} \\ y_{0100} & y_{1100} & y_{0200} & y_{0110} & y_{0101} \\ y_{0010} & y_{1010} & y_{0110} & y_{0020} & y_{0011} \\ y_{0001} & y_{1001} & y_{0101} & y_{0011} & y_{0002} \end{pmatrix}, \tag{10.26}$$

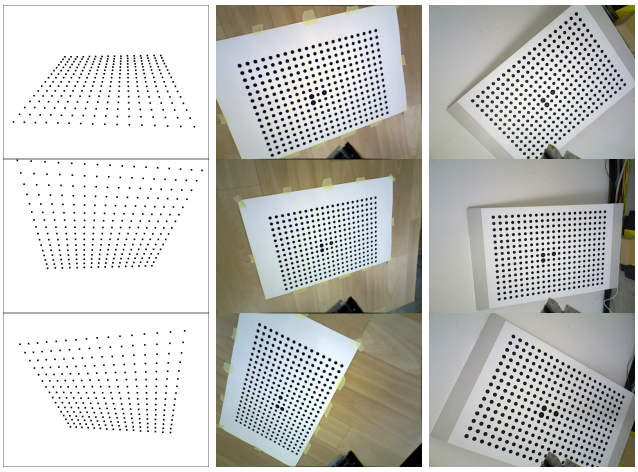
$$\begin{aligned} q_1^2 + q_2^2 + q_3^2 + q_4^2 &\Rightarrow y_{2000} + y_{0200} + y_{0020} + y_{0002}, \\ q_1 &\Rightarrow y_{1000}. \end{aligned} \tag{10.27}$$

Now, we can formulate the second LMI relaxation of 10.2 as

Problem 10.3 (The second LMI relaxation).

$$\begin{aligned} \text{minimize} \quad & F'(\mathbf{y}) \\ \text{subject to} \quad & (y_{2000} + y_{0200} + y_{0020} + y_{0002})\mathtt{M}_1 \succeq 0, \\ & -(y_{2000} + y_{0200} + y_{0020} + y_{0002})\mathtt{M}_1 \succeq 0, \\ & y_{1000}\mathtt{M}_1 \succeq 0, \\ & \mathtt{M}_2 \succeq 0. \end{aligned} \tag{10.28}$$

Problem 10.3 is a semidefinite program and as such solvable by any SDP solver available.



(a) (b) (c)

Figure 10.2: *Examples of source images from the synthetic and real data sequences.* (a) 3 synthetically generated images of a 16×16 calibration grid. No image noise is present. (b) 3 images of the smaller 21×15 calibration grid from the real data experiment. (c) 3 images of the larger calibration grid.

10.5 Experiments

In this section, we present both synthetic and real data experiments to validate the proposed algorithm. For the real data experiment we used a Motoman MA1400 serial 6-DOF manipulator with an Asus Xtion Pro sensor rigidly attached to it. The Xtion Pro sensor is equipped with a camera with the resolution of 640×480 pixels. We simulated the same setup in the synthetic experiment in order to better judge the result of the experiment with real data experiment.

We implemented the calibration algorithm in MATLAB. We used GloptiPoly [79] to convert the problem instances into the LMI relaxations and SeDuMi [168] as the SDP solver. We used the LMI relaxations of the second order in both experiments, obtaining certified global minimum for every calibration instance. We used a 3GHz Intel Core i7 based desktop computer running 64-bit Linux to run the experiments. The mean time the SDP solver took to converge was 0.3 ± 0.006 s.

10.5.1 Synthetic Experiment

In the synthetic experiment we investigated the influence of both image and joint noises on the calibration accuracy. To do that we placed a planar calibration target consisting of 16×16 points in front of a simulated MA1400 serial manipulator. The distance between the points was 12.5 mm in each direction, making the calibration device 200×200 mm in size.

For every calibration task, 9 camera positions $\mathtt{A}_i'$ were randomly generated in a half-sphere of radius of ~ 30 mm facing approximately the center of the calibration device. To better visualize such camera poses, Figure 10.2(a) shows an example of a sequence of 3 image taken by the virtual Xtion Pro sensor. Next, a hand-eye transformation $\mathtt{X}$ was randomly

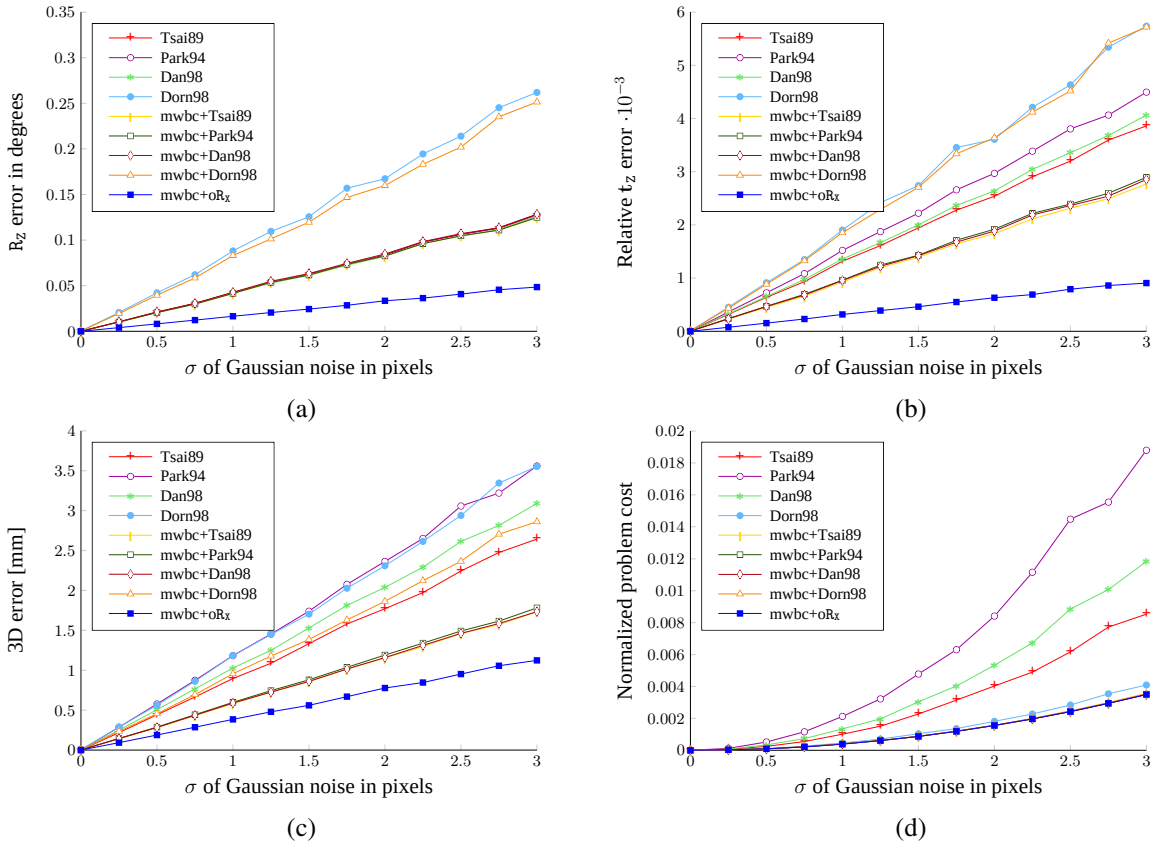


Figure 10.3: *Results of the image noise experiment.* (a) Error of $\mathtt{R}_\mathtt{Z}$ in degrees, (b) relative error of $\mathtt{t}_\mathtt{Z}$, (c) 3D error of the transformation $\mathtt{Z}$ (see Equation 10.29), (d) value of the objective function F'.

generated so that the rotation corresponds to the pose of the camera to a up to 5° degrees difference in each axis. This reflects a typical situation when the camera faces the same direction as the robot's end effector. The translation was randomly generated to move the camera up to ~ 200 mm away from end effector. The end effector poses $\mathtt{B}'_i$ were computed from the camera poses and hand-eye transformation as $\mathtt{X}^{-1}\mathtt{A}'_i$. Finally, we generated a world-robot transformation $\mathtt{Z}$ using a random rotation and a random translation up to ~ 2000 mm in length and transformed the grid points by the inverse $\mathtt{Z}^{-1}$. We generated 10 sets of camera positions, 10 hand-eye transformations and 10 world-robot transformations and combined them into 1000 calibration tasks.

We implemented several published method to pit our algorithm against: "Tsai89" [183], "Park94" [134], "Dan98" [44], and "Dorn98" [46]. There are several points to bear in mind while judging the comparison results. First, all of the above methods work with camera transformations $\mathtt{A}'_i$ rather than with projections $\mathbf{u}_{ij}$ directly. We used EPnP algorithm [101] to recover matrices $\mathtt{A}'_i$ for every calibration task. Next, "Tsai89", "Park94", and "Dan98" are hand-eye calibration method, so Equation 10.4 was needed to cast the world-robot calibration as hand-eye calibration problem. Since Equation 10.4 requires relative rather than absolute transformations, we constructed all the 36 relative transformations as the input for these methods. Finally, unlike our algorithm, none of the methods require prior knowledge of the

hand-eye transformation $\mathtt{X}$. To reflect this fact, we run our algorithm—marked as "mwbc" in the plots—four times using $\mathtt{X}$ recovered by its competitors. We have also added a fifth case that uses the ground truth $\mathtt{X}$, marked as "o$\mathtt{R}_\mathtt{X}$".

Image Noise Experiment. To simulate the influence of image noise on the calibration accuracy we corrupted the projections $\mathbf{u}_{ij}$ in each calibration task with Gaussian noise in the image domain in 13 noise levels, $\sigma \in \langle 0, 3 \rangle$ px in $1/4$ px steps. Figure 10.3(a) shows the mean error of the rotational part $\mathtt{R}_\mathtt{Z}$ as the angle θ of the rotation $\mathtt{R}_\mathtt{Z}^\top \mathtt{R}_\mathtt{Z}^\mathrm{gt}$ such that $0 \leq \theta \leq \pi$ with $\mathtt{R}_\mathtt{Z}^\mathrm{gt}$ being the ground truth rotation. Here "mwbc" slightly outperforms the methods from which it inherited the hand-eye transformation $\mathtt{X}$. Figure 10.3(b) shows the mean error of the translational part $\mathbf{t}_\mathtt{Z}$ as $\left\|\mathbf{t}_\mathtt{Z} - \mathbf{t}_\mathtt{Z}^\mathrm{gt}\right\| / \left\|\mathbf{t}_\mathtt{Z}^\mathrm{gt}\right\|$; $\mathbf{t}_\mathtt{Z}^\mathrm{gt}$ being the ground truth translation. In this regard "mwbc" performs better except for "mwbc+Dorn98" combination. The most informative, however, is Figure 10.3(c), were the quality of the complete transformation $\mathtt{Z}$ is gauged. We uniformly sampled the workspace of the robot with $n = 9240$ points $\mathbf{Y}_j$. Figure 10.3(c) shows the mean distance between points transformed into the coordinate frame of the cameras using recovered transformations $\mathtt{Z}$, $\mathtt{X}$ and using ground truth transformations $\mathtt{Z}^\mathrm{gt}$, $\mathtt{X}^\mathrm{gt}$. Specifically,

$$E = \frac{1}{9n} \sum_{i=1}^{9} \sum_{j=1}^{n} \left\| \mathtt{X}\mathtt{B}_i'\mathtt{Z}\mathbf{Y}_j - \mathtt{X}^\mathrm{gt}\mathtt{B}_i'\mathtt{Z}^\mathrm{gt}\mathbf{Y}_j \right\|. \tag{10.29}$$

Using this measure, we can see "mbwc" to outperform its competitors by the factor of two, with "mwbc+Dorn98" being an exception.

For reference, Figure 10.3(d) shows the mean value of the objective function F' divided by the number of projections of the respective calibration task. We call this measure normalized problem cost.

Joint Noise Experiment. To see how our method performs in the presence of joint noise, we generated 1000 calibration tasks using the same procedure used the image noise experiment. Next, we recovered the joint coordinates [150] of the virtual MA1400 manipulator for every pose $\mathtt{B}_i'$ with respect to the Denavit–Hartenberg convention using the inverse kinematics [150]. Then, we corrupted the joint coordinates by random offsets—we used the same offsets for the joint coordinates of the same task—in 11 noise levels, $\sigma \in \langle 0, 0.25 \rangle$ deg in steps of 0.025 degrees. Finally, we recovered noised poses $\mathtt{B}_i'$ as the forward kinematics task. To further simulate the real world conditions, we corrupted projections $\mathbf{u}_{ij}$ by image noise of $\sigma = 0.5$ px for every joint noise level.

Figures 10.4(a–d) show the resulting plots based on the same error criteria that were used in Figures 10.3(a–d). The results are analogous to the results of the image noise experiment, albeit with smaller performance gain of the "mwbc" based methods and with significantly larger errors. This is to be expected, since this time the input data was significantly noisier. Again, "mwbc+Dorn98" looses on the rest of the "mwbc" methods. However, Figure 10.4(c) shows "mwbc+Tsai89", "mwbc+Park94", and "mwbc+Dorn98" outperforming the rest.

10.5.2 Real Data Experiment

Figure 10.5(a) shows an overview of the setup used for the real data experiment. We used a real MA1400 serial manipulator with a Xtion Pro sensor attached to its 5th link, see Fig-

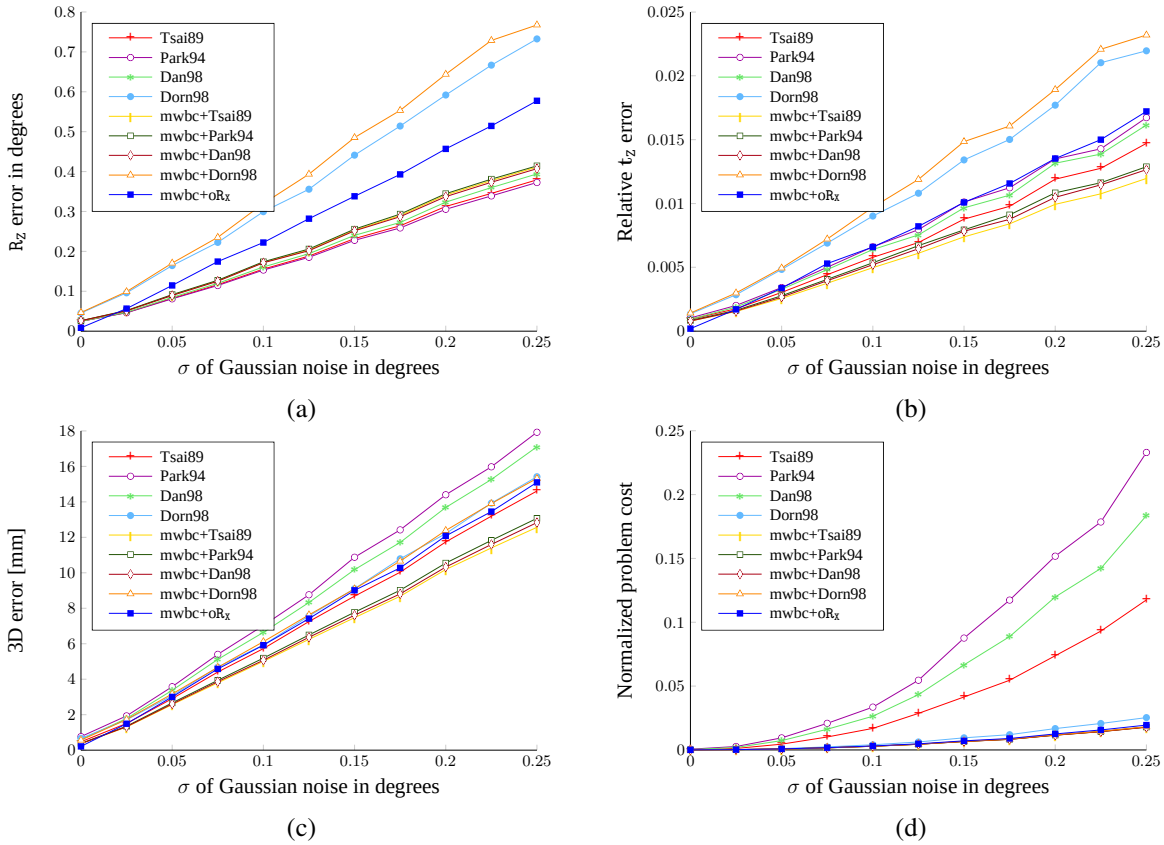


Figure 10.4: *Results of the joint noise experiment*. (a) Error of $\mathtt{R}_\mathtt{Z}$ in degrees, (b) relative error of $\mathtt{t}_\mathtt{Z}$, (c) 3D error of the transformation $\mathtt{Z}$ (see Equation 10.29), (d) value of the objective function F'.

(a) (b)

Figure 10.5: *Real data experiment setup*. (a) "Fisheye" view of the calibration setup. (b) Detail of the Xtion Pro sensor and the robot's end effector.

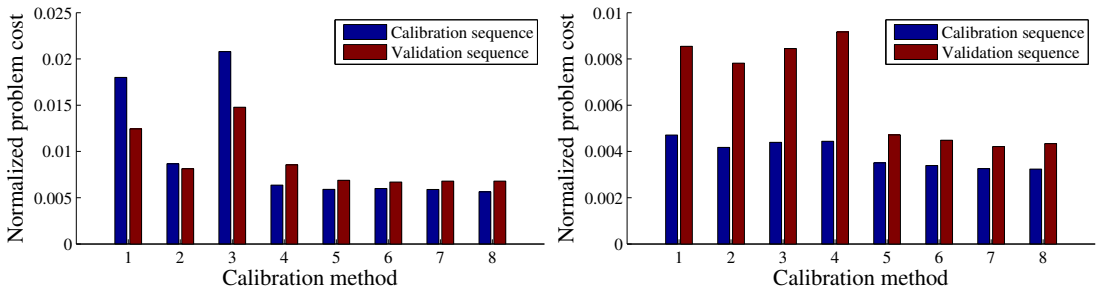


Figure 10.6: *Real data experiment results*. The plots show values of normalized problem cost obtained for smaller (a) and larger (b) calibration grid. The numbers match to the calibration methods as follows: (1) "Tsai89", (2) "Park94", (3) "Dan98", (4) "Dorn98", (5) "mwbc+Tsai89", (6) "mwbc+Park94", (7) "mwbc+Dan98", (8) "mwbc+Dorn98".

ure 10.5(b). OpenCV [2] library was used to obtain the internal calibration of the sensor. We constructed two calibration tasks using calibration grids of different dimensions. In both cases the calibration sequence consisted of 9 images of a 21×15 points calibration grid with an "L" shape in the middle to help determine the grid's orientation. The point spacing was 16 mm in the case of the smaller calibration grid, 40 mm in the case of the larger grid. Validation sequences of 2 images each were also taken. See Figures 10.2(a–b) for sample images from the calibration sequences.

Since in this case there were no ground truth hand-eye and world-base calibrations available, we only show the normalized problem cost measure plots. Figures 10.5(a–b) show such plots for different calibration methods for smaller and larger calibration grid respectively. As in the case of the synthetic data experiments, we used methods "Tsai89", "Park94", "Dan98", and "Dorn98" and their respective combinations with our method to perform the calibration. The plots show that the "mwbc" based methods outperform its competitors on both calibration and validation sequences. The plots also show that the results are systematically better for larger calibration grids. This suggests that larger calibration grids might be preferable for the world-base calibration in case of this setup.

11 Hand-Eye and Robot-World Calibration by LMI Relaxations

There is no subject so old that something new cannot be said about it.

– Fyodor Dostoyevsky

In this chapter, we propose a set of iterative methods to solve hand-eye and robot-world calibration. Unlike the calibration techniques presented in the previous chapters, these methods do not work with image measurements directly and require the absolute camera poses $\mathtt{A}'_i$ to be known. While working with image measurements directly can indeed eliminate the errors resulting from explicit computation of absolute poses $\mathtt{A}'_i$, such an approach is only applicable in situations where the image correspondences are available, which may not always be the case. The proposed methods do not require initial estimates and provide globally optimal solutions in L_2-norm. Further, these methods solve for the rotational and translational part simultaneously. This is achieved by formulating the hand-eye and robot-world calibration problems as multivariate polynomial optimization problems over semialgebraic sets and by solving them using the method of convex linear matrix inequality (LMI) relaxations presented in Section 6.2.1.1.

11.1 Introduction

Let us briefly review the hand-eye and robot-world calibration problems from Sections 5.1 and 5.2. Let's suppose that the hand-eye robotic system has been manipulated into two distinct poses, see Figure 5.2. Let's denote the absolute camera poses $\mathtt{A}'_1$ and $\mathtt{A}'_2$. We can express the camera's relative pose as

$$\mathtt{A} = \begin{pmatrix} \mathtt{R}_\mathtt{A} & \mathbf{t}_\mathtt{A} \\ \mathbf{0}^\top & 1 \end{pmatrix} = \begin{pmatrix} \mathtt{R}_{\mathtt{A}'_2} & \mathbf{t}_{\mathtt{A}'_2} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} \mathtt{R}_{\mathtt{A}'_1} & \mathbf{t}_{\mathtt{A}'_1} \\ \mathbf{0}^\top & 1 \end{pmatrix}^{-1} = \mathtt{A'}_2\mathtt{A'}_1^{-1},$$

where $\mathtt{R}_\mathtt{A} \in SO(3)$ is a 3×3 rotation matrix and $\mathbf{t}_\mathtt{A} \in \mathbb{R}^3$ is a translation vector. Analogically, relative movement of the robotic end-effector can be described as

$$\mathtt{B} = \begin{pmatrix} \mathtt{R}_\mathtt{B} & \mathbf{t}_\mathtt{B} \\ \mathbf{0}^\top & 1 \end{pmatrix} = \begin{pmatrix} \mathtt{R}_{\mathtt{B}'_2} & \mathbf{t}_{\mathtt{B}'_2} \\ \mathbf{0}^\top & 1 \end{pmatrix}^{-1} \begin{pmatrix} \mathtt{R}_{\mathtt{B}'_1} & \mathbf{t}_{\mathtt{B}'_1} \\ \mathbf{0}^\top & 1 \end{pmatrix} = \mathtt{B'}_2^{-1}\mathtt{B}'_1,$$

with $\mathtt{B}'_1, \mathtt{B}'_2 \in \mathbb{R}^{4\times4}$ being the respective transformations from the end-effector's coordinate system to the robot base coordinate system. Assuming we know the transformations $\mathtt{A}$ and $\mathtt{B}$, the problem of finding the rigid transformation $\mathtt{X} \in \mathbb{R}^{4\times4}$ from the coordinate system of end-effector to the coordinate system connected with the camera can be expressed analytically using the following kinematic loop:

$$\mathtt{AX} = \mathtt{XB}. \tag{11.1}$$

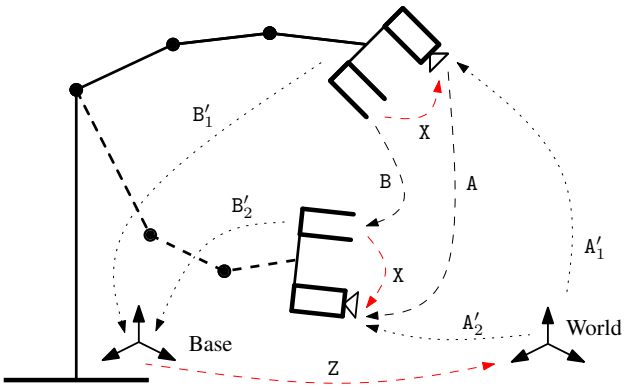


Figure 11.1: Hand-eye (X) and robot-world (Z) transformations. A hand-eye system is depicted in two different poses.

All of the earliest researchers investigating this problem realized that System 11.1 is underdetermined and that two poses are not enough to uniquely determine the transformation X. In [149], Shiu and Ahmad showed that at least two relative motions with non-parallel rotational axes are needed. In practice, several relative motions are executed, leading to a set of matrices $\mathtt{A}_i, \mathtt{B}_i$, $i = 1, \ldots, n$ and to an overdetermined and—unless we can measure $\mathtt{A}_i, \mathtt{B}_i$ with perfect accuracy—noisy system of equations

$$\mathtt{A}_i\mathtt{X} = \mathtt{X}\mathtt{B}_i, \quad i = 1, \ldots, n. \tag{11.2}$$

System 11.2 can be further decomposed in into a rotational matrix equation and translational vector equation

$$\mathtt{R}_{\mathtt{A}_i}\mathtt{R}_\mathtt{X} = \mathtt{R}_\mathtt{X}\mathtt{R}_{\mathtt{B}_i}, \tag{11.3}$$

$$\mathtt{R}_{\mathtt{A}_i}\mathbf{t}_\mathtt{X} + \mathbf{t}_{\mathtt{A}_i} = \mathtt{R}_\mathtt{X}\mathbf{t}_{\mathtt{B}_i} + \mathbf{t}_\mathtt{X}. \tag{11.4}$$

Notice that Equation 11.3 does not depend on the unknown translation $\mathbf{t}_\mathtt{X}$. Once rotation $\mathtt{R}_\mathtt{X}$ is known, Equation 11.4 leads to a system of linear equations in $\mathbf{t}_\mathtt{X}$ and the translation can be easily determined using tools of linear algebra. This fact has been exploited by all of the earliest solution strategies.

In [33], Chen employed the screw motion theory, see Section 3.5, to investigate the necessary and sufficient conditions for the solutions of Equation 11.2. Chen concluded that in the case of noisy inputs the computation of $\mathtt{R}_\mathtt{X}$ and $\mathbf{t}_\mathtt{X}$ should not be decoupled, because otherwise the generality of the result would be negatively affected. In [43, 44], Daniilidis and Bayro-Corrochano showed how to parametrize Equation 11.2 using the algebraic counterparts of screws—dual quaternions—and how to solve for rotational and translational parts of X simultaneously. Another simultaneous solution to Equation 11.2 was proposed by Andreff *et al.* in [8] using the Kronecker product. However, their solution needs to be followed by a orthogonalization of the rotational part $\mathtt{R}_\mathtt{X}$.

Several researchers also proposed iterative solutions to Equation 11.2. In [194], Zhuang and Shiu proposed an iterative non-linear method to minimize function $\sum_{i=1}^{n} \left\|\mathtt{A}_i\mathtt{X} - \mathtt{X}\mathtt{B}_i\right\|^2$ to simultaneously estimate the rotational and translational parts of X. As a part of [81],

Horaud and Dornaika also proposed a simultaneous iterative method based of quaternions and Levenberg-Marquardt non-linear optimization [119]. They observed that the method performed well only after introducing two *ad hoc* selected scaling factors. Both methods need to be provided with an initial solution estimated and depending on its accuracy may not converge to the global optimum. In [191], Zhao suggested an iterative method that is guaranteed to converge globally optimally based on Second Order Cone Programming. However, his iterative method does not enforce the orthogonality of the rotation matrix and thus suffers the same rotational error propagation as does the method [8].

In [193], Zhuang *et al.* extended the hand-eye calibration problem to also include calibration of the robot-world transformation $\mathtt{Z}$, see Figure 5.2. Their method uses quaternion rotation representation to solve an equation analogical to the Equation 11.2,

$$\mathtt{A}_i'^{-1}\mathtt{X} = \mathtt{Z}\mathtt{B}_i', \; i = 1, \ldots, m, \tag{11.5}$$

where $\mathtt{Z} \in \mathbb{R}^{4\times4}$ represents transformation from the robot base coordinate system to the world coordinate system. In this case, the kinematic loop is closed using the absolute camera and robot poses $\mathtt{A}_i', \mathtt{B}_i'$. Dornaika and Horaud [46] suggested a different solution, also based on quaternions. In [7], Li *et al.* proposed two different solutions based on Kronecker product and dual quaternions, analogous to the solutions to the hand-eye calibration problem in [8] and [44], respectively.

In previous chapters, several calibration methods were proposed that use image measurements directly, instead of using them to compute matrices $\mathtt{A}_i'$ as a pre-step. This approach can also be seen in works [146, 91, 14]. While such an approach can indeed eliminate the errors resulting from explicit computation of matrices $\mathtt{A}_i'$, it is only applicable in situations where the image correspondences are available, which may not always be the case.

In this chapter, we propose a set of iterative methods to solve hand-eye and robot-world calibration based on Equations 11.2 and 11.5 that do not require initial estimates and provide globally optimal solutions in L_2-norm. Further, these methods solve for the rotational and translational part simultaneously. In the experimental section, we provide a few implementation details and show the performance of the proposed solution using both synthetic and real data calibration scenarios.

11.2 Hand-Eye Calibration

In this section, three formulation of the hand-eye calibration problem are presented. In the first two, we formulate two parametrizations of of the following minimization problem:

$$\min_{\mathtt{X}\in SE(3)} \sum_{i=1}^{n} \left\|\mathtt{A}_i\mathtt{X} - \mathtt{X}\mathtt{B}_i\right\|^2,$$

i.e., the minimization the Frobenius norm

$$\|\mathtt{M}\| = \sqrt{\textstyle\sum_{i=1}^{m}\sum_{j=1}^{n}|m_{ij}|^2} = \sqrt{\mathrm{tr}(\mathtt{M}^\top\mathtt{M})}$$

on the special Euclidean group

$$SE(3) = \left\{ \begin{pmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix} \middle| \, \mathtt{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3 \right\}.$$

The third formulation uses the dual quaternion parametrization to minimize the vector L_2-norm

$$\min_{\tilde{\mathbf{q}}_\mathtt{X}} \sum_{i=1}^{n} \left\| \tilde{\mathbf{a}}_i \otimes \tilde{\mathbf{q}}_\mathtt{X} - \tilde{\mathbf{q}}_\mathtt{X} \otimes \tilde{\mathbf{b}}_i \right\|^2,$$

where $\tilde{\mathbf{a}}_i, \tilde{\mathbf{b}}_i, \tilde{\mathbf{q}}_\mathtt{X}$ are the dual quaternion representations of $\mathtt{A}_i$,$\mathtt{B}_i$, and $\mathtt{X}$, respectively, and $\otimes$ is the dual quaternion multiplication.

All three formulations lead to the multivariate polynomial optimization problems. However, for the sake of brevity we will not cite the explicit form of the polynomials. The fact that these formulations are indeed polynomial can be easily checked by any tool for symbolic algebra computation.

11.2.1 Orthonormal parametrization

As is the case with all linear maps on finite-dimensional vector spaces, a rotation can be always expressed by a matrix $\mathtt{R}$, in this case of size 3×3. Since a rotation maps orthonormal basis of $\mathbb{R}^3$ to another orthonormal basis, the columns $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$ of the matrix $\mathtt{R} = (\mathbf{u}, \mathbf{v}, \mathbf{w})$ themselves must form an orthonormal basis. The fact that the columns of $\mathtt{R}$ form a orthonormal basis can be also written as

$$\begin{aligned} \mathbf{v}^\top \mathbf{v} = 1, \quad & \mathbf{u}^\top \mathbf{u} = 1, \\ \mathbf{v}^\top \mathbf{u} = 0, \quad & \mathbf{v} \times \mathbf{u} = \mathbf{w}. \end{aligned}$$

This constitutes 6 constraints on the elements of the matrix $\mathtt{R}$, leaving it with 3 degrees of freedom. Using these constraints, we can parametrize the homogeneous transformation $\mathtt{X}$ as

$$\mathtt{X}(\mathbf{u}, \mathbf{v}, \mathbf{t}) = \begin{pmatrix} \mathtt{R}(\mathbf{u}, \mathbf{v}) & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix},$$

where $\mathtt{R}(\mathbf{u}, \mathbf{v}) = (\mathbf{u}, \mathbf{v}, \mathbf{u} \times \mathbf{v})$. This leads to the following parametrization of the hand-eye calibration problem:

Problem 11.1 (*uvhec* method).

$$\begin{aligned} \text{minimize} \quad & f_1(\mathbf{u}_\mathtt{X}, \mathbf{v}_\mathtt{X}, \mathbf{t}_\mathtt{X}) = \\ & \quad \textstyle\sum_{i=1}^{n} \left\| \mathtt{A}_i \mathtt{X}(\mathbf{u}_\mathtt{X}, \mathbf{v}_\mathtt{X}, \mathbf{t}_\mathtt{X}) - \mathtt{X}(\mathbf{u}_\mathtt{X}, \mathbf{v}_\mathtt{X}, \mathbf{t}_\mathtt{X}) \mathtt{B}_i \right\|^2 \\ \text{subject to} \quad & \mathbf{u}_\mathtt{X}^\top \mathbf{u}_\mathtt{X} = 1, \mathbf{v}_\mathtt{X}^\top \mathbf{v}_\mathtt{X} = 1, \mathbf{u}_\mathtt{X}^\top \mathbf{v}_\mathtt{X} = 0. \end{aligned}$$

The objective function f_1 of Problem 11.1 is a polynomial function of degree 4 and it is composed of 123 monomials in 9 variables.

11.2.2 Quaternion parametrization

Quaternions, $\mathbb{Q}$, form a four-dimensional associative normed division algebra over the real numbers. A quaternion $\mathbf{q} \in \mathbb{Q}$ consist of a real part and an imaginary part and is usually denoted as

$$\mathbf{q} = q_1 + q_2\mathbf{i} + q_3\mathbf{j} + q_4\mathbf{k},$$

where $\mathbf{i}, \mathbf{j}, \mathbf{k}$ are the imaginary units such that

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1.$$

As a set, quaternions are equal to $\mathbb{R}^4$ and it is sometimes useful to write them as vectors

$$\mathbf{q} \equiv (q_1, q_2, q_3, q_4)^\top = \left(q_1, \bar{\mathbf{q}}^\top\right)^\top,$$

where $\bar{\mathbf{q}} \in \mathbb{R}^3$ is the imaginary part of the quaternion. Addition of two quaternions $\mathbf{p}, \mathbf{q} \in \mathbb{Q}$, $\mathbf{p} = \left(p_1, \bar{\mathbf{p}}^\top\right)^\top$, $\mathbf{q} = \left(q_1, \bar{\mathbf{q}}^\top\right)^\top$ is equivalent to addition in $\mathbb{R}^4$. Quaternion multiplication, however, does not have a counterpart operation on vector spaces:

$$\mathbf{p} * \mathbf{q} = (p_1 q_1 - \bar{\mathbf{p}}^\top \bar{\mathbf{q}}, (p_1\bar{\mathbf{q}} + q_1\bar{\mathbf{p}} + \bar{\mathbf{p}} \times \bar{\mathbf{q}})^\top)^\top.$$

Because the group of unit quaternions with multiplication, modulo the multiplication by -1, is isomorphic to the group of rotations with composition, they can be used to represent rotations. Rotation about axis $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)^\top$, $||\boldsymbol{\alpha}|| = 1$, by angle θ is represented by $\mathbf{q} \in \mathbb{Q}$ as

$$\mathbf{q} = \cos\frac{\theta}{2} + (\alpha_1\mathbf{i} + \alpha_2\mathbf{j} + \alpha_3\mathbf{k}) \sin\frac{\theta}{2}.$$

The unity of a quaternion can be expressed using the quaternion conjugate $\mathbf{q}^* = \left(q_1, -\bar{\mathbf{q}}^\top\right)^\top$ as $\mathbf{q}^* * \mathbf{q} = 1$ or using the inner product as $\mathbf{q}^\top\mathbf{q} = 1$. Transformation $\mathtt{X}$ can be parametrized using the unit quaternion $\mathbf{q} = (q_1, q_2, q_3, q_4)$ as

$$\mathtt{X}(\mathbf{q}, \mathbf{t}) = \begin{pmatrix} \mathtt{R}(\mathbf{q}) & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix},$$

where

$$\mathtt{R}(\mathbf{q}) = \begin{pmatrix} q_1^2+q_2^2-q_3^2-2q_4^2 & 2q_2q_3-2q_4q_1 & 2q_2q_4+2q_3q_1 \\ 2q_2q_3+2q_4q_1 & q_1^2-q_2^2+q_3^2-q_4^2 & 2q_3q_4-2q_2q_1 \\ 2q_2q_4-2q_3q_1 & 2q_3q_4+2q_2q_1 & q_1^2-q_2^2-q_3^2+2q_4^2 \end{pmatrix}$$

is the quaternion parametrization of a rotation matrix. This leads to the following parametrization of the hand-eye calibration problem:

Problem 11.2 (*qhec* method).

$$\begin{aligned} \text{minimize} \quad & f_2(\mathbf{q}_\mathtt{X}, \mathbf{t}_\mathtt{X}) = \\ & \textstyle\sum_{i=1}^{n} \left\|\mathtt{A}_i\mathtt{X}(\mathbf{q}_\mathtt{X}, \mathbf{t}_\mathtt{X}) - \mathtt{X}(\mathbf{q}_\mathtt{X}, \mathbf{t}_\mathtt{X})\mathtt{B}_i\right\|^2 \\ \text{subject to} \quad & \mathbf{q}_\mathtt{X}^\top\mathbf{q}_\mathtt{X} = 1, \\ & q_{\mathtt{X}1} \geq 0. \end{aligned}$$

The objective function f_2 of Problem 11.2 is a polynomial function of degree 4 and it is composed of 85 monomials in 7 variables. Since the unit quaternions are a double cover of $SO(3)$, function f_2 has at least two global optima $f_2(\mathbf{q}_\mathtt{X}^*, \mathbf{t}_\mathtt{X}^*) = f_2(-\mathbf{q}_\mathtt{X}^*, \mathbf{t}_\mathtt{X}^*)$. To help the SDP solver, we add the semialgebraic constraint $q_{\mathtt{X}1} \geq 0$ to Problem 11.2 to eliminate one of the global optima in the majority of cases, that is when $q_{\mathtt{X}1}^* \neq 0$. The algebraic constraint $\mathbf{q}_\mathtt{X}^\top \mathbf{q}_\mathtt{X} = 1$ enforces the unity of the resulting quaternion.

11.2.3 Dual Quaternion parametrization

Dual quaternions are the algebraic counterparts of screws. They form a Clifford algebra; a dual quaternion $\hat{\mathbf{q}} \in \mathbb{H}$ can be represented in the form

$$\hat{\mathbf{q}} = \mathbf{q} + \epsilon \mathbf{q}',$$

where $\mathbf{q}, \mathbf{q}' \in \mathbb{Q}$ and ϵ is the dual unit, $\epsilon\epsilon = 0$, that commutes with every element of the algebra. It is also convenient to write dual quaternions as vectors $(\mathbf{q}^\top, \mathbf{q}'^\top)^\top$, since the set of dual quaternions is equal to $\mathbb{R}^8$. Addition of two dual quaternions $\hat{\mathbf{p}}, \hat{\mathbf{q}} \in \mathbb{H}$, $\hat{\mathbf{p}} = \left(\mathbf{p}^\top, \mathbf{p}'^\top\right)^\top$, $\hat{\mathbf{q}} = \left(\mathbf{q}^\top, \mathbf{q}'^\top\right)^\top$ is equivalent to addition in $\mathbb{R}^8$. Multiplication can be expressed using quaternion multiplication as

$$\hat{\mathbf{p}} \otimes \hat{\mathbf{q}} = ((\mathbf{p} * \mathbf{q})^\top, (\mathbf{p} * \mathbf{q}' + \mathbf{p}' * \mathbf{q})^\top)^\top.$$

Similar to the way rotations in $\mathbb{R}^3$ can be represented by the quaternions of unit length, rigid motions in $\mathbb{R}^3$ can be represented by unit dual quaternions [122]: rotation represented by a quaternion $\mathbf{p} \in \mathbb{Q}$ followed by translation $\mathbf{t} \in \mathbb{R}^3$ is represented by the dual quaternion

$$\hat{\mathbf{q}}(\mathbf{p}, \mathbf{t}) = (\mathbf{p}^\top, ((0, \tfrac{1}{2}\mathbf{t}^\top)^\top * \mathbf{p})^\top)^\top.$$

Unity of a dual quaternion $\hat{\mathbf{q}}$ can be expressed using its conjugate $\hat{\mathbf{q}}^* = (\mathbf{q}^{*\top}, \mathbf{q}'^{*\top})^\top$ as $\hat{\mathbf{q}}^* \otimes \hat{\mathbf{q}} = 1$ or using the quaternion parts as

$$\mathbf{q}^\top \mathbf{q} = 1 \quad \text{and} \quad q_1 q_5 + q_2 q_6 + q_3 q_7 + q_4 q_8 = 0.$$

Let $\hat{\mathbf{a}}_i$, $\hat{\mathbf{b}}_i$ be the dual quaternion representation of transformations $\mathtt{A}_i$, $\mathtt{B}_i$, respectively. Since Equation 11.2 can be expressed using dual quaternions multiplications as

$$\hat{\mathbf{a}}_i \otimes \hat{\mathbf{q}}_\mathtt{X} = \hat{\mathbf{q}}_\mathtt{X} \otimes \hat{\mathbf{b}}_i, \tag{11.6}$$

where $\hat{\mathbf{q}}_\mathtt{X}$ is the dual quaternion representation of the transformation $\mathtt{X}$, we can recover $\mathtt{X}$ using the following formulation:

Problem 11.3 (*dqhec* method).

$$\begin{aligned} \text{minimize} \quad & f_3(\hat{\mathbf{q}}_\mathtt{X}) = \textstyle\sum_{i=1}^{n} \left\| \hat{\mathbf{a}}_i \otimes \hat{\mathbf{q}}_\mathtt{X} - \hat{\mathbf{q}}_\mathtt{X} \otimes \hat{\mathbf{b}}_i \right\|^2 \\ \text{subject to} \quad & \mathbf{q}_\mathtt{X}^\top \mathbf{q}_\mathtt{X} = 1, \\ & q_{\mathtt{X}1} q_{\mathtt{X}5} + q_{\mathtt{X}2} q_{\mathtt{X}6} + q_{\mathtt{X}3} q_{\mathtt{X}7} + q_{\mathtt{X}4} q_{\mathtt{X}8} = 0. \\ & q_{\mathtt{X}1} \geq 0. \end{aligned}$$

The objective function f_3 of Problem 11.3 is a polynomial function of degree 4 and it is composed of 177 monomials in 8 variables. Again, additional constraint $q_{\mathtt{X}1} \geq 0$ is added to eliminate one of the two global optima in most of the cases. Since we are using dual quaternions directly, we have to take care of the orientation ambiguity of the rotational quaternions $\mathbf{a}_i, \mathbf{b}_i$ when converting matrices $\mathtt{A}_i, \mathtt{B}_i$ into dual quaternions $\hat{\mathbf{a}}_i, \hat{\mathbf{b}}_i$. This is due to the fact that even though quaternions $\mathbf{a}_i, -\mathbf{a}_i$ and $\mathbf{b}_i, -\mathbf{b}_i$ represent the same rotation, the sign matters when Equation 11.2 is expressed using dual quaternions, *i.e.*,

$$\hat{\mathbf{a}}_i(\mathbf{a}_i, \mathbf{t}_{\mathtt{A}_i}) \otimes \hat{\mathbf{q}}_{\mathtt{X}} - \hat{\mathbf{q}}_{\mathtt{X}} \otimes \hat{\mathbf{b}}_i(\mathbf{b}_i, \mathbf{t}_{\mathtt{B}_i}) \neq \hat{\mathbf{a}}_i(-\mathbf{a}_i, \mathbf{t}_{\mathtt{A}_i}) \otimes \hat{\mathbf{q}}_{\mathtt{X}} - \hat{\mathbf{q}}_{\mathtt{X}} \otimes \hat{\mathbf{b}}_i(\mathbf{b}_i, \mathbf{t}_{\mathtt{B}_i}).$$

To check the "compatibility" of the quaternions $\mathbf{a}_i$ and $\mathbf{b}_i$, the *screw congruence theorem* [33] can be used. It provides a necessary condition for the solution of Equation 11.6:

$$\hat{\mathbf{a}}_i \otimes \hat{\mathbf{q}}_{\mathtt{X}} = \hat{\mathbf{q}}_{\mathtt{X}} \otimes \hat{\mathbf{b}}_i \Rightarrow (a_1, a_1')^\top = (b_1, b_1')^\top.$$

Dual quaternions were first applied to the hand-eye calibration problem by Daniilidis and Bayro-Corrochano in [43, 44]. Their method, however, does not make use of all of the information in the camera and robot motions. Since the method assumes that $(a_1, a_1')^\top = (b_1, b_1')^\top$, it only uses the imaginary parts $\bar{\mathbf{a}}_i$, $\bar{\mathbf{a}}_i'$, $\bar{\mathbf{b}}_i$, $\bar{\mathbf{b}}_i'$. In [116], Malti and Barreto proposed a method based on dual quaternions that uses also the real components, however their solution does not solve for rotation and translation simultaneously.

11.3 Simultaneous Hand-Eye and Robot-World Calibration

Due to the apparent similarity of Equations 11.2 and 11.5, analogies to Problems 11.1, 11.2, and 11.3 can be formulated for the simultaneous hand-eye and robot-world calibration problem.

11.3.1 Orthonormal parametrization

Problem 11.4 is based on orthonormal parametrization of Equation 11.5. The objective function f_4 is a polynomial function of degree 4 and it is composed of 280 monomials in 18 variables.

Problem 11.4 (*uvherwc* method).

$$\begin{aligned} \text{minimize} \quad & f_4(\mathbf{u}_{\mathtt{X}}, \mathbf{v}_{\mathtt{X}}, \mathbf{t}_{\mathtt{X}}, \mathbf{u}_{\mathtt{Z}}, \mathbf{v}_{\mathtt{Z}}, \mathbf{t}_{\mathtt{Z}}) = \\ & \textstyle\sum_{i=1}^{m} \left\| \mathtt{A}_i'^{-1} \mathtt{X}(\mathbf{u}_{\mathtt{X}}, \mathbf{v}_{\mathtt{X}}, \mathbf{t}_{\mathtt{X}}) - \mathtt{Z}(\mathbf{u}_{\mathtt{Z}}, \mathbf{v}_{\mathtt{Z}}, \mathbf{t}_{\mathtt{Z}}) \mathtt{B}_i' \right\|^2 \\ \text{subject to} \quad & \mathbf{u}_{\mathtt{X}}^\top \mathbf{u}_{\mathtt{X}} = 1, \mathbf{v}_{\mathtt{X}}^\top \mathbf{v}_{\mathtt{X}} = 1, \mathbf{u}_{\mathtt{X}}^\top \mathbf{v}_{\mathtt{X}} = 0, \\ & \mathbf{u}_{\mathtt{Z}}^\top \mathbf{u}_{\mathtt{Z}} = 1, \mathbf{v}_{\mathtt{Z}}^\top \mathbf{v}_{\mathtt{Z}} = 1, \mathbf{u}_{\mathtt{Z}}^\top \mathbf{v}_{\mathtt{Z}} = 0. \end{aligned}$$

11.3.2 Quaternion parametrization

Problem 11.5 is based on quaternion parametrization of Equation 11.5. The objective function f_5 is a polynomial function of degree 4 and is composed of 209 monomials in 14 variables.

Problem 11.5 (*qherwc* method).

$$\begin{aligned} \text{minimize} \quad & f_5(\mathbf{q}_\mathtt{X}, \mathbf{t}_\mathtt{X}, \mathbf{q}_\mathtt{Z}, \mathbf{t}_\mathtt{Z}) = \\ & \quad \sum_{i=1}^{m} \left\| \mathtt{A}_i'^{-1} \mathtt{X}(\mathbf{q}_\mathtt{X}, \mathbf{t}_\mathtt{X}) - \mathtt{Z}(\mathbf{q}_\mathtt{Z}, \mathbf{t}_\mathtt{Z}) \mathtt{B}_i' \right\|^2 \\ \text{subject to} \quad & \mathbf{q}_\mathtt{X}^\top \mathbf{q}_\mathtt{X} = 1,\ q_{\mathtt{X}1} \geq 0, \\ & \mathbf{q}_\mathtt{Z}^\top \mathbf{q}_\mathtt{Z} = 1,\ q_{\mathtt{Z}1} \geq 0. \end{aligned}$$

11.3.3 Dual Quaternion parametrization

Problem 11.6 is based on quaternion parametrization of Equation 11.5. Note that $\hat{\mathbf{a}}_i'$ is the dual quaternion representation of transformation $\mathtt{A}_i'^{-1}$. The objective function f_6 is a polynomial function of degree 2 and it is composed of 112 monomials in 16 variables.

Problem 11.6 (*dqherwc* method).

$$\begin{aligned} \text{minimize} \quad & f_6(\hat{\mathbf{q}}_\mathtt{X}) = \sum_{i=1}^{m} \left\| \hat{\mathbf{a}}_i' \otimes \hat{\mathbf{q}}_\mathtt{X} - \hat{\mathbf{q}}_\mathtt{Z} \otimes \hat{\mathbf{b}}_i' \right\|^2 \\ \text{subject to} \quad & \mathbf{q}_\mathtt{X}^\top \mathbf{q}_\mathtt{X} = 1,\ q_{\mathtt{X}1} \geq 0, \mathbf{q}_\mathtt{Z}^\top \mathbf{q}_\mathtt{Z} = 1,\ q_{\mathtt{Z}1} \geq 0, \\ & q_{\mathtt{X}1} q_{\mathtt{X}5} + q_{\mathtt{X}2} q_{\mathtt{X}6} + q_{\mathtt{X}3} q_{\mathtt{X}7} + q_{\mathtt{X}4} q_{\mathtt{X}8} = 0, \\ & q_{\mathtt{Z}1} q_{\mathtt{Z}5} + q_{\mathtt{Z}2} q_{\mathtt{Z}6} + q_{\mathtt{Z}3} q_{\mathtt{Z}7} + q_{\mathtt{Z}4} q_{\mathtt{Z}8} = 0. \end{aligned}$$

As is the case of Problem 11.3, we need to take care of the orientation ambiguity of the rotational quaternions $\mathbf{a}_i, \mathbf{b}_i$. This time however, the screw congruence theorem does not apply. The obvious solution it to try all of the 2^m sign combinations and keep the combination with the smallest value of $f_6(\hat{\mathbf{q}}_\mathtt{X}^*)$. Running the SDP optimization 2^m times can be quite computationally expensive, so in our experiments we use the dual quaternion method of Li [7]—which needs to be run 2^m times in practice as well, since it suffers from the same quaternion ambiguity problem—to recover the sign combination before running the SDP solver, because it runs faster. The sign problem is also inherent to the method of Dornaika [46].

11.4 Experiments

In this section we present both synthetic and real data experiments to validate the proposed calibration methods. For the real data experiment we used a Motoman MA1400 serial 6-DOF manipulator with an Asus Xtion Pro sensor rigidly attached to it. The Xtion Pro sensor was equipped with a camera with the resolution of 640×480 pixels. We simulated the same setup in the synthetic experiment in order to better judge the result of the experiment with real data.

11.4.1 Implementation

We implemented all 6 calibration methods in MATLAB using GloptiPoly [79], an interface that automatically constructs LMI relaxations of polynomial problems and converts them into a format understandable by the SDP solver SeDuMi [168]. Further, GloptiPoly can also recover the solution to the original polynomial problem and certify its optimality. Using YALMIP toolbox [106], the SeDuMi format can be further converted into the input formats of several other SDP solvers. In our experiments, we used the LMI relaxations of the second order and SeDuMi 1.3 and MOSEK [1] 7.0 as SDP solvers. GloptiPoly certified that global optimum has been reached by all the methods for all the problem instances encountered during the experiments. The following table sums up the relevant information on the proposed methods:

Method	Variables	Degree	Monomials	Moments	Time[1](s)	Time[2](s)
uvhec	9	4	124	715	1.45	0.45
qhec	7	4	85	330	0.48	0.29
dqhec	8	4	177	495	0.99	0.46
uvherwc	18	4	280	7315	953.76	60.89
qherwc	14	4	209	3060	68.38	7.91
dqherwc	16	2	112	4845	309.25	16.85

[1]SeDuMi,[2]MOSEK

To obtain the timings, we used a 3.5GHz Intel Core i7 based desktop computer running 64-bit Linux. The presented times include time spent in the SDP solver as well as GloptiPoly and YALMIP overheads. To gauge the sizes of the SDP problems involved, the column "Moments" specifies the number of moments (dimension of vector $\mathbf{y}$ in Problem **??**) in the second order LMI relaxation for the respective method. Note, that even though the objective functions f_i depend on n or m, *i.e.*, on the number of relative motions or poses, the number of monomials does not. This means that the sizes of the LMI relaxations and the sizes of the resulting SDP problems also do not depend on n or m and are constant.

As a pre-step to all of the methods, we scale the translations $\mathbf{t}_{\mathtt{A}_i}, \mathbf{t}_{\mathtt{B}_i}$ by the factor of $\alpha = \max_i \{\|\mathbf{t}_{\mathtt{A}_i}\|, \|\mathbf{t}_{\mathtt{B}_i}\|\}$, so that the length of the longest translation is 1. This helps with the convergence of the SDP solver and removes the influence of the chosen physical units on the accuracy of the result. In cases where $\mathbf{t}_{\mathtt{X}}$ and $\mathbf{t}_{\mathtt{Z}}$ are explicitly optimized, we add one or two more constraints $\mathbf{t}_{\mathtt{X}}^\top \mathbf{t}_{\mathtt{X}} \leq 2$, $\mathbf{t}_{\mathtt{Z}}^\top \mathbf{t}_{\mathtt{Z}} \leq 10$. These constraints result from the way our experiments were constructed and are not technically necessary, however, they help the SDP solvers to further speed-up the converge. In the experiments, we set the SeDuMi parameter $\mathtt{eps} = 10^{-20}$ and the MOSEK parameters $\mathtt{MSK\_DPAR\_INTPNT\_CO\_TOL\_\{P|D\}FEAS} = 10^{-20}$.

11.4.2 Synthetic Experiments

In the synthetic experiments, we investigated the influence of both image and joint noises on the calibration accuracy. We placed a virtual planar calibration target consisting of a grid

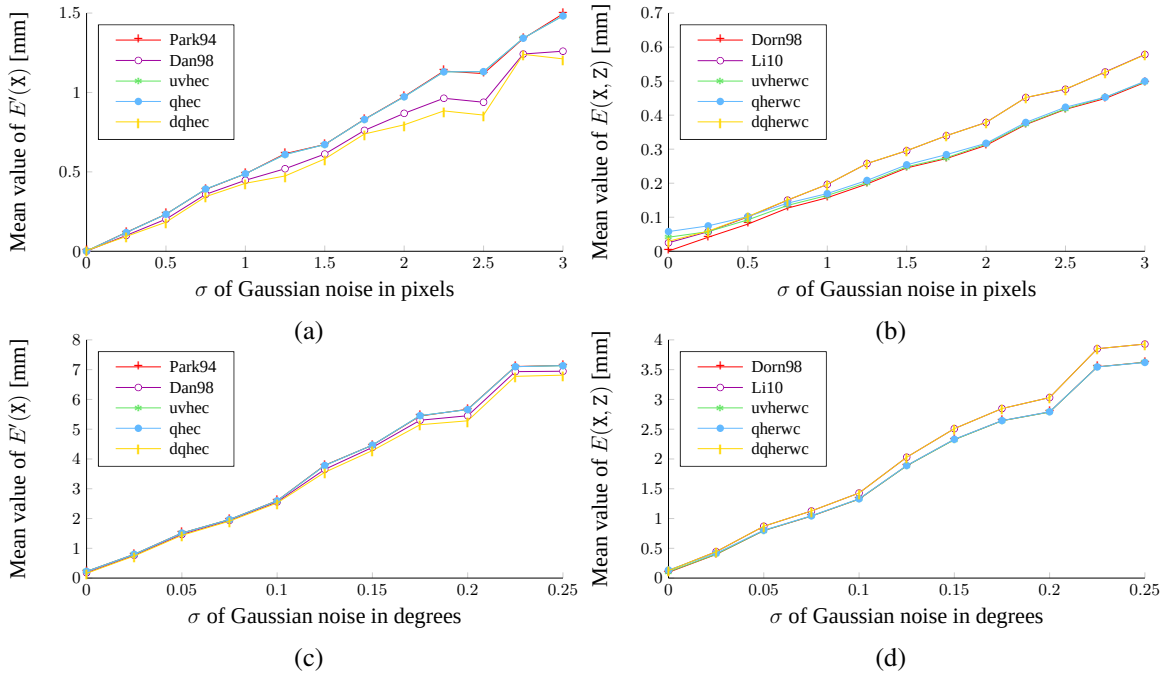
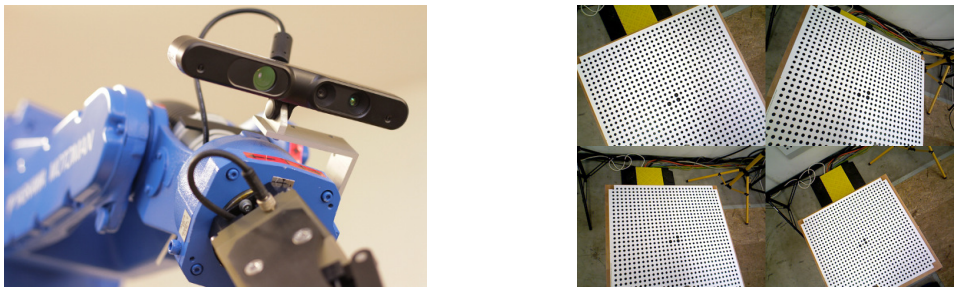


Figure 11.2: *Synthetic data experiment results*. (a–b) Image noise experiment. (c–d) Joint noise experiment.

of 16×16 known points in front of a simulated MA1400 serial manipulator. The distance between the points was set to 12.5 mm in each direction, making the calibration target 200×200 mm in size.

To create a calibration task, we started with 9 camera poses $\mathtt{A}'_i$ randomly generated onto half-sphere of radius of ~ 30 mm oriented as to face approximately the center of the calibration target. To simulate the typical situation when the camera faces approximately the same direction as the robot's end effector, a hand-eye rotation $\mathtt{R}_\mathtt{X}$ was randomly generated so that the rotation corresponded to identity to a up to 5° degrees difference in each axis. The translation $\mathbf{t}_\mathtt{X}$ was randomly generated to move the camera up to ~ 200 mm away from end effector. The arm poses $\mathtt{B}'_i$ were computed using the camera poses and the hand-eye transformation as $\mathtt{X}^{-1}\mathtt{A}'_i$. Finally, we generated a robot-world transformation $\mathtt{Z}$ using a random rotation and a random translation up to ~ 2000 mm in length and used $\mathtt{Z}$ to transform the grid points from the robot into the world coordinate system. We generated 10 sets of 9 camera positions, 10 hand-eye transformations and 1 robot-world transformation and combined them into 100 calibration tasks. For every calibration task, we also computed all possible relative movements $\mathtt{A}_i$, $\mathtt{B}_i$, $i = 1, \ldots, 36$ to be used by the hand-eye calibration methods. In order to better judge the accuracy the proposed methods, we implemented several hand-eye and robot-world calibration methods to compare them against: *Park94* [134] and *Dan98* [44], as representatives of hand-eye calibration methods, and *Dorn98* [46] and *Li10*, the dual quaternion variant of [7], as representatives of simultaneous hand-eye and robot-world calibration methods.



(a) (b)

Figure 11.3: *Real data experiment setup*. (a) Detail of the Xtion Pro sensor and the robot's end effector. (b) Examples of input images.

Image Noise Experiment

To simulate the influence of image noise, we projected the calibration points into 640×480 pixel images using the generated camera poses $\mathtt{A}'_i$ and corrupted the image projections $\mathbf{u}_{ij}$ in each calibration task with Gaussian noise in 13 noise levels, $\sigma \in \langle 0, 3 \rangle$ px in $1/4$ px steps. Finally, we recovered noised camera poses $\mathtt{A}'_i$ using EPnP algorithm [101].

In order to evaluate the performance of the hand-eye and robot-world calibration methods *uvherwc*, *qherwc*, and *dqherwc*, we have sampled the virtual workspace of the robot—a cube of size $70 \times 70 \times 70$ cm around the calibration target—with uniformly distributed $\ell = 9240$ points $\mathbf{Y}_j$. Using these points, measured in the robot coordinate frame, we defined a calibration error measure $E(\mathtt{X}, \mathtt{Z})$ that expresses the error of the calibration as the difference of the positions the workspace points transformed into the camera coordinate frame using the computed transformations $\mathtt{X}$, $\mathtt{Z}$ and the ground truth transformations $\mathtt{X}^{\mathrm{gt}}$, $\mathtt{Z}^{\mathrm{gt}}$. Formally,

$$E(\mathtt{X}, \mathtt{Z}) = \frac{1}{9\ell} \sum_{i=1}^{9} \sum_{j=1}^{\ell} \left\| \mathtt{X}\mathtt{B}_i'^{-1}\mathtt{Z}^{-1}\mathtt{Z}^{\mathrm{gt}}\mathbf{Y}_j - \mathtt{X}^{\mathrm{gt}}\mathtt{B}_i'^{-1}\mathbf{Y}_j \right\| .$$

To evaluate the performance of the hand-eye calibration methods *uvhec*, *qhec*, and *dqhec*, we do not have to transform the points $\mathbf{Y}_j$ into the world coordinate frame first, so we defined a slightly simpler 3D error function

$$E'(\mathtt{X}) = \frac{1}{36\ell} \sum_{i=1}^{36} \sum_{j=1}^{\ell} \left\| \mathtt{X}\mathtt{B}_i'^{-1}\mathbf{Y}_j - \mathtt{X}^{\mathrm{gt}}\mathtt{B}_i'^{-1}\mathbf{Y}_j \right\| .$$

Figure 11.2(a) shows the mean of the values $E'(\mathtt{X})$ for all 100 calibration tasks for different methods and image noise levels. It shows the best performance by the method *dqhec* followed by *Dan98* and finally by *Park94*, *uvhec*, and *qhec*, all with the same performance level. In this experiment, we used SeDuMi as the SDP solver.

Figure 11.2(b) shows the same statistics for the values of $E(\mathtt{X}, \mathtt{Z})$ and suggest quite comparable performance of methods *Dorn98*, *uvhewrc*, and *qhewbc*. This time, the dual quaternion formulations *Li10* and *dqherwc* performed worse that the other methods. In this experiment, we used MOSEK as the SDP solver. This resulted in much faster convergence times, but also in slightly worse accuracy for the lower noise levels.

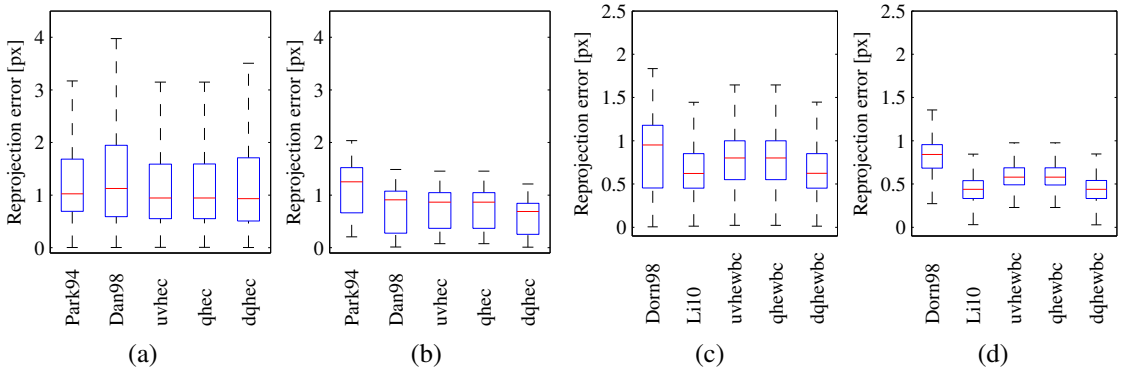


Figure 11.4: *Real data experiment resuts*. (a) Reprojection errors e'_{ij} of the calibration image set (b) Reprojection errors e'_{ij} of the validation image set. (c) Reprojection errors e_{ij} of the calibration image set (d) Reprojection errors e_{ij} of the validation image set. (the red line marks the median, the edges of the box are the 25th and 75th percentiles)

Joint Noise Experiment

In this experiment, we simulated the performance in the presence of joint noise. We started with the same 100 calibration tasks as in the case of the image noise experiment. Further, we recovered the joint coordinates [150] of the virtual MA1400 manipulator for every pose $\mathtt{B}'_i$ with respect to the Denavit–Hartenberg convention. Next, we corrupted the joint coordinates by random offsets—we used the same offsets for the joint coordinates of the same task—in 11 noise levels, $\sigma \in \langle 0, 0.25 \rangle$ deg in steps of 0.025 degrees. Finally, we recovered noised poses $\mathtt{B}'_i$ using the forward kinematics. To further simulate the real world conditions, we corrupted projections $\mathbf{u}_{ij}$ by image noise of $\sigma = 0.5$ px for every joint noise level. We evaluated the performance using the same error measures $E(\mathtt{X}, \mathtt{Z})$ and $E'(\mathtt{X})$.

Figures 11.2(c) and 11.2(d) show the mean of the values $E'(\mathtt{X})$ and $E'(\mathtt{X}, \mathtt{Z})$. Again, the method *qhec* outperforms its competitors, whereas the dual quaternion formulations lose on the rest in the simultaneous hand-eye and robot-world calibration experiment.

11.4.3 Real Data Experiment

For this experiment, we used a real MA1400 serial manipulator with a Xtion Pro sensor attached to its 5th link, see Figure 11.3(a). We manipulated the robotic arm into 10 poses $\mathtt{B}'_i$ and acquired a calibration image set consisting of the same number of images of a 30×30 points calibration grid. The grid was placed ~ 1 m in front of and ~ 0.5 m above the base of the robotic arm. The point spacing of the calibration grid was 24 mm. We also acquired a validation image set consisting of 3 images. See Figure 11.3(b) for example images from the calibration sequences. Next, we used OpenCV [2] library to obtain the internal calibration matrix $\mathtt{K}$ [64] of the sensor as well as the camera poses $\mathtt{A}'_i$, $i = 1, \ldots, 13$.

Since there was no ground truth information available, we had to use a performance measure different from $E(\mathtt{X}, \mathtt{Z})$ and $E'(\mathtt{X})$ used in the synthetic data experiments. Suppose, that a function $\mathcal{P}(\mathbf{Y}, \mathtt{A}'_i, \mathtt{K})$ projects calibration grid points $\mathbf{Y}_j \in \mathbb{R}^3$, $j = 1, \ldots, 900$ in the

world coordinate frame into the i-th image taken by a camera described by its pose $\mathtt{A}'_i$ and internal camera calibration matrix $\mathtt{K}$. The reprojection error of the point $\mathbf{Y}_j$ is defined as $e_{ij}(\mathbf{u}_i, \mathbf{Y}_j, \mathtt{A}'_i, \mathtt{K}) = \|\mathbf{u}_{ij} - \mathcal{P}(\mathbf{Y}_j, \mathtt{A}'_i, \mathtt{K})\|$, where $\mathbf{u}_{ij}$ are the pixel coordinates of the point $\mathbf{Y}_j$ in the i-th image. In case a calibration method recovers both $\mathtt{X}$ and $\mathtt{Z}$, we can judge the quality of the calibration by expressing the reprojection error as $e_{ij}(\mathbf{u}_i, \mathbf{Y}_j, \mathtt{Z}\mathtt{B}_i\mathtt{X}^{-1}, \mathtt{K})$. However, in case only transformation $\mathtt{X}$ is recovered, the camera pose $\mathtt{A}'_i$ can be expressed only with the help of an additional pose $\mathtt{A}'_i = \mathtt{A}'_k\mathtt{X}\mathtt{B}'^{-1}_k\mathtt{B}'_i\mathtt{X}^{-1}$ and we have to define modified reprojection error $e'_{ij} = \frac{1}{m-1}\sum_{k=1,k\neq i}^{m} e_{ij}(\mathbf{u}_{ij}, \mathbf{Y}_j, \mathtt{A}'_k\mathtt{X}\mathtt{B}'^{-1}_k\mathtt{B}'_i\mathtt{X}^{-1}, \mathtt{K})$.

Figure 11.4(a) shows the statistics of the modified reprojection errors e'_{ij} computed from the calibration image set created by the MATLAB function `boxplot` for methods *Park94*, *Dan99*, *uvhec*, *qhec*, and *dqhec*. Figure 11.4(b) shows the same statistics, this time for the validation image set. As is the case of synthetics data experiments, *dqhec* slightly outperforms its competitors. Figures 11.4(c,d) show the reprojection errors e_{ij} computed for the methods *Dorn98*, *Li10*, *uvherwc*, *qherwc*, and *dqherwc* for the calibration and validation image sets, respectively.

12 Conclusions

They've done studies, you know. 60% of the time, it works every time.

– Brian Fantana

In this thesis, the hand-eye and robot-world calibration problems were thoroughly studied. This work contributed to the state of the art by several algorithms, mainly by the branch-and-bound method for hand-eye calibration. This method is the first that does not require camera poses as the input. In the following, novel contributions of each chapter are briefly summarized.

Chapter 7 presented a SfM-based hand-eye calibration method formulated as an L_∞-norm optimization problem. This formulation recovers the hand-eye displacement with the correct scale using image correspondences and robot measurements only. In addition, optimality of the resulting displacement with respect to the reprojection error is guaranteed. The performance of this novel SfM based hand-eye calibration method was successfully validated by both synthetic and real data experiments.

In Chapter 8, we removed the requirement for known camera extrinsics from the hand-eye calibration problem. Since the algorithm is completely independent on the scene geometry and scale, there is no need for a known calibration device and the calibration can be performed solely from a general scene. Not only this makes the method immune to the errors introduced by calibration device manufacturing process but also increases the calibration space to virtually arbitrary size. The theory shows that the algorithm is guaranteed to be globally optimal with respect to L_∞-norm. Further, the experiments show that the algorithm is also practically useful, since it is highly parallelizable and competitive in situation where a calibration device is lacking in precision or is impractical due to the calibration space requirements.

In Chapter 9, we presented the first minimal problem of hand-eye calibration for the situations where the gripper's rotations are not known. We formulated the problem as a system of seven equations in seven unknowns and solved it using the Gröbner basis method for solving systems of polynomial equations providing the first exact algebraic solution to the problem. This solution uses the minimal number of two relative movements. Further, we showed how to select the geometrically correct solution using additional relative movements. Finally, our experiments showed that the proposed solver is numerically stable, fast and—since it can handle noisy inputs—that its results can be successfully used as initialization of subsequent minimization methods.

Chapter 10 presented a novel solution to the robot-world calibration problem. We formulated the world-base calibration problem as a problem of multivariate polynomial minimization and used the method of LMI relaxations to solve it. The experiments showed that LMI relaxation of the second order is enough to lead to certifiably globally optimal solutions

for all calibrations task studied. The experiments also showed that the presented method outperforms the traditional approaches and that it is fast and well-behaved in the presence of both image and joint noise.

Finally in Chapter 11, we showed that the method of convex LMI relaxations can be naturally applied to the hand-eye and robot-world calibration problems. We presented three hand-eye and three hand-eye and robot-world calibration parametrizations and by applying the method of convex LMI relaxations we obtained globally optimal solutions. These formulations provide a new insight into the behavior and complexity of the original problem. The *qhec* parametrization showed the best performance overall. Methods *uvherwc* and *qherwc* do not necessarily provide more accurate results than the previously proposed methods, however, since they do not suffer from the quaternion sign ambiguity, the running time is constant and not exponential in the number of poses.

Bibliography

[1] Mosek: large scale optimization software. www.mosek.com.

[2] Open source computer vision library. http://opencv.org/.

[3] VisualSFM : A visual structure from motion system. http://homes.cs.washington.edu/~ccwu/vsfm/, 2013.

[4] GNU Linear Programming Kit Version 4.47. http://www.gnu.org/software/glpk/glpk.html, Sep. 2011.

[5] Sameer Agarwal, Noah Snavely, Steven M Seitz, and Richard Szeliski. Bundle adjustment in the large. In *European Conference on Computer Vision*, pages 29–42. Springer, 2010.

[6] Motilal Agrawal and Larry S Davis. Camera calibration using spheres: A semi-definite programming approach. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 782–789. IEEE, 2003.

[7] Lin Wang Aiguo Li and Defeng Wu. Simultaneous robot-world and hand-eye calibration using dual-quaternions and kronecker product. *International Journal of the Physical Sciences*, Vol. 5(10):pp. 1530–1536, September 2010.

[8] Nicolas Andreff, Radu Horaud, and Bernard Espiau. On-line hand-eye calibration. In *3-D Digital Imaging and Modeling, 1999. Proceedings. Second International Conference on*, pages 430–436. IEEE, 1999.

[9] Nicolas Andreff, Radu Horaud, and Bernard Espiau. Robot Hand-Eye Calibration using Structure from Motion. *International Journal of Robotics Research*, 20(3):228–248, 2001.

[10] Simon Baker and Shree K Nayar. A theory of single-viewpoint catadioptric image formation. *International Journal of Computer Vision*, 35(2):175–196, 1999.

[11] Hynek Bakstein and Tomas Pajdla. Panoramic mosaicing with a 180 field of view lens. In *Omnidirectional Vision, 2002. Proceedings. Third Workshop on*, pages 60–67. IEEE, 2002.

[12] Joao Barreto and Kostas Daniilidis. Wide area multiple camera calibration and estimation of radial distortion. In *Omnivis-2004, ECCV-2004 workshop*, 2004.

[13] Yalin Bastanlar, Luis Puig, Peter Sturm, Josechu Guerrero, Joao Barreto, et al. Dlt-like calibration of central catadioptric cameras. In *The 8th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras-OMNIVIS*, 2008.

[14] Paul Beardsley, Phil Torr, and Andrew Zisserman. 3d model acquisition from extended image sequences. In *European Conference on Computer Vision*, pages 683–695. Springer, 1996.

[15] Paul A Beardsley, Andrew Zisserman, and David W Murray. Sequential updating of projective and affine structure from motion. *International Journal of Computer Vision*, 23(3):235–259, 1997.

[16] Christoph Bergmeir, Mathias Seitel, Christian Frank, Raffaele De Simone, H-P Meinzer, and Ivo Wolf. Comparing calibration approaches for 3d ultrasound probes. *International journal of computer assisted radiology and surgery*, 4(2):203–213, 2009.

[17] Emad Boctor, Anand Viswanathan, Michael Choti, Russell H Taylor, Gabor Fichtinger, and Gregory Hager. A novel closed form solution for ultrasound calibration. In *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on*, pages 527–530. IEEE, 2004.

[18] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.

[19] Eric Brassart, Laurent Delahoche, Cyril Cauchois, Cyril Drocourt, Claude Pegard, and El Mustapha Mouaddib. Experimental results got with the omnidirectional vision sensor: Syclop. In *Omnidirectional Vision, 2000. Proceedings. IEEE Workshop on*, pages 145–152. IEEE, 2000.

[20] D.C. Brown. Close-range camera calibration. *Photogrammetric engineering*, 37(8):855–866, 1971.

[21] Duane C Brown. Decentering distortion of lenses. *Photometric Engineering*, 32(3):444–462, 1966.

[22] M. Brown, R.I. Hartley, and D. Nister. Minimal solutions for panoramic stitching. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1 –8, 17-22 2007.

[23] Alfred M Bruckstein and Thomas J Richardson. Omniview cameras with curved surface mirrors. In *Omnidirectional Vision, 2000. Proceedings. IEEE Workshop on*, pages 79–84. IEEE, 2000.

[24] M. Bujnak, Z. Kukelova, and T. Pajdla. A general solution to the p4p problem for camera with unknown focal length. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1 –8, 23-28 2008.

[25] Martin Bujnak, Zuzana Kukelova, and Tomas Pajdla. New efficient solution to the absolute pose problem for camera with unknown focal length and radial distortion. In *Computer Vision–ACCV 2010*, pages 11–24. Springer, 2011.

[26] Bruno Burcherger. Ein algorithmus zum auffinden der basiselemente des restklassenrings nach einem nulldimensionalen polynomideal. *Universitat Innsbruck, Austria, Ph. D. Thesis*, 1965.

[27] M. Byrod, M. Brown, and K. Astrom. Minimal solutions for panoramic stitching with radial distortion. pages xx–yy, 2009.

[28] M. Byrod, K. Josephson, and K. Astrom. Improving numerical accuracy of groebner basis polynomial equation solvers. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1 –8, 14-21 2007.

[29] Martin Byröd. Bundle adjustment using conjugate gradients with multiscale preconditioning. In *British Machine Vision Conference*, 2009.

[30] Martin Byröd and Kalle Åström. Conjugate gradient bundle adjustment. In *Computer Vision–ECCV 2010*, pages 114–127. Springer, 2010.

[31] Massimo Caboara, Martin Kreuzer, and Lorenzo Robbiano. Efficiently computing minimal sets of critical pairs. *Journal of Symbolic Computation*, 38(4):1169–1190, 2004.

[32] Michel Chasles. Note sur les propriétés générales du système de deux corps semblables entre eux, placés d'une manière quelconque dans l'espace; et sur le déplacement fini, ou infiniment petit d'un corps solide libre. *Bulletin des Sciences Mathématiques de Férussac*, XIV:321–326, 1831.

[33] Homer H Chen. A screw motion approach to uniqueness analysis of head-eye geometry. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pages 145–151. IEEE, 1991.

[34] Qian Chen, Haiyuan Wu, and Toshikazu Wada. Camera calibration with two arbitrary coplanar circles. In *Computer Vision-ECCV 2004*, pages 521–532. Springer, 2004.

[35] Yanqing Chen, Timothy A Davis, William W Hager, and Sivasankaran Rajamanickam. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software (TOMS)*, 35(3):22, 2008.

[36] Kyuhyoung Choi, Subin Lee, and Yongduek Seo. A branch-and-bound algorithm for globally optimal camera pose and focal length. *Image Vision Comput.*, 28(9):1369–1376, 2010.

[37] Jack C. K. Chou and M. Kamel. Finding the position and orientation of a sensor on a robot manipulator using quaternions. *International Journal of Robotics Research*, 10(3):240–254, 1991.

[38] T.A. Clarke and J.G. Fryer. The development of camera calibration methods and models. *The Photogrammetric Record*, 16(91):51–66, 1998.

[39] David Claus and Andrew W Fitzgibbon. A rational function lens distortion model for general cameras. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 213–219. IEEE, 2005.

[40] A.E. Conrady. Decentered lens-systems. *Monthly notices of the royal astronomical society*, 79:384–390, 1919.

[41] D.A. Cox, J.B. Little, and D. O'Shea. *Using Algebraic Geometry*. Graduate Texts in Mathematics. Springer, 2005.

[42] D.A. Cox, J.B. Little, and D. O'Shea. *Ideals, Varieties, And Algorithms: An Introduction to Computational Algebraic Geometry And Commutative Algebra*. Number v. 10 in Undergraduate Texts in Mathematics. Springer, 2007.

[43] K. Daniilidis and E. Bayro-Corrochano. The dual quaternion approach to hand-eye calibration. In *ICPR '96: Proceedings of the 1996 International Conference on Pattern Recognition (ICPR '96) Volume I*, page 318, Washington, DC, USA, 1996. IEEE Computer Society.

[44] Konstantinos Daniilidis. Hand-eye calibration using dual quaternions. *International Journal of Robotics Research*, 18:286–298, 1998.

[45] J. Denavit and R. S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *J. Appl. Mechanics, June 1955*, 22:215–221, 1955.

[46] Fadi Dornaika and Radu Horaud. Simultaneous robot-world and hand-eye calibration. *IEEE Transactions on Robotics and Automation*, 14:617–622, 1998.

[47] Mohamed Elkadi, Bernard Mourrain, et al. Some applications of bezoutians in effective algebraic geometry. 1998.

[48] Leonhard Euler. Formulae generales pro translatione quacunque corporum rigidorum. *Novi Commentarii academiae scientiarum imperialis Petropolitanae*, 20(E478):pp. 189–207, 1776.

[49] Wolfgang Faig. Calibration of close-range photogrammetric systems: mathematical formulation. *Photogrammetric engineering and remote sensing*, 1975.

[50] Olivier D Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In *European Conference on Computer Vision*, pages 563–578. Springer, 1992.

[51] Jean-Charles Faugère. A new efficient algorithm for computing gröbner bases. *Journal of pure and applied algebra*, 139(1):61–88, 1999.

[52] Jean Charles Faugère. A new efficient algorithm for computing gröbner bases without reduction to zero (f5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. ACM.

[53] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, June 1981.

[54] Andrew W Fitzgibbon and Andrew Zisserman. Automatic camera recovery for closed or open image sequences. In *European Conference on Computer VisionReconstruction from uncalibrated sequences with a hierarchy of trifocal tensors*, pages 311–326. Springer, 1998.

[55] Jan-Michael Frahm, Pierre Fite-Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, et al. Building rome on a cloudless day. In *Computer Vision–ECCV 2010*, pages 368–381. Springer, 2010.

[56] John G Fryer and Duane C Brown. Lens distortion for close-range photogrammetry. *Photogrammetric engineering and remote sensing*, 52(1):51–58, 1986.

[57] Rüdiger Gebauer and H Michael Möller. On an installation of buchberger's algorithm. *Journal of Symbolic Computation*, 6(2):275–286, 1988.

[58] Vladimir P Gerdt and Yuri A Blinkov. Involutive bases of polynomial ideals. *Mathematics and Computers in Simulation*, 45(5):519–541, 1998.

[59] Christopher Geyer and Kostas Daniilidis. A unifying theory for central panoramic systems and practical implications. In *Computer Vision ECCV 2000*, pages 445–461. Springer, 2000.

[60] Christopher Geyer and Kostas Daniilidis. Catadioptric projective geometry. *International Journal of Computer Vision*, 45(3):223–243, 2001.

[61] Christopher Geyer and Henrik Stewenius. A nine-point algorithm for estimating para-catadioptric fundamental matrices. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[62] Alessandro Giovini, Teo Mora, Gianfranco Niesi, Lorenzo Robbiano, and Carlo Traverso. "one sugar cube, please" or selection strategies in the buchberger algorithm. In *Proceedings of the 1991 international symposium on Symbolic and algebraic computation*, pages 49–54. ACM, 1991.

[63] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[64] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[65] R.I. Hartley. Minimizing algebraic error in geometric estimation problems. In *Computer Vision, 1998. Sixth International Conference on*, pages 469–476, 1998.

[66] R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2003.

[67] Richard Hartley. In defense of the eight-point algorithm. volume 19, pages 580 –593, jun 1997.

[68] Richard Hartley and Fredrik Kahl. Optimal algorithms in multiview geometry. In *ACCV'07: Proceedings of the 8th Asian conference on Computer vision*, pages 13–34, Berlin, Heidelberg, 2007. Springer-Verlag.

[69] Richard Hartley and Fredrik Kahl. Global optimization through rotation space search. *International Journal of Computer Vision*, 82(1):64–79, 2009.

[70] Richard Hartley and Peter Sturm. Triangulation. volume 68, pages 146–157, 1997.

[71] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge University, Cambridge, 2nd edition, 2003.

[72] Richard I Hartley. Estimation of relative camera positions for uncalibrated cameras. In *European Conference on Computer Vision*, pages 579–587. Springer, 1992.

[73] S Hayati and M Mirmirani. Improving the absolute positioning accuracy of robot manipulators. *Journal of Robotic Systems*, 2(4):397–413, 1985.

[74] Jan Heller, Michal Havlena, and Tomas Pajdla. A branch-and-bound algorithm for globally optimal hand-eye calibration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1608–1615. IEEE, June 2012.

[75] Jan Heller, Michal Havlena, Akihiro Sugimoto, and Tomas Pajdla. Structure-from-motion based hand-eye calibration using l_∞ minimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3497–3503. IEEE, June 2011.

[76] Jan Heller, Didier Henrion, and Tomas Pajdla. Hand-eye and robot-world calibration by global polynomial optimization. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[77] Jan Heller and Tomas Pajdla. World-base calibration by global polynomial optimization. In Lisa O'Conner, editor, *3D Vision (3DV), 2014 2nd International Conference on*, volume 1, pages 593–600, Piscataway, USA, Dec 2014. IEEE.

[78] Didier Henrion. Optimization on linear matrix inequalities for polynomial systems control. *arXiv:1309.3112*, 2013.

[79] Didier Henrion, Jean-Bernard Lasserre, and Johan Löfberg. Gloptipoly 3: moments, optimization and semidefinite programming. *Optimization Methods & Software*, 24(4-5):761–779, 2009.

[80] Robin Hill. A lens for whole sky photographs. *Quarterly Journal of the Royal Meteorological Society*, 50(211):227–235, 1924.

[81] Radu Horaud and Fadi Dornaika. Hand-eye calibration. *The International Journal of Robotics Research*, 14(3):195–210, 1995.

[82] Tomas Pajdla Jan Heller, Michal Havlena. Globally optimal hand-eye calibration using branch-and-bound. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, accepted for publication, August 2015.

[83] Guang Jiang, Hung-Tat Tsui, Long Quan, and Andrew Zisserman. Single axis geometry by fitting conics. In *Computer Vision ECCV 2002*, pages 537–550. Springer, 2002.

[84] K. Josephson and M. Byrod. Pose estimation with radial distortion and unknown focal length. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2419 –2426, 20-25 2009.

[85] F. Kahl and R. Hartley. Multiple view geometry under the L_∞-norm. *PAMI*, 30(9):1603 – 1617, September 2008.

[86] F. Kahl and D. Henrion. Globally optimal estimates for geometric reconstruction problems. volume 2, pages 978 –985 Vol. 2, oct. 2005.

[87] Fredrik Kahl, Sameer Agarwal, Manmohan Krishna Chandraker, David Kriegman, and Serge Belongie. Practical global optimization for multiview geometry. *Int. J. Comput. Vision*, 79(3):271–284, 2008.

[88] Deepak Kapur and Yagati N Lakshman. Elimination methods: an introduction. symbolic and numerical computation for artificial intelligence b. donald et. al, 1992.

[89] Deepak Kapur, Tushar Saxena, and Lu Yang. Algebraic and geometric reasoning using dixon resultants. In *Proceedings of the international symposium on Symbolic and algebraic computation*, pages 99–107. ACM, 1994.

[90] Qifa Ke and T. Kanade. Quasiconvex optimization for robust geometric reconstruction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(10):1834 –1847, oct. 2007.

[91] Sin-Jung Kim, Mun-Ho Jeong, Joong-Jae Lee, Ji-Yong Lee, Kang-Geon Kim, Bum-Jae You, and Sang-Rok Oh. Robot head-eye calibration using the minimum variance method. In *Robotics and Biomimetics (ROBIO), 2010 IEEE International Conference on*, pages 1446–1451. IEEE, 2010.

[92] E Kruppa. Zur ermittlung eines objektes aus zwei perspektiven mit innerer orientierung. *SitzBer Akad Wiss Wien Math Naturw Abt IIa*, 122(122):1939–1948, 1913.

[93] Z. Kukelova, M. Bujnak, and T. Pajdla. Polynomial eigenvalue solutions to the 5-pt and 6-pt relative pose problems. pages xx–yy, 2008.

[94] Z. Kukelova and T. Pajdla. A minimal solution to the autocalibration of radial distortion. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1 –7, june 2007.

[95] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Automatic generator of minimal problem solvers. In *Proceedings of the 10th European Conference on Computer Vision: Part III*, ECCV '08, pages 302–315, Berlin, Heidelberg, 2008. Springer-Verlag.

[96] Zuzana Kukelova, Martin Byröd, Klas Josephson, Tomas Pajdla, and Kalle ström. Fast and robust numerical solutions to minimal problems for cameras with radial distortion. *Comput. Vis. Image Underst.*, 114(2):234–244, 2010.

[97] Zuzana Kukelova, Jan Heller, and Tomas Pajdla. Hand-eye calibration without hand orientation measurement using minimal solution. In *Asian Conference on Computer Vision (ACCV)*, pages 576–589. Springer, 2013.

[98] Thomas Lange and Sebastian Eulenstein. Calibration of swept-volume 3-d ultrasound. *MIUA Proc*, 99(3):29–32, 2002.

[99] Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11:796–817, 2001.

[100] Monique Laurent. Sums of squares, moment matrices and optimization over polynomials, to appear in emerging applications of algebraic geometry. In *of IMA Volumes in Mathematics and its Applications*. Springer, 2009.

[101] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155–166, 2009.

[102] Hongdong Li. A simple solution to the six-point two-view focal-length problem. In *Computer Vision–ECCV 2006*, pages 200–213. Springer, 2006.

[103] Hongdong Li and Richard Hartley. Five-point motion estimation made easy. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 1, pages 630–633. IEEE, 2006.

[104] Xiaowei Li, Changchang Wu, Christopher Zach, Svetlana Lazebnik, and Jan-Michael Frahm. Modeling and recognition of landmark image collections using iconic scene graphs. In *Computer Vision–ECCV 2008*, pages 427–440. Springer, 2008.

[105] Rong-hua Liang and Jian-fei Mao. Hand-eye calibration with a new linear decomposition algorithm. *Journal of Zhejiang University Science A*, 9(10):1363–1368, 2008.

[106] J. Löfberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[107] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. pages 61–62, 1987.

[108] Manolis I.A. Lourakis. Sparse non-linear least squares optimization for geometric vision. In *European Conference of Computer Vision*, pages 43–56. Springer, 2010.

[109] M.I. A. Lourakis and A.A. Argyros. Sba: A software package for generic sparse bundle adjustment. *ACM Trans. Math. Software*, 36(1):1–30, 2009.

[110] M.I.A. Lourakis. levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. `http://www.ics.forth.gr/~lourakis/levmar/`, Jul. 2004.

[111] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, November 2004.

[112] C-P Lu, Gregory D Hager, and Eric Mjolsness. Fast and globally convergent pose estimation from video images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(6):610–622, 2000.

[113] Fangfang Lu and Richard Hartley. A fast optimal algorithm for l2 triangulation. In *ACCV'07: Proceedings of the 8th Asian conference on Computer vision*, pages 279–288, Berlin, Heidelberg, 2007. Springer-Verlag.

[114] Q-T Luong and Olivier D Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of computer vision*, 22(3):261–289, 1997.

[115] FS Macaulay. Some formulae in elimination. *Proceedings of the London Mathematical Society*, 1(1):3–27, 1902.

[116] Abed Malti and Joao P Barreto. Robust hand-eye calibration for computer aided medical endoscopy. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 5543–5549. IEEE, 2010.

[117] Abed Malti and Joao Pedro Barreto. Hand–eye and radial distortion calibration for rigid endoscopes. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 2013.

[118] P. Mareček. A camera calibration system. Master's thesis, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, 2001.

[119] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial & Applied Mathematics*, 11(2):431–441, 1963.

[120] Daniel Martinec and Tomas Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[121] Stephen J Maybank and Olivier D Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.

[122] J Michael McCarthy. *Introduction to theoretical kinematics*. MIT press, 1990.

[123] J.C. McGlone, E.M. Mikhail, J.S. Bethel, R. Mullen, American Society for Photogrammetry, and Remote Sensing. *Manual of photogrammetry*. American Society for Photogrammetry and Remote Sensing, 2004.

[124] B. Micusik and T. Pajdla. Estimation of omnidirectional camera model from epipolar geometry. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–485–I–490vol.1, 18-20 June 2003.

[125] Branislav Micusik and Tomas Pajdla. Structure from motion with wide circular field of view cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7):1135–1149, 2006.

[126] Kenro Miyamoto. Fish eye lens. *JOSA*, 54(8):1060–1061, 1964.

[127] Joseph L Mundy and Andrew Zisserman. Appendix-projective geometry for machine vision. 1992.

[128] Katta G Murty and Santosh N Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical programming*, 39(2):117–129, 1987.

[129] Jiawang Nie. Optimality conditions and finite convergence of lasserre's hierarchy. *Mathematical Programming*, pages 1–25, 2012.

[130] D. Nistér. An efficient solution to the five-point relative pose problem. *PAMI*, 26(6):756–770, June 2004.

[131] David Nistér. Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors. In *European Conference on Computer Vision*, pages 649–663. Springer, 2000.

[132] David Nister, Richard Hartley, and Henrik Stewenius. Using galois theory to prove structure from motion algorithms are optimal. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:1–8, 2007.

[133] David Nistér and Henrik Stewénius. A minimal solution to the generalised 3-point pose problem. *Journal of Mathematical Imaging and Vision*, 27(1):67–79, 2007.

[134] F.C. Park and B.J. Martin. Robot sensor calibration: solving AX=XB on the euclidean group. *Robotics and Automation, IEEE Transactions on*, 10(5):717 –721, October 1994.

[135] Pablo A. Parrilo. Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization. Phd thesis, 2000.

[136] Michael A. Penna. Camera calibration: A quick and easy way to determine the scale factor. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(12):1240–1245, 1991.

[137] Mihai Putinar. Positive polynomials on compact semi-algebraic sets. *Indiana University Mathematics Journal*, 42(3):969–984, 1993.

[138] N Ragot, J-Y Ertaud, X Savatier, and B Mazari. Calibration of a panoramic stereovision sensor: Analytical vs interpolation-based methods. In *IEEE Industrial Electronics, IECON 2006-32nd Annual Conference on*, pages 4130–4135. IEEE, 2006.

[139] Carsten Rother and Stefan Carlsson. Linear multi view reconstruction and camera recovery using a reference plane. *International journal of computer vision*, 49(2-3):117–141, 2002.

[140] Thomas Ruland, Heidi Loose, Tomas Pajdla, and Lars Kru ger. Hand-eye autocalibration of camera positions on vehicles. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 367–372. IEEE, 2010.

[141] Thomas Ruland, Tomas Pajdla, and Lars Kruger. Globally optimal hand-eye calibration. In *CVPR*, 2012.

[142] D. Scaramuzza, A. Martinelli, and R. Siegwart. A toolbox for easily calibrating omnidirectional cameras. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5695 –5701, oct. 2006.

[143] J. Schmidt, F. Vogt, and H. Niemann. Calibration-free hand-eye calibration: A structure-from-motion approach. In *DAGM*, page 67, 2005.

[144] Jochen Schmidt, Florian Vogt, and Heinrich Niemann. Robust hand–eye calibration of an endoscopic surgery robot using dual quaternions. In *Pattern Recognition*, pages 548–556. Springer, 2003.

[145] G. Schweighofer and A. Pinz. Globally optimal O(n) solution to the PnP problem for general camera models. In *BMVC 2008*, 2008.

[146] Yongduek Seo, Young-Ju Choi, and Sang Wook Lee. A branch-and-bound algorithm for globally optimal calibration of a camera-and-rotation-sensor system. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1173 –1178, sept. 2009.

[147] Yongduek Seo, Hyunjung Lee, and Sang Wook Lee. Outlier removal by convex optimization for L_∞ approaches. In *Proceedings of the 3rd Pacific Rim Symposium on Advances in Image and Video Technology*, PSIVT '09, pages 203–214, Berlin, Heidelberg, 2008. Springer-Verlag.

[148] Mili Shah, Roger D Eastman, and Tsai Hong. An overview of robot-sensor calibration methods for evaluation of perception systems. In *Proceedings of the Workshop on Performance Metrics for Intelligent Systems*, pages 15–20. ACM, 2012.

[149] Y.C. Shiu and S. Ahmad. Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form AX=XB. *Robotics and Automation, IEEE Transactions on*, 5(1):16 –29, feb 1989.

[150] Bruno Siciliano and Oussama Khatib. *Springer handbook of robotics*. Springer, 2008.

[151] Kristy Sim and Richard Hartley. Removing outliers using the L_∞ norm. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1*, pages 485–494, Washington, DC, USA, 2006. IEEE Computer Society.

[152] Chester C Slama, Charles Theurer, Soren W Henriksen, et al. *Manual of photogrammetry*. Number Ed. 4. American Society of photogrammetry, 1980.

[153] N. Snavely, S. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *IJCV*, 80(2):189–210, 2008.

[154] Noah Snavely. Scene reconstruction and visualization from internet photo collections: A survey. *IPSJ Transactions on Computer Vision and Applications*, 3(0):44–66, 2011.

[155] Noah Snavely, Steven M Seitz, and Richard Szeliski. Skeletal graphs for efficient structure from motion. In *CVPR*, volume 1, page 2, 2008.

[156] S. Soatto and R. Brockett. Optimal structure from motion: local ambiguities and global estimates. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 282 –288, 23-25 1998.

[157] Drew Steedly, Irfan Essa, and Frank Dellaert. Spectral partitioning for structure from motion. In *Proceedings of the 2003 9th IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 996–1003, 2003.

[158] R Matt Steele and Christopher Jaynes. Overconstrained linear estimation of radial distortion and multi-view geometry. In *Computer Vision–ECCV 2006*, pages 253–264. Springer, 2006.

[159] Hans J. Stetter. *Numerical polynomial algebra*. SIAM, 2004.

[160] H. Stewenius and K. Astrom. Hand-eye calibration using multilinear constraints. In *ACCV*, 2004.

[161] H. Stewenius, F. Schaffalitzky, and D. Nister. How hard is 3-view triangulation really? In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 686 – 693 Vol. 1, 17-21 2005.

[162] Henrik Stewénius. *Gröbner basis methods for minimal problems in computer vision*. PhD thesis, Lund Inst. for Technology, Centre for Mathematical Sciences, Lund University, 2005.

[163] Henrik Stewénius, Chris Engels, and David Nistér. An efficient minimal solution for infinitesimal camera motion. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[164] Henrik Stewenius, Christopher Engels, and David Nistér. Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60(4):284–294, 2006.

[165] Henrik Stewénius, David Nistér, Fredrik Kahl, and Frederik Schaffalitzky. A minimal solution for relative pose with unknown focal length. *Image and Vision Computing*, 26(7):871–877, 2008.

[166] Henrik Stewénius, David Nistér, Magnus Oskarsson, and Kalle Åström. Solutions to minimal generalized relative pose problems. In *Workshop on omnidirectional vision*, 2005.

[167] K. H. Strobl and G. Hirzinger. Optimal Hand-Eye Calibration. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4647–4653, Beijing, China, October 2006.

[168] J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999.

[169] Peter Sturm, Srikumar Ramalingam, Jean-Philippe Tardif, Simone Gasparini, and Joao Barreto. Camera models and fundamental concepts used in geometric computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(1–2):1–183, 2011.

[170] Peter Sturm and Bill Triggs. A factorization based algorithm for multi-image projective structure and motion. In *European Conference on Computer Vision*, pages 709–720. Springer, 1996.

[171] Peter F Sturm and Stephen J Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1. IEEE, 1999.

[172] Tomáš Svoboda, Tomáš Pajdla, and Václav Hlaváč. Epipolar geometry for panoramic cameras. In *Computer Vision ECCV'98*, pages 218–231. Springer, 1998.

[173] R. Szeliski and Sing Bing Kang. Shape ambiguities in structure from motion. volume 19, pages 506 –512, may 1997.

[174] Richard Szeliski and Sing Bing Kang. Recovering 3d shape and motion from image streams using nonlinear least squares. In *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR'93., 1993 IEEE Computer Society Conference on*, pages 752–753. IEEE, 1993.

[175] William P Tayman. Analytical multicollimator camera calibration. *Photogrammetria*, 34(5):179–197, 1978.

[176] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[177] Akihiko Torii, Michal Havlena, and Tomáš Pajdla. Omnidirectional image stabilization for visual object recognition. *International Journal of Computer Vision*, 91(2):157–174, January 2011.

[178] P.H.S. Torr and A.W. Fitzgibbon. Invariant fitting of two view geometry. volume 26, pages 648 –650, may 2004.

[179] Bill Triggs. Camera pose and calibration from 4 or 5 known 3d points. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 278–284. IEEE, 1999.

[180] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *ICCV '99: Proceedings of the International Workshop on Vision Algorithms*, pages 298–372, London, UK, 2000. Springer-Verlag.

[181] Roger Y Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 1986*, 1986.

[182] R.Y. Tsai and R.K. Lenz. Real time versatile robotics hand/eye calibration using 3d machine vision. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 554 –561 vol.1, April 1988.

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

[184] Anand Viswanathan, Emad M Boctor, Russell H Taylor, Gregory Hager, and Gabor Fichtinger. Immediate ultrasound calibration with three poses and minimal image processing. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2004*, pages 446–454. Springer, 2004.

[185] Florian Vogt, S Kruger, Jochen Schmidt, Dietrich Paulus, Heinrich Niemann, Werner Hohenberger, and CH Schick. Light fields for minimal invasive surgery using an endoscope positioning robot. *Methods of Information in Medicine*, 43(4):403–408, 2004.

[186] C.C. Wang. Extrinsic calibration of a vision sensor mounted on a robot. *Robotics and Automation, IEEE Transactions on*, 8:161–175, 1992.

[187] Robert W Wood. Xxiii. fish-eye views, and vision under water. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 12(68):159–162, 1906.

[188] X Zhang. Pose estimation using l. In *Image and Vision Computing New Zealand*, 2005.

[189] Zhengyou Zhang. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1330–1334, 2000.

[190] Zhengyou Zhang. Camera calibration with one-dimensional objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(7):892–899, 2004.

[191] Zijian Zhao. Hand-eye calibration using convex optimization. In *ICRA*, pages 2947–2952, 2011.

[192] Zijian Zhao and Yuncai Liu. Hand-eye calibration based on screw motions. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 1022–1026. IEEE, 2006.

[193] Hanqi Zhuang, Z.S. Roth, and R. Sudhakar. Simultaneous robot/world and tool/flange calibration by solving homogeneous transformation equations of the form $AX = YB$. *Robotics and Automation, IEEE Transactions on*, 10(4):549 –554, aug 1994.

[194] Hanqi Zhuang and Yiu Cheung Shiu. A noise tolerant algorithm for wrist-mounted robotic sensor calibration with or without sensor orientation measurement. In *lEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1095–1100, 1992.

A Author's Publications

Publications related to the thesis

Impacted journal articles

- **Jan Heller**, Michal Havlena and Tomas Pajdla. Globally Optimal Hand-Eye Calibration Using Branch-and-Bound. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, accepted for publication, August, 2015. [40%-30%-30%]

- **Jan Heller**, Tomas Pajdla. GpoSolver: A Matlab/C++ Toolbox for Global Polynomial Optimization. *Optimization Methods and Software (OMS)*, submitted, after major revision. [50%-50%]

Publications excerpted by WOS

- **Jan Heller**, Michal Havlena, and Tomas Pajdla. A branch-and-bound algorithm for globally optimal hand-eye calibration. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1608–1615. IEEE, June, 2012. [40%]

- **Jan Heller**, Michal Havlena, Akihiro Sugimoto, and Tomas Pajdla. Structure-from-motion based hand-eye calibration using L_∞ minimization. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3497–3503. IEEE, June, 2011. [40%]

Other conference publications

- **Jan Heller**, Didier Henrion, and Tomas Pajdla. Hand-eye and robot-world calibration by global polynomial optimization. *In IEEE International Conference on Robotics and Automation (ICRA)*, pages 3157–3164. IEEE, May, 2014. [40%]

- **Jan Heller**, Tomas Pajdla. World-base calibration by global polynomial optimization. *In 2nd International Conference on 3D Vision (3DV)*, pages 593–600. December, 2014. [50%]

- Zuzana Kukelova, **Jan Heller**, and Tomas Pajdla. Hand-eye calibration without hand orientation measurement using minimal solution. *In Asian Conference on Computer Vision (ACCV)*, pages 576–589. Springer, November, 2012. [30%]

Additional publications

Publications excerpted by WOS

- Tomas Pajdla, Michal Havlena, and **Jan Heller**. Learning from Incongruence. *In Dirac Workshop by The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*. Book Series: Studies in Computational Intelligence, pages 119-127, Springer, 2012.

- Michal Havlena, **Jan Heller**, Hendrik Kayser, Jörg-Hendrik Bach, Jörn Anemüller, and Tomáš Pajdla. Incongruence detection in audio-visual processing. *In Dirac Workshop by The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*. Book Series: Studies in Computational Intelligence, pages 67–75, Springer, 2012.

- **Jan Heller**, Tomas Pajdla. Stereographic rectification of omnidirectional stereo pairs. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1414–1421. IEEE, June, 2009.

Other conference publications

- Zuzana Kukelova, **Jan Heller**, Martin Bujnak, Tomas Pajdla. Radial distortion homography. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, June, 2015.

- **Jan Heller**, Michal Havlena, Michal Jancosek, Akihiko Torii, Tomas Pajdla. 3D Reconstruction from Photographs by CMP SfM Web Service. *In IAPR International Conference on Machine Vision Applications (MVA)*, pages 30–34, 2015.

- **Jan Heller**, Didier Henrion, Tomas Pajdla. Stable radial distortion calibration by polynomial matrix inequalities programming. *In Asian Conference on Computer Vision (ACCV)*, pages 307-321. Springer, November, 2014.

- Zuzana Kukelova, Martin Bujnak, **Jan Heller**, Tomas Pajdla. Singly-bordered block-diagonal form for minimal problem solvers. *In Asian Conference on Computer Vision (ACCV)*, pages 488–502. Springer, November, 2014.

- Michal Havlena, Michal Jancosek, **Jan Heller**, Tomas Pajdla. 3D Surface Models from Opportunity MER NavCam (Extended Abstract), *European Planetary Science Congress*, 2010.

B Citations of Author's Work

[1] Cenek Albl, Zuzana Kukelova, and Tomas Pajdla. R6P - rolling shutter absolute camera pose. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[2] Atsuhiko Banno and Katsushi Ikeuchi. Omnidirectional texturing based on robust 3d registration through euclidean reconstruction from two spherical images. *Computer Vision and Image Understanding*, 114(4):491 – 499, 2010. Special issue on Image and Video Retrieval Evaluation.

[3] Luca Carlone, Pablo Fernandez Alcantarilla, Han-Pang Chiu, Zsolt Kira, and Frank Dellaert. Mining structure fragments for smart bundle adjustment. In *Proceedings of the British Machine Vision Conference. BMVA Press*, 2014.

[4] Zhijun Dai, Yihong Wu, Fengjun Zhang, and Hongan Wang. A novel fast method for l_∞ problems in multiview geometry. In *Computer Vision–ECCV 2012*, pages 116–129. Springer, 2012.

[5] Marek Franaszek. Registration of six degrees of freedom data with proper handling of positional and rotational noise. *Journal of research of the national institute of standards and technology*, 118:280–291, 2013.

[6] Hanchao Jia and Shigang Li. Scene analysis based on horse vision system. In *The Twelfth IAPR Conference on Machine Vision Applications*, pages 267–270, 2011.

[7] Hong Liu, Yu-long Zhou, and Zhao-peng Gu. Inertial measurement unit-camera calibration based on incomplete inertial sensor information. *Journal of Zhejiang University SCIENCE C*, 15(11):999–1008, 2014.

[8] Biao Mei, Weidong Zhu, Kangzheng Yuan, and Yinglin Ke. Robot base frame calibration with a 2D vision system for mobile robotic drilling. *The International Journal of Advanced Manufacturing Technology*, pages 1–15, 2015.

[9] Victor Nevarez and Ron Lumia. Image jacobian estimation using structure from motion on a centralized point. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 773–778. IEEE, 2014.

[10] Gerhard Paar, Lester Waugh, DP Barnes, Tomas Pajdla, Mark Woods, H-R Graf, Yang Gao, Konrad Willner, J-P Muller, and Rongxing Li. Integrated field testing of planetary robotics vision processing: the provisg campaign in tenerife 2011. In *IS&T/SPIE Electronic Imaging*, pages 83010O–83010O. International Society for Optics and Photonics, 2012.

[11] Gerhard Paar, M Woods, D Pullan, and Proviscout Team. Robotics vision for a scouting rover-PRoViScout. In *EPSC-DPS Joint Meeting 2011*, volume 1, page 123, 2011.

[12] Gerhard Paar, Mark Woods, Christiane Gimkiewicz, Frédéric Labrosse, Alberto Medina, Laurence Tyler, David P Barnes, Gerald Fritz, and Konstantinos Kapellos. PRoViScout: a planetary scouting rover demonstrator. In *IS&T/SPIE Electronic Imaging*, pages 83010A–83010A. International Society for Optics and Photonics, 2012.

[13] Thomas Ruland, Tomas Pajdla, and Lars Krüger. Global optimization of extended hand-eye calibration. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 740–745. IEEE, 2011.

[14] Thomas Ruland, Tomas Pajdla, and Lars Kruger. Globally optimal hand-eye calibration. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1035–1042. IEEE, 2012.

[15] Peter Sturm, Srikumar Ramalingam, Jean-Philippe Tardif, Simone Gasparini, and Joao Barreto. Camera models and fundamental concepts used in geometric computer vision. *Foundations and Trends in Computer Graphics and Vision*, 6(1–2):1–183, 2011.

[16] Zachary Taylor and Juan Nieto. Motion-based calibration of multimodal sensor arrays. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4843–4850, 2015.

[17] Haixia Wang, Xiao Lu, Zhanyi Hu, and Yuxia Li. A vision-based fully-automatic calibration method for hand-eye serial robot. *Industrial Robot: An International Journal*, 42(1):64–73, 2015.

[18] Yanchang Wang, Xiaojin Gong, Ying Lin, and Jilin Liu. Stereo calibration and rectification for omnidirectional multi-camera systems. *International Journal of Advanced Robotic Systems*, 9(143), 2012.

[19] Ying-mei Wei, Lai Kang, Bing Yang, et al. Applications of structure from motion: a survey. *Journal of Zhejiang University SCIENCE C*, 14(7):486–494, 2013.

[20] Daphna Weinshall, Hynek Hermansky, Alon Zweig, Jie Luo, Holly Jimison, Frank Ohl, and Misha Pavel. Beyond novelty detection: Incongruent events, when general and specific classifiers disagree. In *Advances in Neural Information Processing Systems*, pages 1745–1752, 2009.

[21] Jiaolong Yang, Hongdong Li, and Yunde Jia. Optimal essential matrix estimation via inlier-set maximization. In *Computer Vision–ECCV 2014*, pages 111–126. Springer, 2014.

[22] Junchao Zhu, Yongchen Li, Zhijun Ma, Yingkui Jiao, and Baofeng Zhang. Study on multiple image processing hardware system based on dsp. In *Mechatronics and Automation (ICMA), 2014 IEEE International Conference on*, pages 1844–1849. IEEE, 2014.