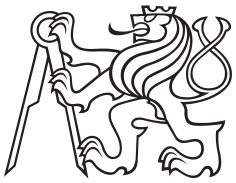
Master Thesis



Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Efficient Homotopy Continuation for Multi-View Geometry

Bc. Petr Hrubý

Supervisor: doc. Ing. Tomáš Pajdla Ph.D.
Field of study: Open Informatics
Subfield: Computer Vision and Digital Image
May 2021

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Hrubý Petr** Personal ID number: **466292**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Open Informatics**

Specialisation: **Computer Vision and Image Processing**

II. Master's thesis details

Master's thesis title in English:

Efficient Homotopy Continuation for Multi-View Geometry

Master's thesis title in Czech:

Efektivní výpočet geometrie kamer s použitím homotopických metod

Guidelines:

1) Study the approach to computing multi-view geometry using homotopy continuation [1,2,3].
2) Suggest a modification of the standard homotopy continuation method to achieve an efficient solution for problems in multi-view geometry.
3) Implement the approach and evaluate it on real data.

Bibliography / sources:

[1] Ricardo Fabbri, Timothy Duff, Hongyi Fan, Margaret H. Regan, David da Costa de Pinho, Elias P. Tsigaridas, Charles Wampler, Jonathan D. Hauenstein, Benjamin B. Kimia, Anton Leykin, Tomáš Pajdla: Trifocal Relative Pose from Lines at Points and its Efficient Solution. CoRR abs/1903.09755 (2019)
[2] Duff Timothy, et. al.: Solving polynomial systems via homotopy continuation and monodromy
[3] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler: Numerically Solving Polynomial Systems with Bertini, Software, Environments, and Tools 25, SIAM, 2013.

Name and workplace of master's thesis supervisor:

doc. Ing. Tomáš Pajdla, Ph.D., Applied Algebra and Geometry, CIIRC

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **04.01.2021** Deadline for master's thesis submission: **21.05.2021**

Assignment valid until: **30.09.2022**

doc. Ing. Tomáš Pajdla, Ph.D.	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor's signature	Head of department's signature	Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

© ČVUT v Praze, Design: ČVUT v Praze, VIC

Acknowledgements

I would like to thank my supervisor Tomáš Pajdla for his guidance, his valuable advice, and for finding me an interesting research topic. I would also like to thank Anton Leykin and Tim Duff for useful discussions about the project and for introducing me to the Homotopy Continuation method. We intend to use the results presented in this thesis as a part of our joint work and future publications. Finally, I would like to thank my family for their support.

This research was supported by the EU Structural and Investment Funds, Operational Programe Research, Development and Education under the project IMPACT (reg. no. CZ.02.1.01/0.0/0.0/15_003/0000468), EU H2020 ARtwin No. 856994, and EU H2020 SPRING No. 871245.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, 20. May 2021

Abstract

Homotopy Continuation is a method from numerical algebraic geometry for solving systems of polynomial equations. During the Homotopy Continuation, the starting system is gradually transformed into the final problem and the solution to the system is tracked. When the Homotopy Continuation is finished, we obtain the solution to the final problem.

In this thesis, we present two solvers based on Homotopy Continuation, which are used for the estimation of the relative pose of two views from five points and the relative pose of three views from four points. Until now, the Homotopy Continuation solvers have been too slow to be used in RANSAC. The running time of the solvers proposed in this thesis is comparable to the time of the Nistér solver, which makes them eligible for RANSAC. We have achieved the low running time by selecting the starting problem with a Multilayer Perceptron, by tracking in the real domain, and by an efficient evaluation of systems of linear equations which arise in the Homotopy Continuation method.

Keywords: homotopy continuation, solution of equations, polynomial equations, minimal geometric problems

Supervisor: doc. Ing. Tomáš Pajdla Ph.D.
Aplikovaná algebra a geometrie CIIRC,
Jugoslávských partyzánů 1580/3,
Praha 6

Abstrakt

Homotopické pokračování je metoda numerické algebraické geometrie pro řešení soustav polynomiálních rovnic. Během homotopického pokračování se počáteční soustava postupně transformuje na koncovou soustavu, během čehož se sleduje řešení transformované soustavy. Když je homotopické pokračování ukončeno, získáme řešení koncové soustavy.

V této práci navrhujeme řešení úloh nalezení relativní polohy dvou kamer z pěti bodů a nalezení relativní polohy tří kamer ze čtyř bodů pomocí homotopického pokračování. Doposud bylo řešení pomocí homotopického pokračování příliš pomalé na to, aby se dalo efektivně použít ve schématu RANSAC. Čas potřebný pro řešení prezentovaná v této práci je srovnatelný s časem potřebným pro Nistérův pětibodový algoritmus, tudíž jsou tato řešení vhodná pro RANSAC. Takto nízkých časů jsme dosáhli výběrem počátečního řešení pomocí neuronové sítě, sledováním řešení pouze v reálném oboru a efektivním řešením soustav lineárních rovnic, které se objevují při homotopickém pokračování.

Klíčová slova: homotopické metody, řešení rovnic, polynomiální rovnice, minimální geometrické problémy

Překlad názvu: Efektivní výpočet geometrie kamer s použitím homotopických metod

Contents

Figures

Tables

ctuthesis t1606152353

Chapter 1

Introduction

A system of polynomial equations is called a minimal problem if it has a finite nonzero number of complex solutions. Minimal problems arise in numerous problems in computer vision, such as pose estimation or camera calibration. The solution of minimal problems is often used in the RANSAC scheme [BF81] to obtain an exact solution for a sample of points, which is then verified using the remaining points. Until now, the solvers based on Homotopy Continuation [SI05] have been too slow to be used in RANSAC. Therefore, the solvers used in the RANSAC loop have been based on the symbolical methods in Algebraic Geometry [CLO07], while the Homotopy Continuation Solvers have been used to study the minimal problems In this thesis, we propose two solvers based on Homotopy Continuation, which are used to solve two minimal problems arising in the computer vision: the Five-Point problem and the relaxed version of the Four-Point problem. The running time of these solvers is comparable to the running time of the Nistér algorithm, and therefore, they are suitable to be used in RANSAC.

1.1 Structure of the Thesis

In Chapter 1 the State of the art is reviewed, the notation used in the thesis is presented, and the contributions of the thesis are summarized.

In Chapter 2 the Point-Line problems of five points in two views and four points in three views are described. The depth formulation, which allows

to convert the problems into systems of parametrized polynomial equations, is introduced. For both problems, a generator of problem-solution pairs is given.

In Chapter 3 the Homotopy Continuation method for solving systems of polynomial equations is described. The predictor and corrector steps of the homotopy continuation, as well as the concepts of Parametric Homotopy Continuation and the Real Homotopy Continuation are given.

In Chapter 4 the Homotopy Continuation solver for the problem of five points in two views is proposed. In Chapter 5 the Homotopy Continuation solver for the problem of four points in three views is proposed.

In Chapter 6 the experiments, whose purpose is to evaluate the solvers described in Chapters 4 and 5, are described.

1.2 Contributions

This work brings the following main contributions.

1. We propose a new solver for the problem of five points in two views (Five-Point problem) [Nis04] and for the problem of four points in three views (Four-Point problem) [NS06]. Our solvers are based on Real Homotopy Continuation [SI05] and their running time is comparable with the state-of-the-art Nistér solver for the Five-Point problem [Nis04], which makes them suitable for the use in RANSAC loop.
2. In the solvers, we track only one solution to each input problem from a starting problem-solution pair which is selected using a Neural Network classifier. We show that it is possible to learn a very few starting points to be able to solve realistic real data sets. This, together with the use of Real Homotopy Continuation instead of Complex Homotopy Continuation and with an efficient evaluation of the predictor and corrector, allows us to achieve such low running time in tens of micro seconds.
3. So far, the homotopy continuation solvers have been used predominantly for an offline study of the minimal problems, as they have been too slow for the use in the RANSAC loop. In this work, we present Homotopy Continuation solvers, which are fast enough for RANSAC. To our

best knowledge, this is the first time a Homotopy Continuation solver for a minimal problem in Computer Vision achieves times below 100 microseconds.

4. We evaluate our Five-Point Homotopy Continuation solver on a benchmark [Mys20]. The experiments show that the Homotopy Continuation solver is slightly faster than the Nistér solver [Nis04], while its precision is not much worse.

5. A symbolic solver for the Four-Point problem has been proposed in [NS06] but, to our best knowledge, there is no available implementation of the solver. In this work, we propose a solver to the Four-Point problem based on the Homotopy Continuation, whose average running time is below 100 microseconds. We demonstrate this on the proposed Four-Point Homotopy Continuation solver. The formulation of this problem is simple but the solution is difficult.

6. We demonstrate that Homotopy Continuation solvers may be generalized to different minimal problems [DKLP19], [DKLP20], for which a symbolic solution is difficult to find. Efficient Homotopy Continuation solvers for other problems may be designed using the principles described in this work.

1.3 State of the Art

1.3.1 Homotopy continuation

Homotopy Continuation [SI05], [Mor09], [BSHW13] is a method from the Numerical Algebraic Geometry, which is used for solving systems of polynomial equations. The principle of Homotopy Continuation is further described in Chapter 3. Over the time, numerous general-purpose Homotopy Continuation solvers have been developed, such as Bertini [BHSW], Hom4PS-3 [CLL14], Numerical Algebraic Geometry for Macaulay2 [Ley09], PHPack [Ver99].

The Homotopy Continuation was used in the 1990s in the Computer Vision for tracing of curves and surfaces in 3D [KP91], [KP92], computing aspect graphs [KP89], [Pet99], motion estimation [HNH90], [BHN94], pose estimation [RJH97], [HN94], camera calibration [MF92], [Luo92], [FLM92], [PG99], Markov Random Field models for image restoration [NDP94]. More recently, Homotopy Continuation has been used for [EJ10] shape from shading, and for energy minimization [Sal13]. The implementation [Pol] of the solver [KP91] is available as a part of the library VNL.

1.3.2 Minimal Problems Solving

Minimal problems arise in numerous problems of Computer Vision, such as pose estimation or camera calibration. The State-of-the-art minimal solvers rely on symbolic methods of Algebraic Geometry [CLO07], [LOÅ+18]. A solution to the Five-Point problem using Essential matrix has been proposed in [Nis04].

The problem of estimation of the trifocal tensor has been studied in [QTAM01] [QTM06], [Rod15], [LTD15], [Mar17], [AO14], [AT10]. Problems of estimation of relative poses between more than three cameras have been studied in [Mat16], [Oed15]. In [QTM06], a depth formulation of the problem of the problem of four points in three views is described.

A solution to the problem of four points in three views has been proposed in [NS06]. A solver for the estimation of a trifocal tensor from 10, 11, and 12 lines was proposed in [KOÅ14]. More recently, a minimal solver [LÅO17] for estimation of a trifocal tensor from 9 lines was proposed. Solvers for the estimation of a trifocal pose from a combination of points and lines are proposed in papers [OZÅ04], [QN17]. In [VLZ19] a relative pose between two stereo pairs is obtained from two points and two lines. In works [AL87], [RF91], [CG00], [SZ00] the relative pose between three cameras is obtained from edges or curves. In works [JOÅ02], [ZKHM20] the relative pose is computed from points with one, two, or three lines incident to them. The orientation of the SIFT features may be used as the line incident to a point [Fab10]. In work [FKG12], the points incident to lines are used for the estimation of the trifocal tensor in a calibrated setting. To our best knowledge, none of these three view solvers is used in practice.

The Homotopy Continuation method has been used by the Computer Vision community as an offline tool to study the structure of the solutions of minimal problems, such as symmetry, number of real solutions, and decomposition of varieties into irreducible components [Kor20]. A minimal problem with six lines in three views was studied with homotopy continuation in [HN94]. In [Kil17], minimal problems in three views, including the problem described in Section 2.2.1, have been studied using a Homotopy Continuation system Bertini [BHSW]. In [KGB19] the general homotopy continuation system PHPack [Ver99] is compared against a symbolic solver based on resultants.

A more specific Homotopy Continuation procedure [DHJ+16] is used in [FDF+20] to solve a problem "Chicago" of three points and two incident lines in three views. In [DKLP19], a catalogue of all minimal problems in full

visibility is given. Work [DKLP20] shows a similar catalogue of minimal problems in partial visibility. The numbers of solutions of the problems in both catalogues have been found using homotopy continuation.

1.3.3 Real Homotopy Continuation

In most of the applications, only the real solutions to the polynomial systems are interesting. However, usually a vast majority of the solutions are nonreal. In [HR18], the homotopy continuation is truncated if it appears to end in a nonreal solution. In work [HR20] the structure of real solutions in a real parametric space is studied. In [BHM+20], the structure of cells in the parametric space, whose number of solutions is the same, is learned using a neural network. If the real homotopy continuation tracks within one cell, the track avoids the singularities between the cells, and therefore, there hold similar guarantees as in the case of complex homotopy. The process of determining the cell and tracking the real solutions is used to solve a Kuramoto model with 3 and 4 oscillators. In [Die19], the real homotopy continuation is used to solve the kinematics of a robot.

1.4 Notation

Now, we are going to describe the notation used in the thesis. If not stated otherwise, the elements of matrices are indexed from 1. Sections 4.2 and 5.2, which describe a closed form solution of the sparse linear equations, use indexing from 0.

A set of n points of dimension d is denoted as a matrix $X \quad \mathsf{R}^{d,n}$ whose columns are the points in the set. i-th coordinate of j-th point is denoted as $X_{i,j}$. The j-th point is denoted as $X_{:,j}$ (like in MATLAB, j-th column of the matrix) or by enumerating the coordinates like:

$$\begin{bmatrix} X_{1,j} \\ \dots \\ X_{d,j} \end{bmatrix}$$

The depths are denoted by a matrix λ whose i-th row contains the depths

in the i-th view. For the Five-Point problem, $\lambda \quad \mathsf{R}^{2,5}$, for the Four-Point problem $\lambda \quad \mathsf{R}^{3,4}$.

The relative poses are denoted as (R, t), where $R \quad SO(3)$, $t \quad \mathsf{R}^3$.

The invariantized points (Section 4.4, Section 5.4) are denoted as $\bar{x}$, aligned points (Section 4.5, Section 5.5) are denoted as $\hat{x}$, and the anchors (Section 4.6.1, Section 5.6.1) are denoted as $\breve{x}$. The corresponding depths are denoted likewise. Homogeneous points are denoted with the superscript h as x^h.

Chapter 2

Minimal problems in Multi-View Geometry

The **Point-Line-Problem** [DKLP19] is given by m cameras. The projections of p points, and l lines onto the cameras, as well as a set $I \quad 1,...,l \times 1,...,p$ of incidences between the points and the lines, are known. The task is to recover the position of the p points and l lines in the 3D space, as well as to obtain the set of relative poses between the m cameras. In the partial visibility version of the problem [DKLP20], some of the points or lines may not be visible in all m cameras. Then, we define for every camera $i \quad \{1,...,m\}$ a set O_i of the points and lines observed in the camera i. Every Point-Line-Problem is defined by the numbers m, p, l and sets L, O. A calibrated setting is assumed.

The Point-Line-Problem is **minimal**, if a generic instance of the problem has a finite positive number of complex solutions.

2.1 Problem of five points in two views

The best known calibrated Point-Line Minimal Problem is the Five-Point problem. The code of the Five-Point problem according to [DKLP19] is 5000_2, the number of complex solutions to this problem is 20. The problem consists of projections of five points on two cameras. The task is to compute the relative pose between the cameras and the coordinates of the 3D points in the coordinate system of the first camera. No lines are assumed in the problem and the visibility is complete. First, we will give the description of

the Five-Point problem. Let us introduce the notation which we will use in this section.

X $\mathsf{R}^{3,5}$ Matrix whose rows represent five points in 3D.

(R_j, t_j), j $\{1, 2\}$ The poses of the cameras.

x $\mathsf{R}^{2,5}$ Matrix whose rows are 2D projections of the points onto the first camera.

y $\mathsf{R}^{2,5}$ Matrix whose rows are 2D projections of the points onto the second camera.

λ $\mathsf{R}^{2,5}$ Matrix of the depths of the points from X. Element $\lambda_{j,i}$ is the **depth** of the i-th point in the camera j. Let X_i^j be the coordinates of point $X_{:,i}$ in the coordinate system of camera j, i.e. $X_i^j = R_j X_{:,i} + t_j$. The depth $\lambda_{j,i}$ is the third coordinate of X_i^j.

Let us fix the first pose as $R_1 = I$, $t_1 = \vec{0}$. We know x, y. The task is to compute R_2, t_2, X, λ, such that there holds:

$$\lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} = R_1 \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix} + t_1 = \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix}, \quad i \quad \{1, ..., 5\} \tag{2.1}$$

$$\lambda_{i,2} \begin{bmatrix} y_{1,i} \\ y_{2,1} \\ 1 \end{bmatrix} = R_2 \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix} + t_2, \quad i \quad \{1, ..., 5\} \tag{2.2}$$

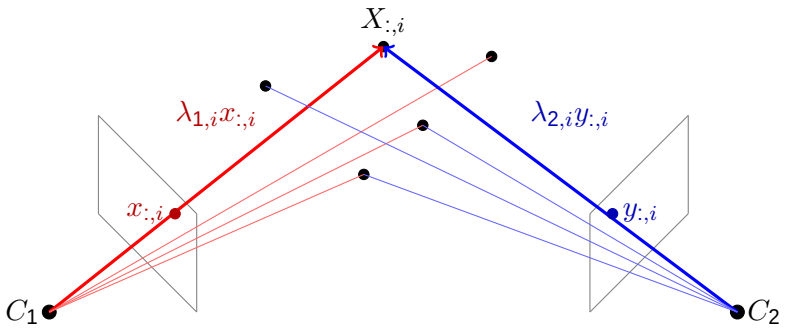


Figure 2.1: An example of the Five-Point problem. Five points X_i, i $\{1, ..., 5\}$ are projected onto two cameras C_1, C_2. The projections of the points are $x_{:,i}, y_{:,i}$, and the depths are $\lambda_{1,i}$ in the first view and $\lambda_{2,i}$ in the second view.

2.1.1 Nistér Solver

Now, we are going to briefly describe the solution to the Five-Point problem due to Nistér [Nis04]. Let us introduce the notation which we will use in this section.

x $\mathsf{R}^{2,5}$ Five calibrated points in the first camera.

y $\mathsf{R}^{2,5}$ Five calibrated points in the second camera.

(R, t) Relative pose between the cameras.

E $\mathsf{R}^{3,3}$ Essential matrix consistent with the points in x, y. There holds $E = [t]_\times R$

The Nistér solver finds the solution to the Five-Point problem by computing the essential matrix E which is consistent with the points x, y, i.e.:

$$\begin{bmatrix} y_{1,i} & y_{2,i} & 1 \end{bmatrix} E \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} = 0, \quad i \quad \{1, ..., 5\} \tag{2.3}$$

In addition to that, the essential matrix fulfills the following conditions:

$$\begin{aligned} \det E &= 0 \\ 2EE^T E - trace(EE^T)E &= 0 \end{aligned} \tag{2.4}$$

We know the points x, y. The goal is to find the essential matrix E, which would fulfill the equations (2.3), (2.4). Nistér solver solves this problem in two steps. First, a four-dimensional linear subspace of matrices fulfilling equation (2.3) is found. In the second step, the matrix from the linear subspace which fulfills equations 2.4 is found.

When the essential matrix is obtained by the Nistér solver, the Essential matrix can be decomposed into the rotation R and translation t according to [Paj21]. We obtain two possible rotations R_1, R_2 and two possible translations $\pm t$, out of which we select the pair of rotation and translation, for which all five points are triangulated in front of both cameras.

2.1.2 Depth formulation

There are different ways to convert the Five-Point problem to a set of polynomial equations. One of them is a depth formulation. In this formulation,

the parameters are the projections x, y of the points to the cameras and the unknowns are the depths $\lambda_{j,i}$, $i \quad \{1,...,5\}, j \quad \{1,2\}$. If the depths are known, we can easily compute the relative pose of the cameras (Sec. 4.7). Now, we are going to formulate the depth formulation of the Five-Point problem.

If the depths $\lambda_{j,i}$ correspond to the correct solution, then the distances between two points X_i, X_i are the same in both cameras for every pair of indices $i, i \quad \{1,...,5\}, i = i$. The situation is shown in Figure 2.2 The constraint can be written as the following set of polynomial equations:

$$\left\| \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} - \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} \right\|^2 = \left\| \lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} - \lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} \right\|^2 \tag{2.5}$$

$$i \quad \{1,...,5\}, i \quad \{1,...,5\}, i < i$$

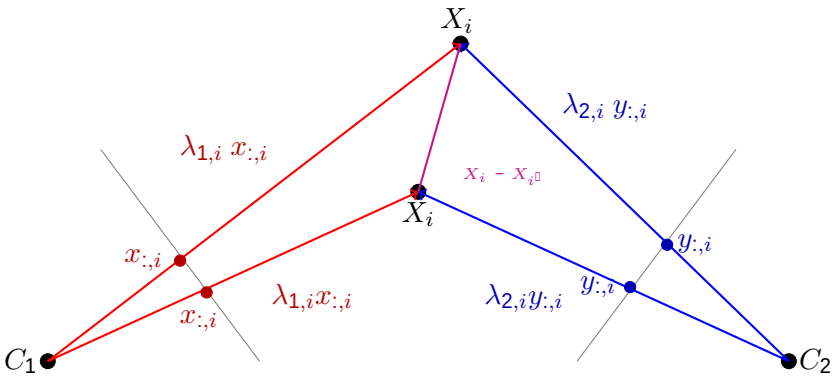


Figure 2.2: The illustration of the depth formulation of the five point problem. Two points X_i and $X_{i'}$ together with their projections are depicted in the image. If the depths λ are correct, the value $X_i - X_{i'}$ is equal to both $\lambda_{1,i} x_{:,i} - \lambda_{1,i'} x_{:,i'}$, and $\lambda_{2,i} y_{:,i} - \lambda_{2,i'} y_{:,i'}$. Therefore, these values are equal.

The problem is scale-invariant, i.e., if we multiply all correct depths $\lambda_{i,j}$ by the same coefficient $\alpha \quad \mathbb{R}$, we obtain another solution to the equations (2.5). Therefore, we have to fix the scale in order to have a set of equations with a finite number of solutions. This is done by fixing the depth of the first point in the first camera to unity: $\lambda_{1,1} = 1$. After that, the equations (2.5) have 9 variables.

2.1.3 Generator of the instances of Five-Point problem

Now, we will describe the way the instances of the Five-Point problem together with their solutions are generated. These generated solutions may be used as the starting points for the Homotopy Continuation Solver (Section 4.1.3) or for the training and evaluation of the solver (Section 4.6.3).

We generate the problems from the real 3D models from the ETH 3D dataset [1]. We use the datasets "courtyard", "meadow", and "pipes" for the training and "delivery_area" for the validataion of the solvers. We generate problems for every pair j, j of cameras in the dataset. Now, we will show how to generate one problem with its solution from one pair of cameras j, j. Let us introduce the notation which we will use in this section.

n		Number of points observed by both camera j and camera j.
(R_j, t_j)		Ground truth pose of camera j.
(R_j, t_j)		Ground truth pose of camera j.
(R, t)		Ground truth relative pose between cameras j, j. The relative rotation R is equal to $R = R_j (R_j)^T$, the relative translation is equal to $t = t_j - (R_j)^T t_j$.
X	$\mathsf{R}^{3,n}$	A matrix whose columns are n 3D points observed by both cameras.
x	$\mathsf{R}^{2,n}$	A matrix whose columns are correct 2D projections of the points from X onto the first camera j.
y	$\mathsf{R}^{2,n}$	A matrix whose columns are correct 2D projections of the points from X onto the second camera j.
ϵ^x	$\mathsf{R}^{2,n}$	Measurement errors of the observations x of points X by the first camera j.
ϵ^y	$\mathsf{R}^{2,n}$	Measurement errors of the observations y of points X by the second camera j.
x^ϵ	$\mathsf{R}^{2,n}$	Observations of points from X by the first camera. There holds $x^\epsilon = x + \epsilon^x$.
y^ϵ	$\mathsf{R}^{2,n}$	Observations of points from X by the second camera. There holds $y^\epsilon = y + \epsilon^y$.

[1] https://www.eth3d.net/datasets

I	$\{1,...,n\}^5$	A set of five indices of the selected points.
x	$\mathbb{R}^{2,5}$	A matrix whose columns are the columns from x^ϵ indexed by I.
y	$\mathbb{R}^{2,5}$	A matrix whose columns are the columns from y^ϵ indexed by I. The points are therefore in correspondence with points from x.
X	$\mathbb{R}^{3,5}$	A matrix whose columns are the 3D points triangulated from the points x, y in the coordinate system of the first camera.
λ	$\mathbb{R}^{2,5}$	A matrix of the depths of points from x, y. The depths are such that the values in λ are the solutions to the depth formulation of the Five-Point problem (2.5) parametrized by x, y, all depths are positive and the relative pose consistent with the depths is close to the ground truth relative pose $(R\ , t\ )$.

We know the observations x^ϵ, y^ϵ, and the ground truth poses (R_j, t_j), $(R_{j\ }, t_{j\ })$ of both cameras. The task is to find the samples x, y of five corresponding points from x^ϵ, y^ϵ and to compute the corresponding depths λ, such that the relative pose consistent with the depths is close to the ground truth relative pose $(R_{j,j\ }, t_{j,j\ })$. This procedure may be repeated to obtain a set of problem-solution pairs.

First, we sample five elements I from the set of indices $\{1,...,n\}$. We obtain the matrix x whose columns are the columns from x^ϵ indexed by I. Likewise, we obtain the matrix y whose columns are the columns from x^ϵ indexed by I. Then, we call the Nistér algorithm (Section 2.1.1) to obtain a set of $k \leq 10$ essential matrices $E_i, i \quad \{1,...,k\}$ between points x, y. We decompose each essential matrix E_i according to [NS06] to obtain two rotations R_1^i, R_2^i and two translations t^i, $-t^i$.

Out of these four potential combinations of rotation and translation for a given essential matrix E_i, we select the rotation R_i and translation t_i, for which all five 3D points X^i are triangulated in front of both cameras. Then, we select the relative pose $(R_{i\ }, t_{i\ }), i \quad \{1,...,k\}$ which minimizes the distance (6.3) from the ground truth relative pose $(R\ , t\ )$. Finally, we obtain the depths $\lambda_{1,a}, a \quad \{1,...,5\}$ in the first view from the last row of matrix X^i and the depths $\lambda_{2,a}, a \quad \{1,...,5\}$ in the second view from the last row of matrix $R_{i\ } X^i\ + t_i\ .$ Then, the depths λ are a solution to the depth formulation (2.5) parametrized by points x, y. The sampling procedure is described in Algorithm 1.

Algorithm 1: Five-Point Problem Generator

input : x^ϵ, y^ϵ, (R_j, t_j), (R_j, t_j)
output: x, y, λ
Find the ground truth relative pose (R, t);
Sample five elements I from $\{1, ..., n\}$;
$x :=$ points from x^ϵ indexed by I;
$y :=$ points from y^ϵ indexed by I;
Compute $k < 10$ essential matrices E_i between points x, y using Nistér algorithm (Section 2.1.1);
$i := -1$;
$dist := \infty$;
for $i \in \{1, ..., k\}$ **do**
  Decompose E_i into two rotations R_1^i, R_2^i and two translations $\pm t^i$;
  $(R_i, t_i) :=$ the relative pose for which all 5 3D points are triangulated in front of both cameras;
  $X^i :=$ points triangulated from points x, y and pose (R_i, t_i);
  $dist :=$ Distance (6.3) between (R_i, t_i) and (R, t);
  if $dist < dist$ **then**
    $i := i$;
    $dist := dist$;
  end
end
Set the first row of λ as the last row of X^i;
Set the second row of λ as the last row of $R_i X^i + t_i$;

2.2 Problem of four points in three views

Now, we are going to describe the problem of four points in three views. As the name suggests, we know the projections of four points into three cameras and the task is to compute the relative poses between these cameras. This problem, however, is overconstrained, i.e., a generic instance of the problem has no solution.

First, we will describe the problem, then (Section 2.2.1), we will relax the problem to a point-line minimal problem, which can be solved by homotopy continuation. We will describe the depth formulation of the minimal relaxed problem in Section 2.2.2. Finally, we will describe the generator of the instances of the relaxed minimal problem together with their solutions in Section 2.2.3. Let us introduce the notation which we will use in this section.

X $\mathsf{R}^{3,4}$ Matrix whose rows represent four points in 3D.

(R_j, t_j), $j \in \{1, 3\}$ The poses of the cameras.

x $\mathsf{R}^{2,4}$ Matrix whose rows are 2D projections of the points onto the first camera.

y $\mathsf{R}^{2,4}$ Matrix whose rows are 2D projections of the points onto the second camera.

w $\mathsf{R}^{2,4}$ Matrix whose rows are 2D projections of the points onto the third camera.

λ $\mathsf{R}^{3,5}$ Matrix of the depths of the points from X. Element $\lambda_{j,i}$ is the **depth** of the i-th point in the camera j. Let X_i^j be the coordinates of point $X_{:,i}$ in the coordinate system of camera j, i.e. $X_i^j = R_j X_{:,i} + t_j$. The depth $\lambda_{j,i}$ is the third coordinate of X_i^j.

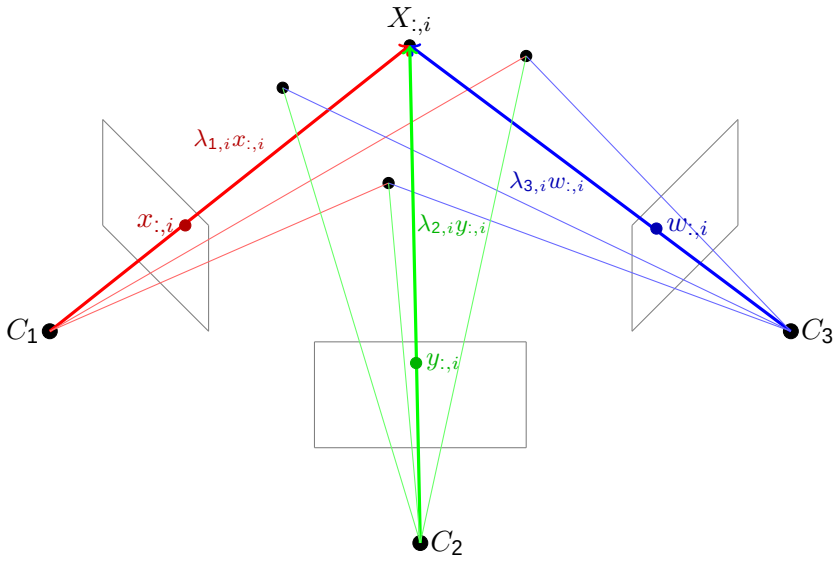


Figure 2.3: An example of the Four-Point problem. Four points $X_i, i \in \{1, ..., 4\}$ are projected onto three cameras C_1, C_2, C_3. The projections of the points are $x_{:,i}, y_{:,i}, w_{:,i}$, and the depths are $\lambda_{1,i}$ in the first view, $\lambda_{2,i}$ in the second view, and $\lambda_{3,i}$ in the third view.

Let us fix the first pose as $R_1 = I$, $t_1 = \vec{0}$. We know x, y. The task is to compute R_2, t_2, R_3, t_3, X, λ, such that there holds:

$$\lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} = R_1 \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix} + t_1 = \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix}, \quad i \quad \{1, ..., 4\} \tag{2.6}$$

$$\lambda_{i,2} \begin{bmatrix} y_{1,i} \\ y_{2,1} \\ 1 \end{bmatrix} = R_2 \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix} + t_2, \quad i \quad \{1, ..., 4\} \tag{2.7}$$

$$\lambda_{i,3} \begin{bmatrix} w_{1,i} \\ w_{2,1} \\ 1 \end{bmatrix} = R_3 \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix} + t_3, \quad i \quad \{1, ..., 4\} \tag{2.8}$$

2.2.1 Relaxation to a minimal problem

The system of equations (2.6), (2.7), (2.8) is overconstrained, therefore, its generic instance does not have any solution. Now, we will describe how this problem is relaxed to a partial visibility minimal problem from [DKLP20]. This problem consists of four points and one line incident to one of the points. The line is observed by one of the cameras, while the point incident to the line is observed by the other two cameras. The points which are not incident to the line are observed by all three cameras. This problem is minimal and has 272 complex solutions in a generic case. The problem is depicted in Fig. 2.4. Let us introduce the notation which we will use in this section.

X $\mathsf{R}^{3,4}$ Matrix whose rows represent four points in 3D.

(R_j, t_j), j $\{1, 3\}$ The poses of the cameras.

x $\mathsf{R}^{2,4}$ Matrix whose rows are 2D projections of X onto the first camera.

y $\mathsf{R}^{2,4}$ Matrix whose rows are 2D projections of X onto the second camera.

w $\mathsf{R}^{2,4}$ Matrix whose rows are 2D projections of X onto the third camera.

λ $\mathsf{R}^{3,5}$ Matrix of the depths of the pts. from X. $\lambda_{j,i}$ is the **depth** of the i-th point in cam. j. Let $X_i^j = R_j X_{:,i} + t_j$ be the coordinates of pt. $X_{:,i}$ in the coord. system of cam. j. Then, $\lambda_{j,i}$ is the third coord of X_i^j.

l R The oriented distance from the point $w_{:,A}$ to the true projection of the last point onto the last camera.

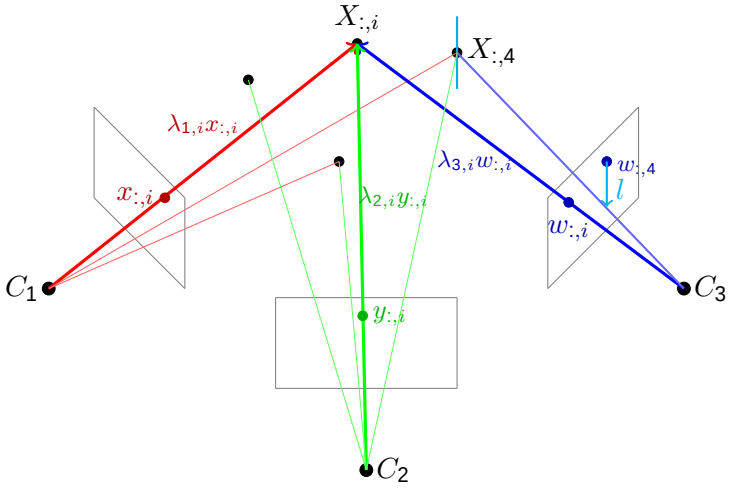


Figure 2.4: Relaxation of the Four-Point problem to a minimal problem. Four points $X_i, i \quad \{1, ..., 4\}$ are projected onto three cameras C_1, C_2, C_3. The projections of the points are $x_{:,i}, y_{:,i}, w_{:,i}$, and the depths are $\lambda_{1,i}$ in the first view, $\lambda_{2,i}$ in the second view, and $\lambda_{3,i}$ in the third view. The last point $X_{:,i}$ projects in the third camera to a point with the same x-coordinate as $w_{:,4}$, the oriented distance from $w_{:,4}$ to the projection is l.

The principle of the relaxation of the Four-Point problem to the minimal problem is that the last observation $w_{:,4}$ in the last view is relaxed to a vertical line passing through the point $w_{:,4}$. The last point $X_{:,4}$ is then projected to a point on this line. Apart from the variables λ, we introduce another variable l, which is the oriented distance from the point $w_{:,4}$ to the true projection of the last point $X_{:,4}$ to the last camera. The task is to compute R_2, t_2, R_3, t_3, λ, $l \quad \mathsf{R}$, such that there holds:

$$\lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} = R_1 \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix} + t_1 = \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix}, \quad i \quad \{1, ..., 4\} \tag{2.9}$$

$$\lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} = R_2 \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix} + t_2, \quad i \quad \{1, ..., 4\} \tag{2.10}$$

$$\lambda_{3,i} \begin{bmatrix} w_{1,i} \\ e_{2,i} \\ 1 \end{bmatrix} = R_3 \begin{bmatrix} X_{1,i} \\ X_{2,i} \\ X_{3,i} \end{bmatrix} + t_3, \quad i \quad \{1, ..., 3\} \tag{2.11}$$

$$\lambda_{3,4} \left(\begin{bmatrix} w_{1,4} \\ w_{2,4} \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ l \\ 0 \end{bmatrix} \right) = R_3 \begin{bmatrix} X_{1,4} \\ X_{2,4} \\ X_{3,4} \end{bmatrix} + t_3 \tag{2.12}$$

2.2.2 Depth formulation

Similarly to the Five-Point problem (Section 2.1.2), we convert the relaxed Four-Point problem (2.9), (2.10), (2.11), (2.12) to a set of polynomial equations using a depth formulation. In this formulation, the unknowns are the depths λ and the oriented distance l, the rotations and translations are eliminated. If the depths and the oriented distance are known, we can compute the relative poses according to Section 5.7.

If the depths λ and the oriented distance l correspond to the correct solution, then the distances between two points $X_{:,i}$, $X_{:,i}$ are the same in all three cameras for every pair of indices $i, i \quad \{1,...,4\}, i = i$. The situation, including the relaxed point $w_{:,4}$, is shown in Figure 2.5. The constraint can be written as the following system of polynomial equations:

$$\left\| \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} - \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} \right\|^2 = \left\| \lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} - \lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} \right\|^2$$
$$i \quad \{1,...,4\}, i \quad \{1,...,4\}, i < i$$

$$\left\| \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} - \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} \right\|^2 = \left\| \lambda_{3,i} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} - \lambda_{3,i} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} \right\|^2$$
$$i \quad \{1,...,3\}, i \quad \{1,...,3\}, i < i$$

$$\left\| \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} - \lambda_{1,4} \begin{bmatrix} x_{1,4} \\ x_{2,4} \\ 1 \end{bmatrix} \right\|^2 = \left\| \lambda_{3,i} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} - \lambda_{3,4} \left(\begin{bmatrix} w_{1,4} \\ w_{2,4} \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ l \\ 0 \end{bmatrix} \right) \right\|^2 \quad (2.13)$$
$$i \quad \{1,...,3\}$$

$$\left\| \lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} - \lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} \right\|^2 = \left\| \lambda_{3,i} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} - \lambda_{3,i} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} \right\|^2$$
$$i \quad \{1,...,3\}, i \quad \{1,...,3\}, i < i$$

$$\left\| \lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} - \lambda_{2,4} \begin{bmatrix} y_{1,4} \\ y_{2,4} \\ 1 \end{bmatrix} \right\|^2 = \left\| \lambda_{3,i} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} - \lambda_{3,4} \left(\begin{bmatrix} w_{1,4} \\ w_{2,4} \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ l \\ 0 \end{bmatrix} \right) \right\|^2$$
$$i \quad \{1,...,3\}$$

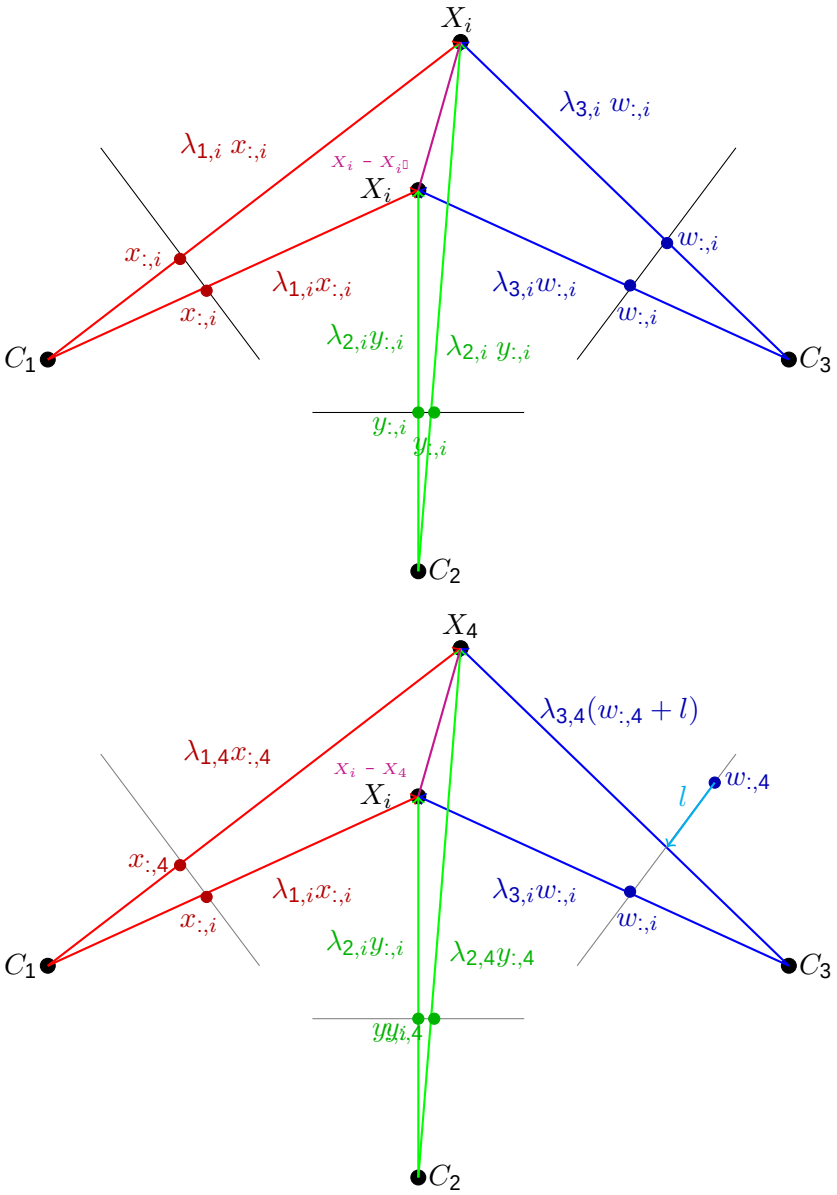


Figure 2.5: The illustration of the depth formulation of the four point problem. The upper image depicts the situation where $i < 4$, $i \ < 4$, while the lower image depicts the situation with the fourth point, which is projected onto a line in the third camera and the oriented distance from $w_{:,4}$ to the projection is l. If the depths λ are correct, the equations in (2.13) hold.

The problem is scale-invariant, i.e., if we multiply all correct depths λ by the same coefficient α R, we obtain another solution to the equations (2.13). Therefore, we fix the scale in order to obtain a system of equations with a finite number of solutions. Like in the case of the Five-Point problem, this is

done by fixing the depth of the first point in the first camera to one: $\lambda_{1,1} = 1$. After that, the equations (2.13) have 12 variables and a finite number of solutions.

2.2.3 Generator of problem-solution pairs of Four-Point problem

Now, we will describe how the instances of the relaxed Four-Point problem (Section 2.2.1) together with their solutions are generated. These generated solutions are supposed to be used as the starting points for the Homotopy Continuation Solver (Section 5). Therefore, we need to have the problems paired with the exact solution. The homotopy continuation is tracked after both the start and the final problems are invariantized (Sec 5.4). The rotation of the views, which is performed during the invariantization may change the solution to the equations (2.13). Therefore, we first invariantize the sampled points and then compute the depths of the invariantized points.

Like in the case of the Five-Point problem generator, we generate the problems from the real 3D models from the ETH 3D dataset [2]. We generates problems from every triplet j_1, j_2, j_3 of cameras in the model for which a set of 3D points observed by all three cameras exists. Now, we will show how to generate one instance of the relaxed problem described in Section 2.2.1 together with its solution. Let us introduce the notation which we will use in this section.

n Number of points observed by both camera j and camera j .

(R_1, t_1) Ground truth pose of camera j_1.

(R_2, t_2) Ground truth pose of camera j_2.

(R_3, t_3) Ground truth pose of camera j_3.

$(R_{1,2}, t_{1,2})$ Ground truth relative pose between cameras j_1, j_2. The relative rotation $R_{1,2}$ is equal to $R_{1,2} = R_2(R_1)^T$, the relative translation is equal to $t_{1,2} = t_2 - R_2(R_1)^T t_1$.

$(R_{1,3}, t_{1,3})$ Ground truth relative pose between cameras j_1, j_3. The relative rotation $R_{1,3}$ is equal to $R_{1,3} = R_3(R_1)^T$, the relative translation is equal to $t_{1,3} = t_3 - R_3(R_1)^T t_1$.

[2] https://www.eth3d.net/datasets

$(R_{2,3}, t_{2,3})$		Ground truth relative pose between cameras j_2, j_3. The relative rotation $R_{2,3}$ is equal to $R_{2,3} = R_3(R_2)^T$, the relative translation is equal to $t_{2,3} = t_3 - R_3(R_2)^T t_2$.
X	$\mathsf{R}^{3,n}$	A matrix whose columns are n 3D points observed by all three cameras.
x	$\mathsf{R}^{2,n}$	A matrix whose columns are correct 2D projections of the points from X onto the first camera j_1.
y	$\mathsf{R}^{2,n}$	A matrix whose columns are correct 2D projections of the points from X onto the second camera j_2.
w	$\mathsf{R}^{2,n}$	A matrix whose columns are correct 2D projections of the points from X onto the third camera j_3.
ϵ^x	$\mathsf{R}^{2,n}$	Measurement errors of the observations x of points X by the first camera j_1.
ϵ^y	$\mathsf{R}^{2,n}$	Measurement errors of the observations y of points X by the first camera j_2.
ϵ^w	$\mathsf{R}^{2,n}$	Measurement errors of the observations w of points X by the first camera j_3.
x^ϵ	$\mathsf{R}^{2,n}$	Observations of points from X by the first camera. There holds $x^\epsilon = x + \epsilon^x$.
y^ϵ	$\mathsf{R}^{2,n}$	Observations of points from X by the second camera. There holds $y^\epsilon = y + \epsilon^y$.
w^ϵ	$\mathsf{R}^{2,n}$	Observations of points from X by the third camera. There holds $w^\epsilon = w + \epsilon^w$.
I	$\{1, ..., n\}^5$	A set of five indices of the sampled points.
x^I	$\mathsf{R}^{2,5}$	A matrix whose columns are the columns from x^ϵ indexed by I.
y^I	$\mathsf{R}^{2,5}$	A matrix whose columns are the columns from y^ϵ indexed by I. The points are therefore in correspondence with points from x^I.
w^I	$\mathsf{R}^{2,5}$	A matrix whose columns are the columns from w^ϵ indexed by I. The points are therefore in correspondence with points from x^I.
$\bar{x}^I$	$\mathsf{R}^{2,5}$	Invariantized representation of five sampled points in the first view obtained by Section 5.4.
$\bar{y}^I$	$\mathsf{R}^{2,5}$	Invariantized representation of five sampled points in the second view obtained by Section 5.4.
$\bar{w}^I$	$\mathsf{R}^{2,5}$	Invariantized representation of five sampled points in the third view obtained by Section 5.4.

s $Sym(\{1,2,3\})$ The permutation of the views induced by the invariantization according to Section 5.4.

R_x^I $SO(3)$ 3D Rotation matrix induced by the invariantization, which transforms the homogeneous coordinates of the first view to the homogeneous coordinates of $\bar{x}$.

R_y^I $SO(3)$ 3D Rotation matrix induced by the invariantization, which transforms the homogeneous coordinates of the second view to the homogeneous coordinates of $\bar{y}$.

R_w^I $SO(3)$ 3D Rotation matrix induced by the invariantization, which transforms the homogeneous coordinates of the third view to the homogeneous coordinates of $\bar{w}$.

$(\bar{R}_{1,2}, \bar{t}_{1,2})$ Ground truth relative pose between invariantized views $j_{s(1)}$, $j_{s(2)}$.

$(\bar{R}_{1,3}, \bar{t}_{1,3})$ Ground truth relative pose between invariantized views $j_{s(1)}$, $j_{s(3)}$.

$(\bar{R}_{1,2}, \bar{t}_{1,2})$ Obtained relative pose between invariantized points $\bar{x}^I$, $\bar{y}^I$.

$(\bar{R}_{1,3}, \bar{t}_{1,3})$ Obtained relative pose between invariantized points $\bar{x}^I$, $\bar{w}^I$.

$\bar{X}$ $\mathsf{R}^{3,5}$ Matrix whose columns are 3D points triangulated using the points $\bar{x}^I$, $\bar{y}^I$ and the relative pose $(\bar{R}_{1,2}, \bar{t}_{1,2})$ between the views.

$\bar{x}$ $\mathsf{R}^{2,4}$ The output of the generator, the invariantized sample of four points in the first view.

$\bar{y}$ $\mathsf{R}^{2,4}$ The output of the generator, the invariantized sample of four points in the second view.

$\bar{w}$ $\mathsf{R}^{2,4}$ The output of the generator, the invariantized sample of four points in the third view.

$\bar{X}^w_{:,4}$ The coordinates of the last point $\bar{X}_{:,4}$ in the coordinate system of the third camera.

λ $\mathsf{R}^{3,5}$ A matrix of the depths of points from $\bar{x}, \bar{y}, \bar{w}$. The depths are such, that the values in λ are the solutions to the depth formulation of the Four-Point problem (2.13) parametrized by x, y, w, all depths are positive and the relatives pose consistent with the depths is close to the ground truth relative poses $(\bar{R}_{1,2}, \bar{t}_{1,2})$, $(\bar{R}_{1,3}, \bar{t}_{1,3})$.

l R The output of the generator, oriented distance from the observation $\bar{w}_{:,4}$ to the exact projection of the last point $\bar{X}_{:,4}$ onto the last invariantized camera $j_{s(3)}$.

There is no exact solver available for the relaxed Four-Point problem. Therefore, we are going to show how the problem-solution pairs of the relaxed

Four-Point problem are generated without an access to the solver. We know the observations x^ϵ, y^ϵ, w^ϵ, and the ground truth poses (R_1, t_1), (R_2, t_2), (R_3, t_3) of all three cameras. The task is to find the invariantized samples $\bar{x}$, $\bar{y}$, $\bar{w}$ of four corresponding points from x^ϵ, y^ϵ, w^ϵ and to compute the corresponding depths λ and the oriented distance l.
First, we sample **five** elements I from the set of indices $\{1, ..., n\}$ and obtain matrices x^I, y^I, w^I whose columns are the columns of matrices x^ϵ, y^ϵ, w^ϵ indexed by I. Then, we invariantize the points in x^I, y^I, w^I according to Section 5.4 to obtain $\bar{x}^I$, $\bar{y}^I$, $\bar{w}^I$. This invariantization moves the center of mass of the first four points to zero, permutes the views according to the distance to the furthest point, orders the first four points counterclockwise in the last view, and rotates the fourth view to the x-axis. During the invariantization, we also obtain the permutation s of the views and the rotation matrices R_x^I, R_y^I, R_w^I which transform the homogeneous coordinates of points x^I, y^I, w^I to the homogeneous coordinates of points $\bar{x}^I$, $\bar{y}^I$, $\bar{w}^I$.

We obtain the ground truth relative poses $(\bar{R}_{1,2}, \bar{t}_{1,2})$, $(\bar{R}_{1,3}, \bar{t}_{1,3})$ between the invariantized views in the following way. If the index of the new first view $s(1)$ is smaller than the index of the new second view $s(2)$, then the relative pose $(\bar{R}_{1,2}, \bar{t}_{1,2})$ between the views $\bar{x}^I$, $\bar{y}^I$ is obtained as:

$$\begin{aligned} \bar{R}_{1,2} &= R_y^I R_{s(1),s(2)} (R_x^I)^T \\ \bar{t}_{1,2} &= R_y^I t_{s(1),s(2)} \end{aligned} \tag{2.14}$$

If, on the other hand, the index $s(1)$ of the new first view is larger than the index $s(2)$ of the new second view, then the relative pose $(\bar{R}_{1,2}, \bar{t}_{1,2})$ is obtained as:

$$\begin{aligned} \bar{R}_{1,2} &= R_y^I (R_{s(2),s(1)})^T (R_x^I)^T \\ \bar{t}_{1,2} &= -R_y^I (R_{s(2),s(1)})^T t_{s(2),s(1)} \end{aligned} \tag{2.15}$$

Likewise, if the index $s(1)$ of the new first view is smaller than the index $s(3)$ of the new third view, then the relative pose $(\bar{R}_{1,3}, \bar{t}_{1,3})$ between the views $\bar{x}^I$, $\bar{w}^I$ is obtained as:

$$\begin{aligned} \bar{R}_{1,3} &= R_w^I R_{s(1),s(3)} (R_x^I)^T \\ \bar{t}_{1,3} &= R_w^I t_{s(1),s(3)} \end{aligned} \tag{2.16}$$

And if the index index $s(1)$ of the new first view is larger than the index $s(3)$, then relative pose $(\bar{R}_{1,3}, \bar{t}_{1,3})$ is computed as:

$$\begin{aligned} \bar{R}_{1,3} &= R_w^I (R_{s(3),s(1)})^T (R_x^I)^T \\ \bar{t}_{1,3} &= -R_w^I (R_{s(3),s(1)})^T t_{s(3),s(1)} \end{aligned} \tag{2.17}$$

The relative pose $(\bar{R}_{2,3}, \bar{t}_{2,3})$ between the new second and the new third view is obtained analogously.

Then, we compute the set of $k \leq 10$ essential matrices $E_i, i \quad \{1, ..., k\}$ between views $\bar{x}^I$, $\bar{y}^I$ using Nistér algorithm 2.1.1. We decompose each essential matrix E_i according to [Paj21] to obtain two rotation matrices R_1^i, R_2^i and two translations t^i, $-t_i$. Out of these four potential combinations of a rotation and a translation we select the rotation R_i and the translation t_i, for which all five 3D points $\bar{X}^i$ are triangulated in front of both cameras. Out of all relative poses $(R_i, t_i), i \quad \{1, ..., k\}$, we select the relative pose $(\bar{R}_{1,2}, \bar{t}_{1,2})$ which minimizes the distance (6.3) from the ground truth relative pose $(\bar{R}_{1,2}, \bar{t}_{1,2})$. 3D points $\bar{X}$ are the points in the coordinate system of the first camera which have been triangulated from the observations $\bar{x}^I$, $\bar{y}^I$ with the relative pose $(\bar{R}_{1,2}, \bar{t}_{1,2})$.

After that, we register the first three observations from $\bar{w}^I$ to the first three 3D points from $\bar{X}$ using PNP [Paj21]. This process gives us up to 3 real poses. $(\bar{R}_{1,3}, \bar{t}_{1,3})$ is the pose obtained by PNP which minimizes the distance (6.3) from the ground truth pose $(\bar{R}_{1,3}, \bar{t}_{1,3})$.

We obtain the resulting projections $\bar{x}$ by taking the first four columns of $\bar{x}^I$ and the projections $\bar{y}$ by taking the first four columns of $\bar{y}^I$. The first three columns of $\bar{w}$ are the first three columns of $\bar{w}^I$. The last column of $\bar{w}$ is obtained in the following way. First, the coordinates $X^w_{:,4}$ of the fourth 3D point $\bar{X}_{:,4}$ in the coordinate system of the last camera are obtained as:

$$\bar{X}^w_{:,4} = \bar{R}_{1,3} \begin{bmatrix} \bar{X}_{1,4} \\ \bar{X}_{2,4} \\ \bar{X}_{3,4} \end{bmatrix} + \bar{R}_{1,3} \tag{2.18}$$

Then, the fourth column of the observations $\bar{w}$ in the last view is obtained as:

$$\bar{w}_{:,4} = \begin{bmatrix} \frac{\bar{X}^w_{1,4}}{\bar{X}^w_{3,4}} \\ 0 \end{bmatrix} \tag{2.19}$$

And the oriented distance l from the projection of $\bar{X}_{:,4}$ to the last observation in $\bar{w}$ is obtained as:

$$l = \frac{\bar{X}^w_{2,4}}{\bar{X}^w_{3,4}} \tag{2.20}$$

Then, the fourth point $\bar{X}_{:,4}$ is projected to the third camera $s(3)$ to the point

$$\begin{bmatrix} \frac{\bar{X}^w_{1,4}}{\bar{X}^w_{3,4}} \\ \frac{\bar{X}^w_{2,4}}{\bar{X}^w_{3,4}} \end{bmatrix} = \begin{bmatrix} \frac{\bar{X}^w_{1,4}}{\bar{X}^w_{3,4}} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{\bar{X}^w_{2,4}}{\bar{X}^w_{3,4}} \end{bmatrix} = \bar{w}_{:,4} + \begin{bmatrix} 0 \\ l \end{bmatrix} \tag{2.21}$$

which corresponds to the equations (2.13) which define the depth formulation of the relaxed Four-Point problem. The depths $\lambda_{1,a}, a \quad \{1, ..., 4\}$ of the first 4 points from $\bar{X}$ in the first view are obtained from the last row of $\bar{X}$. The

depths $\lambda_{2,a}, a \quad \{1, ..., 4\}$ of the first 4 points from $\bar{X}$ in the second view are obtained from the last row of $\bar{R}_{1,2}\bar{X} + \bar{t}_{1,2}$ and the depths $\lambda_{3,a}, a \quad \{1, ..., 4\}$ of the first 4 points from $\bar{X}$ in the third view are obtained from the last row of $\bar{R}_{1,3}\bar{X} + \bar{t}_{1,3}$. The whole procedure of generating problem-solution pairs of the Four-Point problem is described in Algorithm 2.

Algorithm 2: Four-Point Problem-Solution Pair Generator

input : Sets of matched observations x^ϵ, y^ϵ, z^ϵ, Relative poses $(R_{1,2}, t_{1,2})$, $(R_{1,3}, t_{1,3})$, $(R_{2,3}, t_{2,3})$

output : Four invariantized sampled points in three views: $\bar{x}$, $\bar{y}$, $\bar{z}$, Depths λ, Oriented distance l

Find ground truth relative poses $(R_{1,2}, t_{1,2})$, $(R_{1,3}, t_{1,3})$, $(R_{2,3}, t_{2,3})$;
Sample five corresponding points $x^\prime$, $y^\prime$, $w^\prime$ from x^ϵ, y^ϵ, w^ϵ;
$(\bar{x}, \bar{y}, \bar{w})$:= Points x, y, w invariantized according to Section 5.4;
R^I_x, R^I_z, R^I_w := Rotations transforming x, y, w to $(\bar{x}, \bar{y}, \bar{w})$;
s := permutation of views induced by the invariantization;
Find the invariantized poses $(\bar{R}_{1,2}, \bar{t}_{1,2})$, $(\bar{R}_{1,3}, \bar{t}_{1,3})$ according to (2.14), (2.15), (2.16), (2.17);
Compute $k \leq 10$ essential matrices E_i between points $\bar{x}^\prime$, $\bar{y}^\prime$ using Nistér algorithm (Section 2.1.1);
$i := -1$, $dist := \infty$;
for $i \in \{1, ..., k\}$ **do**
  Decompose E_i into two rotations R^i_1, R^i_2 and two translations $\pm t^i$;
  (R_i, t_i) := the relative pose for which all 5 3D points are triangulated in front of both cameras;
  $\bar{X}^i$:= 3D points triangulated from points $\bar{x}^\prime, \bar{y}^\prime$ and pose (R_i, t_i);
  $dist$:= Distance (6.3) between (R_i, t_i) and $(\bar{R}_{1,2}, \bar{t}_{1,2})$;
  if $dist < dist$ **then**
    $i := i$, $dist := dist$;
  end
end
$(\bar{R}_{1,2}, \bar{t}_{1,2}) := (R_i, t_i)$, $\bar{X} := \bar{X}^i$;
Find $k \leq 3$ relative poses (R_i, t_i) of first three points from $\bar{w}^\prime$ towards $\bar{X}$ using PNP;
$i := -1$, $dist := \infty$;
for $i \in \{1, ..., k\}$ **do**
  $dist$:= Distance (6.3) between (R_i, t_i) and $(\bar{R}_{1,3}, \bar{t}_{1,3})$;
  if $dist < dist$ **then**
    $i := i$, $dist := dist$;
  end
end
$(\bar{R}_{1,3}, \bar{t}_{1,3}) := (R_i, t_i)$;
Set $\bar{x}, \bar{y}, \bar{w}$ as the first 3 columnss of $\bar{x}^\prime, \bar{y}^\prime, \bar{w}^\prime$;
Set the first row of λ as the first 4 entries in the last row of $\bar{X}$;
Set the second row of λ as the last row of $\bar{R}_{1,2}\bar{X} + \bar{t}_{1,2}$;
Set the third row of λ as the last row of $\bar{R}_{1,3}\bar{X} + \bar{t}_{1,3}$;
Set the last column of $\bar{w}$ and l according to (2.18), (2.19), (2.20);

2.2.4 Generator of testing and training problems of Four-Point problem

Now, we will describe how to generate the instances of the Four-Point which are used for the training and testing of the Four-Point solver (Chapter 5). The training and validation data should arise from a real distribution. The problems obtained in Section 2.2.3 are modified by reprojection of the last point into the last camera. Therefore, they do not reflect the distribution of the real data exactly. In order to obtain the labels used in the training of the classifier (Section 5.6.3), we compare the poses instead of the depths. Therefore, we do not need to know the correct depths of the training problems. Because of these reasons, we pair the points together with the ground truth relative poses. Let us introduce the notation which we will use in this section.

n		Number of points observed by both camera j and camera j .
(R_1, t_1)		Ground truth pose of camera j_1.
(R_2, t_2)		Ground truth pose of camera j_2.
(R_3, t_3)		Ground truth pose of camera j_3.
$(R_{1,2}, t_{1,2})$		Ground truth relative pose between cameras j_1, j_2. The relative rotation $R_{1,2}$ is equal to $R_{1,2} = R_2(R_1)^T$, the relative translation is equal to $t_{1,2} = t_2 - R_2(R_1)^T t_1$.
$(R_{1,3}, t_{1,3})$		Ground truth relative pose between cameras j_1, j_3. The relative rotation $R_{1,3}$ is equal to $R_{1,3} = R_3(R_1)^T$, the relative translation is equal to $t_{1,3} = t_3 - R_3(R_1)^T t_1$.
$(R_{2,3}, t_{2,3})$		Ground truth relative pose between cameras j_2, j_3. The relative rotation $R_{2,3}$ is equal to $R_{2,3} = R_3(R_2)^T$, the relative translation is equal to $t_{2,3} = t_3 - R_3(R_2)^T t_2$.
X	$\mathsf{R}^{3,n}$	A matrix whose columns are n 3D points observed by all three cameras.
x	$\mathsf{R}^{2,n}$	A matrix whose columns are correct 2D projections of the points from X onto the first camera j_1.
y	$\mathsf{R}^{2,n}$	A matrix whose columns are correct 2D projections of the points from X onto the second camera j_2.
w	$\mathsf{R}^{2,n}$	A matrix whose columns are correct 2D projections of the points from X onto the third camera j_3.

ϵ^x $\mathsf{R}^{2,n}$ Measurement errors of the observations x of points X by the first camera j_1.

ϵ^y $\mathsf{R}^{2,n}$ Measurement errors of the observations y of points X by the first camera j_2.

ϵ^w $\mathsf{R}^{2,n}$ Measurement errors of the observations w of points X by the first camera j_3.

x^ϵ $\mathsf{R}^{2,n}$ Observations of points from X by the first camera. There holds $x^\epsilon = x + \epsilon^x$.

y^ϵ $\mathsf{R}^{2,n}$ Observations of points from X by the second camera. There holds $y^\epsilon = y + \epsilon^y$.

w^ϵ $\mathsf{R}^{2,n}$ Observations of points from X by the third camera. There holds $w^\epsilon = w + \epsilon^w$.

I $\{1, ..., n\}^4$ A set of four indices of the sampled points.

x^I $\mathsf{R}^{2,4}$ A matrix whose columns are the columns from x^ϵ indexed by I.

y^I $\mathsf{R}^{2,4}$ A matrix whose columns are the columns from y^ϵ indexed by I. The points are therefore in correspondence with points from x^I.

w^I $\mathsf{R}^{2,4}$ A matrix whose columns are the columns from w^ϵ indexed by I. The points are therefore in correspondence with points from x^I.

$\bar{x}$ $\mathsf{R}^{2,4}$ Invariantized representation of four sampled points in the first view obtained by Section 5.4.

$\bar{y}$ $\mathsf{R}^{2,4}$ Invariantized representation of four sampled points in the second view obtained by Section 5.4.

$\bar{z}$ $\mathsf{R}^{2,4}$ Invariantized representation of four sampled points in the third view obtained by Section 5.4.

s $Sym(\{1, 2, 3\})$ The permutation of the views induced by the invariantization according to Section 5.4.

R_x^I $SO(3)$ 3D Rotation matrix induced by the invariantization, which transforms the homogeneous coordinates of the first view to the homogeneous coordinates of $\bar{x}$.

R_y^I $SO(3)$ 3D Rotation matrix induced by the invariantization, which transforms the homogeneous coordinates of the second view to the homogeneous coordinates of $\bar{y}$.

R_w^I $SO(3)$ 3D Rotation matrix induced by the invariantization, which transforms the homogeneous coordinates of the third view to the homogeneous coordinates of $\bar{w}$.

$(\bar{R}_{1,2}, \bar{t}_{1,2})$ Ground truth relative pose between invariantized views $j_{s(1)}$, $j_{s(2)}$.

$(\bar{R}_{1,3}, \bar{t}_{1,3})$ Ground truth relative pose between invariantized views $j_{s(1)}$, $j_{s(3)}$.

$(\bar{R}_{2,3}, \bar{t}_{2,3})$ Ground truth relative pose between invariantized views $j_{s(2)}$, $j_{s(3)}$.

We know the observations x^ϵ, y^ϵ, w^ϵ, and the ground truth poses (R_1, t_1), (R_2, t_2), (R_3, t_3) of all three cameras. The task is to find the invariantized samples $\bar{x}$, $\bar{y}$, $\bar{w}$ of four corresponding points from x^ϵ, y^ϵ, w^ϵ and the ground truth poses $(\bar{R}_{1,2}, \bar{t}_{1,2})$, $(\bar{R}_{1,3}, \bar{t}_{1,3})$, $(\bar{R}_{2,3}, \bar{t}_{2,3})$ between the invariantized views.

First, we sample four elements I from the set of indices $\{1, ..., n\}$ and obtain matrices x^I, y^I, w^I whose columns are the columns of matrices x^ϵ, y^ϵ, w^ϵ indexed by I. Then, we invariantize the points in x^I, y^I, w^I according to Section 5.4 to obtain $\bar{x}^I$, $\bar{y}^I$, $\bar{w}^I$. We also obtain the permutation s of the views and the rotation matrices R_x^I, R_y^I, R_w^I which transform the homogeneous coordinates of points x^I, y^I, w^I to the homogeneous coordinates of points $\bar{x}^I$, $\bar{y}^I$, $\bar{w}^I$.

Then, we obtain the ground truth relative poses $(\bar{R}_{1,2}, \bar{t}_{1,2})$, $(\bar{R}_{1,3}, \bar{t}_{1,3})$, $(\bar{R}_{2,3}, \bar{t}_{2,3})$ between the invariantized views according to equations (2.14), (2.15), (2.16), (2.17). The generator of the training and testing problems is described in Algorithm 3.

Algorithm 3: Four-Point Training Problem Generator

input : Sets of matched observations x^ϵ, y^ϵ, z^ϵ, Relative poses $(R_{1,2}, t_{1,2})$, $(R_{1,3}, t_{1,3})$, $(R_{2,3}, t_{2,3})$

output : Four invariantized sampled points in three views: $\bar{x}$, $\bar{y}$, $\bar{z}$, Ground truth poses $(\bar{R}_{1,2}, \bar{t}_{1,2})$, $(\bar{R}_{1,3}, \bar{t}_{1,3})$, $(\bar{R}_{2,3}, \bar{t}_{2,3})$ between the invariantized points

Find ground truth relative poses $(R_{1,2}, t_{1,2})$, $(R_{1,3}, t_{1,3})$, $(R_{2,3}, t_{2,3})$;
Sample four corresponding points x^I, y^I, w^I from x^ϵ, y^ϵ, w^ϵ;
$(\bar{x}, \bar{y}, \bar{w})$:= Points x, y, w invariantized according to Section 5.4;
R_x^I, R_z^I, R_w^I := Rotations transforming x, y, w to $\bar{x}, \bar{y}, \bar{w}$;
s := permutation of views induced by the invariantization;
Find the invariantized poses $(\bar{R}_{1,2}, \bar{t}_{1,2})$, $(\bar{R}_{1,3}, \bar{t}_{1,3})$, $(\bar{R}_{2,3}, \bar{t}_{2,3})$ according to (2.14), (2.15), (2.16), (2.17);

Chapter 3

Homotopy continuation

In this section, we will describe the **homotopy continuation** [SI05], [Mor09], [BSHW13]. It is a method from numerical algebraic geometry which can solve sets of polynomial equations by transforming the start problem with a known solution to the final problem whose solutions we want to obtain. Let us introduce the notation which we will use in this section.

$f(z)$ A starting system of s polynomials in s variables.

s The size of the system.

$g(z)$ A final system of s polynomials in s variables.

z_0 C A solution of the starting set of polynomials, $f(z_0) = 0$.

$H(z,t)$ A homotopy which transforms the starting set $f(z)$ to the final set $g(z)$.

t $0,1$ A "time" parameter of the homotopy.

$z(t)$ $,t$ $0,1$ A path of solutions for which there holds $H(z(t)\ ,t) = 0$.

$H_z(z,t)$ $\mathsf{C}^{s,s}$ Jacobian matrix of homotopy $H(z,t)$ w.r.t. parameters z.

$H_t(z,t)$ $\mathsf{C}^{s,1}$ Jacobian matrix of homotopy $H(z,t)$ w.r.t. parameter t.

We know the starting square set of polynomials $f(z)$ with its solutions z_0 and the final square set of polynomials $g(z)$. We assume that the Jacobian

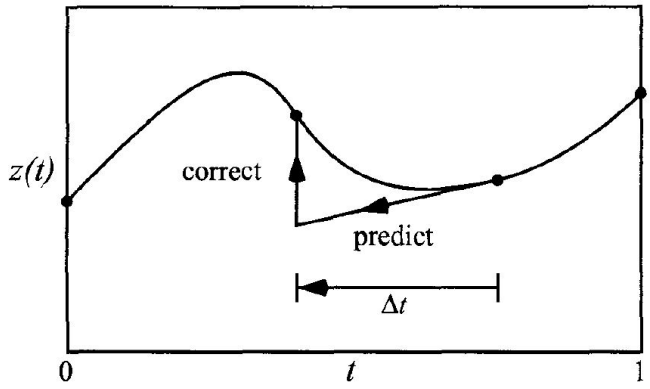


Figure 3.1: The illustration of the prediction and correction step of the homotopy continuation. The image is taken from [SI05].

matrix $H_z(z(t)\ ,t)$ is invertible for every $t \quad 0,1$. The task is to find a path of solutions $z(t)$ such that $z(0) = z_0$. Then, $z(1)$ is a solution to the final set of polynomials $g(z)$.

Let us set a homotopy:

$$H(z,t) = (1 - t)f(z) + tg(z) \tag{3.1}$$

If the parameter t is equal to zero, there holds $H(z,0) = f(z)$. If $t = 1$, then $H(z,1) = g(z)$.

Because there holds $H(z(t)\ ,t) = 0$ for every $t \quad 0,1$, the values of the homotopy over the track $z(t)$ are constant and the derivative of $H(z(t)\ ,t) = 0$ with respect to t is equal to zero, as well. We can use the chain rule to write:

$$\frac{dH(z(t)\ ,t)}{dt} = H_z(z(t)\ ,t)\frac{dz(t)}{dt} + H_t(z(t)\ ,t) = 0 \tag{3.2}$$

Differential equation (3.2) is called Davidenko equation [SI05].

The goal is to find a track $z(t)$ which satisfies the Davidenko equation (3.2). This is accomplished by alternating two steps, the prediction and the correction. In the prediction step, the correct value of $z(t)$ in time t is known and the task is to find the approximate value of $z(t + \Delta t)$ in time $t + \Delta t$. In the prediction step, we know the approximate value of $z(t)$ in time t and the task is to find the corrected value of $z(t)$ in the same time t. The homotopy continuation is summarized in Algorithm 4.

Algorithm 4: Homotopy Continuation

```
input  : Starting polynomial system f(z), Final polynomial system
         g(z), Starting solution z_0, Initial step size Δt_0, Maximal
         number of corrector steps k_max, Corrector tolerance ε,
         Number of successful steps for increase step succ_max,
         Increase step factor β, Δ_min Minimal step size
output : Final solution z_1
z := z_0, t := 0, Δt := Δt_0, succ := 0;
while t < 1 do
    Predict the solution z at time t + Δt according to Section 3.1;
    k := 0;
    while H(z, t + Δt) > ε and k < k_max do
        Correct the solution z according to Section 3.2;
        k := k + 1;
    end
    if H(z, t + Δt) ≤ ε then
        z := z, t := t + Δt, succ := succ + 1 ;          // Update
        if succ == succ_max then
            Δt := βΔt ;                          // Increase step size
            succ := 0;
        end
    else
        Δt := (1/β)Δt, succ := 0 ;                   // Decrease step size
        if Δt < Δ_min then
            return ;
        end
    end
end
return z ;
```

The parameters of the homotopy continuation are as follows:

Δt_0 R Initial step size of the homotopy continuation.

k_{max} N Maximal number of corrector steps.

ϵ R Corrector tolerance. If z is the output of the corrector and $H(z, t + \Delta t) \leq \epsilon$, the output of the corrector is accepted.

$succ_{max}$ N Number of consecutive successful steps after which the step size is increased.

β R Increase step factor. Step size Δt is increased to $\beta \Delta t$ after $succ_{max}$ consecutive successful steps, step size Δt is decreased to

$\frac{1}{\beta}\Delta t$ after one unsuccessful step.

Δ_{min} Minimal step size.

3.1 Predictor step

Now, we will describe the predictor step of the homotopy continuation. We know the correct value of $z(t)$ in time t. The task is to find an approximation $z(t+\Delta t)$ of the value $z(t+\Delta t)$. The value $z(t+\Delta t)$ is the solution of the differential equation (3.2). Therefore, we will obtain its approximation $z(t+\Delta t)$ with one step of a numerical method for solving differential equations.

One of the possibilities is to set up a Taylor polynomial of order 1 for the homotopy $H(z,t)$ as:

$$H(z(t) + \Delta z, t+\Delta t) \approx H(z(t), t) + H_z(z(t), t)\Delta z + H_t(z(t), t)\Delta t \quad (3.3)$$

We search for such a value of Δz that the Taylor polynomial for the time $t+\Delta t$ is equal to zero. Because we know that $H(z(t), t) = 0$, we can set up the equation as:

$$H_z(z(t), t)\Delta z + H_t(z(t), t)\Delta t = 0 \quad (3.4)$$

This is a simple linear equation, whose solution is:

$$\Delta z = -H_z(z(t), t)^{-1} H_t(z(t), t)\Delta t = 0 \quad (3.5)$$

This is a step of the Euler method. The approximation of $z(t+\Delta t)$ is obtained as:

$$z(t+\Delta t) = z(t) + \Delta z \quad (3.6)$$

Another possibility is to use a step of the Runge-Kutta method instead of the Euler method. Then, the change of variables Δz is obtained as:

$$\begin{aligned}
dz_1 &= -H_z(z(t), t)^{-1} H_t(z(t), t)\Delta t \\
dz_2 &= -H_z(z(t) + \frac{1}{2}dz_1, t+\frac{1}{2}\Delta t)^{-1} H_t(z(t) + \frac{1}{2}dz_1, t+\frac{1}{2}\Delta t)\Delta t \\
dz_3 &= -H_z(z(t) + \frac{1}{2}dz_2, t+\frac{1}{2}\Delta t)^{-1} H_t(z(t) + \frac{1}{2}dz_2, t+\frac{1}{2}\Delta t)\Delta t \\
dz_4 &= -H_z(z(t) + dz_3, t+\Delta t)^{-1} H_t(z(t) + dz_3, t+\Delta t)\Delta t \\
\Delta z &= \frac{1}{6}(dz_1 + dz_2 + dz_3 + dz_4)
\end{aligned} \quad (3.7)$$

The prediction $z(t + \Delta t)$ is then obtained with Equation (3.6).

One step of the Runge-Kutta method is typically slower then one step of the Euler method because Runge-Kutta method requires to solve 4 linear equations while Euler method requires only one. However, Runge-Kutta method is more precise than Euler, and therefore allows to use a larger step size Δt and computes the value of $z(1)$ using fewer steps than the Euler method. In practice, the Runge-Kutta method is faster and produces more accurate results.

3.2 Corrector step

Now, we will describe the corrector step of the homotopy continuation. We know an approximate value of $z(t)$ in time t which has been obtained during the predictor step. The task is to compute the corrected value $z(t)$ such that $H(z(t), t) = 0$. We assume that the approximate value $z(t)$ is close to the correct value $z(t)$.

One step of the solution is again based on the Taylor polynomial of order 1 for the homotopy $H(z, t)$:

$$H(z(t) + \Delta z, t + \Delta t) \approx H(z(t), t) + H_z(z(t), t)\Delta z + H_t(z(t), t)\Delta t \quad (3.8)$$

This time, the parameter t is fixed, therefore the value Δt is equal to zero. The goal is to compute a value of Δz such that the value of the Taylor polynomial is equal to zero. The equation can be set up as:

$$\Delta z = -H_z(z(t), t)^{-1} H(z(t), t) \quad (3.9)$$

This is a step of a Newton method. The value $z(t)$ is updated according to equation 3.6. The corrector steps are repeated until the norm of $H(z(t), t)$ is smaller than a threshold ϵ. If the threshold is not reached in k steps (typically $k = 3$), the corrector fails and the prediction is performed again with a smaller step size Δt.

3.3 Parametric homotopy continuation

If both the start and the final problems are parametrized sets of polynomial equations with the same shape, the method is called a **parametric homotopy continuation**. The parametric homotopy continuation has several desirable properties, such that for the generic case, the number of complex solutions of the start and final problems are the same. Let us introduce the notation which we will use in this section.

s The size of the system.

$h(z, p)$ A parametric system of s polynomials in s variables.

p_s Parameters of the start system. There holds $f(z) = h(z, p_s)$.

p_f Parameters of the final system. There holds $g(z) = h(z, p_f)$.

In this case, we can set up the homotopy as:

$$H(z, p, t) = (1 - t)h(z, p_s) + th(z, p_f) \tag{3.10}$$

This can be simplified as:

$$H(z, p, t) = h(z, (1 - t)p_s + tp_f) \tag{3.11}$$

3.4 Real homotopy continuation

The homotopy continuation is typically performed in the complex space even if only real solutions are required because even the tracks from a real solution to another real solution may pass through complex values [SI05]. However, in some cases, the track between two real solutions may lie entirely in real numbers. In these cases, tracking in real space may lead to an acceleration of the solver because the complex solutions are not tracked and because the computations in real numbers are faster than the computations in complex numbers.

The real homotopy $H(z, t)$ is set up according to Equation (3.1), just as the complex homotopy. In the case of the real homotopy continuation, however,

the coefficients of the sets of polynomials $f(z)$, $g(z)$, as well as the starting solution $z(0)$ are real. The derivatives of the homotopy $H_z(z,t)$, $H_t(z,t)$ are also real. Therefore, the computed steps Δz, predicted solutions $z(t)$ and corrected solutions $z(t)$ are real as well.

The complex homotopy continuation is guaranteed to successfully finish for a generic triplet of start system $f(z)$, start solution $z(0)$, and final system $g(z)$. No such guarantee exists for the real homotopy continuation, if no real track $z(t)$ starting z_0 in exists between the sets of polynomials $f(z)$, $g(z)$, the real homotopy continuation fails.

3.4.1 Possible results of the real homotopy continuation

For the start set of polynomials $f(z)$, final set of polynomials $g(z)$, starting solution z_0 to the system $f(z)$ and expected solution z_1 to the system z_1, there are three different possible results of the real homotopy continuation:

- **Correct result** The track finishes successfully and the result $z(1)$ is equal to the expected result z_1.
- **Incorrect result** The track finishes successfully, but the result $z(1)$ is different from the expected result z_1.
- **Failed track** The track fails and does not finish.

Chapter 4

Efficient Homotopy Continuation Solver for the Five-Point Problem

4.1 Description of the solver

In this section, we will propose a new solver for the Five-Point calibrated problem based on the real homotopy continuation. The solver is meant to be used in the RANSAC loop, therefore, it should be fast but it does not have to finish correctly for all inputs. Moreover, it is desirable not to track the inputs which consist of mismatched points. Therefore, we have decided to use the real homotopy continuation (Section 3.4) in the solver. We use the depth formulation of the Five-Point problem (Section 2.1.2).

The parametrization of the problem and the solution is described in Section 4.1.1. The homotopy continuation used in the solver is described in Section 4.1.2 and the solver itself is described in Section 4.1.3. Let us introduce the notation which we will use in this section.

x $\mathsf{R}^{2,5}$ Five calibrated 2D points in the first camera of the final problem.

y $\mathsf{R}^{2,5}$ Five calibrated 2D points in the second camera of the final problem.

λ $\mathsf{R}^{2,5}$ The depths of the points in both cameras. $\lambda_{j,i}$ is the depth of point i in camera j. λ are the solutions to the depth formulation of the Five-Point problem (Equation 2.5) parametrized by x, y.

$(R\ ,t\ )$ A relative pose between the first and the second camera.

m The number of starting problem-solution pairs.

$A = (\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)_{,}\ a\quad \{1, ..., m\}, \breve{x}_a\quad \mathsf{R}^{2,5}, \breve{y}_a\quad \mathsf{R}^{2,5}, \breve{\lambda}_a\quad \mathsf{R}^{2,5}$ A sequence of **anchors**, i.e. starting problem-solution pairs. For every a, the values in $\breve{\lambda}_a$ are the solutions to the depth formulation of the Five-Point problem (Equation 2.5) parametrized by $\breve{x}_a, \breve{y}_a$.

We know the calibrated points x, y in the first and the second cameras, as well as the number of anchors m and the anchors (problem-solution pairs) A. The task is to find the depths $\lambda\ $ and the relative pose $(R\ ,t\ )$, such that:

$$\lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} = \lambda_{2,i} R \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} + t \tag{4.1}$$

4.1.1 Parametrization of the problem and the solution

Now, we are going to describe the parametrizations of the problem and of the solution vector. Let us introduce the notation which we will use in this section.

$p_s\quad \mathsf{R}^{20}$ The parametrization of the initial problem.

$p_f\quad \mathsf{R}^{20}$ The parametrization of the final problem.

$h(z, p)$ A square parametric system of polynomials from the depth formulation of the Five-Point problem.

$z_0\quad \mathsf{R}^{9}$ The solution to the initial problem parametrized by p_s. There holds $h(z_0, p_s) = 0$.

The parametrization of the instance of a Five-Point problem is a twenty-dimensional vector p. This vector contains flattened points x, y which define the instance, i.e., the first five elements are the x-coordinates of the points in the first camera, the next five elements are the y-coordinates of the points in the first camera. The last ten elements are the coordinates of the points in the second camera in the same order. Let us have an instance $x\quad \mathsf{R}^{2,5}, y\quad \mathsf{R}^{2,5}$. The parametrization p of the instance is:

$$i\quad \{1, ..., 5\} : p_i = x_{1,i}, p_{i+5} = x_{2,i}, p_{i+10} = y_{1,i}, p_{i+15} = y_{2,i} \tag{4.2}$$

The depth formulation (2.5) of the Five-Point problem is scale-invariant, therefore, the scale of the depths λ has to be fixed in order for the system of equations (2.5) to have a finite number of solutions. We fix the depths by setting the first depth in the first image $\lambda_{1,1}$ to 1. The solution vector z R^9 is therefore:

$$i \quad \{1,...4\} : z_i = \frac{\lambda_{1,i+1}}{\lambda_{1,1}}; \quad i \quad \{1,...5\} : z_{i+4} = \frac{\lambda_{2,i}}{\lambda_{1,1}} \tag{4.3}$$

The depths λ $\mathsf{R}^{2,5}$ of the problem may be recovered from the solution vector z R^9 up to scale by setting $\lambda_{1,1}$ to one and by copying the values from the solution vector as:

$$i \quad \{1,...4\} : \lambda_{1,i+1} = z_i; \quad i \quad \{1,...5\} : \lambda_{2,i} = z_{i+4} \tag{4.4}$$

4.1.2 Homotopy continuation used in the solver

Now, we are going to describe the parametric homotopy $H(z,p,t)$ used to track the solution z of the depth formulation (2.5) of the Five-Point problem. The solution vector z has 9 elements but there are 10 equations in the depth formulation (2.5). Therefore, one of the equations from (2.5) has to be dropped in order to obtain a square system of independent equations. The square system has the following form:

$$\left\| \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} - \lambda_{i,1} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} \right\|^2 = \left\| \lambda_{i,2} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} - \lambda_{i,2} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} \right\|^2, \tag{4.5}$$
$$(i,i) \quad \{(1,2),(1,3),(1,4),(1,5),(2,3),(2,4),(2,5),(3,4),(3,5)\}$$

If the values of matrices x, y, λ are replaced by the corresponding values of the parametrization p and solution vector z according to (4.2) and (4.3), we obtain the square parametrized system of polynomial equations $h(z,p)$. Because the parametrization of the start problem is p_s and the parametrization of the final problem is p_f, we can set the parametric homotopy $H(z,p,t)$ according to:

$$H(z,p,t) = h(z,(1-t)p_s + tp_f) \tag{4.6}$$

The tracking of the homotopy $H(z,p,t)$ with the start parametrization z_0 is performed according to Algorithm 4. The derivatives $H_z(z(t),p,t)$ and $H_z(z(t),p,t)$ of the homotopy are obtained with a Straight-Line Program (SLP) generated in Macaulay2.

The largest amount of time on the solver is spent on the solution of the linear equations (3.9), (3.7), (3.5) which are solved as part of the predictor and the corrector. The left side of the equations is always the Jacobian matrix $H_z(z(t), p, t)$ of the value of the homotopy w.r.t. the variables z. Because the matrix $H_z(z(t), p, t)$ is relatively sparse, the closed form of the solution can be obtained, which can significantly reduce the time spent on one track. This closed-form solution is described in Section 4.2.

4.1.3 Overview of the solver

Let us introduce the notation which we will use in this section.

x $\mathsf{R}^{2,5}$ Five calibrated 2D points in the first camera of the final problem.

y $\mathsf{R}^{2,5}$ Five calibrated 2D points in the second camera of the final problem.

λ $\mathsf{R}^{2,5}$ The depths of the points in both cameras. $\lambda_{j,i}$ is the depth of point i in camera j. λ are the solutions to the depth formulation of the Five-Point problem (Equation 2.5) parametrized by x, y.

(R, t) A relative pose between the first and the second camera.

m The number of starting problem-solution pairs.

$A = (\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)$, a $\{1, ..., m\}$, $\breve{x}_a$ $\mathsf{R}^{2,5}$, $\breve{y}_a$ $\mathsf{R}^{2,5}$, $\breve{\lambda}_a$ $\mathsf{R}^{2,5}$ A sequence of **anchors**, i.e. starting problem-solution pairs. For every a, the values in $\breve{\lambda}_a$ are the solutions to the depth formulation of the Five-Point problem (Equation 2.5) parametrized by $\breve{x}_a, \breve{y}_a$.

$\bar{x}$ $\mathsf{R}^{2,5}$ Points in the first view invariantized according to Section 4.4.

$\bar{y}$ $\mathsf{R}^{2,5}$ Points in the second view invariantized according to Section 4.4.

D R^{14} Low-dimensional representation of the problem defined by $(\bar{x}, \bar{y})$. The low-dimensional representation is obtained according to Section 4.6.2.

L $\mathsf{A}^{14,20}$ A transformation matrix which transforms a 20-dimensional (invariantized or aligned) representation p to a 14-dimensional representation D.

a $\{0, ..., m\}$ Index of the selected anchor. If $a = 0$, the problem is not tracked, otherwise, it is tracked from anchor a.

$c : \mathsf{R}^{14} \to \{0, ..., m\}$ The classifier which for a low-dimensional representation D of a problem gives the index a of the selected anchor. The classifier is described in Section 4.6.4.

$\hat{x}$ $\mathsf{R}^{2,5}$ Points in the first view aligned to the selected anchor a according to Section 4.5.

$\hat{y}$ $\mathsf{R}^{2,5}$ Points in the second view aligned to the selected anchor a according to Section 4.5.

R_x^I $SO(3), R_y^I$ $SO(3)$:= Rotation matrices induced by the invariantization according to Section 4.5.

s $Sym\{1,2\}$ Permutation of the views induced by the invariantization (Section 4.5).

$H(z,p,t)$ Parametric homotopy (4.6) for the square system (4.5). z R^9 is the parametrization of the solution, p R^{20} is the parametrization of the problem and t $< 0, 1 >$ is the time parameter.

p_s R^{20} A parametrization (4.2) of the starting problem $\breve{x}_a$, $\breve{y}_a$

z_0 R^9 A parametrization (4.3) of the starting solution $\breve{\lambda}_a$

p_f R^{20} A parametrization (4.2) of the aligned final problem $\hat{x}$, $\hat{y}$

$\hat{\lambda}$ The depths of the aligned problem defined by $\hat{x}$, $\hat{y}$.

Now, we are going to describe the solver. The structure of the homotopy continuation procedure used in the solver is inspired by the MINUS solver [FDF+20], however, our solver is much faster because it operates in real numbers only, because it tracks only one solution per problem and because the LU-decomposition is replaced by a closed-form solution of the linear equations, which is described in Section 4.2. Before the solver can be used, the set of anchors $\mathcal{A}$ (starting problem-solution pairs) has to be generated and the classifier c has to be trained. The generating of the anchors is described in Section 4.6.1, the training of the classifier c is described in Section 4.6.4.

First, the invariantized representation $\bar{x}, \bar{y}$ of points x, y is obtained (Section 4.4) in order to have a unified representation of the problems, which can simplify the training of the anchor selector and the alignment. Then, one anchor a $\{1, ..., m\}$ is selected (Section 4.6). The aligned points $\hat{x}, \hat{y}$ are obtained by permutation and rotation of the invariantized points $\bar{x}, \bar{y}$ in order to minimize the Euclidean distance from the selected anchor $\breve{x}_a$, $\breve{y}_a$ (Section 4.5), and therefore, to increase the probability of the successful track from the selected anchor.

After that, the parametrizations p_s of the start problem $\breve{x}_a$, $\breve{y}_a$ and p_f of the aligned final problem $\hat{x}, \hat{y}$ are obtained by (4.2) and the parametrization z_0 of the start solution $\breve{\lambda}_a$ is obtained by (4.3). Then, the parametric homotopy

$H(z, p, t)$ from Equation (4.6) is set up and tracked from z_0 using Algorithm 4 with parameters from Section 4.3. If the track is successful, the solution $\hat{\lambda}$ is obtained from the final solution $z(1)$ by (4.4) and the relative pose (R, t) is obtained by the procedure described in Section 4.7. The solver is described in Algorithm 5.

Algorithm 5: Five-Point solver

input : Five points x, y in two views, Anchors A obtained according to Section 4.6.1, Transformation matrix L (Section 4.6.2), classifier c trained by Section 4.6.4

output : Relative pose (R, t) between x, y

`/* Invariantize the problem */`

$\bar{x}, \bar{y}$:= Invariantized representation of x, y obtained by Section 4.4;
R_x^I, R_y^I := Rotation matrices induced by the invariantization;
s := Permutation of the views induced by the invariantization;

`/* Select the starting anchor */`

D := Low dimensional representation of x, y (by Section 4.6.2, use L);
a := $c(D)$; `// Select the starting point`
if $a = 0$ **then**
  return
end

`/* Align the problem */`

$\hat{x}, \hat{y}$:= Points from x, y aligned to anchor $\breve{x}_a, \breve{y}_a$ by Section 4.5;
R_x, R_y := Rotation matrices induced by the alignment;

`/* Parametrize the problems and the starting solution */`

p_s := parametrization of $\breve{x}_a, \breve{y}_a$ by (4.2);
p_f := parametrization of $\hat{x}, \hat{y}$ by (4.2);
z_0 := parametrization of $\breve{\lambda}_a$ by (4.3);

`/* TRACK the solution */`

z_1 := Track homotopy (4.6) from z_0 by Algorithm 4 (Chapter 3);

`/* Extract the depths and the relative pose */`

if $z_1 =$ **then**
  $\hat{\lambda}$:= Extract depths from z_1 by (4.4);
  (R, t) := Get relative pose from z_1, R_x, R_y, R_x^I, R_y^I by Sec. 4.7;
end

4.2 Efficient evaluation of the predictor and the corrector

We have noticed that the largest amount of time of the MINUS solver [FDF+20] is spent on the LU-decomposition, which is used to solve linear equations (3.7), (3.9) which arise in the predictor and the corrector steps of the homotopy continuation. We have therefore replaced the LU-decomposition with the closed-form solution of the linear equations, which exploits the sparsity of the linear equations in the Multi-View Geometry problems. The average time for one track is $26.6\mu s$ if the method proposed in this section is used. If the LU decomposition in real domain is used, the average time for one track is $124.1\mu s$. If the LU decomposition in complex domain is used, the average time for one track is around $400\mu s$. The number of successful tracks is the same for both methods.

4.2.1 Structure of the Linear Equations in the Two-View Problem

The linear equations (3.7), (3.9) are computed in order to perform the predictor and corrector steps of the homotopy continuation. The linear equations have the form $Ax = b$ where the matrix A is always the derivative $H_z(z, p, t)$ of the homotopy $H(z, p, t)$ (4.6) w.r.t. the solution parameters z. Irrespective of the parameters z, p, t, the matrix $H_z(z, p, t)$ has the following form:

$$\begin{bmatrix} A_{0,0} & 0 & 0 & 0 & A_{0,4} & A_{0,5} & 0 & 0 & 0 \\ 0 & A_{1,1} & 0 & 0 & A_{1,4} & 0 & A_{1,6} & 0 & 0 \\ A_{2,0} & A_{2,1} & 0 & A_{2,3} & 0 & 0 & A_{2,6} & 0 & 0 \\ 0 & 0 & A_{3,2} & 0 & A_{3,4} & 0 & 0 & A_{3,7} & 0 \\ A_{4,0} & 0 & A_{4,2} & 0 & 0 & A_{4,5} & 0 & A_{4,7} & 0 \\ 0 & A_{5,1} & A_{5,2} & 0 & 0 & 0 & A_{5,6} & A_{5,7} & 0 \\ 0 & 0 & 0 & A_{6,3} & A_{6,4} & 0 & 0 & 0 & A_{6,8} \\ A_{7,0} & 0 & 0 & A_{7,3} & 0 & A_{7,5} & 0 & 0 & A_{7,8} \\ 0 & A_{8,1} & 0 & A_{8,3} & 0 & 0 & A_{8,6} & 0 & A_{8,8} \end{bmatrix} \tag{4.7}$$

The vector x of the variables can be written as:

$$\begin{bmatrix} x_0 & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \end{bmatrix}^T \tag{4.8}$$

The vector b of the right side can be written as:

$$\begin{bmatrix} b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7 & b_8 \end{bmatrix}^T \tag{4.9}$$

4.2.2 Closed-Form Solution of the Linear Equations in the Two-View Problem

Let us have a system of linear equations $Ax = b$ whose matrix A has the form of (4.7), vector x has the form of (4.8) and vector b has the form of (4.9). Now, we are going to show the closed-form solution to this system.

First, we have noticed that the rows $0, 1, 3, 6$ of matrix (4.7) have only three nonzero entries. We can easily obtain the variables x_1, x_2, x_3, x_5 from these equations and substitute them to the rest of the equations to obtain a square system with 5 variables. The variables x_1, x_2, x_3, x_5 are obtained as:

$$\begin{aligned} x_1 &= \frac{b_1 - A_{1,4}x_4 - A_{1,6}x_6}{A_{1,1}} \\ x_2 &= \frac{b_3 - A_{3,4}x_4 - A_{3,7}x_7}{A_{3,2}} \\ x_3 &= \frac{b_6 - A_{6,4}x_4 - A_{6,8}x_8}{A_{6,3}} \\ x_5 &= \frac{b_0 - A_{0,0}x_0 - A_{0,4}x_4}{A_{0,5}} \end{aligned} \tag{4.10}$$

After substituting of these into the rest of the equations, we obtain:

$$\begin{aligned} &A_{2,0}x_0 + \frac{A_{2,1}}{A_{1,1}}(b_1 - A_{1,4}x_4 - A_{1,6}x_6) + \frac{A_{2,5}}{A_{0,5}}(b_0 - A_{0,0}x_0 - A_{0,4}x_4) + A_{2,6}x_6 = b_2 \\ &A_{4,0}x_0 + \frac{A_{4,2}}{A_{3,2}}(b_3 - A_{3,4}x_4 - A_{3,7}x_7) + \frac{A_{4,5}}{A_{0,5}}(b_0 - A_{0,0}x_0 - A_{0,4}x_4) + A_{4,7}x_7 = b_4 \\ &\frac{A_{5,1}}{A_{1,1}}(b_1 - A_{1,4}x_4 - A_{1,6}x_6) + \frac{A_{5,2}}{A_{3,2}}(b_3 - A_{3,4}x_4 - A_{3,7}x_7) + A_{5,6}x_6 + A_{5,7}x_7 = b_5 \\ &A_{7,0}x_0 + \frac{A_{7,3}}{A_{6,3}}(b_6 - A_{6,4}x_4 - A_{6,8}x_8) + \frac{A_{7,5}}{A_{0,5}}(b_0 - A_{0,0}x_0 - A_{0,4}x_4) + A_{7,8}x_8 = b_7 \\ &\frac{A_{8,1}}{A_{1,1}}(b_1 - A_{1,4}x_4 - A_{1,6}x_6) + \frac{A_{8,3}}{A_{6,3}}(b_6 - A_{6,4}x_4 - A_{6,8}x_8) + A_{8,6}x_6 + A_{8,8}x_8 = b_8 \end{aligned} \tag{4.11}$$

These five equations can be rewritten as:

$$\begin{aligned}
\left(A_{2,0} - \frac{A_{2,5}}{A_{0,5}}A_{0,0}\right)x_0 - \left(\frac{A_{2,1}}{A_{1,1}}A_{1,4} + \frac{A_{2,5}}{A_{0,5}}A_{0,4}\right)x_4 + \left(A_{2,6} - \frac{A_{2,1}}{A_{1,1}}A_{1,6}\right)x_6 \\
= b_2 - \frac{A_{2,1}}{A_{1,1}}b_1 - \frac{A_{2,5}}{A_{0,5}}b_0 \\
\left(A_{4,0} - \frac{A_{4,5}}{A_{0,5}}A_{0,0}\right)x_0 - \left(\frac{A_{4,2}}{A_{3,2}}A_{3,4} + \frac{A_{4,5}}{A_{0,5}}A_{0,4}\right)x_4 + \left(A_{4,7} - \frac{A_{4,2}}{A_{3,2}}A_{3,7}\right)x_7 \\
= b_4 - \frac{A_{4,2}}{A_{3,2}}b_3 - \frac{A_{4,5}}{A_{0,5}}b_0 \\
\left(-\frac{A_{5,1}}{A_{1,1}}A_{1,4} - \frac{A_{5,2}}{A_{3,2}}A_{3,4}\right)x_4 + \left(A_{5,6} - \frac{A_{5,1}}{A_{1,1}}A_{1,6}\right)x_6 + \left(A_{5,7} - \frac{A_{5,2}}{A_{3,2}}A_{3,7}\right)x_7 \\
= b_5 - \frac{A_{5,1}}{A_{1,1}}b_1 - \frac{A_{5,2}}{A_{3,2}}b_3 \\
\left(A_{7,0} - \frac{A_{7,5}}{A_{0,5}}A_{0,0}\right)x_0 - \left(\frac{A_{7,3}}{A_{6,3}}A_{6,4} + \frac{A_{7,5}}{A_{0,5}}A_{0,4}\right)x_4 + \left(A_{7,8} - \frac{A_{7,3}}{A_{6,3}}A_{6,8}\right)x_8 \\
= b_7 - \frac{A_{7,3}}{A_{6,3}}b_6 - \frac{A_{7,5}}{A_{0,5}}b_0 \\
\left(-\frac{A_{8,1}}{A_{1,1}}A_{1,4} - \frac{A_{8,3}}{A_{6,3}}A_{6,4}\right)x_4 + \left(A_{8,6} - \frac{A_{8,1}}{A_{1,1}}A_{1,6}\right)x_6 + \left(A_{8,8} - \frac{A_{8,3}}{A_{6,3}}A_{6,8}\right)x_8 \\
= b_8 - \frac{A_{8,1}}{A_{1,1}}b_1 - \frac{A_{8,3}}{A_{6,3}}b_6
\end{aligned} \tag{4.12}$$

We can simplify these equations by replacing the coefficients by $C_{i,j}$ and the right sides by d_i, where $C_{i,j}$ is the coefficient in equation i at x_j and d_i is the right side of equation i. We obtain:

$$\begin{aligned}
C_{0,0}x_0 - C_{0,4}x_4 + C_{0,6}x_6 &= d_0 \\
C_{1,0}x_0 - C_{1,4}x_4 + C_{1,7}x_7 &= d_1 \\
C_{2,4}x_4 + C_{2,6}x_6 + C_{2,7}x_7 &= d_2 \\
C_{3,0}x_0 - C_{3,4}x_4 + C_{3,8}x_8 &= d_3 \\
C_{4,4}x_4 + C_{4,6}x_6 + C_{4,8}x_8 &= d_4
\end{aligned} \tag{4.13}$$

Notice that every equation in this system has only three variables. We can express the variables x_6, x_7, x_8 from the equations 0, 1, 3 and substitute the results to the remaining two equations to obtain a system of two equations with two variables, which can be easily solved. The variables x_6, x_7, x_8 are

obtained as:

$$\begin{aligned} x_6 &= \frac{d_0 - C_{0,0}x_0 + C_{0,4}x_4}{C_{0,6}} \\ x_7 &= \frac{d_1 - C_{1,0}x_0 + C_{1,4}x_4}{C_{1,7}} \\ x_8 &= \frac{d_3 - C_{3,0}x_0 + C_{3,4}x_4}{C_{3,8}} \end{aligned} \tag{4.14}$$

After substituting these into the two remaining equations, we obtain:

$$\begin{aligned} C_{2,4}x_4 + \frac{C_{2,6}}{C_{0,6}}(d_0 - C_{0,0}x_0 + C_{0,4}x_4) + \frac{C_{2,7}}{C_{1,7}}(d_1 - C_{1,0}x_0 + C_{1,4}x_4) = d_2 \\ C_{4,4}x_4 + \frac{C_{4,6}}{C_{0,6}}(d_0 - C_{0,0}x_0 + C_{0,4}x_4) + \frac{C_{4,8}}{C_{3,8}}(d_3 - C_{3,0}x_0 + C_{3,4}x_4) = d_4 \end{aligned} \tag{4.15}$$

These equations can be rewritten as:

$$\begin{aligned} -\left(\frac{C_{2,6}}{C_{0,6}}C_{0,0} + \frac{C_{2,7}}{1,7}C_{1,0}\right)x_0 + \left(C_{2,4} + \frac{C_{2,6}}{C_{0,6}}C_{0,4} + \frac{C_{2,7}}{C_{1,7}}C_{1,4}\right)x_4 \\ = d_2 - \frac{C_{2,6}}{C_{0,6}}d_0 - \frac{C_{2,7}}{C_{1,7}}d_1 \\ -\left(\frac{C_{4,6}}{C_{0,6}}C_{0,0} + \frac{C_{4,8}}{3,8}C_{3,0}\right)x_0 + \left(C_{4,4} + \frac{C_{4,6}}{C_{0,6}}C_{0,4} + \frac{C_{4,8}}{C_{3,8}}C_{3,4}\right)x_4 \\ = d_4 - \frac{C_{4,6}}{C_{0,6}}d_0 - \frac{C_{4,8}}{C_{3,8}}d_3 \end{aligned} \tag{4.16}$$

We can simplify these two equations by replacing the coefficients by $E_{i,j}$ and the right sides with f_i, such, that $E_{i,j}$ is the coefficient in equation i at x_j and f_i is the right side of equation i. We obtain:

$$\begin{aligned} -E_{0,0}x_0 + E_{0,4}x_4 = f_0 \\ -E_{1,0}x_0 + E_{1,4}x_4 = f_1 \end{aligned} \tag{4.17}$$

We can express the variable x_4 from the first equation of (4.17) as:

$$x_4 = \frac{f_0 + E_{0,0}x_0}{E_{0,4}} \tag{4.18}$$

Finally, we can substitute this to the second equation from (4.17) to obtain:

$$-E_{1,0}x_0 + \frac{E_{1,4}}{E_{0,4}}(f_0 + E_{0,0}x_0) = f_1 \tag{4.19}$$

Now, we can easily obtain the variable x_0 from (4.19) as:

$$x_0 = \frac{f_1 - \frac{E_{1,4}}{E_{0,4}}f_0}{\frac{E_{1,4}}{E_{0,4}}E_{0,0} - E_{1,0}} \tag{4.20}$$

The full evaluation of the vector x (4.8) goes as follows: first, the coefficients $C_{i,j}$ and right sides d_i of the 5×5 system (4.13) are evaluated. Then, the coefficients $E_{i,j}$ and right sides f_i of the 2×2 system (4.17) are evaluated. After that, the variable x_0 is obtained according to (4.20). When the value of x_0 is known, we can gradually evaluate the variables x_4 from (4.18), x_6, x_7, x_8 from (4.14) and x_1, x_2, x_3, x_5 from (4.10).

4.3 Optimization of the homotopy continuation parameters

Now, we are going to show the parameters of the homotopy continuation described in Chapter 3. We have set the parameters empirically in order to increase the number of correct tracks and decrease the time for one track. The parameters are set as follows:

- $\Delta t_0 = 0.05$
- $k_{max} = 3$
- $\epsilon = 1e - 5$
- $succ_{max} = 4$
- $\beta = 3$
- $\Delta_{min} = 1e - 4$

4.4 Invariantization of the problems

Now, we are going to describe the invariantization of the problems. The purpose of the invariantization is to obtain a unified representation of the problem and to prepare the problem for the alignment 4.5 which may contain fewer steps and take a shorter time if the problem is properly invariantized before the alignment. If the depths $\lambda \quad \mathsf{R}^{2,5}$ are known, the invariantization changes them as well. Let us introduce the notation which we will use in this section.

$x \quad \mathsf{R}^{2,5}$ Five calibrated 2D points in the first camera.

y	$\mathsf{R}^{2,5}$	Five calibrated 2D points in the second camera.
λ	$\mathsf{R}^{2,5}$	The depths of the points in both cameras.
$\bar{x}$	$\mathsf{R}^{2,5}$	Invariantized representation of points x.
$\bar{y}$	$\mathsf{R}^{2,5}$	Invariantized representation of points y.
$\bar{\lambda}$	$\mathsf{R}^{2,5}$	Depths consistent with the invariantized points $\bar{x}$, $\bar{y}$.
R_x^I	$SO(3)$	A 3D rotation matrix which transforms the homogeneous coordinates of x to the homogeneous coordinates of $\bar{x}$.
R_y^I	$SO(3)$	A 3D rotation matrix which transforms the homogeneous coordinates of y to the homogeneous coordinates of $\bar{y}$.
s	$Sym(\{1,2\})$	The permutation of the views. $s = \{2,1\}$ if the views have been swapped, $s = \{1,2\}$ otherwise.

We know the points x, y, and optionally also the depths λ. The task is to obtain invariantized points $\bar{x}$, $\bar{y}$, such that the center of mass of the points in every view is in zero, the point farthest from the center of mass lies on y-axis, and the rest of the points are ordered counterclockwise in the first view. The point farthest from the center of mass should be more distant from the center in the first view. Further, the task is to obtain the rotation matrices R_x^I, R_y^I, and the permutation s of the views. If the depths λ are known (such as during the invariantization of an anchor), the part of the task is also to find new depths $\bar{\lambda}$ consistent with the invariantized points $\bar{x}$, $\bar{y}$. The invariantization is described in Algorithm 6 and depicted in Figure 4.1. The transformation of the depths is described in Section 4.4.4.

Algorithm 6: Invariantize Five-Point problem

input : Five points x, y in two views
output : Invariantized representation $\bar{x}, \bar{y}$ of x, y, Rotation matrices R_x^I, R_y^I, Permutations s, r
$x^{h,\mu}, y^{h,\mu}$:= Rotate x, y such that the center of mass of the points is in zero (Section 4.4.1);
R_x^μ, R_y^μ := Rotation matrices which rotate the center of mass to zero;
$x^{h,\mu}, y^{h,\mu}$:= Permute the points, views and rotation matrices R_x^I, R_x^I according to Section 4.4.2;
s := permutation of the views, r := permutation of the points;
$\bar{x}, \bar{y}$:= Rotate the first point to y-axis (Section 4.4.3);
R_x^a, R_y^a := Rotation matrices which rotate the first point to y-axis;
$R_x^I := R_x^a R_x^\mu$, $R_y^I := R_y^a R_y^\mu$;

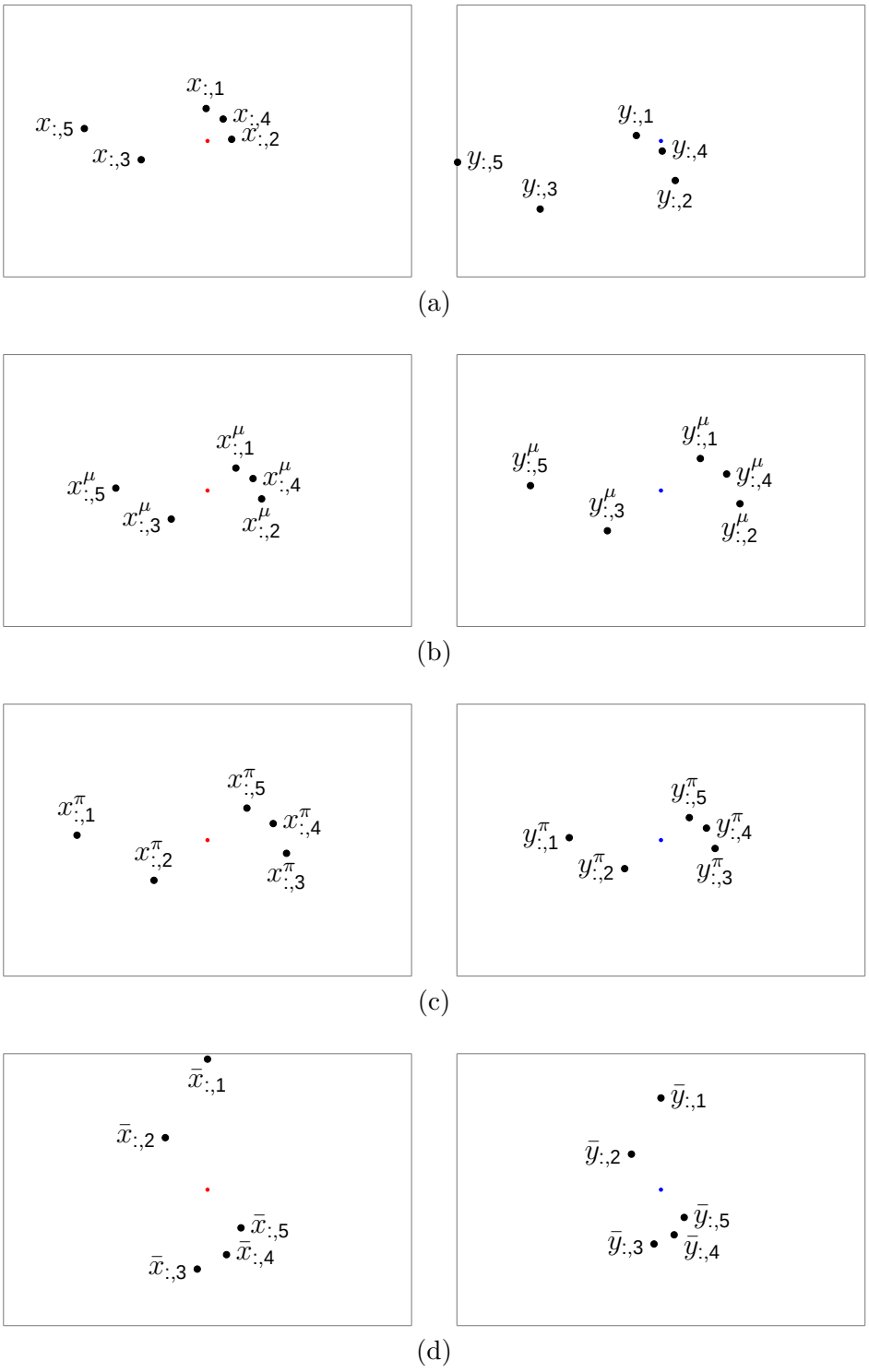


Figure 4.1: An illustration of the invariantization procedure. (a) The input points x, y. (b) The points rotated such, that their center of mass is zero. (Section 4.4.1) (c) The points permuted according to Section 4.4.2 (d) The invariantized points $\bar{x}, \bar{y}$. The red and blue points symbolize the zero point.

4.4.1 Moving the center of mass to zero

Let us introduce the notation which we will use in this section.

x	$\mathsf{R}^{2,5}$	Five calibrated 2D points in the first camera.
y	$\mathsf{R}^{2,5}$	Five calibrated 2D points in the second camera.
x^h	$\mathsf{R}^{3,5}$	Homogeneous representation of columns of x.
y^h	$\mathsf{R}^{3,5}$	Homogeneous representation of columns of y.
R_x^μ	$SO(3)$	A 3D rotation matrix which transforms the homogeneous coordinates x^h of x such that their center of mass is equal to zero.
R_y^μ	$SO(3)$	A 3D rotation matrix which transforms the homogeneous coordinates y^h of y such that their center of mass is equal to zero.
$x^{h,\mu}$	$\mathsf{R}^{3,5}$	Homogeneous representation of points from x rotated such, that their point of mass is in zero. There holds $x^{h,\mu} \approx R_x^\mu x^h$.
$y^{h,\mu}$	$\mathsf{R}^{3,5}$	Homogeneous representation of points from y rotated such, that their point of mass is in zero. There holds $y^{h,\mu} \approx R_y^\mu y^h$.

Now, we are going to describe how to move the center of mass to zero. We know the points $x \in \mathsf{R}^{2,5}$, $y \in \mathsf{R}^{2,5}$. The goal is to find rotation matrices $R_x^\mu \in SO(3)$, $R_y^\mu \in SO(3)$, such that the center of mass of the homogeneous representative of columns of x multiplied by R_x^μ is equal to the homogeneous representative of zero, and the center of mass of the homogeneous representative of columns of y multiplied by R_y^μ is equal to the homogeneous representative of zero.

The procedure is performed iteratively. In every step, the center of mass μ is computed and rotated to zero. This does not guarantee that the center of mass of the transformed points is zero, however, if after about 3 - 4 iterations, the distance between the center of mass and the zero vector is acceptable.

This procedure is the same for both views. We are going to describe it for the first view, where the center of mass of points x is rotated to zero and the accumulated rotation is stored into R_x^μ. The invariantization of points y is done analogously, while the accumulated rotation is stored into R_y^μ. First, the homogeneous representation $x^h \in \mathsf{R}^{3,5}$ of points x is obtained.

$$x^h = \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} & x_{1,5} \\ x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} & x_{2,5} \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{4.21}$$

The matrix R_x^μ is initialized to identity $R_x^\mu = I$, the matrix of transformed points $x^{h,\mu}$ is initialized as $x^{h,\mu} = x^h$.

At the beginning of every iteration, the center of mass μ is computed as:

$$\mu = \frac{1}{5}\sum_{i=1}^{5}\begin{bmatrix} x_{1,i}^{h,\mu} \\ x_{2,i}^{h,\mu} \\ x_{3,i}^{h,\mu} \end{bmatrix} \tag{4.22}$$

With the knowledge of the center of mass μ, we can build the matrix M as:

$$M = \begin{bmatrix} \mu_1 & 1 & 0 \\ \mu_2 & 0 & 1 \\ \mu_3 & 0 & 0 \end{bmatrix} \tag{4.23}$$

Because $\mu_3 = 1$, matrix M is full-rank. Matrix M is decomposed using a QR-decomposition into an orthogonal matrix Q and an upper-triangular matrix R, such as $M = QR$. Because the entries $R_{2,1}$, $R_{3,1}$ of matrix R are equal to zero, the center of mass μ, which is the first column of matrix M is equal to the first column of matrix Q multiplied by a scalar $R_{1,1}$ R. Because the matrix Q is orthogonal, its second and third columns are orthogonal to the center of mass μ. There holds:

$$\begin{bmatrix} Q_{1,i} & Q_{2,i} & Q_{3,i} \end{bmatrix} \mu = 0, i \quad \{2, 3\} \tag{4.24}$$

If we multiply the matrix Q^T with the center of mass μ, we obtain:

$$Q^T\mu = \begin{bmatrix} Q_{1,1} & Q_{2,1} & Q_{3,1} \\ Q_{1,2} & Q_{2,2} & Q_{3,2} \\ Q_{1,3} & Q_{2,3} & Q_{3,3} \end{bmatrix} \mu = \begin{bmatrix} \sigma \\ 0 \\ 0 \end{bmatrix} \tag{4.25}$$

For some σ R.The goal is, however, to find a rotation matrix R_{cur}, such that

$$R_{cur}\mu = \begin{bmatrix} 0 \\ 0 \\ \alpha \end{bmatrix}, \alpha \quad \mathsf{R} \tag{4.26}$$

This can be achieved by multiplying the matrix Q^T by a matrix M_1, such as:

$$M_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{4.27}$$

Then, there holds:

$$R_{cur}\mu = M_1 Q^T \mu = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \sigma \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \sigma \end{bmatrix} \tag{4.28}$$

Both matrices M_1, Q^T are orthogonal, therefore, matrix $R_{cur} = M_1 Q^T$ is orthogonal as well. However, we search for a rotation matrix, which is an orthogonal matrix with a determinant equal to 1. If the determinant $\det R_{cur} = -1$, we can multiply R_{cur} from left by a matrix M_2, such as:

$$M_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{4.29}$$

After that, we obtain the rotation matrix R_{cur} as:

$$R_{cur} = M_2 R_{cur} = M_2 M_1 Q^T \tag{4.30}$$

Now, R_{cur} is a rotation matrix because it is orthogonal and its determinant is equal to 1.

This process is repeated 4 times, after each step, we update the rotation matrix R_x^μ as $R_{cur} R_x^\mu$ and the homogeneous representation $x^{h,\mu}$ according to (4.31). The whole procedure is described in Algorithm 7.

$$\begin{aligned} x^{h,\mu} &= R_{cur} x^{h,\mu} \\ \begin{bmatrix} x_{1,i}^{h,\mu} \\ x_{2,i}^{h,\mu} \\ x_{3,i}^{h,\mu} \end{bmatrix} &= \frac{1}{x_{3,i}^{h,\mu}} \begin{bmatrix} x_{1,i}^{h,\mu} \\ x_{2,i}^{h,\mu} \\ x_{3,i}^{h,\mu} \end{bmatrix} \end{aligned} \tag{4.31}$$

Algorithm 7: Rotate the center of mass to zero

input : Five points x
output : $x^{h,\mu}$ Homogeneous representation of points x rotated such, that their point of mass is zero, R_x^μ Rotation matrix which transforms x to $x^{h,\mu}$

$x^h :=$ Homogeneous representation of x;
$x^{h,\mu} := x^h$, $R_x^\mu := I$; `// Initialize the values`
$M_1 =$ Matrix (4.27), $M_2 =$ Matrix (4.29);
for $j \quad \{1, ..., 4\}$ **do**
  $\mu :=$ Point of mass of $x^{h,\mu}$ according to (4.22);
  $M :=$ Matrix (4.23);
  Decompose M into $M = QR$ using QR-Decomposition;
  $R_{cur} := M_1 Q^T$;
  if $\det R_{cur} < 1$ **then**
    $R_{cur} := M_2 R_{cur}$;
  end
  $R_x^\mu := R_{cur} R_x^\mu$, Update $x^{h,\mu}$ according to (4.31);
end

4.4.2 Permutation of the points

Let us introduce the notation which we will use in this section.

x	$\mathsf{R}^{2,5}$	Five calibrated 2D points in the first camera.
y	$\mathsf{R}^{2,5}$	Five calibrated 2D points in the second camera.
R_x^μ	$SO(3)$	The output of Section 4.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of x such that their center of mass is zero.
R_y^μ	$SO(3)$	The output of Section 4.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of y such that their center of mass is zero.
$x^{h,\mu}$	$\mathsf{R}^{3,5}$	The output of Section 4.4.1. Homogeneous representation of points from x rotated such, that their point of mass is in zero. There holds $x^{h,\mu} \approx R_x^\mu x^h$.
$y^{h,\mu}$	$\mathsf{R}^{3,5}$	The output of Section 4.4.1. Homogeneous representation of points from y rotated such, that their point of mass is in zero. There holds $y^{h,\mu} \approx R_y^\mu y^h$.

r $Sym(\{1,...,5\})$ The permutation of the points.

s $Sym(\{1,2\})$ The permutation of the views. $s = \{2,1\}$ if the views have been swapped, $s = \{1,2\}$ otherwise.

$x^{h,\pi}$ $\mathsf{R}^{3,5}$ If $s = \{1,2\}$, then $x^{h,\pi}$ are the points from $x^{h,\mu}$ permuted by r, otherwise $x^{h,\pi}$ are the points from $y^{h,\mu}$ permuted by r.

$y^{h,\pi}$ $\mathsf{R}^{3,5}$ If $s = \{1,2\}$, then $x^{h,\pi}$ are the points from $x^{h,\mu}$ permuted by r, otherwise $x^{h,\pi}$ are the points from $y^{h,\mu}$ permuted by r.

Now, we assume that we have found matrices R_x^μ, R_y^μ (Section 4.4.1) which transform the points of mass of homogeneous representations of points x, y to zero. We further assume that we know the homogeneous representations $x^{h,\mu}$ $\mathsf{R}^{3,5}$, $y^{h,\mu}$ $\mathsf{R}^{3,5}$ of the points x, y rotated such that their points of mass are in zero.

The goal is to permute the points in such a way that the point whose distance from the center of mass is the largest is the first and the remaining points are ordered counterclockwise. Furthermore, we swap the views if the first point is more distant in the second view. We want to find the permutations r of the points and s of the views, as well as the permuted points $x^{h,\pi}$, $y^{h,\pi}$.

First, we identify the index i_1 of the point in the first view, whose distance from zero is maximal:

$$i_1 = \underset{i \ \{1,...,5\}}{\arg\max} \left\| \begin{bmatrix} x_{1,i}^{h,\mu} \\ x_{2,i}^{h,\mu} \end{bmatrix} \right\|^2 \tag{4.32}$$

Analogously, we identify the index i_2 of the point in the second view, whose distance from zero is maximal:

$$i_2 = \underset{i \ \{1,...,5\}}{\arg\max} \left\| \begin{bmatrix} y_{1,i}^{h,\mu} \\ y_{2,i}^{h,\mu} \end{bmatrix} \right\|^2 \tag{4.33}$$

If the point i_1 in the first view is further from zero than the point i_2 in the second view, we set $i = i_1$ and $s = \{1,2\}$. Otherwise, we set $i = i_2$ and $s = \{2,1\}$, and we swap the views, such, that the new $x^{h,\mu}$ is equal to the original $y^{h,\mu}$ and vice versa, new $y^{h,\mu}$ is equal to the original $x^{h,\mu}$. In addition to that, we also swap the rotation matrices R_x^μ and R_y^μ.

Now, we are going to order the remaining points counterclockwise. We compute the angles α_i, $i \quad \{1, ..., 5\}$ of every point in the first view as:

$$\alpha_i = atan2(x_{2,i}^{h,\mu}, x_{2,i}^{h,\mu}) \tag{4.34}$$

If $\alpha_i < 0$, add π to it to obtain:

$$\alpha_i = \alpha_i + \pi \tag{4.35}$$

For points $i \quad \{1, ..., 5\}$, $i = i$, we compute the angle α_i relative to point i as:

$$\alpha_i = \alpha_i - \alpha_i \tag{4.36}$$

If $\alpha_i < 0$, add π to it to obtain:

$$\alpha_i = \alpha_i + \pi \tag{4.37}$$

Now, we find the permutation $r \quad Sym(\{1, 2, 3, 4, 5\})$, such that the first element $r(1)$ of the permutation is the index of the most distant point i and the remaining elements are sorted with an increasing value of α_i (4.36), (4.37). We permute the columns of matrices $x^{h,\mu}$, $y^{h,\mu}$ with this permutation to obtain $x^{h,\pi}$, $y^{h,\pi}$.

4.4.3 Moving the first point to y-axis

Let us introduce the notation which we will use in this section.

x $\mathsf{R}^{2,5}$ Five calibrated 2D points in the first camera.

y $\mathsf{R}^{2,5}$ Five calibrated 2D points in the second camera.

R_x^μ $SO(3)$ The output of Section 4.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of x such that their center of mass is zero.

R_y^μ $SO(3)$ The output of Section 4.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of y such that their center of mass is zero.

$x^{h,\pi}$ $\mathsf{R}^{3,5}$ Points in the first view rotated such that their center of mass is zero and permuted according to Section 4.4.2.

$y^{h,\pi}$ $\mathsf{R}^{3,5}$ Points in the second view rotated such that their center of mass is zero and permuted according to Section 4.4.2.

R_x^a $SO(3)$ A rotation matrix which rotates the first point of $x^{h,\pi}$ to the y-axis.

R_y^a $SO(3)$ A rotation matrix which rotates the first point of $y^{h,\pi}$ to the y-axis.

$\bar{x}$ $\mathsf{R}^{2,5}$ Invariantized representation of points x.

$\bar{y}$ $\mathsf{R}^{2,5}$ Invariantized representation of points y.

R_x^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of x to the homogeneous coordinates of $\bar{x}$.

R_y^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of y to the homogeneous coordinates of $\bar{y}$.

We know the permuted points $x^{h,\pi}$, $y^{h,\pi}$ and the rotation matrices R_x^μ, R_y^μ which transform the center of mass to zero. The goal is to transform the permuted points such that the first point lies on the y-axis. The outputs of this section are the rotation matrices R_x^a, R_y^a, the invariantized points $\bar{x}$, $\bar{y}$ and the rotation matrices R_x^I, R_y^I.

Now, we are going to find the matrix R_x^a which would transform the first row of matrix $x^{h,\pi}$ to y-axis. This matrix represents a rotation around the optical axis, therefore, the property that the center of mass of the points is zero is preserved. Matrix R_y^a which rotates the first point in the second view to the y-axis, is found analogously.

Let $\sigma = \sqrt{(x_{1,1}^\pi)^2 + (x_{2,1}^\pi)^2}$ be the norm of the first Euclidean (not homogeneous) point in the first view. The rotation matrix R_x^a is constructed as follows:

$$R_x^a = \begin{bmatrix} \frac{x_{2,1}^{h,\pi}}{\sigma} & \frac{-x_{1,1}^{h,\pi}}{\sigma} & 0 \\ \frac{x_{1,1}^{h,\pi}}{\sigma} & \frac{x_{2,1}^{h,\pi}}{\sigma} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.38}$$

Matrix R_x^a is a rotation matrix because its columns are orthonormal and because the determinant of R_x^a is equal to 1. The first point is transformed by the matrix R_x^a as follows:

$$R_x^a \begin{bmatrix} x_{1,1}^{h,\pi} \\ x_{2,1}^{h,\pi} \\ x_{3,1}^{h,\pi} \end{bmatrix} = \begin{bmatrix} \frac{x_{2,1}^{h,\pi}}{\sigma} & \frac{-x_{1,1}^{h,\pi}}{\sigma} & 0 \\ \frac{x_{1,1}^{h,\pi}}{\sigma} & \frac{x_{2,1}^{h,\pi}}{\sigma} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1,1}^{h,\pi} \\ x_{2,1}^{h,\pi} \\ x_{3,1}^{h,\pi} \end{bmatrix} = \begin{bmatrix} 0 \\ \sigma \\ x_{3,1}^{h,\pi} \end{bmatrix} \tag{4.39}$$

We can see from equation (4.39), that matrix R_x^a rotates the first point to zero. Because R_x^a is an extended 2D matrix, it represents a rotation around the optical axis which passes through the center of mass of the points. Therefore, the center of mass of points $x^{h,\pi}$ stays in zero after being transformed by R_x^a.

The rotation matrix R_x^I is obtained as:

$$R_x^I = R_x^a R_x^\mu \tag{4.40}$$

The homogeneous representation $\bar{x}^h$ of the invariantized points $\bar{x}$ is obtained as:

$$\bar{x}^h = R_x^a x^{h,\pi} \tag{4.41}$$

The Cartesian representation $\bar{x}$ of the invariantized points is then obtained by taking the first two rows of $\bar{x}^h$ because the procedures described in Sections 4.4.1 and 4.4.2 and the character of matrix R_x^a guarantee that all entries in the last row of $\bar{x}^h$ are equal to 1.

Matrix R_y^a which rotates the first point in the second view to the y-axis is obtained analogously to (4.38). The rotation matrix R_y^I is obtained as:

$$R_y^I = R_y^a R_y^\mu \tag{4.42}$$

The homogeneous representation $\bar{y}^h$ of the invariantized points $\bar{y}$ is obtained as:

$$\bar{y}^h = R_y^a y^{h,\pi} \tag{4.43}$$

The Cartesian representation $\bar{y}$ of the invariantized points is then obtained by taking the first two rows of $\bar{y}^h$.

4.4.4 Obtaining invariantized depths

Let us introduce the notation which we will use in this section.

x	$\mathsf{R}^{2,5}$	Five calibrated 2D points in the first camera.
y	$\mathsf{R}^{2,5}$	Five calibrated 2D points in the second camera.
λ	$\mathsf{R}^{2,5}$	The depths of the points in both cameras.
$\bar{x}$	$\mathsf{R}^{2,5}$	Invariantized representation of points x.
$\bar{y}$	$\mathsf{R}^{2,5}$	Invariantized representation of points y.

R_x^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of x to the homogeneous coordinates of $\bar{x}$.

R_y^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of y to the homogeneous coordinates of $\bar{y}$.

r $Sym(\{1,...,5\})$ The permutation of the points, the output of Section 4.4.2.

s $Sym(\{1,2\})$ The output of Section 4.4.2. The permutation of the views. $s = \{2,1\}$ if the views have been swapped, $s = \{1,2\}$ otherwise.

$\bar{\lambda}$ $\mathsf{R}^{2,5}$ Depths consistent with the invariantized points $\bar{x}$, $\bar{y}$.

$x^{h,P}$ $\mathsf{R}^{3,5}$ Points in the first view permuted according to r, s.

$y^{h,P}$ $\mathsf{R}^{3,5}$ Points in the second view permuted according to r, s.

λ^P $\mathsf{R}^{2,5}$ The depths from λ permuted by permutations r and s.

$X_{i,j}^1$ $\mathsf{R}^{3,5}$ Coordinates of the 3D points in the coordinate system of the first camera.

$\bar{X}_{i,j}^1$ $\mathsf{R}^{3,5}$ Coordinates of the 3D points in the coordinate system of the first invariantized camera.

$X_{i,j}^2$ $\mathsf{R}^{3,5}$ Coordinates of the 3D points in the coordinate system of the second camera.

$\bar{X}_{i,j}^2$ $\mathsf{R}^{3,5}$ Coordinates of the 3D points in the coordinate system of the second invariantized camera.

We know the points x, y, as well as their invariantized representations $\bar{x}$, $\bar{y}$ and the transformation matrices R_x^I, R_y^I. We also assume, that the depths λ are known. The goal is to find the depths $\bar{\lambda}$ of the invariantized points.

In order to obtain the depths of the invariantized points, we first obtain the homogeneous representatives x^h and y^h according to (4.21). Then, we permute the columns of x^h and y^h according to r to obtain $x^{h,P}$, $y^{h,P}$. If the permutation s of views is equal to $s = \{2,1\}$, we swap the values from $x^{h,P}$ and $y^{h,P}$. Then, we permute the columns of λ by r and the rows by s to obtain permuted depths λ^P.

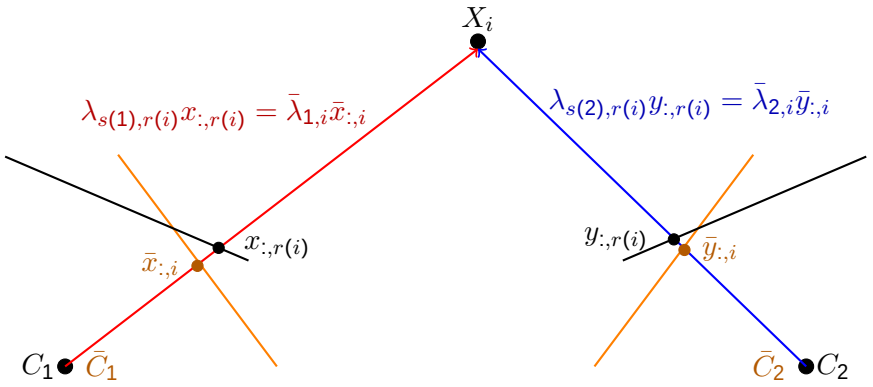


Figure 4.2: An illustration of the change of depths induced by the invariantization. Original cameras C_1, C_2 are black and the invariantized cameras $\bar{C}_1$, $\bar{C}_2$ are orange. The rotation of the camera changes the point where the projection plane of the camera intersects the ray containing the point. Therefore, the depth of the point changes.

We multiply every column of $x^{h,P}$ by the corresponding depth from matrix λ^P to obtain the coordinates X^1 of the 3D points in the coordinate system of camera 1 as:

$$X^1_{i,j} = \lambda^P_{1,j} x^{h,P}_{i,j}, j \quad \{1, ..., 5\}, i \quad \{1, 2, 3\} \tag{4.44}$$

After that, we obtain the coordinates $\bar{X}^1$ of the 3D points in the coordinate system of the first camera after the invariantization as:

$$\bar{X}^1 = R^I_x X^1 \tag{4.45}$$

And we obtain the depths $\bar{\lambda}$ of the invariantized points from the last row of matrix $\bar{X}^1$ as:

$$\bar{\lambda}_{1,j} = \bar{X}^1_{3,j}, j \quad \{1, ..., 5\} \tag{4.46}$$

Similarly, we obtain the coordinates X^2 of the 3D points in the coordinate system of camera 2 as:

$$X^2_{i,j} = \lambda^P_{2,j} y^{h,P}_{i,j}, j \quad \{1, ..., 5\}, i \quad \{1, 2, 3\} \tag{4.47}$$

Then, we obtain the coordinates $\bar{X}^2$ of the 3D points in the coordinate system of the second camera after the invariantization as:

$$\bar{X}^2 = R^I_y X^2 \tag{4.48}$$

And we obtain the depths $\bar{\lambda}$ of the invariantized points from the last row of matrix $\bar{X}^2$ as:

$$\bar{\lambda}_{2,j} = \bar{X}^2_{3,j}, j \quad \{1, ..., 5\} \tag{4.49}$$

4.5 Alignment of the problems on the anchors

The invariantization step transforms the problems to a uniform shape. However, we have noticed that the sum of squared distances between two invariantized problems can often be suboptimal 4.3. Therefore, we add another step of the problem preprocessing which would minimize the sum of squared distances between the anchor and the final problem.

This alignment step increases the number of successful tracks significantly. We have performed the experiment described in Algorithm 16 (Section 4.6.1) with 2000 problem-solution pairs and counted the successful tracks. The number of successful tracks with alignment was equal to 71014 successful tracks out of 1999000, while the number of successful tracks without alignment was equal to 13006 successful tracks out of 1999000. Let us introduce the notation which we will use in this section.

$\bar{x}$ $\mathsf{R}^{2,5}$ Invariantized representation of five calibrated 2D points in the first camera.

$\bar{y}$ $\mathsf{R}^{2,5}$ Invariantized representation of five calibrated 2D points in the second camera.

$\bar{\lambda}$ $\mathsf{R}^{2,5}$ Depths consistent with the invariantized points $\bar{x}, \bar{y}$.

R_x^I $SO(3)$ The output of Section 4.4. A 3D rotation matrix which transforms the homogeneous coordinates of x to the homogeneous coordinates of invariantized points $\bar{x}$.

R_y^I $SO(3)$ The output of Section 4.4. A 3D rotation matrix which transforms the homogeneous coordinates of y to the homogeneous coordinates of invariantized points $\bar{y}$.

$(\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)$, $\breve{x}_a$ $\mathsf{R}^{2,5}$, $\breve{y}_a$ $\mathsf{R}^{2,5}$, $\breve{\lambda}_a$ $\mathsf{R}^{2,5}$ The selected **anchor** (problem-solution pair) from which the homotopy continuation should be tracked. We assume that the points $\breve{x}_a, \breve{y}_a$ have been invariantized according to section 4.4.

$\hat{x}$ $\mathsf{R}^{2,5}$ Points in the first camera aligned to the anchor $(\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)$

$\hat{y}$ $\mathsf{R}^{2,5}$ Points in the second camera aligned to the anchor $(\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)$

$\hat{\lambda}$ $\mathsf{R}^{2,5}$ Depths consistent with the aligned points $\hat{x}, \hat{y}$.

r_A $Sym(\{1, ..., 5\})$ A permutation of the points induced by the alignment.

R_x $SO(3)$ A 3D rotation matrix which transforms the homogeneous representation of invariantized points $\bar{x}$ to the homogeneous representation of aligned points $\hat{x}$.

R_y $SO(3)$ A 3D rotation matrix which transforms the homogeneous representation of invariantized points $\bar{y}$ to the homogeneous representation of aligned points $\hat{y}$.

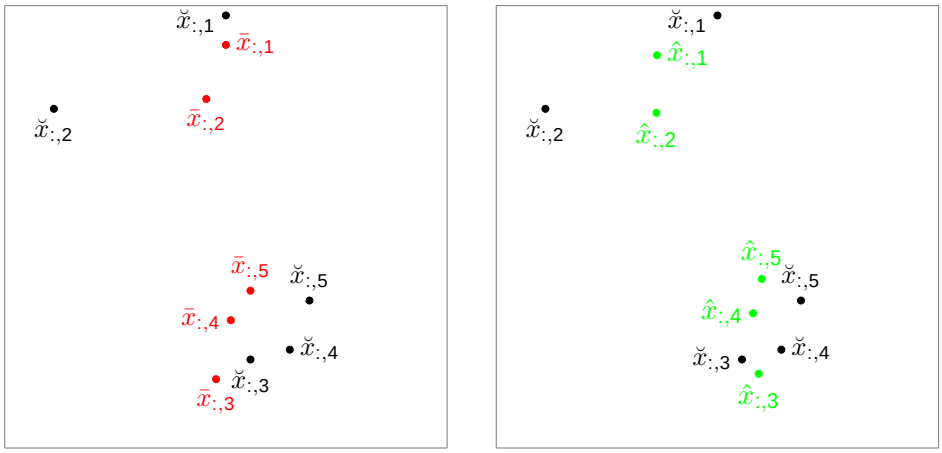


Figure 4.3: The black points are the anchor $\breve{x}$. The red points on the left are the input points $\bar{x}$. The green points on the right are the points $\hat{x}$ aligned to $\breve{x}$.

We know the invariantized points $\bar{x}$, $\bar{y}$ and the selected anchor $(\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)$. The goal is to find the rotation matrices R_x, R_y, the permutation r_A and the aligned points $\hat{x}$, $\hat{y}$, such that the sum of squared distances between the aligned points and the anchor $(\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)$ is minimized. This can be formulated as the following optimization task:

$$R_x, R_y, r_A = \underset{R_x, R_y, r_A}{\arg\min} \sum_{i=1}^{5} \left\| \begin{bmatrix} \breve{x}^a_{1,i} \\ \breve{x}^a_{2,i} \\ 1 \end{bmatrix} - R_x \begin{bmatrix} \bar{x}_{1,r_A(i)} \\ \bar{x}_{2,r_A(i)} \\ 1 \end{bmatrix} \right\|^2 + \sum_{i=1}^{5} \left\| \begin{bmatrix} \breve{y}^a_{1,i} \\ \breve{y}^a_{2,i} \\ 1 \end{bmatrix} - R_y \begin{bmatrix} \bar{y}_{1,r_A(i)} \\ \bar{y}_{2,r_A(i)} \\ 1 \end{bmatrix} \right\|^2 \tag{4.50}$$

Now, we are going to show how the optimization task (4.50) is solved. We consider a subset $S \quad Sym(\{1,...,5\})$ of all permutations in the group $Sym(\{1,...,5\})$. For every permutation $r_A \quad S$ we fix the permutation and find the optimal rotations R_x, R_y, which would minimize the expression (4.50). Then, we select the permutation r_A whose value of (4.50) is minimal.

In Section 4.5.1 we show, how the task (4.50) is minimized with a fixed permutation r_A. We also show that the optimal rotations R_x, R_y rotate

around the optical axis. Therefore, the depths do not have to be adjusted after the alignment, and the depths $\hat{\lambda}$ are obtained by permuting the columns of $\bar{\lambda}$ by r_A. In Section 4.5.2 we discuss the dependence of the result on the size of the set S of all considered permutations. The alignment is described in Algorithm 8.

Algorithm 8: Align Five-Point problem

input : Invariantized representation $\bar{x}, \bar{y}$ of x, y, Selected anchor $(\breve{x}_a\ , \breve{y}_a\ , \breve{\lambda}_a\ )$, Set of permutations S (from Section 4.5.2)
output : Aligned points $\hat{x}, \hat{y}$, Rotation matrices R_x, R_y transforming $\bar{x}, \bar{y}$ to $\hat{x}$, $\hat{y}$, Permutation r_A

$dist_{best} := \infty$;
for $r_A \quad S$ **do**
  $R_x, R_y :=$ Rotation matrices minimizing (4.50) with fixed permutation r_A (Section 4.5.1);
  $dist :=$ Distance (4.50) given by R_x, R_y, r_A;
  if $dist < dist_{best}$ **then**
    $dist := dist_{best}$, $R_x := R_x$, $R_y := R_y$, $r_A := r_A$
  end
end
$\hat{x}, \hat{y} :=$ Rotate the points $\bar{x}, \bar{y}$ with R_x, R_y and permute them with r_A;

4.5.1 Minimization of the Squared Distance Between the Problems

We know the invariantized points $\bar{x}$, $\bar{y}$ and the selected anchor $(\breve{x}_a\ , \breve{y}_a\ , \breve{\lambda}_a\ )$. We assume that the permutation r_A is fixed. The goal is to find the rotation matrices R_x, R_y, which would minimize the value of (4.50) with a fixed r_A.

Then, the optimization task (4.50) breaks up into two independent problems, one for each view:

$$R_x = \underset{R_x \quad SO(3)}{\arg\min} \sum_{i=1}^{5} \left\| \begin{bmatrix} \breve{x}^a_{1,i} \\ \breve{x}^a_{2,i} \\ 1 \end{bmatrix} - R_x \begin{bmatrix} \bar{x}_{1,r_A(i)} \\ \bar{x}_{2,r_A(i)} \\ 1 \end{bmatrix} \right\|^2 \tag{4.51}$$

$$R_y = \underset{R_y \quad SO(3)}{\arg\min} \sum_{i=1}^{5} \left\| \begin{bmatrix} \breve{y}^a_{1,i} \\ \breve{y}^a_{2,i} \\ 1 \end{bmatrix} - R_y \begin{bmatrix} \bar{y}_{1,r_A(i)} \\ \bar{y}_{2,r_A(i)} \\ 1 \end{bmatrix} \right\|^2 \tag{4.52}$$

This is exactly the Orthogonal Procrustes problem [HC62]. Now, we are going to show how the Orthogonal Procrustes problem is solved. We are going to use the example of (4.51), problem (4.52) is solved analogously. The Let $\breve{x}^{a\ ,h}$ $\mathsf{R}^{3,5}$ be the homogeneous representation of points from $\breve{x}^a$ and $\bar{x}^h$ be the homogeneous representation of points from $\bar{x}$. Matrix M is built as:

$$M = \breve{x}^{a\ ,h}\ (\bar{x}^h)^T \tag{4.53}$$

Then, the matrix M is decomposed using SVD into $M = U\Sigma V^T$ and the optimal rotation matrix R_x is obtained as:

$$R_x = UV^T \tag{4.54}$$

Efficient evaluation of SVD

Now, we are going to describe how to obtain the optimal matrix R_x efficiently. Let $\breve{\mu}_a$ R^2 be the center of mass of points in $\breve{x}^a$ and $\bar{\mu}$ be the center of mass of points in $\bar{x}$. Because the last row of both $\breve{x}^{a\ ,h}$ and $\bar{x}^h$ contain only ones, matrix M (4.53) can be evaluated as:

$$M = \begin{bmatrix} \breve{x}^a & (\bar{x})^T & 5\breve{\mu}_a \\ & 5\bar{\mu}^T & 5 \end{bmatrix} \tag{4.55}$$

Because we assume that both $\breve{x}^a$ and $\bar{x}$ are invariantized, their centers of mass $\breve{\mu}_a$, $\bar{\mu}$ are equal to zero and matrix M is equal to:

$$M = \begin{bmatrix} \breve{x}^a & (\bar{x})^T & 0 \\ & 0 & 5 \end{bmatrix} \tag{4.56}$$

Let $M\ = \breve{x}^a\ \ (\bar{x})^T$ be a top-left 2×2 submatrix of M and let $U\ \Sigma\ V^T$ be its SVD decomposition. Then, the SVD decomposition of matrix M is equal to:

$$M = U\Sigma V^T = \begin{bmatrix} U & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \Sigma & 0 \\ 0 & 5 \end{bmatrix} \begin{bmatrix} V^T & 0 \\ 0 & 1 \end{bmatrix} \tag{4.57}$$

This means that the rotation R_x which minimizes expression (4.51) represents a rotation around the optical axis. We can obtain the rotation R_x by a SVD decomposition of matrix M and building:

$$R_x = UV^t = \begin{bmatrix} U\ V^T & 0 \\ 0 & 1 \end{bmatrix} \tag{4.58}$$

The SVD decomposition of a 2×2 matrix M can be obtained in a closed form according to [Bli96]. First, the scalar values E, F, G, H, Q, R are computed as:

$$\begin{aligned} E &= \frac{M_{0,0} + M_{1,1}}{2} \\ F &= \frac{M_{0,0} - M_{1,1}}{2} \\ G &= \frac{M_{1,0} + M_{0,1}}{2} \\ H &= \frac{M_{1,0} - M_{0,1}}{2} \\ Q = \sqrt{E^2 + H^2}&, R = \sqrt{F^2 + G^2} \end{aligned} \tag{4.59}$$

Then, the singular values s_x, s_y are obtained as:

$$s_x = Q + R, s_y = Q - R \tag{4.60}$$

In order to obtain the angles ϕ, θ of the rotation matrices U , V^T, we first compute the values a_1, a_2 as:

$$a_1 = atan2(G, F), a_2 = atan2(H, E) \tag{4.61}$$

Then, the angles ϕ, θ are computed as:

$$\phi = \frac{a_2 + a_1}{2}, \theta = \frac{a_2 - a_1}{2} \tag{4.62}$$

The SVD decomposition of matrix M goes as follows:

$$M = U \, \Sigma \, V^T = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{4.63}$$

This procedure can be further simplified. We are only interested in computing the matrix $U \, V^T$, therefore, matrix Σ and values s_x, s_y are not interesting for us. The rotation matrix $U \, V^T$ R^2, which is the top-left submatrix of R_x, can be obtained as:

$$\begin{aligned} U \, V^T &= \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \\ &= \begin{bmatrix} \cos\phi\cos\theta - \sin\phi\sin\theta & -\cos\phi\sin\theta - \sin\phi\cos\theta \\ \sin\phi\cos\theta + \cos\phi\sin\theta & \cos\phi\cos\theta - \sin\phi\sin\theta \end{bmatrix} \end{aligned} \tag{4.64}$$

Using trigonometric identities, the matrix $U \, V^T$ can be simplified as:

$$U \, V^T = \begin{bmatrix} \cos(\phi + \theta) & -\sin(\phi + \theta) \\ \sin(\phi + \theta) & \cos(\phi + \theta) \end{bmatrix} \tag{4.65}$$

Angle $\phi + \theta$ can be evaluated as:

$$\phi + \theta = \frac{a_2 + a_1}{2} + \frac{a_2 - a_1}{2} = a_2 = atan2(H, E) \tag{4.66}$$

Because $a_2 = atan2(H, E) = \arctan \frac{H}{E}$, there holds:

$$\tan a_2 = \frac{\sin a_2}{\cos a_2} = \frac{H}{E} \tag{4.67}$$

And the values of $\sin a_2$ and $\cos a_2$ are proportional to values of H, E. Therefore, we introduce scalar values r_1, r_2, such that there holds:

$$r_1 = \cos a_2 = \frac{E}{\sqrt{H^2 + E^2}}, r_2 = \sin a_2 = \frac{H}{\sqrt{H^2 + E^2}} \tag{4.68}$$

Then, we compute the matrix $U\,V^T$ as:

$$U\,V^T = \begin{bmatrix} r_1 & -r_2 \\ r_2 & r_1 \end{bmatrix} \tag{4.69}$$

With the knowledge of matrix $U\,V^T$, we compute matrix R_x according to (4.58).

4.5.2 Selecting the Subset of the Available Permutations

During the alignment, we try all permutations from a set S, which is a subset of the symmetric group $Sym(\{1, ..., 5\})$. Let us introduce the notation which we will use in this section.

r_A $Sym(\{1, ..., 5\})$ A permutation of the points induced by the alignment.

S $Sym(\{1, ..., 5\})$ The subset of the possible permutations r_A which are considered in the alignment.

S^i $Sym(\{1, ..., 5\}), i \quad \{1, ..., 120\}$ Set of i most successful permutations from $Sym(\{1, ..., 5\})$

P A set of 2000 problem-solution pairs generated according to Section 2.1.3.

If we try all permutations ($\mathsf{S} = Sym(\{1, ..., 5\})$), we can obtain the global minimum of function (4.50). However, although the procedure described in 4.5.1 is fairly fast, it is not efficient to evaluate it 120 times for every track.

We have conducted the following experiment to select a meaningful subset S of permutations which will be considered in the alignment.

First, we have generated a set P of 2000 problems according to Section 2.1.3, then we have invariantized them (Section 4.4), aligned (Algorithm 8) with a full symmetric group $S = Sym(\{1, ..., 5\})$ and tracked from every problem $p \quad P$ to every other problem $p \quad P, p = p$. After that, we have recorded how many successful tracks have been performed after the permutation r_A has been selected in the alignment step. The result for up to 20 most successful permutations is shown in Table 4.1.

rank	r_A	# successful tracks
1	{0, 1, 2, 3, 4}	11964
2	{0, 4, 1, 2, 3}	7813
3	{0, 2, 3, 4, 1}	7534
4	{0, 3, 4, 1, 2}	6980
5	{0, 2, 1, 3, 4}	2315
6	{0, 1, 2, 4, 3}	2177
7	{0, 1, 3, 2, 4}	1990
8	{0, 4, 3, 1, 2}	1767
9	{0, 2, 4, 3, 1}	1624
10	{0, 3, 4, 2, 1}	1619
11	{0, 4, 1, 3, 2}	1602
12	{0, 4, 2, 1, 3}	1578
13	{0, 3, 2, 4, 1}	1561
14	{4, 0, 1, 2, 3}	1538
15	{1, 2, 3, 4, 0}	1518
16	{4, 1, 2, 3, 0}	1213
17	{0, 3, 1, 2, 4}	1179
18	{2, 3, 4, 0, 1}	1113
19	{1, 0, 2, 3, 4}	1113
20	{0, 1, 4, 2, 3}	1112
rest	-	29688
total	-	88998

Table 4.1: Numbers of successful tracks after selected permutations r_A in the alignment step

After that, we have ordered the permutations according to the number of successful tracks in the experiment. For every $i \quad \{1, ..., 20\}$, we have considered the set S^i of the i most successful permutations. We have tracked from every problem $p \quad P$ to every other problem $p \quad P, p = p$, while the set S^i was considered in the alignment. The dependence between the size i of the set of considered permutations S, the number of successful tracks and the time needed for the alignment is shown in Table 4.2. We have decided

to use the first four permutations as the set of permutations considered in the alignment because the number of successful tracks in Table 4.2 grows fast until number 4, while it grows much faster after it. Therefore, the set S is equal to $S = \{\{0, 1, 2, 3, 4\}, \{0, 4, 1, 2, 3\}, \{0, 2, 3, 4, 1\}, \{0, 3, 4, 1, 2\}\}$.

# Permutations	Successful Tracks	Alignment Time (μs)
1	39554	1.38
2	53456	1.45
3	64755	1.51
4	71014	1.65
5	72090	1.70
6	72981	1.92
7	73062	2.25
8	73711	2.32
9	74358	2.38
10	75102	2.38
11	75660	2.40
12	76325	2.65
13	76978	2.75
14	78199	2.85
15	79097	3.35
16	79912	3.48
17	80411	3.60
18	80416	3.62
19	81289	3.72
20	81587	3.74

Table 4.2: Numbers of successful tracks after the alignment where S contains i top permutations from Table 4.1, $i \quad \{1, ..., 20\}$.

4.6 Selection of the Anchor

Let us introduce the notation which we will use in this section.

$(\bar{x}, \bar{y}), \bar{x} \quad \mathsf{R}^{2,5}, \bar{y} \quad \mathsf{R}^{2,5}$ The invariantized points in the first and the second view.

$m \quad \mathsf{N}$ The expected number of anchors.

$A = (\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)_I \; a \quad \{1, ..., m\}, \breve{x}_a \quad \mathsf{R}^{2,5}, \breve{y}_a \quad \mathsf{R}^{2,5}, \breve{\lambda}_a \quad \mathsf{R}^{2,5}$ A sequence of **anchors**, i.e. starting problem-solution pairs. For every a, the values in λ_a are the solutions to the depth formulation

of the Five-Point problem (2.5) parametrized by $\breve{x}_a, \breve{y}_a$. We assume that the problem $\breve{x}_a, \breve{y}_a$ is invariantized.

D R^{14} A low-dimensional representation of problem $(\bar{x}, \bar{y})$

$c(D, \theta) : \mathsf{R}^{14} \to \mathsf{R}^{m+1}$ The classifier which for every low-dimensional representation D of problem $(\bar{x}, \bar{y})$ outputs a vector of probabilities that the problem is successfully tracked from each anchor. θ is a vector of parameters to the classifier.

We use a real homotopy continuation (Section 3.4) in our solver. In contrast to the complex homotopy continuation, the real homotopy continuation has no guarantee that the track will be successful. In addition to that, we perform only one track and even if we succeed, there is no guarantee that we will end up in the expected solution.

Therefore, we maintain a set $\mathcal{A}$ of **anchors** (problem-solution pairs), which serve as the starting points for the homotopy continuation. We also train a classifier c, which accepts an invariantized problem $(\bar{x}, \bar{y})$ and outputs a number a $\{0, ..., m\}$. If $a = 0$, we do not track the problem. Otherwise, we track the problem from the anchor $(\breve{x}^a, \breve{y}^a, \breve{\lambda}^a)$.

In Section 4.6.1 we describe how the set $\mathcal{A}$ of the anchors is generated. In Section 4.6.2 we describe the preprocessing of the problem $(\bar{x}, \bar{y})$, which generates a low-dimensional representation D R^{14} of the problem. In Section 4.6.3 we describe how the training data is generated. In Section 4.6.4 we describe the structure of the classifier c.

4.6.1 Anchor set generation

Now, we are going to describe how the set of anchors $\mathcal{A}$ is generated. Let us introduce the notation which we will use in this section.

n The number of considered problems.

$\mathcal{P} = (x_i, y_i, \lambda_i)_i$ i $\{1, ..., m\}, x_i$ $\mathsf{R}^{2,5}, y_i$ $\mathsf{R}^{2,5}, \lambda_i$ $\mathsf{R}^{2,5}$ A set of n problem-solution pairs generated by Section 2.1.3.

$\bar{\mathcal{P}} = (\bar{x}_i, \bar{y}_i, \bar{\lambda}_i)_i$ i $\{1, ..., m\}, \bar{x}_i$ $\mathsf{R}^{2,5}, \bar{y}_i$ $\mathsf{R}^{2,5}, \bar{\lambda}_i$ $\mathsf{R}^{2,5}$ A set of problem-solution pairs invariantized according to 4.4.

m N The expected number of anchors.

$\mathcal{A} = (\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)_{,}$ a $\{1, ..., m\}, \breve{x}_a$ $\mathsf{R}^{2,5}, \breve{y}_a$ $\mathsf{R}^{2,5}, \breve{\lambda}_a$ $\mathsf{R}^{2,5}$ A sequence of **anchors**, i.e. starting problem-solution pairs. For every a, the values in $\breve{\lambda}_a$ are the solutions to the depth formulation of the Five-Point problem (Equation 2.5) parametrized by $\breve{x}_a, \breve{y}_a$. We assume that the problem $\breve{x}_a, \breve{y}_a$ is invariantized.

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$ An undirected connectivity graph on the set $\mathcal{P}$. The vertices of $\mathcal{G}$ are equal to the problem-solution pairs in $\mathcal{P}$ ($\mathcal{V} = \mathcal{P}$). Two problems p_i, p_i $\mathcal{P}$ are connected by an edge in $\mathcal{E}$ if p_i is correctly tracked from p_i or vice versa.

We know the set $\mathcal{P}$ of n problem-solution pairs. We say that the problem-solution pair (x_i, y_i, λ_i) is **correctly tracked** from (x_a, y_a, λ_a) if the homotopy continuation is successfully tracked and the resulting depth is equal to the expected depth λ_i. The task is to find a subset of anchors $\mathcal{A}$ $\mathcal{P}, |A| = m$, such that the number of problem-solution pairs from $\mathcal{P}$ which are correctly tracked at least from one anchor from $\mathcal{A}$ is maximized.

We solve this problem in two stages. First, we build a connectivity graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, whose vertices are equal to $\mathcal{P}$. Two problems p_i, p_i $\mathcal{P}$ are connected by an edge in $\mathcal{E}$ if p_i is correctly tracked from p_i or vice versa. The building of the connectivity graph is described in Algorithm 9. The graph is undirected because, according to theory [SI05], if there is a correct track from problem p_i to p_i , then there is also a correct track from p_i to p_i.

In the second stage, we find a subset $\mathcal{A}$ of vertices in $\mathcal{G}$, such that $|\mathcal{A}| = m$ and the number of vertices from $\mathcal{G}$ which are covered by the vertices from $\mathcal{A}$ is maximized. This is an NP-hard task, therefore, we use the heuristic described in Algorithm 10.

Algorithm 9: Building connectivity graph

input : Set P of n problem-solution pairs, Threshold θ
output : Connectivity graph $G = (V, E)$. Two problem-solution pairs p_i, p_i are connected if we can track correctly p_i from p_i

$V := \{1, ..., n\}$, $E :=$;
for i $\{1, ..., n\}$ **do**
  $\bar{x}_i, \bar{y}_i :=$ Invariantize x_i, y_i (Section 5.4);
  $\bar{\lambda}_i :=$ Transform depths λ_i according to Section 4.4.4;
end
for i $\{1, ..., n\}$ **do**
  for j $\{i + 1, ..., n\}, j = i$ **do**
    $\hat{x}_j, \hat{y}_j :=$ Align x_j, y_j on $\bar{x}_i, \bar{x}_j$ (Section 4.5);
    $r_A :=$ Permutation of the points induced by the alignment;
    $\hat{\lambda}_j :=$ Permute $\bar{\lambda}_j$ according to r_A and fix scale (Section 4.1.1);
    $p_s :=$ Parametrize $(\bar{x}_i, \bar{y}_i)$ according to (4.2);
    $p_f :=$ Parametrize $(\hat{x}_j, \hat{y}_j)$ according to (4.2);
    $z_0 :=$ Parametrize $\bar{\lambda}_i$ according to (4.3);
    $z_1 :=$ Track homotopy (4.6) from z_0 using Algorithm 4;
    if $z_1 =$ **then**
      $\lambda_1 :=$ Depths obtained from z_1 according to (4.4);
      if $\lambda_1 - \hat{\lambda}_j < \theta$ **then**
        $E := E$ $(\{i, j\}$;
      end
    end
  end
end

Algorithm 10: Heuristic for finding the anchors

```
input  : Graph G = (V, E) obtained by Algorithm 9, number of
         anchors m, Invariantized problems P̄
output : Sequence A of m anchors
n := number of vertices in G;
for i  {1, ..., n} do
    active_i := 1 ;                    // Set all vertices as active
end
for i  {1, ..., m} do
    /* Find the node connected to most active nodes      */
    j  := 0, N_A :=  ;
    for j  {1, ..., n} do
        /* Find all active nodes connected with j        */
        N := {v  V | (j, v)  E};       // Neighbors of vertex j
        N_A :=  ;                      // Active neighbors
        for v  N do
            if active_v == 1 then
                N_A := N  {v};
            end
        end
        if |N_A| > |N_A| then
            N_A := N_A, j  := j;
        end
    end
    (x̆_i, y̆_i, λ̆_i) := (x̄_j , ȳ_j , λ̄_j );          // Add a new anchor
    for v  N_A do
        active_v := 0;     // Deactivate nodes connected with j
    end
end
```

4.6.2 Problem preprocessing for the classifier

In this section, we are going to discuss how to uniquely represent the problem (x, y) with a vector D R^{14}, which will be used as the input to the classifier c. We want to design the preprocessing such that the number of problems correctly classified by the classifier c is as large as possible. Let us introduce the notation which we will use in this section.

$(x, y), x$ $\mathsf{R}^{2,5}, y$ $\mathsf{R}^{2,5}$ The points in the first and the second view which define an instance of the five-point problem.

$(\bar{x}, \bar{y}), \bar{x}$ $\mathsf{R}^{2,5}, \bar{y}$ $\mathsf{R}^{2,5}$ The invariantized points in the first and the second view.

p R^{20} A 20-dimensional representation of five-point problem instance $(\bar{x}, \bar{y})$ obtained by (4.2).

D R^{14} A unique 14-dimensional representation of five-point problem instance $(\bar{x}, \bar{y})$.

L $\mathsf{A}^{14,20}$ A transformation matrix which transforms a 20-dimensional (invariantized or aligned) representation p to a 14-dimensional representation D.

$\mathsf{P} = (x_i, y_i, \lambda_i)_i$ i $\{1, ..., n\}, x_i$ $\mathsf{R}^{2,5}, y_i$ $\mathsf{R}^{2,5}, \lambda_i$ $\mathsf{R}^{2,5}$ A set of n problem-solution pairs generated by Section 2.1.3.

A A set of anchors obtained by the procedure described in Section 4.6.1.

$\bar{K}$ $\mathsf{R}^{20,20}$ A covariance matrix of the invariantized points from P.

$\hat{K}$ $\mathsf{R}^{20,20}$ A covariance matrix of the points from P aligned to the first anchor $(\breve{x}_1, \breve{y}_1)$ A.

We know the problem (x, y), we want to find the its representation D and the transformation matrix L, which transforms the 20-dimensional representation p of the problem to a 14-dimensional representation D. We are going to describe two different low-dimensional representations of the points. One is based on the invariantized points $(\bar{x}, \bar{y})$, and the other one is based on the points $(\hat{x}, \hat{y})$ aligned on the first anchor $(\breve{x}_1, \breve{y}_1)$.

Low dimensional representation of invariantized points

First, we invariantize the problem (x, y) according to Section 4.4 to obtain the invariantized problem $(\bar{x}, \bar{y})$. We are going to show, that all invariantized instances $(\bar{x}, \bar{y})$ of the Five-Point Problem live in a 14-dimensional linear subspace of the linear space R^2.

Points $(\bar{x}, \bar{y})$ are invariantized according to Section 4.4, which means, that the first coordinates of the first point $\bar{x}_{1,1}$, $\bar{y}_{1,1}$ are equal to zero and the centers

of mass of the points in both views are equal to zero as well. There holds:

$$\begin{aligned} \bar{x}_{1,1} &= 0 \\ \bar{y}_{1,1} &= 0 \\ \sum_{i=1}^{5} \bar{x}_{1,i} &= 0 \\ \sum_{i=1}^{5} \bar{x}_{2,i} &= 0 \\ \sum_{i=1}^{5} \bar{y}_{1,i} &= 0 \\ \sum_{i=1}^{5} \bar{y}_{2,i} &= 0 \end{aligned} \tag{4.70}$$

The invariantized points have to satisfy the equations (4.70). This is a set of six homogeneous linear equations. The set of solutions to these equations is a 14-dimensional linear subspace of the 20-dimensional linear space R^{20}. For every invariantized problem $(\bar{x}, \bar{y})$ there exists a vector $D \quad \mathsf{R}^{14}$ that can represent the invariantized problem uniquely. Now, we are going to describe how this vector can be obtained.

Let us have a set $\mathcal{P}$ of n problem-solution pairs generated according to Section 2.1.3. First, we invariantize the points in $\mathcal{P}$ to obtain $\bar{\mathcal{P}}$. Then, we obtain a 20-dimensional parametrization $p_i, i \quad \{1, ..., n\}$ of every problem in $\bar{\mathcal{P}}$ according to (4.2) and compute the covariance matrix $\bar{K} \quad \mathsf{R}^{20,20}$ of the invariantized points as:

$$\bar{K} = \frac{1}{n} \sum_{i=1}^{n} p_i p_i^T \tag{4.71}$$

Because the points $p_i, i \quad \{1, ..., n\}$ live in a 14-dimensional subspace of the linear space R^{20}, the covariance matrix K has a rank 14. The matrix K is symmetric, therefore, there exists a real eigendecomposition of matrix K:

$$\bar{K} = V \Lambda V^T \tag{4.72}$$

Because the matrix K has rank 14, its first six eigenvalues are equal to zero. Let $V \quad \mathsf{R}^{20,14}$ be the matrix whose columns are the last 14 eigenvectors of K, i.e. the last 14 columns of V. Let $\Lambda \quad \mathsf{R}^{14,14}$ be a diagonal matrix whose entries are the nonzero eigenvalues of K. Then, the matrix K can be represented as:

$$\bar{K} = V \, \Lambda \, V^T \tag{4.73}$$

Let us have a parametrization $p \quad \mathsf{R}^{20}$ of an invariantized problem $(\bar{x}, \bar{y})$ according to (4.2). This parametrization lives in a 14-dimensional subspace generated by V . The first six columns of V are orthogonal to p, therefore, the first six coordinates of vector $V^T p$ are equal to zero. Vector $d \quad \mathsf{R}^{14}, d = V^T p$

is therefore a unique representation of an invariantized problem p. We have found empirically, that we can learn a better classifier if the resulting vector d is multiplied by matrix $\Lambda^{-\frac{1}{2}}$. The transformation matrix L is equal to:

$$L = \Lambda^{-\frac{1}{2}} V^T \tag{4.74}$$

Therefore, the 14-dimensional representation D of a problem $(\bar{x}, \bar{y})$ is obtained as:

$$D = Lp = \Lambda^{-\frac{1}{2}} V^T p \tag{4.75}$$

Low dimensional representation of aligned points

Now, we are going to describe the second low-dimensional representation of the problem (x, y) based on the points aligned on the first anchor $(\breve{x}_1, \breve{y}_1)$.

We invariantize the points in $\mathbf{P}$ to obtain $\bar{\mathbf{P}}$ and align the points in $\bar{\mathbf{P}}$ to the first anchor $(\breve{x}^1, \breve{y}^1)$ according to (4.5) to obtain $\hat{\mathbf{P}}$. Then, we obtain a 20-dimensional parametrization $p_i, i \quad \{1, ..., n\}$ of every problem in $\hat{\mathbf{P}}$ according to (4.2) and compute the covariance matrix $\hat{K} \quad \mathbb{R}^{20,20}$ of the invariantized points as: invariantized points as:

$$\hat{K} = \sum_{i=1}^{n} p_i p_i^T \tag{4.76}$$

We have found empirically, that the rank of matrix $\hat{K}$ is equal to 14, therefore, the points aligned to the first anchor live in a 14-dimensional subspace of $\mathbb{R}^{20}$. We can obtain the eigendecomposition of the matrix $\hat{K} = V \Lambda V^T$. Then, we get the matrix $V \quad \mathbb{R}^{20,14}$ of the last 14 eigenvectors and the diagonal matrix $\Lambda \quad \mathbb{R}^{14,14}$ with the 14 largest eigenvalues. The transformation matrix L is equal to:

$$L = \Lambda^{-\frac{1}{2}} V^T \tag{4.77}$$

Therefore, we obtain the 14-dimensional representation D of problem $(\hat{x}, \hat{y})$ as:

$$D = Lp = \Lambda^{-\frac{1}{2}} V^T p \tag{4.78}$$

The explanation of this approach is the same as in the low-dimensional representation of invariantized points described in the previous subsection.

Comparison of the approaches

We have found empirically, that if the approach using the low-dimensional representation of aligned points is used, then the resulting classifier is able

to classify correctly approximately 5-10 % more problems than if the low-dimensional representation of the invariantized points.

4.6.3 Training data generation

Let us introduce the notation which we will use in this section.

n The number of training data.

$P_t = (x_i, y_i, \lambda_i)_i$ i $\{1, ..., n\}, x_i$ $\mathsf{R}^{2,5}, y_i$ $\mathsf{R}^{2,5}, \lambda_i$ $\mathsf{R}^{2,5}$ Training data generated according to Section 2.1.3.

$A = (\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)_a$ $\{1, ..., m\}, \breve{x}_a$ $\mathsf{R}^{2,5}, \breve{y}_a$ $\mathsf{R}^{2,5}, \breve{\lambda}_a$ $\mathsf{R}^{2,5}$ A sequence of **anchors**, i.e. starting problem-solution pairs. The anchors are generated using a procedure from Section 4.6.1. For every a, the values in $\breve{\lambda}_a$ are the solutions to the depth formulation of the Five-Point problem (Equation 2.5) parametrized by $\breve{x}_a, \breve{y}_a$. We assume that the problem $\breve{x}_a, \breve{y}_a$ is invariantized.

$X = D_i, i$ $\{1, ..., n\}, D_i$ R^{14} The set of input observations for the training of the classifier. D_i is the 14 dimensional representation of problem (x_i, y_i) from P_t obtained by procedure from Section 4.6.2.

$Y = a_i, i$ $\{1, ..., n\}, a_0$ $\{0, ..., m\}$ The set of the labels for the training of the classifier. The problem (x_i, y_i, λ_i) can be correctly tracked from the anchor $(\breve{x}_{a_i}, \breve{y}_{a_i}, \breve{\lambda}_{a_i})$ A. If the problem cannot be tracked correctly from any anchor in A, then $a_i = 0$.

Now, we are going to describe how the training data for the classifier are generated. We want to find the training observations X and the training labels Y.

We start with empty sequences X, Y. First, we generate a set of n problems P_t according to Section 2.1.3. For every problem (x_i, y_i, λ_i) P_t we obtain the 14-dimensional representation D_i according to Section (4.6.2). For every anchor $(\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)$ A, we invariantize the problem and align it to the anchor $(\breve{x}_a, \breve{y}_a, \breve{\lambda}_a)$ according to Section 4.5 to obtain an aligned problem $(\hat{x}_i, \hat{y}_i, \hat{\lambda}_i)$.

Then, we track the homotopy (4.6) from the parametrization p_s of anchor $(\breve{x}_a, \breve{y}_a)$ to the parametrization p_f of problem $(\hat{x}, \hat{y})$. If the homotopy continuation is **correctly tracked**, i.e. it is successfully tracked and the resulting depth λ is equal to the expected depth $\hat{\lambda}$, then we add the 14-dimensional representation D_i of problem (x_i, y_i, λ_i) to the end of the sequence X and the index a of the current anchor to the end of the sequence Y. If the current problem cannot be successfully tracked from any anchor, we add its 14-dimensional representation D_i to the end of X and zero to the end of Y. The procedure of generating the training data is described in Algorithm 11.

Algorithm 11: Generating train data

input : Set P_t of n problem-solution pairs, Set A of m anchors, Threshold θ

output : X Sequence of Low-rank representations of points in P, Y Sequence of ids of anchors from which the problems from P can be tracked

$\mathsf{V} := \{1, ..., n\}$, $\mathsf{E} :=$;
for i $\{1, ..., n\}$ **do**
  $\bar{x}_i, \bar{y}_i :=$ Invariantize x_i, y_i (Section 5.4);
  $\bar{\lambda}_i :=$ Transform depths λ_i according to Section 4.4.4;
  $D_i :=$ Low-dim representation of x_i, y_i (Sec. 4.6.2);
  $tracked := 0$;
  for j $\{1, ..., m\}$ **do**
    $\hat{x}_i, \hat{y}_i :=$ Align x_i, y_i on $\breve{x}_i, \breve{y}_i$ (Section 4.5);
    $r_A :=$ Permutation of the points induced by the alignment;
    $\hat{\lambda}_i :=$ Permute $\bar{\lambda}_i$ according to r_A and fix scale (Section 4.1.1);
    $p_s :=$ Parametrize $(\breve{x}_j, \breve{y}_j)$ according to (4.2);
    $p_f :=$ Parametrize $(\hat{x}_i, \hat{y}_i)$ according to (4.2);
    $z_0 :=$ Parametrize $\breve{\lambda}_j$ according to (4.3);
    $z_1 :=$ Track homotopy (4.6) from z_0 using Algorithm 4;
    if $z_1 =$ **then**
      $\lambda_1 :=$ Depths obtained from z_1 according to (4.4);
      if $\lambda_1 - \hat{\lambda}_i < \theta$ **then**
        $tracked := 1$;
        $\mathsf{X} := \mathsf{X}$ $\{D_i\}$;
        $\mathsf{Y} := \mathsf{Y}$ $\{j\}$;
      end
    end
  end
  if $tracked == 0$ **then**
    $\mathsf{X} := \mathsf{X}$ $\{D_i\}$;
    $\mathsf{Y} := \mathsf{Y}$ $\{0\}$;
  end
end

4.6.4 The Anchor Selection Classifier

Let us introduce the notation which we will use in this section.

$(\bar{x}, \bar{y}), \bar{x} \quad \mathsf{R}^{2,5}, \bar{y} \quad \mathsf{R}^{2,5}$ The invariantized points in the first and the second view.

$m \quad \mathsf{N}$ The expected number of anchors.

$\mathcal{A} = (\breve{x}_a, \breve{y}_a, \breve{\lambda}_a), a \quad \{1, ..., m\}, \breve{x}_a \quad \mathsf{R}^{2,5}, \breve{y}_a \quad \mathsf{R}^{2,5}, \breve{\lambda}_a \quad \mathsf{R}^{2,5}$ A sequence of **anchors**, i.e. starting problem-solution pairs. For every a, the values in $\breve{\lambda}_a$ are the solutions to the depth formulation of the Five-Point problem (Equation 2.5) parametrized by $\breve{x}_a, \breve{y}_a$. We assume that the problem $\breve{x}_a, \breve{y}_a$ is invariantized.

$D \quad \mathsf{R}^{14}$ A low-dimensional representation of problem $(\bar{x}, \bar{y})$

$c(D, \theta) : \mathsf{R}^{14} \rightarrow \mathsf{R}^{m+1}$ The classifier which for every low-dimensional representation D of problem $(\bar{x}, \bar{y})$ outputs a vector of probabilities that the problem is successfully tracked from each anchor. θ is a vector of parameters to the classifier.

$s \quad \mathsf{R}^{m+1}$ The vector of the outputs of the classifier $c(D, \theta)$. a is the index of the largest entry of vector s minus 1. If $a = 0$, the problem is not tracked. Otherwise, the problem is tracked from the anchor a .

$\mathcal{X} = D_i, i \quad \{1, ..., n\}, D_i \quad \mathsf{R}^{14}$ The set of input observations for the training of the classifier obtained according to Section 4.6.3. D_i is the 14 dimensional representation of a problem (x_i, y_i) from $\mathcal{P}_t$ obtained by procedure from Section 4.6.2.

$\mathcal{Y} = a_i, i \quad \{1, ..., n\}, a_0 \quad \{0, ..., m\}$ The set of the labels for the training of the classifier obtained according to Section 4.6.3. The problem (x_i, y_i, λ_i) can be correctly tracked from the anchor $(\breve{x}_{a_i}, \breve{y}_{a_i}, \breve{\lambda}_{a_i}) \quad \mathcal{A}$. If the problem cannot be tracked correctly from any anchor in $\mathcal{A}$, then $a_i = 0$.

In this section, we describe the structure and the training of the classifier for the selection of the anchor. The input to the classifier is a 14 dimensional representation $D \quad \mathsf{R}^{14}$ of a problem (x, y) obtained according to Section 4.6.2. The output of the classifier is a vector $s \quad \mathsf{R}^{n+1}$ of probabilities that the problem can be successfully tracked from each anchor. When the vector s is known, we can obtain the index $a \quad \{0, ..., m\}$ of the selected anchor as the index of the largest entry of vector s minus 1. If $a = 0$, the problem is not tracked. Otherwise, the problem is tracked from the anchor a . The

task is to find such parameters θ of the classifier $c(D, \theta)$, that the classifier maximizes the probability that the correct anchor a is selected.

Now, we are going to describe the loss of the classifier. We use the cross-entropy loss. Let s_i R^n be the output of the classifier $c(D_i, \theta)$ on the i-th element of the training data X. Let s_i be a vector with one on the position a_i Y and zeros on the other positions. The loss is defined as:

$$L(\mathsf{X}, \mathsf{Y}, \theta) = -\sum_{i=1}^{n}\sum_{j=1}^{m} s_{i,j} \log s_{i,j} \tag{4.79}$$

Now, we are going to describe the classifier $c(D, \theta)$. We use a small fully connected neural network with 7 linear layers as the classifier. The reason for this is that the typical time for one homotopy continuation track is about 20 to 30 microseconds and the speed of the neural network is expected to be similar or higher. If the neural network was too slow, it would be advantageous to try all anchors instead. Another reason for using the fully connected network is that the entries in the input vector D R^{14} do not form any spatial sequence, therefore, the use of convolutional neural network does not make sense. The size of the input layer is 14. The sizes of the hidden layers are $200, 200, 200, 200, 100, 100$. The size of the output layer is $m + 1$. The nonlinearity used after the linear layers is PReLU. There is a dropout before the last linear layer to prevent overfitting.

Now, we are going to describe how the classifier c is trained. Let us have the training data X, Y generated according to Section 4.6.3. The goal is to find the parameters θ of the classifier $c(D, \theta)$ which would minimize the loss (4.79). This is achieved with the Stochastic Gradient Descent (SGD) optimizer. The classifier is modeled and trained using PyTorch.

4.7 Computation of the Relative Pose

Let us introduce the notation which we will use in this section.

$(x, y), x$ $\mathsf{R}^{2,5}, y$ $\mathsf{R}^{2,5}$ The points in the first and the second view.

$(\bar{x}, \bar{y}), \bar{x}$ $\mathsf{R}^{2,5}, \bar{y}$ $\mathsf{R}^{2,5}$ The invariantized points in the first and the second view.

$(\hat{x}, \hat{y}), \hat{x}$ $\mathsf{R}^{2,5}, \hat{y}$ $\mathsf{R}^{2,5}$ The aligned points in the first and the second view.

$(\hat{x}^h, \hat{y}^h), \hat{x}^h$ $\mathsf{R}^{3,5}, \hat{y}^h$ $\mathsf{R}^{3,5}$ Homogeneous representation of he aligned points $(\hat{x}, \hat{y})$ in the first and the second view.

R_x^I $SO(3)$ The 3D rotation matrix which transforms the homogeneous representation of points x in the first view to the homogeneous representations of the invariantized points $\bar{x}$. (Section 4.4)

R_y^I $SO(3)$ The 3D rotation matrix which transforms the homogeneous representation of points y in the second view to the homogeneous representations of the invariantized points $\bar{y}$. (Section 4.4)

R_x $SO(3)$ The 3D rotation matrix which transforms the homogeneous representation of invariantized points $\bar{x}$ in the first view to the homogeneous representations of the aligned points $\hat{x}$. (Section 4.5)

R_y $SO(3)$ The 3D rotation matrix which transforms the homogeneous representation of invariantized points $\bar{y}$ in the first view to the homogeneous representations of the aligned points $\hat{y}$. (Section 4.5)

s $Sym(\{1, 2\})$ The permutation of the views. $s = \{2, 1\}$ if the views have been swapped during the invariantization, $s = \{1, 2\}$ otherwise.

$\hat{\lambda}$ $\mathsf{R}^{2,5}$ The depths of the aligned points $(\hat{x}, \hat{y})$. This is the output of the homotopy continuation (Algorithm 4)

$(\hat{R}, \hat{t})$ The relative pose of the cameras of the aligned problem $(\hat{x}, \hat{y})$.

$(R\ , t\ )$ The relative pose of the cameras of the original problem (x, y).

$\hat{X}$ $\mathsf{R}^{3,5}$ The coordinates of the 3D points in the coordinate system of the first aligned camera.

$\hat{Y}$ $\mathsf{R}^{3,5}$ The coordinates of the 3D points in the coordinate system of the second aligned camera.

In this section, we are going to describe how the relative pose $(R\ , t\ )$ is obtained from the depths $\hat{\lambda}$. In Section 4.7.1 we describe how the relative pose of the aligned problem $(\hat{x}, \hat{y})$ is obtained and in Section 4.7.2 we describe how the relative pose of the aligned problem is transformed to the relative pose of the original problem (x, y).

4.7.1 Getting the Relative Pose From the Depths

We know the aligned points $(\hat{x}, \hat{y})$ and the depths $\hat{\lambda}$ which are the result of the depth formulation of the Five-Point problem (4.5) parametrized by

the aligned points $(\hat{x}, \hat{y})$. The depths have been obtained by the homotopy continuation (Chapter 3). The goal is to find the rotation $\hat{R}$ and translation $\hat{t}$ consistent with the points and with the depths, i.e. such (R, t), that the equations (2.1), (2.2) hold.

This procedure is taken from [Paj21]. First, we obtain the coordinates of the 3D points $\hat{X}$ in the coordinate system of the first aligned camera and $\hat{Y}$ in the coordinate system of the second camera by multiplying the homogeneous aligned points $\hat{x}^h$, $\hat{y}^h$ by the depths $\hat{\lambda}$ as:

$$\hat{X}_{i,j} = \hat{\lambda}_{1,j}\hat{x}^h, \ \hat{Y}_{i,j} = \hat{\lambda}_{2,j}\hat{y}^h \quad i \quad \{1,2,3\}, \quad j \quad \{1,...,5\} \tag{4.80}$$

We find such rotation $\hat{R}$ and translation $\hat{t}$, that:

$$\begin{bmatrix} \hat{Y}_{1,j} \\ \hat{Y}_{2,j} \\ \hat{Y}_{3,j} \end{bmatrix} = \hat{R} \begin{bmatrix} \hat{X}_{1,j} \\ \hat{X}_{2,j} \\ \hat{X}_{3,j} \end{bmatrix} + \hat{t} \quad j \quad \{1,...,5\} \tag{4.81}$$

Let us denote:

$$\begin{aligned} Z_2 = \begin{bmatrix} \hat{X}_{1,2} \\ \hat{X}_{2,2} \\ \hat{X}_{3,2} \end{bmatrix} - \begin{bmatrix} \hat{X}_{1,1} \\ \hat{X}_{2,1} \\ \hat{X}_{3,1} \end{bmatrix}, \quad Z_3 = \begin{bmatrix} \hat{X}_{1,3} \\ \hat{X}_{2,3} \\ \hat{X}_{3,3} \end{bmatrix} - \begin{bmatrix} \hat{X}_{1,1} \\ \hat{X}_{2,1} \\ \hat{X}_{3,1} \end{bmatrix} \\ Z_2 = \begin{bmatrix} \hat{Y}_{1,2} \\ \hat{Y}_{2,2} \\ \hat{Y}_{3,2} \end{bmatrix} - \begin{bmatrix} \hat{Y}_{1,1} \\ \hat{Y}_{2,1} \\ \hat{Y}_{3,1} \end{bmatrix}, \quad Z_3 = \begin{bmatrix} \hat{Y}_{1,3} \\ \hat{Y}_{2,3} \\ \hat{Y}_{3,3} \end{bmatrix} - \begin{bmatrix} \hat{Y}_{1,1} \\ \hat{Y}_{2,1} \\ \hat{Y}_{3,1} \end{bmatrix} \end{aligned} \tag{4.82}$$

If (4.81) holds, then there holds:

$$\begin{aligned} Z_2 = \begin{bmatrix} \hat{Y}_{1,2} \\ \hat{Y}_{2,2} \\ \hat{Y}_{3,2} \end{bmatrix} - \begin{bmatrix} \hat{Y}_{1,1} \\ \hat{Y}_{2,1} \\ \hat{Y}_{3,1} \end{bmatrix} = \hat{R} \begin{bmatrix} \hat{X}_{1,2} \\ \hat{X}_{2,2} \\ \hat{X}_{3,2} \end{bmatrix} + \hat{t} - \hat{R} \begin{bmatrix} \hat{X}_{1,1} \\ \hat{X}_{2,1} \\ \hat{X}_{3,1} \end{bmatrix} - \hat{t} = \hat{R}Z_2 \\ Z_3 = \begin{bmatrix} \hat{Y}_{1,3} \\ \hat{Y}_{2,3} \\ \hat{Y}_{3,3} \end{bmatrix} - \begin{bmatrix} \hat{Y}_{1,1} \\ \hat{Y}_{2,1} \\ \hat{Y}_{3,1} \end{bmatrix} = \hat{R} \begin{bmatrix} \hat{X}_{1,3} \\ \hat{X}_{2,3} \\ \hat{X}_{3,3} \end{bmatrix} + \hat{t} - \hat{R} \begin{bmatrix} \hat{X}_{1,1} \\ \hat{X}_{2,1} \\ \hat{X}_{3,1} \end{bmatrix} - \hat{t} = \hat{R}Z_3 \end{aligned} \tag{4.83}$$

We search for a rotation $\hat{R}$, such that $Z_2 = \hat{R}Z_2$ and $Z_3 = \hat{R}Z_3$. Because $\hat{R}$ is a rotation, it transforms the cross-product of two vectors to a cross-product of the transformed vectors, i.e.:

$$Z_2 \times Z_3 = \hat{R}(Z_2 \times Z_3) \tag{4.84}$$

Therefore, we can find the rotation $\hat{R}$ as:

$$\begin{aligned} \hat{R} \begin{bmatrix} Z_2 & Z_3 & Z_2 \times Z_3 \end{bmatrix} &= \begin{bmatrix} Z_2 & Z_3 & Z_2 \times Z_3 \end{bmatrix} \\ \hat{R} &= \begin{bmatrix} Z_2 & Z_3 & Z_2 \times Z_3 \end{bmatrix} \begin{bmatrix} Z_2 & Z_3 & Z_2 \times Z_3 \end{bmatrix}^{-1} \end{aligned} \tag{4.85}$$

Then, we can find the translation $\hat{t}$ according to (4.81) as:

$$\hat{t} = \begin{bmatrix} \hat{Y}_{1,1} \\ \hat{Y}_{2,1} \\ \hat{Y}_{3,1} \end{bmatrix} - \hat{R} \begin{bmatrix} \hat{X}_{1,1} \\ \hat{X}_{2,1} \\ \hat{X}_{3,1} \end{bmatrix} \tag{4.86}$$

4.7.2 Passing From the Solution of the Aligned Problem

Now, we are going to describe how the relative pose $(\hat{R}, \hat{t})$ of the aligned problem $(\hat{x}, \hat{y})$ is transformed to the relative pose $(R\ , t\ )$ of the original problem (x, y).

We know the relative pose $(\hat{R}, \hat{t})$ of the aligned problem, the rotation matrices R_x^I, R_y^I transforming the original points to the invariantized points, the rotation matrices R_x, R_y transforming the invariantized points to the aligned points and the permutation $s \quad Sym(\{1, 2\})$ of the views during the invariantization.

If the views have not been swapped ($s = \{1, 2\}$), then the relative rotation R of the original problem is obtained as:

$$R = (R_y^I)^T R_y^T \hat{R} R_x R_x^I \tag{4.87}$$

And the relative translation t of the original problem is obtained as:

$$t = (R_y^I)^T R_y^T \hat{t} \tag{4.88}$$

If, on the other hand, the views have been swapped, i.e., $s = \{2, 1\}$, then the relative rotation R of the original problem is obtained as:

$$R = (R_x^I)^T R_x^T \hat{R}^T R_y R_y^I \tag{4.89}$$

And the relative translation t is obtained as:

$$t = -(R_x^I)^T R_x^T \hat{R}^T \hat{t} \tag{4.90}$$

Chapter 5

Efficient Homotopy Continuation Solver for the Four-Point Problem

5.1 Description of the solver

In this section, we will propose a new solver for the Four-Point calibrated problem (Section 2.2) based on the real homotopy continuation. The solver uses the same principles as the Five-Point solver described in Chapter 4. The solver is meant to be used in the RANSAC loop, therefore, it should be fast but it does not have to finish correctly for all inputs. Moreover, it is desirable not to track the inputs which consist of mismatched points. Therefore, we have decided to use the real homotopy continuation (Section 3.4) in the solver. We use the depth formulation of the relaxed minimal version of the Four-Point problem (Section 2.2.2).

The parametrization of the problem and the solution is described in Section 5.1.1. The homotopy continuation used in the solver is described in Section 5.1.2 and the solver itself is described in Section 5.1.3. Let us introduce the notation which we will use in this section.

x $\mathsf{R}^{2,4}$ Four calibrated 2D points in the first camera of the final problem.

y $\mathsf{R}^{2,4}$ Four calibrated 2D points in the second camera of the final problem.

w $\mathsf{R}^{2,4}$ Four calibrated 2D points in the third camera of the final problem.

λ $\mathsf{R}^{3,4}$ The depths of the points in all three. $\lambda_{j,i}$ is the depth of point i in camera j. λ are the solutions to the depth formulation of the relaxed minimal Four-Point problem (Equation 2.13) parametrized by x, y, w.

l R The oriented distance between the observation $w_{:,4}$ of the last point in the last camera and the actual projection of the last point in the last camera.

$(R_{1,2}, t_{1,2})$ A relative pose between the first and the second camera.

$(R_{1,3}, t_{1,3})$ A relative pose between the first and the third camera.

m The Number of starting problem-solution pairs.

$\mathcal{A} = (\breve{x}_a, \breve{y}_a, \breve{w}_a, \breve{\lambda}_a, \breve{l}_a)_l$ a $\{1, ..., m\}, \breve{x}_a$ $\mathsf{R}^{2,4}, \breve{y}_a$ $\mathsf{R}^{2,4}, \breve{w}_a$ $\mathsf{R}^{2,4}, \breve{\lambda}_a$ $\mathsf{R}^{2,4}, \breve{l}_a$ R A sequence of **anchors**, i.e. starting problem-solution pairs. For every a, the values in $\breve{\lambda}_a$ and $\breve{l}_a$ are the solutions to the depth formulation of the Four-Point problem (Equation 2.13) parametrized by $\breve{x}_a, \breve{y}_a, \breve{w}_a$.

We know the calibrated points x, y, w in all three cameras, as well as the number of anchors m and the anchors (problem-solution pairs) $\mathcal{A}$. The task is to obtain the depths λ , oriented distance l and the relative poses $(R_{1,2}, t_{1,2})$, $(R_{1,3}, t_{1,3})$, such that:

$$\begin{aligned}
\lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} &= \lambda_{2,i} R_{1,2} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} + t_{1,2}, \quad i \quad \{1, ..., 4\} \\
\lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} &= \lambda_{3,i} R_{1,3} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} + t_{1,3}, \quad i \quad \{1, 2, 3\} \\
\lambda_{1,4} \begin{bmatrix} x_{1,4} \\ x_{2,4} \\ 1 \end{bmatrix} &= \lambda_{3,4} R_{1,3} \left(\begin{bmatrix} w_{1,4} \\ w_{2,4} \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ l \\ 0 \end{bmatrix} \right) + t_{1,2}
\end{aligned} \tag{5.1}$$

Structure of the set of solutions

The relaxed minimal Four-Point problem (Section 2.2.1) has 272 solutions. Some of these solutions are often similar but not equal to the expected solution. In addition to that, if the views or the points are permuted or if the last view is rotated, then the relative position of the line towards the points

is changed, which may change the solution, as well. Therefore, the definition of a **correct solution** is more relaxed when compared with the Five-Point problem solver. Namely, the solution is viewed as correct, if the distance (6.3) between the expected relative poses $(R_{1,2})$, $(R_{1,3})$ and the relative poses obtained by the solver is smaller than 5 degrees.

5.1.1 Parametrization of the problem and the solution

Now, we are going to describe the parametrizations of the problem and of the solution vector. Let us introduce the notation which we will use in this section.

p_s R^{24} The parametrization of the initial problem.

p_f R^{24} The parametrization of the final problem.

$h(z, p)$ A square parametric system of polynomials from the depth formulation of the Four-Point problem (Section 2.2.2).

z_0 R^{12} The solution to the initial problem parametrized by p_s. There holds $h(z_0, p_s) = 0$.

The parametrization of the instance of a Four-Point problem is a twenty-four-dimensional vector p. This vector contains flattened projections x, y, w which define the instance, i.e., the first four elements are the x-coordinates of the points in the first camera, the next four elements are the y-coordinates of the points in the first camera. The next eight elements are the coordinates of the points in the second camera in the same order, and the last eight elements are the coordinates of the points in the third camera. Let us have an instance $x \quad \mathsf{R}^{2,4}, y \quad \mathsf{R}^{2,4}, w \quad \mathsf{R}^{2,4}$. The parametrization p of the instance is:

$$i \quad \{1, ..., 4\} :$$
$$p_i = x_{1,i}, p_{i+4} = x_{2,i}, p_{i+8} = y_{1,i}, p_{i+12} = y_{2,i}, p_{i+16} = w_{1,i}, p_{i+20} = w_{2,i} \tag{5.2}$$

The depth formulation (2.13) of the Four-Point problem is scale-invariant, therefore, the scale of the depths λ has to be fixed in order for the system of equations (2.13) to have a finite number of solutions. We fix the depths by setting the first depth in the first image $\lambda_{1,1}$ to 1. The last element of

the solution vector z is the oriented depth l. The solution vector z R^{12} is therefore:

$$\begin{aligned} i \quad \{1,...3\} : z_i &= \frac{\lambda_{1,i+1}}{\lambda_{1,1}} \\ i \quad \{1,...4\} : z_{i+3} &= \frac{\lambda_{2,i}}{\lambda_{1,1}} \\ i \quad \{1,...4\} : z_{i+7} &= \frac{\lambda_{3,i}}{\lambda_{1,1}} \\ z_{12} &= l \end{aligned} \tag{5.3}$$

The depths λ $\mathsf{R}^{3,4}$ of the problem may be recovered from the solution vector z R^{12} up to scale by setting $\lambda_{1,1}$ to one and by copying the values from the solution vector as:

$$\begin{aligned} i \quad \{1,...3\} : \lambda_{1,i+1} &= z_i \\ i \quad \{1,...4\} : \lambda_{2,i} &= z_{i+3} \\ i \quad \{1,...4\} : \lambda_{3,i} &= z_{i+7} \end{aligned} \tag{5.4}$$

5.1.2 Homotopy continuation used in the solver

Now, we are going to describe the parametric homotopy $H(z,p,t)$ used to track the solution z of the depth formulation (2.13) of the relaxed Four-Point problem. The solution vector z has 12 elements but there are 18 equations in the depth formulation (2.13). The last 6 equations, which relate the second and the third views, may be expressed from the first 12 equations. If we drop these last 6 equations, we obtain the following square system:

$$\left\| \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} - \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} \right\|^2_i = \left\| \lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} - \lambda_{2,i} \begin{bmatrix} y_{1,i} \\ y_{2,i} \\ 1 \end{bmatrix} \right\|^2_{\{1,...,4\},i \quad \{1,...,4\},i<i}$$

$$\left\| \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} - \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} \right\|^2_i = \left\| \lambda_{3,i} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} - \lambda_{3,i} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} \right\|^2_{\{1,...,3\},i \quad \{1,...,3\},i<i} \tag{5.5}$$

$$\left\| \lambda_{1,i} \begin{bmatrix} x_{1,i} \\ x_{2,i} \\ 1 \end{bmatrix} - \lambda_{1,4} \begin{bmatrix} x_{1,4} \\ x_{2,4} \\ 1 \end{bmatrix} \right\|^2 = \left\| \lambda_{3,i} \begin{bmatrix} w_{1,i} \\ w_{2,i} \\ 1 \end{bmatrix} - \lambda_{3,4} \left(\begin{bmatrix} w_{1,4} \\ w_{2,4} \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ l \\ 0 \end{bmatrix} \right) \right\|^2_{i \quad \{1,...,3\}}$$

If the values from x, y, w, λ, l are replaced by the corresponding values of the parametrization p and solution vector z according to (5.2) and (5.3), we obtain the square parametrized system of polynomial equations $h(z, p)$. Because the parametrization of the start problem is p_s and the parametrization of the final problem is p_f, we can set the parametric homotopy $H(z, p, t)$ according to:

$$H(z, p, t) = h(z, (1 - t)p_s + tp_f) \tag{5.6}$$

The tracking of the homotopy $H(z, p, t)$ with the start parametrization z_0 is performed according to 4. The derivatives $H_z(z(t), p, t)$ and $H_z(z(t), p, t)$ of the homotopy are obtained with a Straight-Line Program (SLP) generated in Macaulay2.

The largest amount of time on the solver is spent on the solution of the linear equations (3.9), (3.7), (3.5) which are solved as a part of the predictor and the corrector. The left side of the equations is always the Jacobian matrix $H_z(z(t), p, t)$ of the value of the homotopy w.r.t. the variables z. Because the matrix $H_z(z(t), p, t)$ is relatively sparse, the closed form of the solution can be obtained, which can significantly reduce the time spent on one track. This closed-form solution is described in Section 5.2.

5.1.3 Overview of the solver

Let us introduce the notation which we will use in this section.

x $\mathsf{R}^{2,4}$ Four calibrated 2D points in the first camera of the final problem.

y $\mathsf{R}^{2,4}$ Four calibrated 2D points in the second camera of the final problem.

w $\mathsf{R}^{2,4}$ Four calibrated 2D points in the third camera of the final problem.

λ $\mathsf{R}^{2,5}$ The depths of the points in all three cameras. $\lambda_{j,i}$ is the depth of point i in camera j. λ are the solutions to the depth formulation of the relaxed Four-Point problem (Equation 2.13) parametrized by x, y, w.

l R The oriented distance between the observation $w_{:,4}$ of the last point in the last camera and the actual projection of the last point in the last camera. Together with the depths λ , this is the solution to the relaxed Four-Point problem parametrized by x. y, w.

$(R_{1,2}, t_{1,2})$ A relative pose between the first and the second camera.

$(R_{1,3}, t_{1,3})$ A relative pose between the first and the third camera.

m The number of starting problem-solution pairs.

$A = (\breve{x}_a, \breve{y}_a, \breve{w}_a, \breve{\lambda}_a, \breve{l}_a)_l$ a $\{1, ..., m\}$, $\breve{x}_a$ $\mathsf{R}^{2,4}$, $\breve{y}_a$ $\mathsf{R}^{2,4}$, $\breve{w}_a$ $\mathsf{R}^{2,4}$, $\breve{\lambda}_a$ $\mathsf{R}^{2,4}$, $\breve{l}_a$ R A sequence of **anchors**, i.e. starting problem-solution pairs. For every a, the values in $\breve{\lambda}_a$ and $\breve{l}_a$ are the solutions to the depth formulation of the Four-Point problem (Equation 2.13) parametrized by $\breve{x}_a, \breve{y}_a, \breve{w}_a$.

$\bar{x}$ $\mathsf{R}^{2,4}$ Points in the first view invariantized according to Section 5.4.

$\bar{y}$ $\mathsf{R}^{2,4}$ Points in the second view invariantized according to Section 5.4.

$\bar{w}$ $\mathsf{R}^{2,4}$ Points in the third view invariantized according to Section 5.4.

D R^{15} Low-dimensional representation of the problem defined by $(\bar{x}, \bar{y}, \bar{w})$. The low-dimensional representation is obtained according to Section 5.6.2.

L $\mathsf{A}^{15,24}$ A transformation matrix which transforms a 24-dimensional invariantized representation p to a 15-dimensional representation D.

a $\{0, ..., m\}$ Index of the selected anchor. If $a = 0$, the problem is not tracked, otherwise, it is tracked from anchor a .

$c : \mathsf{R}^{15} \to \{0, ..., m\}$ The classifier which for a low-dimensional representation D of a problem gives the index a of the selected anchor. The classifier is described in Section 5.6.4.

$\hat{x}$ $\mathsf{R}^{2,4}$ Points in the first view aligned to the selected anchor a according to Section 4.5.

$\hat{y}$ $\mathsf{R}^{2,4}$ Points in the second view aligned to the selected anchor a according to Section 4.5.

$\hat{w}$ $\mathsf{R}^{2,4}$ Points in the second view aligned to the selected anchor a according to Section 4.5.

$H(z,p,t)$ Parametric homotopy (5.6) for the square system (5.5). z R^{12} is the parametrization of the solution, p R^{24} is the parametrization of the problem and t $< 0, 1 >$ is the time parameter.

p_s R^{24} A parametrization (5.2) of the starting problem $\breve{x}_a$, $\breve{y}_a$, $\breve{w}_a$

z_0 R^{12} A parametrization (5.3) of the starting solution $\breve{\lambda}_a$, $\breve{l}_a$.

p_f R^{24} A parametrization (5.2) of the aligned final problem $\hat{x}$, $\hat{y}, \hat{w}$.

$\hat{\lambda}$ The depths of the aligned problem defined by $\hat{x}$, $\hat{y}$, $\hat{w}$.

$\hat{l}$ R The oriented distance between the observation $\hat{w}_{:,4}$ of the last point in the last camera and the actual projection of the last point onto the last camera.

Now, we are going to describe the solver. The solver is based on the same principles as the solver for the Five-Point problem, which is described in Chapter 4. The solver operates in real numbers only and tracks only one solution per problem. The predictor and corrector steps of the homotopy continuation used in the solver use a closed-form solution of the linear equations (3.9), (3.7), (3.5), which is described in Section 5.2. Before the solver can be used, the set of anchors $\mathcal{A}$ (starting problem-solution pairs) has to be generated and the classifier c has to be trained. The generating of the anchors is described in Section 5.6.1, the training of the classifier c is described in Section 5.6.4.

First, the invariantized representation $\bar{x}, \bar{y}, \bar{w}$ of points x, y, w is obtained (Section 5.4) in order to have a unified representation of the problems, which can simplify the training of the anchor selector and the alignment. Then, one anchor a $\{1, ..., m\}$ is selected (Section 5.6). The aligned points $\hat{x}, \hat{y}, \hat{w}$ are obtained by permutation of the invariantized points $\bar{x}, \bar{y}, \bar{w}$ in order to minimize the Euclidean distance from the selected anchor $\breve{x}_a$, $\breve{y}_a$, $\breve{w}_a$ (Section 5.5), and therefore, to increase the probability of the successful track from the selected anchor.

After that, the parametrizations p_s of the start problem $\breve{x}_a$, $\breve{y}_a$, $\breve{w}_a$ and p_f of the aligned final problem $\hat{x}, \hat{y}, \hat{w}$ are obtained by (5.2) and the parametrization z_0 of the start solution $\breve{\lambda}_a$, $\breve{l}_a$ is obtained by (5.3). Then, the parametric

homotopy $H(z, p, t)$ from Equation (5.6) is set up and tracked from z_0. If the track is successful, the solution $\hat{\lambda}$, $\hat{l}$ is obtained from the final solution $z(1)$ by (5.4) and the relative pose (R , t) is obtained by the procedure described in Section 5.7. The solver is described in Algorithm 12.

Algorithm 12: Five-Point solver

input : Four points x, y, w in three views, Anchors A obtained according to Section 5.6.1, Transformation matrix L (Section 5.6.2), classifier c trained by Section 5.6.4

output : Relative poses $(R_{1,2}, t_{1,2})$, $(R_{1,3}, t_{1,3})$, $(R_{2,3}, t_{2,3})$ between x, y, w

`/* Invariantize the problem */`

$\bar{x}, \bar{y}, \bar{w}$:= Invariantized representation of x, y, w obtained by Sec. 5.4;
R_x^I, R_y^I, R_w^I := Rotation matrices induced by the invariantization;
s := Permutation of the views induced by the invariantization;

`/* Select the starting anchor */`

D := Low-dim representation of x, y, w (by Section 5.6.2, use L);
a := $c(D)$; `// Select the starting point`
if a $= 0$ **then**
  return
end

`/* Align the problem */`

$\hat{x}, \hat{y}, \hat{w}$:= Points from x, y, w aligned to anchor $\breve{x}_a, \breve{y}_a, \breve{w}_a$ by Sec. 5.5;

`/* Parametrize the problems and the starting solution */`

p_s := parametrization of $\breve{x}_a$, $\breve{y}_a$, $\breve{w}_a$ by (5.2);
p_f := parametrization of $\hat{x}, \hat{y}, \hat{w}$ by (5.2);
z_0 := parametrization of $\breve{\lambda}_a$, $\breve{l}_a$ by (5.3);

`/* TRACK the solution */`

z_1 := Track homotopy (5.6) from z_0 by Algorithm 4 (Chapter 3);

`/* Extract the depths and the relative poses */`

if $z_1 =$ **then**
  $\hat{\lambda}$:= Extract depths from z_1 by (5.4);
  $(R_{1,2}, t_{1,2}), (R_{1,3}, t_{1,3}), (R_{2,3}, t_{2,3})$:= Get relative poses from z_1, R_x^I, R_y^I, R_w^I by Sec. 5.7;
end

5.2 Efficient evaluation of the predictor and the corrector

We have noticed that the largest amount of time of the MINUS solver [FDF+20] is spent on the LU-decomposition, which is used to solve linear equations (3.7), (3.9) which arise in the predictor and the corrector steps of the homotopy continuation. We have solved this in the same way as in the Five-Point problem solved (Chapter 4) and we have replaced the LU-decomposition with the closed-form solution of the linear equations, which exploits the sparsity of the linear equations in the relaxed Four-Point problem. If the method described in this section is used, the average time for one track is $36.1\mu s$. If the LU decomposition is used, the average time for one track is $172.7\mu s$.

5.2.1 Structure of the Linear Equations in the Three-View Problem

The linear equations (3.7), (3.9) are computed in order to perform the predictor and corrector steps of the homotopy continuation. The linear equations have the form $Ax = b$ where the matrix A is always the derivative $H_z(z, p, t)$ of the homotopy $H(z, p, t)$ (5.6) w.r.t. the solution parameters z. Irrespective of the parameters z, p, t, the matrix $H_z(z, p, t)$ has the following form:

$$\begin{bmatrix}
A_{0,0} & 0 & 0 & A_{0,3} & A_{0,4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & A_{1,1} & 0 & A_{1,3} & 0 & A_{1,5} & 0 & 0 & 0 & 0 & 0 & 0 \\
A_{2,0} & A_{2,1} & 0 & 0 & A_{2,4} & A_{2,5} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & A_{3,2} & A_{3,3} & 0 & 0 & A_{3,6} & 0 & 0 & 0 & 0 & 0 \\
A_{4,0} & 0 & A_{4,2} & 0 & A_{4,4} & 0 & A_{4,6} & 0 & 0 & 0 & 0 & 0 \\
0 & A_{5,1} & A_{5,2} & 0 & 0 & A_{5,5} & A_{5,6} & 0 & 0 & 0 & 0 & 0 \\
A_{6,0} & 0 & 0 & 0 & 0 & 0 & 0 & A_{6,7} & A_{6,8} & 0 & 0 & 0 \\
0 & A_{7,1} & 0 & 0 & 0 & 0 & 0 & A_{7,7} & 0 & A_{7,9} & 0 & 0 \\
A_{8,0} & A_{8,1} & 0 & 0 & 0 & 0 & 0 & 0 & A_{8,8} & A_{8,9} & 0 & 0 \\
0 & 0 & A_{9,2} & 0 & 0 & 0 & 0 & A_{9,7} & 0 & 0 & A_{9,10} & A_{9,11} \\
A_{10,0} & 0 & A_{10,2} & 0 & 0 & 0 & 0 & 0 & A_{10,8} & 0 & A_{10,10} & A_{10,11} \\
0 & A_{11,1} & A_{11,2} & 0 & 0 & 0 & 0 & 0 & 0 & A_{11,9} & A_{11,10} & A_{11,11}
\end{bmatrix} \tag{5.7}$$

The vector x of the variables can be written as:

$$\begin{bmatrix} x_0 & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} & x_{11} \end{bmatrix}^T \tag{5.8}$$

The vector b of the right side can be written as:

$$\begin{bmatrix} b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7 & b_8 & b_9 & b_{10} & b_{11} \end{bmatrix}^T \tag{5.9}$$

5.2.2 Closed-Form Solution of the Linear Equations in the Three-View Problem

Let us have a system of linear equations $Ax = b$ whose matrix A has the form of (5.7), vector x has the form of (5.8) and vector b has the form of (5.9). Now, we are going to show the closed-form solution to this system.

First, we have noticed that the rows $0, 1, 3, 6, 7$ of matrix (5.7) have only three nonzero entries. We can easily obtain the variables x_4, x_5, x_6, x_8, x_9 from these equations and substitute them to the rest of the equations to obtain a square system with 7 variables. The variables x_4, x_5, x_6, x_8, x_9 are obtained as:

$$\begin{aligned} x_4 &= \frac{b_0 - A_{0,0}x_0 - A_{0,3}x_3}{A_{0,4}} \\ x_5 &= \frac{b_1 - A_{1,1}x_1 - A_{1,3}x_3}{A_{1,5}} \\ x_6 &= \frac{b_3 - A_{3,2}x_2 - A_{3,3}x_3}{A_{3,6}} \\ x_8 &= \frac{b_6 - A_{6,0}x_0 - A_{6,7}x_7}{A_{6,8}} \\ x_9 &= \frac{b_7 - A_{7,1}x_1 - A_{7,7}x_7}{A_{7,9}} \end{aligned} \tag{5.10}$$

Now, we are going to split the remaining equations into two groups. The first group consists of equations 2, 4, 5, the second one consists of equations 8, 9, 10, 11. After substituting the expressions from (5.10) into the first group of equations, we obtain:

$$\begin{aligned} A_{2,0}x_0 + A_{2,1}x_1 + \frac{A_{2,4}}{A_{0,4}}(b_0 - A_{0,0}x_0 - A_{0,3}x_3) + \frac{A_{2,5}}{A_{1,5}}(b_1 - A_{1,1}x_1 - A_{1,3}x_3) &= b_2 \\ A_{4,0}x_0 + A_{4,2}x_2 + \frac{A_{4,4}}{A_{0,4}}(b_0 - A_{0,0}x_0 - A_{0,3}x_3) + \frac{A_{4,6}}{A_{3,6}}(b_3 - A_{3,2}x_2 - A_{3,3}x_3) &= b_4 \\ A_{5,1}x_1 + A_{5,2}x_2 + \frac{A_{5,5}}{A_{1,5}}(b_1 - A_{1,1}x_1 - A_{1,3}x_3) + \frac{A_{5,6}}{A_{3,6}}(b_3 - A_{3,2}x_2 - A_{3,3}x_3) &= b_5 \end{aligned} \tag{5.11}$$

These three equations can be rewritten as:

$$\begin{aligned}
\left(A_{2,0} - \frac{A_{2,4}A_{0,0}}{A_{0,4}}\right)x_0 + \left(A_{2,1} - \frac{A_{2,5}A_{1,1}}{A_{1,5}}\right)x_1 + \left(-\frac{A_{2,4}A_{0,3}}{A_{0,4}} - \frac{A_{2,5}A_{1,3}}{A_{1,5}}\right)x_3 \\
= b_2 - \frac{A_{2,4}}{A_{0,4}}b_0 - \frac{A_{2,5}}{A_{1,5}}b_1 \\
\left(A_{4,0} - \frac{A_{4,4}A_{0,0}}{A_{0,4}}\right)x_0 + \left(A_{4,2} - \frac{A_{4,6}A_{3,2}}{A_{3,6}}\right)x_2 + \left(-\frac{A_{4,4}A_{0,3}}{A_{0,4}} - \frac{A_{4,6}A_{3,3}}{A_{3,6}}\right)x_3 \\
= b_4 - \frac{A_{4,4}}{A_{0,4}}b_0 - \frac{A_{4,6}}{A_{3,6}}b_3 \\
\left(A_{5,1} - \frac{A_{5,5}A_{1,1}}{A_{1,5}}\right)x_1 + \left(A_{5,2} - \frac{A_{5,6}A_{3,2}}{A_{3,6}}\right)x_2 + \left(-\frac{A_{5,5}A_{1,3}}{A_{1,5}} - \frac{A_{5,6}A_{3,3}}{A_{3,6}}\right)x_3 \\
= b_5 - \frac{A_{5,5}}{A_{1,5}}b_1 - \frac{A_{5,6}}{A_{3,6}}b_3
\end{aligned} \tag{5.12}$$

We can simplify these equations by replacing the coefficients by $C_{i,j}$ and the right sides by d_i, where $C_{i,j}$ is the coefficient in equation i at x_j and d_i is the right side of equation i. We obtain:

$$\begin{aligned}
C_{0,0}x_0 + C_{0,1}x_1 + C_{0,3}x_3 &= d_0 \\
C_{1,0}x_0 - C_{1,2}x_2 + C_{1,3}x_3 &= d_1 \\
C_{2,1}x_1 + C_{2,2}x_2 + C_{2,3}x_3 &= d_2
\end{aligned} \tag{5.13}$$

Now, we are going to express the variables x_1, x_2 from the first two equations from (5.13):

$$\begin{aligned}
x_1 &= \frac{d_0 - C_{0,0}x_0 - C_{0,3}x_3}{C_{0,1}} \\
x_2 &= \frac{d_1 - C_{1,0}x_0 - C_{1,3}x_3}{C_{1,2}}
\end{aligned} \tag{5.14}$$

If we substitute these expressions to the third equation from (5.13), we obtain:

$$\frac{C_{2,1}}{C_{0,1}}(d_0 - C_{0,0}x_0 - C_{0,3}x_3) + \frac{C_{2,2}}{C_{1,2}}(d_1 - C_{1,0}x_0 - C_{1,3}x_3) + C_{2,3}x_3 = d_2 \tag{5.15}$$

We can rewrite this equation as:

$$-\left(\frac{C_{2,1}}{C_{0,0}}C_{0,1} + \frac{C_{2,2}}{C_{1,0}}C_{1,2}\right)x_0 + \left(-\frac{C_{2,1}C_{0,3}}{C_{0,1}} - \frac{C_{2,2}}{C_{1,3}}C_{1,2} + C_{2,3}\right)x_3 = d_2 - \frac{C_{2,1}}{C_{0,1}}d_0 - \frac{C_{2,2}}{C_{1,2}}d_1 \tag{5.16}$$

And we can simplify it by replacing the coefficients with $e_{3,0}$, $e_{3,3}$ and the right side with f_3 to obtain:

$$-e_{3,0}x_0 + e_{3,3}x_3 = f_3 \tag{5.17}$$

Now, we can express the variable x_3 from the equation as:

$$x_3 = \frac{e_{3,0}x_0 + f_3}{e_{3,3}} \tag{5.18}$$

Now, we are going to substitute the expression (5.18) into the expressions (5.14). Using this, we are able to express the variables x_1, x_2 using only x_0 as:

$$\begin{aligned} x_1 &= \frac{d_0 - C_{0,0}x_0}{C_{0,1}} - \frac{C_{0,3}(e_{3,0}x_0 + f_3)}{C_{0,1}e_{3,3}} \\ x_2 &= \frac{d_1 - C_{1,0}x_0}{C_{1,2}} - \frac{C_{1,3}(e_{3,0}x_0 + f_3)}{C_{1,2}e_{3,3}} \end{aligned} \tag{5.19}$$

We can rewrite these equations as:

$$\begin{aligned} x_1 &= \left(-\frac{C_{0,0}}{C_{0,1}} - \frac{C_{0,3}e_{3,0}}{C_{0,1}e_{3,3}}\right)x_0 + \left(\frac{d_0}{C_{0,1}} - \frac{C_{0,3}f_3}{C_{0,1}e_{3,3}}\right) \\ x_2 &= \left(-\frac{C_{1,0}}{C_{1,2}} - \frac{C_{1,3}e_{3,0}}{C_{1,2}e_{3,3}}\right)x_0 + \left(\frac{d_1}{C_{1,2}} - \frac{C_{1,3}f_3}{C_{1,2}e_{3,3}}\right) \end{aligned} \tag{5.20}$$

And we can simplify them by replacing the coefficients by e_1, e_2 and the constant terms by f_1, f_2 to obtain:

$$\begin{aligned} x_1 &= e_1x_0 + f_1 \\ x_2 &= e_2x_0 + f_2 \end{aligned} \tag{5.21}$$

Now, we can return back to the second group of the original equation $Ax = b$, which consists of the equations number 8, 9, 10, 11. First, we are going to substitute the expressions from (5.10), (5.21) into the equation 8 to obtain:

$$(A_{8,0}+A_{8,1}e_1)x_0+A_{8,1}f_1+\frac{A_{8,8}}{A_{6,8}}(b_6-A_{6,0}x_0-A_{6,7}x_7)+\frac{A_{8,9}}{A_{7,9}}(b_7-A_{7,1}(e_1x_0+f_1)-A_{7,7}x_7) = b_8 \tag{5.22}$$

We can rewrite this equation as:

$$\begin{aligned} \left(-\frac{A_{8,8}}{A_{6,8}}A_{6,7} - \frac{A_{8,9}}{A_{7,9}}A_{7,7}\right)x_7 = &\left(-A_{8,0} - A_{8,1}e_1 + \frac{A_{8,8}}{A_{6,8}}A_{6,0} + \frac{A_{8,9}}{A_{7,9}}A_{7,1}e_1\right)x_0 \\ &+ \left(b_8 - A_{8,1}f_1 - \frac{A_{8,8}}{A_{6,8}}b_6 - \frac{A_{8,9}}{A_{7,9}}(b_7 - A_{7,1}f_1)\right) \end{aligned} \tag{5.23}$$

And we can simplify it by replacing the coefficients with $C_{4,0}$, $C_{4,7}$ and the constant term with d_4 to obtain:

$$C_{4,7}x_7 = C_{4,0}x_0 + d_4 \tag{5.24}$$

We can express the variable x_7 as:

$$x_7 = \frac{C_{4,0}x_0 + d_4}{C_{4,7}} \tag{5.25}$$

We are going to substitute the expressions (5.10), (5.21), (5.25) into the last remaining equations from $Ax = b$, i.e. to the equations 9, 10, 11. After that, we obtain:

$$A_{9,2}(e_2x_0 + f_2) + \frac{C_{9,7}}{C_{4,7}}(C_{4,0}x_0 + d_4) + A_{9,10}x_{10} + A_{9,11}x_{11} = b_9$$

$$A_{10,0}x_0 + A_{10,2}(e_2x_0 + f_2) + \frac{A_{10,8}}{A_{6,8}}(b_6 - A_{6,0}x_0 - \frac{A_{6,7}}{C_{4,7}}(C_{4,0}x_0 + d_4)) + A_{10,10}x_{10} + A_{10,11}x_{11} = b_{10}$$

$$A_{11,1}(e_1x_0 + f_1) + A_{11,2}(e_2x_0 + f_2) + A_{11,10}x_{10} + A_{11,11}x_{11} + \frac{A_{11,9}}{A_{7,9}}\left(b_7 - A_{7,1}(e_1x_0 + f_1) - \frac{A_{7,7}}{A_{4,7}}(C_{4,0}x_0 + d_4)\right) = b_{11} \tag{5.26}$$

We group all coefficients belonging to the same variable together to obtain:

$$\left(A_{9,2}e_2 + \frac{A_{9,7}C_{4,0}}{C_{4,7}}\right)x_0 + A_{9,10}x_{10} + A_{9,11}x_{11} = b_9 - A_{9,2}f_2 - \frac{A_{9,7}}{C_{4,7}}d_4$$

$$\left(A_{10,0} + A_{10,2}e_2 - \frac{A_{10,8}A_{6,0}}{A_{6,8}} - \frac{A_{10,8}A_{6,7}C_{4,0}}{A_{6,8}C_{4,7}}\right)x_0 + A_{10,10}x_{10} + A_{10,11}x_{11} = b_{10} - A_{10,2}f_2 - \frac{A_{10,8}}{A_{6,8}}b_6 + \frac{A_{10,8}A_{6,7}}{A_{6,8}c_{4,7}}d_4$$

$$\left(A_{11,1}e_1 + A_{11,2}e_2 - \frac{A_{11,9}A_{7,1}e_1}{A_{7,9}} - \frac{A_{11,9}A_{7,7}C_{4,0}}{A_{7,9}C_{4,7}}\right)x_0 + A_{11,10}x_{10} + A_{11,11}x_{11} = b_{11} - A_{11,1}f_1 - A_{11,2}f_2 - \frac{A_{11,9}}{A_{7,9}}b_7 + \frac{A_{11,9}A_{7,1}}{A_{7,9}}f_1 + \frac{A_{11,9}A_{7,7}}{A_{7,9}C_{4,7}}d_4 \tag{5.27}$$

This is a system of 3 equations with 3 variables. We can simplify it by replacing the coefficients of the variable x_0 with $C_{i,0}$ and the right sides with d_i, where $C_{i,0}$ is the coefficient in equation i at x_0 and d_i is the right side of equation i. We obtain:

$$\begin{aligned} C_{9,0}x_0 + A_{9,10}x_{10} + A_{9,11}x_{11} &= d_9 \\ C_{10,0}x_0 + A_{10,10}x_{10} + A_{10,11}x_{11} &= d_{10} \\ C_{11,0}x_0 + A_{11,10}x_{10} + A_{11,11}x_{11} &= d_{11} \end{aligned} \tag{5.28}$$

We express the variable x_{11} from the first equation of (5.28) as:

$$x_{11} = \frac{d_9 - c_{9,0}x_0 - A_{9,10}x_{10}}{A_{9,11}} \tag{5.29}$$

Then, we substitute the expression (5.29) to the second equation of (5.28) to obtain:

$$\left(C_{10,0} - \frac{A_{10,11}c_9}{A_{9,11}}\right)x_0 + \left(C_{10,10} - \frac{A_{10,11}A_{9,10}}{A_{9,11}}\right)x_{10} = d_{10} - \frac{A_{10,11}d_9}{A_{9,11}} \quad (5.30)$$

We simplify this equation by replacing the variables with $e_{0,10}$, $e_{10,10}$ and the right side with f_{10}:

$$e_{0,10}x_0 + e_{10,10}x_{10} = f_{10} \quad (5.31)$$

We express the variable x_{10} from this equation as:

$$x_{10} = \frac{f_{10} - e_{0,10}x_0}{e_{10,10}} \quad (5.32)$$

Now, we substitute the expressions (5.29) and (5.32) to the last equation from (5.28) to obtain an equation in one variable x_0:

$$\begin{aligned}\left(C_{11,0} - \frac{A_{11,10}e_{0,10}}{e_{10,10}} - \frac{A_{11,11}C_{9,0}}{A_{9,11}} + \frac{A_{11,11}A_{9,10}e_{0,10}}{A_{9,11}e_{10,10}}\right)x_0 \\ = d_{11} - \frac{A_{11,10}f_{10}}{e_{10,10}} - \frac{A_{11,11}d_9}{A_{9,11}} + \frac{A_{11,11}A_{9,10}f_{10}}{A_{9,11}e_{10,10}}\end{aligned} \quad (5.33)$$

We replace the coefficient in this equation with g and the right side with h and obtain the variable x_0 as:

$$x_0 = \frac{g}{h} \quad (5.34)$$

The full evaluation of the vector x (5.8) goes as follows: first, the coefficients $C_{i,j}$ and the right sides d_i of the equations (5.13) are evaluated. Then, the coefficients $e_{3,0}$, $e_{3,3}$ and the constant f_3 are obtained from 5.17 and the coefficients e_1, e_2 and constants f_1, f_2 are obtained from (5.21). The coefficients $C_{4,0}$, $C_{4,7}$ and the constant d_4 are obtained from (5.24). After that, the coefficients $C_{i,0}$ and the right sides d_i are obtained from (5.28), the coefficients $e_{0,10}$, $e_{10,10}$ and the right side f_{10} are obtained from (5.31) and the coefficients g, h are obtained from (5.34).

After that, the variable x_0 is obtained from the equation (5.34). The variable x_{10} is obtained from (5.32) and the variable x_{11} is obtained from (5.29). Then, the variable x_7 is obtained from (5.25), variable x_3 is obtained from (5.18) and variables x_1, x_2 are obtained from (5.21). Finally, we evaluate the equations (5.10) to obtain the variables x_4, x_5, x_6, x_8, x_9.

5.3 Optimization of the homotopy continuation parameters

Now, we are going to show the parameters of the homotopy continuation described in Chapter 3. We have set the parameters empirically in order to increase the number of correct tracks and decrease the time for one track. The parameters are set as follows:

- $\Delta t_0 = 0.05$
- $k_{max} = 9$
- $\epsilon = 4e - 2$
- $succ_{max} = 4$
- $\beta = 3$
- $\Delta_{min} = 1e - 4$

5.4 Invariantization of the problems

Now, we are going to describe the invariantization of the problems. The purpose of the invariantization is to obtain a unified representation of the problems, to make the problems "more similar" and to put the last point, through which the line passes to x-axis, which would increase the probability of a successful track. If the depths λ $\mathsf{R}^{3,4}$ and oriented distance l R are known, the invariantization changes them as well. Let us introduce the notation which we will use in this section.

x	$\mathsf{R}^{2,4}$	Four calibrated 2D points in the first camera.
y	$\mathsf{R}^{2,4}$	Four calibrated 2D points in the second camera.
w	$\mathsf{R}^{2,4}$	Four calibrated 2D points in the third camera.
λ	$\mathsf{R}^{2,5}$	The depths of the points in both cameras.
l	R	The oriented distance between the observation $w_{:,4}$ and the projection of the last point onto the last camera.

$\bar{x}$ $\mathsf{R}^{2,4}$ Invariantized representation of points x.

$\bar{y}$ $\mathsf{R}^{2,4}$ Invariantized representation of points y.

$\bar{w}$ $\mathsf{R}^{2,4}$ Invariantized representation of points w.

R_x^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of x to the homogeneous coordinates of $\bar{x}$.

R_y^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of y to the homogeneous coordinates of $\bar{y}$.

R_w^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of w to the homogeneous coordinates of $\bar{w}$.

s $Sym(\{1,2,3\})$ The permutation of the views. The new view at position i is equal to the old view at position $s(i)$.

We know the points x, y, w. The task is to obtain invariantized points $\bar{x}$, $\bar{y}$, $\bar{w}$, such that the center of mass of the points in every view is zero, the point farthest from the center of mass lies on x-axis, and the rest of the points are ordered counterclockwise in the last view. The views should be ordered in such a way that the first view contains the point farthest from the center of mass. The two following views are ordered according to the distance of this point from the center of mass. Furthermore, the task is to obtain the rotation matrices R_x^I, R_y^I, R_z^I, and the permutation s of the views. The depths λ and the oriented distance l change if the invariantization is applied. However, when the problem-solution pairs are generated (Section 2.2.3), the values λ, l are determined after the invariantization and therefore, we do not need to know the way the change of λ, l induced by the invariantization. The invariantization is described in Algorithm 13.

Algorithm 13: Invariantize Five-Point problem

input : Four points x, y, w in three views
output : Invariantized representation $\bar{x}, \bar{y}, \bar{w}$ of x, y, w, Rotation matrices R_x^I, R_y^I, R_w^I, Permutations s, r
$x^{h,\mu}, y^{h,\mu}, w^{h,\mu}$:= Rotate x, y such that the center of mass of the points is in zero (Section 5.4.1);
$R_x^\mu, R_x^\mu, R_w^\mu$:= Rotation matrices which rotate the center of mass to zero;
$x^{h,\mu}, y^{h,\mu}, w^{h,\mu}$:= Permute the points, views and rotation matrices R_x^I, R_y^I, R_w^I according to Section 5.4.2;
s := permutation of the views, r := permutation of the points;
$\bar{x}, \bar{y}, \bar{w}$:= Rotate the first point to x-axis (Section 5.4.3);
R_x^a, R_y^a, R_w^a := Rotation matrices which rotate the first point to y-axis;
$R_x^I := R_x^a R_x^\mu$, $R_y^I := R_y^a R_y^\mu$, $R_w^I := R_w^a R_w^\mu$;

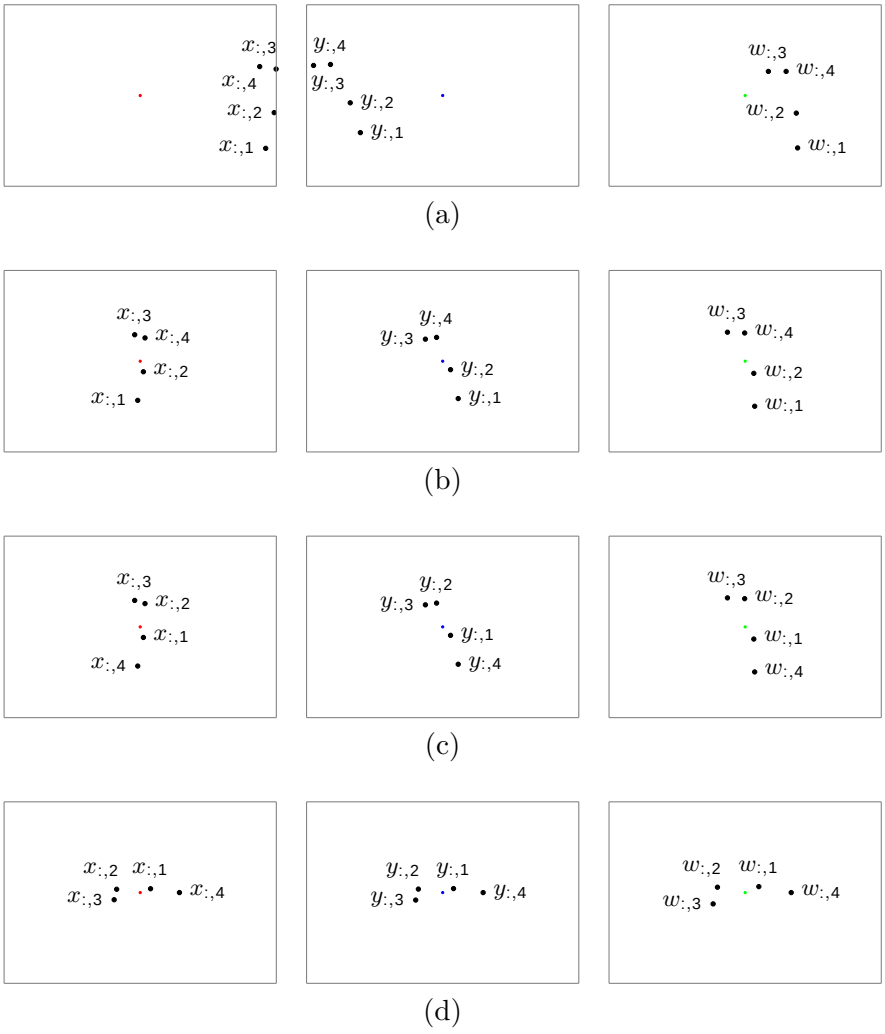


Figure 5.1: An illustration of the invariantization procedure. (a) The input points x, y. (b) The points rotated such, that their center of mass is zero. (Section 5.4.1) (c) The points permuted according to Section 5.4.2 (d) The invariantized points $\bar{x}, \bar{y}$. The red, green, and blue dots depict the zero point.

5.4.1 Moving the center of mass to zero

Let us introduce the notation which we will use in this section.

x $\mathsf{R}^{2,4}$ Four calibrated 2D points in the first camera.

y $\mathsf{R}^{2,4}$ Four calibrated 2D points in the second camera.

w $\mathsf{R}^{2,4}$ Four calibrated 2D points in the third camera.

x^h $\mathsf{R}^{3,4}$ Homogeneous representation of columns of x.

y^h $\mathsf{R}^{3,4}$ Homogeneous representation of columns of y.

w^h $\mathsf{R}^{3,4}$ Homogeneous representation of columns of w.

R_x^μ $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates x^h of x such that their center of mass is equal to zero.

R_y^μ $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates y^h of y such that their center of mass is equal to zero.

R_w^μ $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates w^h of w such that their center of mass is equal to zero.

$x^{h,\mu}$ $\mathsf{R}^{3,4}$ Homogeneous representation of points from x rotated such, that their point of mass is in zero. There holds $x^{h,\mu} \approx R_x^\mu x^h$.

$y^{h,\mu}$ $\mathsf{R}^{3,4}$ Homogeneous representation of points from y rotated such, that their point of mass is in zero. There holds $y^{h,\mu} \approx R_y^\mu y^h$.

$w^{h,\mu}$ $\mathsf{R}^{3,4}$ Homogeneous representation of points from w rotated such, that their point of mass is in zero. There holds $w^{h,\mu} \approx R_w^\mu w^h$.

Now, we are going to describe how to move the center of mass to zero. We know the points x $\mathsf{R}^{2,4}$, y $\mathsf{R}^{2,4}$, w $\mathsf{R}^{2,4}$. The goal is to find rotation matrices R_x^μ $SO(3)$, R_y^μ $SO(3)$, R_w^μ $SO(3)$, such that the center of mass of the homogeneous representative of columns of x multiplied by R_x^μ, the center of mass of the homogeneous representative of columns of y multiplied by R_y^μ, and the center of mass of the homogeneous representative of columns of w multiplied by R_w^μ are all equal to the homogeneous representative of zero.

This procedure is the same as the procedure described in Section 4.4.1, which is part of the invariantization of the Five-Point problem. The only difference is that the procedure in Section 4.4.1 operates with 5 points, while this procedure operates with 4 points. The procedure is performed iteratively. In every step, the center of mass μ is computed and rotated to zero. After about 3 - 4 iterations, the distance between the center of mass and the zero vector is acceptable.

This procedure is the same for all three views. We are going to describe it for the first view, where the center of mass of points x is rotated to zero and the accumulated rotation is stored into R_x^μ. The invariantization of points y and w is done analogously, while the accumulated rotations are stored into

R_y^μ and into R_w^μ. First, the homogeneous representation $x^h \quad \mathsf{R}^{3,4}$ of points x is obtained. The matrix R_x^μ is initialized to identity $R_x^\mu = I$, the matrix of transformed points $x^{h,\mu}$ is initialized as $x^{h,\mu} = x^h$.

At the beginning of every iteration, the center of mass μ is computed as:

$$\mu = \frac{1}{4}\sum_{i=1}^{4}\begin{bmatrix} x_{1,i}^{h,\mu} \\ x_{2,i}^{h,\mu} \\ x_{3,i}^{h,\mu} \end{bmatrix} \tag{5.35}$$

And the matrix M is built as:

$$M = \begin{bmatrix} \mu_1 & 1 & 0 \\ \mu_2 & 0 & 1 \\ \mu_3 & 0 & 0 \end{bmatrix} \tag{5.36}$$

Then, the matrix M is decomposed using a QR-decomposition into an orthogonal matrix Q and an upper-triangular matrix R, such as $M = QR$. We obtain the rotation matrix R_{cur} for the current iteration step by multiplying matrix Q^T by a matrix M_1, such as:

$$M_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{5.37}$$

Then, there holds:

$$R_{cur} = M_1 Q^T \tag{5.38}$$

Both matrices M_1, Q^T are orthogonal, therefore, matrix $R_{cur} = M_1 Q^T$ is orthogonal as well. However, we search for a rotation matrix, which is an orthogonal matrix with a determinant equal to 1. If the determinant $\det R_{cur} = -1$, we can multiply R_{cur} from left by a matrix M_2, such as:

$$M_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{5.39}$$

After that, we obtain the rotation matrix R_{cur} as:

$$R_{cur} = M_2 R_{cur} = M_2 M_1 Q^T \tag{5.40}$$

Now, R_{cur} is a rotation matrix because it is orthogonal and its determinant is equal to 1.

This process is repeated 4 times, after each step, we update the rotation matrix R_x^μ as $R_{cur} R_x^\mu$ and the homogeneous representation $x^{h,\mu}$ according to

(5.41). The whole procedure is described in Algorithm 14. The reason why this works is explained in Section 4.4.1.

$$x^{h,\mu} = R_{cur} x^{h,\mu}$$
$$\begin{bmatrix} x_{1,i}^{h,\mu} \\ x_{2,i}^{h,\mu} \\ x_{3,i}^{h,\mu} \end{bmatrix} = \frac{1}{x_{3,i}^{h,\mu}} \begin{bmatrix} x_{1,i}^{h,\mu} \\ x_{2,i}^{h,\mu} \\ x_{3,i}^{h,\mu} \end{bmatrix} \tag{5.41}$$

Algorithm 14: Rotate the center of mass to zero

input : Four points x

output : $x^{h,\mu}$ Homogeneous representation of points x rotated such, that their point of mass is zero, R_x^μ Rotation matrix which transforms x to $x^{h,\mu}$

$x^h :=$ Homogeneous representation of x;

$x^{h,\mu} := x^h$, $R_x^\mu := I$; `// Initialize the values`

$M_1 =$ Matrix (5.37), $M_2 =$ Matrix (5.39);

for $j \quad \{1, ..., 4\}$ **do**

  $\mu :=$ Point of mass of $x^{h,\mu}$ according to (5.35);

  $M :=$ Matrix (5.36);

  Decompose M into $M = QR$ using QR-Decomposition;

  $R_{cur} := M_1 Q^T$;

  if $\det R_{cur} < 1$ **then**

    $R_{cur} := M_2 R_{cur}$;

  end

  $R_x^\mu := R_{cur} R_x^\mu$, Update $x^{h,\mu}$ according to (5.41);

end

5.4.2 Permutation of the points

Let us introduce the notation which we will use in this section.

x	$\mathsf{R}^{2,4}$	Four calibrated 2D points in the first camera.
y	$\mathsf{R}^{2,4}$	Four calibrated 2D points in the second camera.

w $\mathsf{R}^{2,4}$ Four calibrated 2D points in the third camera.

R_x^μ $SO(3)$ The output of Section 5.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of x such that their center of mass is zero.

R_y^μ $SO(3)$ The output of Section 5.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of y such that their center of mass is zero.

R_y^μ $SO(3)$ The output of Section 5.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of y such that their center of mass is zero.

R_w^μ $SO(3)$ The output of Section 5.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of w such that their center of mass is zero.

$x^{h,\mu}$ $\mathsf{R}^{3,4}$ The output of Section 5.4.1. Homogeneous representation of points from x rotated such, that their point of mass is in zero. There holds $x^{h,\mu} \approx R_x^\mu x^h$.

$y^{h,\mu}$ $\mathsf{R}^{3,4}$ The output of Section 5.4.1. Homogeneous representation of points from y rotated such, that their point of mass is in zero. There holds $y^{h,\mu} \approx R_y^\mu y^h$.

$w^{h,\mu}$ $\mathsf{R}^{3,4}$ The output of Section 5.4.1. Homogeneous representation of points from w rotated such, that their point of mass is in zero. There holds $w^{h,\mu} \approx R_w^\mu w^h$.

r $Sym(\{1,...,4\})$ The permutation of the points.

s $Sym(\{1,2,3\})$ The new view at the position i is equal to the old view at the position $s(i)$.

$x^{h,\pi}$ $\mathsf{R}^{3,4}$ The points from the original view $s(1)$. The points are permuted by r.

$y^{h,\pi}$ $\mathsf{R}^{3,4}$ The points from the original view $s(2)$. The points are permuted by r.

$w^{h,\pi}$ $\mathsf{R}^{3,4}$ The points from the original view $s(3)$. The points are permuted by r.

Now, we assume that we have found matrices R_x^μ, R_y^μ, R_w^μ (Section 5.4.1) which transform the points of mass of homogeneous representations of points x, y, w to zero. We further assume that we know the homogeneous representations $x^{h,\mu}$ $\mathsf{R}^{3,4}$, $y^{h,\mu}$ $\mathsf{R}^{3,4}$, $w^{h,\mu}$ $\mathsf{R}^{3,4}$ of the points x, y, w rotated such that their points of mass are in zero.

The goal is to permute the **points** in such a way, that the point with the largest distance from the center of mass is the last and the remaining points are ordered counterclockwise. Further, we permute the **views** according to the distance from the first point to zero. We want to find the permutations r of the points and s of the views, as well as the permuted points $x^{h,\pi}$, $y^{h,\pi}$, $w^{h,\pi}$.

First, we identify the index i_1 of the point in the first view, whose distance from zero is maximal:

$$i_1 = \arg\max_{i \in \{1,\dots,4\}} \left\| \begin{bmatrix} x^{h,\mu}_{1,i} \\ x^{h,\mu}_{2,i} \end{bmatrix} \right\|^2 \tag{5.42}$$

Analogously, we identify the index i_2 of the point in the second view, whose distance from zero is maximal and the index i_3 of the point in the third view, whose distance from zero is maximal:

$$i_2 = \arg\max_{i \in \{1,\dots,4\}} \left\| \begin{bmatrix} y^{h,\mu}_{1,i} \\ y^{h,\mu}_{2,i} \end{bmatrix} \right\|^2 \tag{5.43}$$

$$i_3 = \arg\max_{i \in \{1,\dots,4\}} \left\| \begin{bmatrix} w^{h,\mu}_{1,i} \\ w^{h,\mu}_{2,i} \end{bmatrix} \right\|^2 \tag{5.44}$$

If the point i_1 in the first view is further from zero than the points i_2 in the second view and i_3 in the third view, then we set $i = i_1$. If the point i_2 in the second view is further from zero than the points i_1 in the first view and i_3 in the third view, then we set $i = i_2$. Otherwise, we set $i = i_3$. We find the permutation $s \in Sym(\{1,2,3\})$, such that the permuted views are ordered according to the descending distance of the point indexed by i from zero. We permute the rotation matrices R^{μ}_{x}, R^{μ}_{y} and R^{μ}_{w} according to s.

Now, we find the permutation $r \in Sym(\{1,2,3,4\})$, such that the last element $r(4)$ of the permutation is the index of the most distant point i and the remaining elements are sorted counterclockwise in the new third view. For further details on how the permutation r is obtained, please see Section 4.4.2. We permute the columns of matrices $x^{h,\mu}$, $y^{h,\mu}$, $w^{h,\mu}$ with this permutation to obtain $x^{h,\pi}$, $y^{h,\pi}$, $w^{h,\pi}$.

5.4.3 Moving the last point to x-axis

Let us introduce the notation which we will use in this section.

x $\mathsf{R}^{2,4}$ Four calibrated 2D points in the first camera.

y $\mathsf{R}^{2,4}$ Four calibrated 2D points in the second camera.

w $\mathsf{R}^{2,4}$ Four calibrated 2D points in the third camera.

R_x^μ $SO(3)$ The output of Section 5.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of x such that their center of mass is zero.

R_y^μ $SO(3)$ The output of Section 5.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of y such that their center of mass is zero.

R_w^μ $SO(3)$ The output of Section 5.4.1. A 3D rotation matrix which transforms the homogeneous coordinates of w such that their center of mass is zero.

$x^{h,\pi}$ $\mathsf{R}^{3,4}$ Points in the first view rotated such that their center of mass is zero and permuted according to Section 5.4.2.

$y^{h,\pi}$ $\mathsf{R}^{3,4}$ Points in the second view rotated such that their center of mass is zero and permuted according to Section 5.4.2.

$w^{h,\pi}$ $\mathsf{R}^{3,4}$ Points in the third view rotated such that their center of mass is zero and permuted according to Section 5.4.2.

R_x^a $SO(3)$ A rotation matrix which rotates the last point of $x^{h,\pi}$ to the x-axis.

R_y^a $SO(3)$ A rotation matrix which rotates last point of $y^{h,\pi}$ to the x-axis.

R_w^a $SO(3)$ A rotation matrix which rotates last point of $w^{h,\pi}$ to the x-axis.

$\bar{x}$ $\mathsf{R}^{2,5}$ Invariantized representation of points x.

$\bar{y}$ $\mathsf{R}^{2,5}$ Invariantized representation of points y.

$\bar{w}$ $\mathsf{R}^{2,5}$ Invariantized representation of points w.

R_x^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of x to the homogeneous coordinates of $\bar{x}$.

R_y^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of y to the homogeneous coordinates of $\bar{y}$.

R_w^I $SO(3)$ A 3D rotation matrix which transforms the homogeneous coordinates of w to the homogeneous coordinates of $\bar{w}$.

We know the permuted points $x^{h,\pi}$, $y^{h,\pi}$, $w^{h,\pi}$ and the rotation matrices R_x^μ, R_y^μ, R_w^μ which transform the center of mass to zero. The goal is to transform

the permuted points such that the last point lies on the x-axis. The outputs of this section are the rotation matrices R_x^a, R_y^a, R_w^a, the invariantized points $\bar{x}$, $\bar{y}$, $\bar{w}$ and the rotation matrices R_x^I, R_y^I, R_w^I.

Now, we are going to find the matrix R_x^a which would transform the last row of matrix $x^{h,\pi}$ to x-axis. This matrix represents a rotation around the optical axis, therefore, the property that the center of mass of the points is zero is preserved. Matrices R_y^a and R_w^a which rotate the last point in the second view to the x-axis, are found analogously.

Let $\sigma = \sqrt{(x_{1,4}^{\pi})^2 + (x_{2,4}^{\pi})^2}$ be the norm of the last Euclidean (not homogeneous) point in the first view. The rotation matrix R_x^a is constructed as follows:

$$R_x^a = \begin{bmatrix} \frac{x_{1,4}^{h,\pi}}{\sigma} & \frac{x_{2,4}^{h,\pi}}{\sigma} & 0 \\ \frac{-x_{2,4}^{h,\pi}}{\sigma} & \frac{x_{1,4}^{h,\pi}}{\sigma} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.45}$$

Matrix R_x^a is a rotation matrix because its columns are orthonormal and because the determinant of R_x^a is equal to 1. Matrix R_x^a rotates the last point to zero. Because R_x^a is an extended 2D matrix, it represents a rotation around the optical axis which passes through the center of mass of the points. Therefore, the center of mass of points $x^{h,\pi}$ stays in zero after being transformed by R_x^a.

The rotation matrix R_x^I is obtained as:

$$R_x^I = R_x^a R_x^{\mu} \tag{5.46}$$

The homogeneous representation $\bar{x}^h$ of the invariantized points $\bar{x}$ is obtained as:

$$\bar{x}^h = R_x^a x^{h,\pi} \tag{5.47}$$

The Cartesian representation $\bar{x}$ of the invariantized points is then obtained by taking the first two rows of $\bar{x}^h$ because the procedures described in Sections 5.4.1 and 5.4.2 and the character of matrix R_x^a guarantee that all entries in the last row of $\bar{x}^h$ are equal to 1.

5.5 Alignment of the problem on the anchors

The invariantization step transforms the problems to a uniform shape. However, the sum of squared distances between two invariantized problems may

sometimes be improved. Therefore, we add another step of the problem preprocessing, after which the sum of squared distances between the anchor and the final problem decreases, which may improve the probability of a successful track from the anchor to the final problem.

The alignment for the Five-Point problem described in Section 4.5 minimizes the distance between two views by rotation and permutation of the views. In the case of the Four-Point problem, the rotation of the views decreases the probability of a successful track, probably because the last point, through which the line passes, is displaced. Therefore, we introduce here a simplified version of the alignment, which only selects a permutation of the points for which the sum of squared distances between the corresponding points in the anchor and in the final problem is minimal. Although the effect of the alignment is smaller than in the case of the Five-Point problem, it still increases the number of successful tracks from 32456 out of 1398306 to 36615 out of 1398306. Let us introduce the notation which we will use in this section.

$\bar{x}$ $\mathsf{R}^{2,4}$ Invariantized representation of four calibrated 2D points in the first camera.

$\bar{y}$ $\mathsf{R}^{2,4}$ Invariantized representation of four calibrated 2D points in the second camera.

$\bar{w}$ $\mathsf{R}^{2,4}$ Invariantized representation of four calibrated 2D points in the second camera.

R_x^I $SO(3)$ The output of Section 5.4. A 3D rotation matrix which transforms the homogeneous coordinates of x to the homogeneous coordinates of invariantized points $\bar{x}$.

R_y^I $SO(3)$ The output of Section 5.4. A 3D rotation matrix which transforms the homogeneous coordinates of y to the homogeneous coordinates of invariantized points $\bar{y}$.

R_w^I $SO(3)$ The output of Section 5.4. A 3D rotation matrix which transforms the homogeneous coordinates of w to the homogeneous coordinates of invariantized points $\bar{y}$.

$(\breve{x}_a\ ,\breve{y}_a\ ,\breve{w}_a\ ,\breve{\lambda}_a\ ,\breve{l}_a\ )$, $\breve{x}_a$ $\mathsf{R}^{2,4}, \breve{y}_a$ $\mathsf{R}^{2,4}, \breve{w}_a$ $\mathsf{R}^{2,4}, \breve{\lambda}_a$ $\mathsf{R}^{3,4}, \breve{l}_a$ R The selected **anchor** (problem-solution pair) from which the homotopy continuation should be tracked. We assume that the points $\breve{x}_a\ ,\breve{y}_a\ ,\breve{w}_a$ have been invariantized according to section 5.4.

$\hat{x}$ $\mathsf{R}^{2,4}$ Points in the first camera aligned to the anchor $(\breve{x}_a\ ,\breve{y}_a\ ,\breve{\lambda}_a\ )$

$\hat{y}$ $\mathsf{R}^{2,4}$ Points in the second camera aligned to the anchor $(\breve{x}_a\ ,\breve{y}_a\ ,\breve{\lambda}_a\ )$

$\hat{w}$ $\mathsf{R}^{2,4}$ Points in the third camera aligned to the anchor ($\breve{x}_a$, $\breve{y}_a$, $\breve{\lambda}_a$)

r_A $Sym(\{1,...,4\})$ A permutation of the points induced by the alignment.

We know the invariantized points $\bar{x}$, $\bar{y}$, $\bar{w}$ and the selected anchor ($\breve{x}_a$, $\breve{y}_a$, $\breve{w}_a$, $\breve{\lambda}_a$, $\breve{l}_a$). The goal is to find the permutation r_A and the aligned points $\hat{x}$, $\hat{y}$, $\hat{w}$, such that the sum of squared distances between the aligned points and the anchor ($\breve{x}_a$, $\breve{y}_a$, $\breve{w}_a$, $\breve{\lambda}_a$, $\breve{l}_a$) is minimized. We consider a subset S $Sym(\{1,...,4\})$ of permutations in the group $Sym(\{1,...,4\})$ which keep the last point on its original position, i.e. $\mathsf{S} = \{\{1,2,3,4\},\{1,3,2,4\},\{2,1,3,4\},\{2,3,1,4\},\{3,1,2,4\},\{3,2,1,4\}\}$. This task can be formulated as the following optimization problem:

$$r_A = \arg\min_{r_A \in \mathsf{S}} \sum_{i=1}^{4} \left\| \begin{bmatrix} \breve{x}^a_{1,i} \\ \breve{x}^a_{2,i} \\ 1 \end{bmatrix} - \begin{bmatrix} \bar{x}_{1,r_A(i)} \\ \bar{x}_{2,r_A(i)} \\ 1 \end{bmatrix} \right\|^2 + \sum_{i=1}^{4} \left\| \begin{bmatrix} \breve{y}^a_{1,i} \\ \breve{y}^a_{2,i} \\ 1 \end{bmatrix} - \begin{bmatrix} \bar{y}_{1,r_A(i)} \\ \bar{y}_{2,r_A(i)} \\ 1 \end{bmatrix} \right\|^2 + \sum_{i=1}^{4} \left\| \begin{bmatrix} \breve{w}^a_{1,i} \\ \breve{w}^a_{2,i} \\ 1 \end{bmatrix} - \begin{bmatrix} \bar{w}_{1,r_A(i)} \\ \bar{w}_{2,r_A(i)} \\ 1 \end{bmatrix} \right\|^2 \tag{5.48}$$

For every permutation r_A S we evaluate the expression (4.50). Then, we select the permutation r_A whose value of (4.50) is minimal. Because the last point, through which the vertical line passes, remains in its place, the oriented distance l does not change and the depths λ are only permuted by the optimal permutation r_A. The alignment is summarized in Algorithm 15.

Algorithm 15: Align Four-Point problem

```
input  : Invariantized representation x̄, ȳ, w̄ of x, y, w, Selected
         anchor (x̆_a , y̆_a , w̆_a , λ̆_a , l̆_a ), Set of permutations S
output : Aligned points x̂, ŷ, ŷ, Permutation r_A
dist_best := ∞;
for r_A  S do
    dist := Distance (5.48) given by r_A;
    if dist < dist_best then
        dist := dist_best, r_A := r_A
    end
end
x̂, ŷ, ŵ := Permute the points x̄, ȳ, w̄ with r_A;
```

5.6 Selection of the Anchor

Let us introduce the notation which we will use in this section.

$(\bar{x}, \bar{y}, \bar{w}), \bar{x}$ $\mathsf{R}^{2,4}, \bar{y}$ $\mathsf{R}^{2,4}, \bar{w}$ $\mathsf{R}^{2,4}$ The invariantized points in the first, the second and the third view.

m N The expected number of anchors.

$\mathcal{A} = (\breve{x}_a, \breve{y}_a, \breve{w}_a, \breve{\lambda}_a, \breve{l}_{a\prime}$ a $\{1, ..., m\}, \breve{x}_a$ $\mathsf{R}^{2,4}, \breve{y}_a$ $\mathsf{R}^{2,4}, \breve{w}_a$ $\mathsf{R}^{2,4}, \breve{\lambda}_a$ $\mathsf{R}^{3,4}, \breve{l}_a$ R A sequence of **anchors**, i.e. starting problem-solution pairs. For every a, the values in λ_a and l are the solutions to the depth formulation of the Four-Point problem (2.13) parametrized by $\breve{x}_a, \breve{y}_a, \breve{w}_a$. We assume that the problem $\breve{x}_a, \breve{y}_a, \breve{w}_a$ is invariantized.

D R^{15} A low-dimensional representation of problem $(\bar{x}, \bar{y}, \bar{w})$

$c(D, \theta) : \mathsf{R}^{15} \to \mathsf{R}^m$ The classifier which for every low-dimensional representation D of problem $(\bar{x}, \bar{y}, \bar{w})$ outputs a vector of probabilities that the problem is successfully tracked from each anchor. θ is a vector of parameters to the classifier.

We use a real homotopy continuation (Section 3.4) in our solver. The real homotopy continuation does not guarantee that the track will be successful. In addition to that, we perform only one track and even if we succeed, there is no guarantee that we will end up in the expected solution.

Like in the Five-Point solver, we maintain a set $\mathcal{A}$ of **anchors** (problem-solution pairs), which serve as the starting points for the homotopy continuation. We also train a classifier c, which accepts an invariantized problem $(\bar{x}, \bar{y}, \bar{w})$ and outputs a number a $\{1, ..., m\}$. Then, we track the problem from the anchor $(\breve{x}^a, \breve{y}^a, \breve{\lambda}^a)$.

In Section 5.6.1 we describe how the set $\mathcal{A}$ of the anchors is generated. In Section 5.6.2 we describe the preprocessing of the problem $(\bar{x}, \bar{y}, \bar{w})$, which generates a low-dimensional representation D R^{15} of the problem. In Section 5.6.3 we describe how the training data is generated. In Section 5.6.4 we describe the structure of the classifier c.

5.6.1 Anchor set generation

Let us introduce the notation which we will use in this section.

n The number of considered problems.

$\bar{P} = (\bar{x}_i, \bar{y}_i, \bar{w}_i, \bar{\lambda}_i, \bar{l}_i)_i \; i \quad \{1, ..., m\}, \bar{x}_i \quad \mathsf{R}^{2,4}, \bar{y}_i \quad \mathsf{R}^{2,4}, \bar{w}_i \quad \mathsf{R}^{2,4}, \bar{\lambda}_i \quad \mathsf{R}^{3,4}, \bar{l}_i \quad \mathsf{R}$
A set of n problem-solution pairs generated by Section 2.2.3. These problem-solution pairs are invariantized according to 5.4.

$m \quad \mathsf{N}$ The expected number of anchors.

$A = (\breve{x}_a, \breve{y}_a, \breve{w}_a, \breve{\lambda}_a), \breve{l}_a)_i \; a \quad \{1, ..., m\}, \breve{x}_a \quad \mathsf{R}^{2,4}, \breve{y}_a \quad \mathsf{R}^{2,4}, \breve{w}_a \quad \mathsf{R}^{2,4}, \breve{\lambda}_a \quad \mathsf{R}^{3,4}, \breve{l}_a \quad \mathsf{R}$
A sequence of **anchors**, i.e. starting problem-solution pairs. For every a, the values in $\breve{\lambda}_a$, $\breve{l}_a$ are the solutions to the depth formulation of the Four-Point problem (Equation 2.13) parametrized by $\breve{x}_a, \breve{y}_a, \breve{w}_a$. We assume that the problem $\breve{x}_a, \breve{y}_a, \breve{w}_a$ is invariantized.

$G = (V, E)$ A directed connectivity graph on the set P. The vertices of G are equal to the problem-solution pairs in $\bar{P}$ ($V = \bar{P}$). There is an edge in E from a problem $p_i \quad \bar{P}$ to a problem $p_i \quad \bar{P}$ p_i is correctly tracked from p_i.

Now, we are going to describe how the set of anchors A is generated. The procedure is similar to the procedure of anchor generation for the Five-Point problem solver, which is described in Section 4.6.1.

We know the set $\bar{P}$ of n invariantized problem-solution pairs. We say that the problem-solution pair $(x_i, y_i, w_i, \lambda_i, l_i)$ is **correctly tracked** from $(x_a, y_a, w_a, \lambda_a, l_a)$ if the homotopy continuation is successfully tracked and the distance (6.3) between the relative poses obtained from the resulting depths and the relative poses obtained from the ground truth depth λ_i is $\leq 5^\circ$. Both relative poses are obtained according to Sec. 5.7. We want to find a subset of anchors $A \quad \bar{P}, |A| = m$, such that the number of problem-solution pairs from P which are correctly tracked at least from one anchor from A is maximized.

Like in Section 4.6.1, we solve this problem in two stages. First, we build a directed connectivity graph $G = (V, E)$ according to Algorithm 16. In the case of the Four-Point problem, the graph is directed because the relaxed definition of the correct track allows the existence of a correct track in one direction but not in another. In the second stage, we find a subset A of vertices in G, such that $|A| = m$ and the number of vertices from G which are

covered by the vertices from A is maximized. This is an NP-hard task, for which we use the same heuristic as in the Five-Point Solver (Section 4.6.1). The heuristic is described in Algorithm 10.

Algorithm 16: Building connectivity graph

input : Set $\hat{P}$ of n invariantized problem-solution pairs
output : Connectivity graph $G = (V, E)$. Two problem-solution pairs p_i, p_i are connected if we can track correctly p_i from p_i

$V := \{1, ..., n\}$, $E :=$;
for i $\{1, ..., n\}$ **do**
  for j $\{1, ..., n\}, j = i$ **do**
    $\hat{x}_j, \hat{y}_j, \hat{w}_j :=$ Align x_j, y_j, w_j on $\bar{x}_i, \bar{x}_j, \bar{w}_j$ (Section 5.5);
    $p_s :=$ Parametrize $(\bar{x}_i, \bar{y}_i, \bar{w}_i)$ according to (5.2);
    $p_f :=$ Parametrize $(\hat{x}_j, \hat{y}_j, \hat{w}_j)$ according to (5.2);
    $z_0 :=$ Parametrize $\bar{\lambda}_i, \bar{l}_i$ according to (5.3);
    $z_1 :=$ Track homotopy (5.6) from z_0 using Algorithm 4;
    if $z_1 =$ **then**
      $\lambda_1 :=$ Depths obtained from z_1 according to (5.4);
      $(R_{1,2}, t_{1,2}), (R_{1,3}, t_{1,3}), (R_{2,3}, t_{2,3}) :=$ Relative poses obtained from depths λ_1 according to Section 5.7;
      $(R_{1,2}, t_{1,2}), (R_{1,3}, t_{1,3}), (R_{2,3}, t_{2,3}) :=$ Ground Truth relative poses obtained from depths $\bar{\lambda}_j$ according to Section 5.7;
      $d_1 :=$ Distance (6.3) between $(R_{1,2}, t_{1,2})$, $(R_{1,2}, t_{1,2})$;
      $d_2 :=$ Distance (6.3) between $(R_{1,3}, t_{1,3})$, $(R_{1,3}, t_{1,3})$;
      $d_3 :=$ Distance (6.3) between $(R_{2,3}, t_{2,3})$, $(R_{2,3}, t_{2,3})$;
      if $\frac{1}{3}(d_1 + d_2 + d_3) < 5^\circ$ **then**
        $E := E$ (i, j);
      end
    end
  end
end

5.6.2 Problem preprocessing for the classifier

In this section, we are going to discuss how to uniquely represent the problem (x, y) with a vector D R^{15}, which will be used as the input to the classifier c. We want to design the preprocessing such that the number of problems correctly classified by the classifier c is as large as possible. Let us introduce the notation which we will use in this section.

$(x, y, w), x \quad \mathsf{R}^{2,4}, y \quad \mathsf{R}^{2,4}, w \quad \mathsf{R}^{2,4}$ The points in the first, the second and the third view which define an instance of the Four-Point problem.

$(\bar{x}, \bar{y}, \bar{w}), \bar{x} \quad \mathsf{R}^{2,4}, \bar{y} \quad \mathsf{R}^{2,4}, \bar{w} \quad \mathsf{R}^{2,4}$ The invariantized points in all three views

$D \quad \mathsf{R}^{15}$ A unique 15-dimensional representation of Four-Point problem instance $(\bar{x}, \bar{y}, \bar{w})$.

$p \quad \mathsf{R}^{24}$ A 20-dimensional representation of Four-Point problem instance $(\bar{x}, \bar{y}, \bar{w})$ obtained by (5.2).

$L \quad \mathsf{A}^{15,24}$ A transformation matrix which transforms a 24-dimensional invariantized representation p to a 15-dimensional representation D.

$\bar{P} = (x_i, y_i, w_i, \lambda_i, l_i)_{\prime} \; i \quad \{1, ..., n\}, x_i \quad \mathsf{R}^{2,4}, y_i \quad \mathsf{R}^{2,4}, w_i \quad \mathsf{R}^{2,4}, \lambda_i \quad \mathsf{R}^{2,4}, l \quad \mathsf{R}$ A set of n problem-solution pairs generated by Section 2.2.3. These problem-solution pairs are invariantized according to Section 5.4.

A A set of anchors obtained by the procedure described in Section 5.6.1.

$\bar{K} \quad \mathsf{R}^{24,24}$ A covariance matrix of the invariantized points from P.

We know the problem (x, y, w), we want to find its representation D. We are going to describe a low-dimensional representation of the points based on the invariantized points $(\bar{x}, \bar{y}, \bar{w})$.

First, we invariantize the problem (x, y) according to Section 4.4 to obtain the invariantized problem $(\bar{x}, \bar{y})$. We are going to show, that all invariantized instances $(\bar{x}, \bar{y})$ of the Five-Point Problem live in a 14-dimensional linear subspace of the linear space R^2.

Points $(\bar{x}, \bar{y}, \bar{w})$ are invariantized according to Section 5.4, which means, that the second coordinates of the last point $\bar{x}_{2,4}$, $\bar{y}_{2,4}$, $\bar{w}_{2,4}$ are equal to zero and the centers of mass of the points in both views are equal to zero, as well.

There holds:

$$\begin{aligned}
\bar{x}_{2,4} &= 0 \\
\bar{y}_{2,4} &= 0 \\
\bar{w}_{2,4} &= 0 \\
\sum_{i=1}^{4} \bar{x}_{1,i} &= 0 \\
\sum_{i=1}^{4} \bar{x}_{2,i} &= 0 \\
\sum_{i=1}^{4} \bar{y}_{1,i} &= 0 \\
\sum_{i=1}^{4} \bar{y}_{2,i} &= 0 \\
\sum_{i=1}^{4} \bar{w}_{1,i} &= 0 \\
\sum_{i=1}^{4} \bar{w}_{2,i} &= 0
\end{aligned} \tag{5.49}$$

The invariantized points have to satisfy the equations (5.49). This is a set of nine homogeneous linear equations. The set of solutions to these equations is a 15-dimensional linear subspace of the 24-dimensional linear space R^{24}. For every invariantized problem $(\bar{x}, \bar{y}, \bar{w})$ there exists a vector $D \quad \mathsf{R}^{15}$ that can represent the invariantized problem uniquely. Now, we are going to describe how this vector can be obtained.

Let us have a set $\bar{P}$ of n problem-solution pairs generated according to Section 2.2.3. These problems are invariantized according to Section 5.4. We obtain a 24-dimensional parametrization $p_i, i \quad \{1, ..., n\}$ of every problem in $\bar{P}$ according to (5.2) and compute the covariance matrix $\bar{K} \quad \mathsf{R}^{24,24}$ of the invariantized points as:

$$\bar{K} = \sum_{i=1}^{n} p_i p_i^T \tag{5.50}$$

Matrix $\bar{K}$ is symmetric and has rank 15. There exists a real eigendecomposition of $\bar{K}$:

$$\bar{K} = V \Lambda V^T \tag{5.51}$$

Because the matrix K has rank 15, its first 9 eigenvalues are equal to zero. Let $V \quad \mathsf{R}^{20,15}$ be the matrix whose columns are the last 15 eigenvectors of K, i.e. the last 15 columns of V. Let $\Lambda \quad \mathsf{R}^{15,15}$ be a diagonal matrix whose entries are the nonzero eigenvalues of K. Then, the matrix K can be represented as:

$$\bar{K} = V \, \Lambda \, V^T \tag{5.52}$$

The transformation matrix L is equal to:

$$L = \Lambda^{-\frac{1}{2}} V^T \tag{5.53}$$

The 15-dimensional representation D of a problem $(\bar{x}, \bar{y}, \bar{w})$ is then obtained as:

$$D = Lp = \Lambda^{-\frac{1}{2}} V^T p \tag{5.54}$$

The reason this gives the desired result is the same as in Section 4.6.2.

5.6.3 Training data generation

Let us introduce the notation which we will use in this section.

n	The number of training data.
$\bar{P}_t = (\bar{x}_i, \bar{y}_i, \bar{w}, (R_{1,2}, t_{1,2})_i, (R_{1,3}, t_{1,3})_i, (R_{2,3}, t_{2,3})_i)_i$ i $\{1, ..., n\}, \bar{x}_i$ $\mathsf{R}^{2,4}, \bar{y}_i$ $\mathsf{R}^{2,4}$	Invariantized training data generated together with their ground truth poses according to Section 2.2.4.
$A = (\breve{x}_a, \breve{y}_a, \breve{w}_a, \breve{\lambda}_a, \breve{l}_a)_i$ a $\{1, ..., m\}, \breve{x}_a$ $\mathsf{R}^{2,4}, \breve{y}_a$ $\mathsf{R}^{2,4}, \breve{w}_a$ $\mathsf{R}^{2,4}, \breve{\lambda}_a$ $\mathsf{R}^{3,4}, \breve{l}_a$ R	A sequence of **anchors**, i.e. starting problem-solution pairs. The anchors are generated using a procedure from Section 5.6.1. For every a, the values in $\breve{\lambda}_a$, $\breve{l}_a$ are the solutions to the depth formulation of the Four-Point problem (Equation 2.13) parametrized by $\breve{x}_a, \breve{y}_a, \breve{w}_a$. The problem $\breve{x}_a, \breve{y}_a, \breve{w}_a$ is invariantized.
$X = D_i, i$ $\{1, ..., n\}, D_i$ R^{15}	The set of input observations for the training of the classifier. D_i is the 15 dimensional representation of problem (x_i, y_i, w_i) from P_t obtained by procedure from Section 5.6.2.
$Y = a_i, i$ $\{1, ..., n\}, a_0$ $\{1, ..., m\}$	The set of the labels for the training of the classifier. The problem $(\bar{x}_i, \bar{y}_i, \bar{w}, \bar{\lambda}_i, \bar{l}_i)$ can be correctly tracked from the anchor $(\breve{x}_{a_i}, \breve{y}_{a_i}, \breve{w}_{a_i}, \breve{\lambda}_{a_i}, \breve{l}_{a_i})$ A.

Now, we are going to describe how the training data for the classifier are generated. We want to find the training observations X and the training labels Y. The principle of the training data generation is the same as in the case of the Five-Point problem (Section 4.6.3).

We start with empty sequences X, Y. First, we generate a set of n problems $\bar{P}_t$ according to Section 2.2.4. For every problem $(x_i, y_i, \lambda_i) \quad \bar{P}_t$ we obtain the 15-dimensional representation D_i according to Section (5.6.2). For every anchor $(\breve{x}_a, \breve{y}_a, \breve{w}_a, \breve{\lambda}_a, \breve{l}_a) \quad A$, we align the problem to the anchor according to Section 5.5 to obtain an aligned problem $(\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{\lambda}_i \hat{l}_i)$.

Then, we track the homotopy (5.6) from the parametrization p_s of anchor $(\breve{x}_a, \breve{y}_a, \breve{w}_a)$ to the parametrization p_f of problem $(\hat{x}, \hat{y}, \hat{w})$. If the homotopy continuation is **correctly tracked**, i.e. it is successfully tracked and the distance (6.3) between the obtained pose and the ground truth pose is smaller than 5 degrees, then we insert the 15-dimensional representation D_i of problem $(\bar{x}_i, \bar{y}_i, \bar{w}_i, \bar{\lambda}_i, \bar{l}_i)$ to the end of the sequence X and the index a of the current anchor to the end of the sequence Y. The procedure of generating the training data is described in Algorithm 17

Algorithm 17: Generating training data

input : Set P_t of n invariantized problems with Ground Truth relative poses, Set A of m anchors

output : X Sequence of Low-rank representations of points in $\bar{P}_t$, Y Sequence of ids of anchors from which the problems from $\bar{P}_t$ can be tracked

$V := \{1, ..., n\}$, $E :=$;

for $i \quad \{1, ..., n\}$ **do**

  $D_i :=$ Low-dim representation of $\bar{x}_i, \bar{y}_i, \bar{w}_i$ (Sec. 5.6.2);

  for $j \quad \{1, ..., m\}$ **do**

    $\hat{x}_i, \hat{y}_i, \hat{w}_i :=$ Align $\bar{x}_i, \bar{y}_i, \bar{w}_i$ on $\breve{x}_i, \breve{y}_i, \breve{w}_i$ (Section 5.5);

    $p_s :=$ Parametrize $(\breve{x}_j, \breve{y}_j, \breve{w}_j)$ according to (5.2);

    $p_f :=$ Parametrize $(\hat{x}_i, \hat{y}_i, \hat{w}_i)$ according to (5.2);

    $z_0 :=$ Parametrize $\breve{\lambda}_j, \breve{l}_j$ according to (5.3);

    $z_1 :=$ Track homotopy (5.6) from z_0 using Algorithm 4;

    if $z_1 =$ **then**

      $\lambda_1 :=$ Depths obtained from z_1 according to (4.4);

      $(R_{1,2}, t_{1,2}), (R_{1,3}, t_{1,3}), (R_{2,3}, t_{2,3}) :=$ Relative poses obtained from depths λ_1 according to Section 5.7;

      $d_1 :=$ Distance (6.3) between $(R_{1,2}, t_{1,2})$, $(R_{1,2}, t_{1,2})_i$;

      $d_2 :=$ Distance (6.3) between $(R_{1,3}, t_{1,3})$, $(R_{1,3}, t_{1,3})_i$;

      $d_3 :=$ Distance (6.3) between $(R_{2,3}, t_{2,3})$, $(R_{2,3}, t_{2,3})_i$;

      if $\frac{1}{3}(d_1 + d_2 + d_3) < 5^\circ$ **then**

        $X := X \quad \{D_i\}$;

        $Y := Y \quad \{j\}$;

      end

    end

  end

end

5.6.4 The Anchor Selection Classifier

Let us introduce the notation which we will use in this section.

$(\bar{x}, \bar{y}, \bar{w}), \bar{x}$ $\mathsf{R}^{2,4}, \bar{y}$ $\mathsf{R}^{2,4}, \bar{w}$ $\mathsf{R}^{2,4}$ The invariantized points in all three views.

m N The expected number of anchors.

$A = (\breve{x}_a, \breve{y}_a, \breve{w}_a, \breve{\lambda}_a, \breve{l}_a)$, a $\{1, ..., m\}, \breve{x}_a$ $\mathsf{R}^{2,4}, \breve{y}_a$ $\mathsf{R}^{2,4}, \breve{w}_a$ $\mathsf{R}^{2,4}, \breve{\lambda}_a$ $\mathsf{R}^{3,4}, \breve{l}_a$ R A sequence of **anchors**, i.e. starting problem-solution pairs. The anchors are generated using a procedure from Section 5.6.1. For every a, the values in $\breve{\lambda}_a$, $\breve{l}_a$ are the solutions to the depth formulation of the Four-Point problem (Equation 2.13) parametrized by $\breve{x}_a, \breve{y}_a, \breve{w}_a$. The problem $\breve{x}_a, \breve{y}_a, \breve{w}_a$ is invariantized.

D R^{15} A low-dimensional representation of problem $(\bar{x}, \bar{y}, \bar{w})$

$c(D, \theta) : \mathsf{R}^{15} \to \mathsf{R}^m$ The classifier which for every low-dimensional representation D of problem $(\bar{x}, \bar{y}, \bar{w})$ outputs a vector of probabilities that the problem is successfully tracked from each anchor. θ is a vector of parameters to the classifier.

s R^m The vector of the outputs of the classifier $c(D, \theta)$. a is the index of the largest entry of vector s. The problem is tracked from anchor a .

$X = D_i, i$ $\{1, ..., n\}, D_i$ R^{15} The set of input observations for the training of the classifier obtained according to Section 5.6.3. D_i is the 15 dimensional representation of a problem (x_i, y_i, w_i) from $\bar{P}_t$ obtained by procedure from Section 5.6.2.

$Y = a_i, i$ $\{1, ..., n\}, a_0$ $\{1, ..., m\}$ The set of the labels for the training of the classifier obtained according to Section 5.6.3. The problem $(x_i, y_i, w_i, \lambda_i, l_i)$ can be correctly tracked from the anchor $(\breve{x}_a, \breve{y}_a, \breve{w}_a, \breve{\lambda}_a, \breve{l}_a)$ A.

In this section, we describe the structure and the training of the classifier for the selection of the anchor. The input to the classifier is a 15 dimensional representation D R^{15} of a problem (x, y) obtained according to Section 5.6.2. The output of the classifier is a vector s R^n of probabilities that the problem can be successfully tracked from each anchor. When the vector s is known, we obtain the index a $\{0, ..., m\}$ of the selected anchor as the index of the largest entry of vector s. The problem is then tracked from the anchor a . The task is to find such parameters θ of the classifier $c(D, \theta)$, that the classifier maximizes the probability that the correct anchor a is selected.

Like in the Five-Point problem case (Section 4.6.4), we use the cross-entropy loss (4.79). The classifier $c(D, \theta)$ itself is a fully connected network with a similar architecture as in the Five-Point problem case. The size of the input layer is 15. The neural network has six hidden layers, whose sizes are $200, 200, 200, 200, 100, 100$. The size of the output layer is m. We use PReLU as the nonlinearity. There is a dropout layer before the last linear layer in order to prevent overfitting.

Now, we are going to describe how the classifier c is trained. Let us have the training data X, Y generated according to Section 5.6.3. The goal is to find the parameters θ of the classifier $c(D, \theta)$ which would minimize the loss (4.79). This is achieved with the Stochastic Gradient Descent (SGD) optimizer. The classifier is modeled and trained in PyTorch.

5.7 Computation of the relative poses

Let us introduce the notation which we will use in this section.

$(x, y, w), x \quad \mathsf{R}^{2,4}, y \quad \mathsf{R}^{2,4}, w \quad \mathsf{R}^{2,4}$ The points in all three views.

$(\bar{x}, \bar{y}, \bar{w}), \bar{x} \quad \mathsf{R}^{2,4}, \bar{y} \quad \mathsf{R}^{2,4}, \bar{w} \quad \mathsf{R}^{2,4}$ The invariantized points in all three views.

$(\hat{x}, \hat{y}, \hat{w}), \hat{x} \quad \mathsf{R}^{2,4}, \hat{y} \quad \mathsf{R}^{2,4}, \hat{w} \quad \mathsf{R}^{2,4}$ The aligned points in all three views.

$(\hat{x}^h, \hat{y}^h, \hat{w}^h), \hat{x}^h \quad \mathsf{R}^{3,4}, \hat{y}^h \quad \mathsf{R}^{3,4}, \hat{w}^h \quad \mathsf{R}^{3,4}$ Homogeneous representation of he aligned points $(\hat{x}, \hat{y}, \hat{w})$.

$R_x^I \quad SO(3)$ The 3D rotation matrix which transforms the homogeneous representation of points x in the first view to the homogeneous representations of the invariantized points $\bar{x}$. (Section 5.4)

$R_y^I \quad SO(3)$ The 3D rotation matrix which transforms the homogeneous representation of points y in the second view to the homogeneous representations of the invariantized points $\bar{y}$. (Section 5.4)

$R_w^I \quad SO(3)$ The 3D rotation matrix which transforms the homogeneous representation of points w in the third view to the homogeneous representations of the invariantized points $\bar{w}$. (Section 5.4)

$s \quad Sym(\{1, 2, 3\})$ The permutation of the views induced by the invariantization. The new view at position i is equal to the old view at position $s(i)$.

$\hat{\lambda}$ $\mathsf{R}^{3,4}$ The depths of the aligned points $(\hat{x}, \hat{y}, \hat{w})$. This is the output of the homotopy continuation (4.6).

$(\hat{R}_{1,2}, \hat{t}_{1,2})$ The relative pose of the cameras between the first two aligned views $\hat{x}, \hat{y}$.

$(\hat{R}_{1,3}, \hat{t}_{1,3})$ The relative pose of the cameras between the first aligned view $\hat{x}$ and the third aligned view $\hat{w}$.

$(\hat{R}_{2,3}, \hat{t}_{2,3})$ The relative pose of the cameras between the second aligned view $\hat{y}$ and the third aligned view $\hat{w}$.

$(R_{1,2}, t_{1,2})$ The relative pose of the cameras between the first two original views x, y.

$(R_{1,3}, t_{1,3})$ The relative pose of the cameras between the first orignal view x and the third original view w.

$(R_{2,3}, t_{2,3})$ The relative pose of the cameras between the second original view y and the third original view w.

$\hat{X}$ $\mathsf{R}^{3,4}$ The coordinates of the 3D points in the coordinate system of the first aligned camera.

$\hat{Y}$ $\mathsf{R}^{3,4}$ The coordinates of the 3D points in the coordinate system of the second aligned camera.

$\hat{W}$ $\mathsf{R}^{3,4}$ The coordinates of the 3D points in the coordinate system of the third aligned camera.

In this section, we are going to describe how the relative poses $(R_{1,2}, t_{1,2})$, $(R_{1,2}, t_{1,3})$ are obtained from the depths $\hat{\lambda}$. In Section 5.7.1 we describe how the relative poses of the aligned problem $(\hat{x}, \hat{y}, \hat{w})$ is obtained and in Section 5.7.2 we describe how the relative poses of the aligned problem are transformed to the relative pose of the original problem (x, y, w).

5.7.1 Getting the Relative Pose From the Depths

We know the aligned points $(\hat{x}, \hat{y}, \hat{w})$ and the depths $\hat{\lambda}$ which are the result of the depth formulation of the Four-Point problem (2.13) parametrized by the aligned points $(\hat{x}, \hat{y}, \hat{w})$. The depths have been obtained by the homotopy continuation (5.6). The goal is to find the relative poses $(\hat{R}_{1,2}, \hat{t}_{1,2})$, $(\hat{R}_{1,3}, \hat{t}_{1,3})$, $(\hat{R}_{2,3}, \hat{t}_{2,3})$ consistent with the points and with the depths, i.e. such relative poses that the equations (2.6), (2.7), (2.8) hold.

This procedure is taken from [Paj21]. First, we obtain the coordinates of the 3D points $\hat{X}$ in the coordinate system of the first aligned camera, $\hat{Y}$ in the coordinate system of the second camera and $\hat{W}$ in the coordinate system of the third camera by multiplying the homogeneous aligned points $\hat{x}^h$, $\hat{y}^h$, $\hat{w}^h$ by the depths $\hat{\lambda}$ as:

$$\hat{X}_{i,j} = \hat{\lambda}_{1,j}\hat{x}^h,\ \hat{Y}_{i,j} = \hat{\lambda}_{2,j}\hat{y}^h\ \hat{W}_{i,j} = \hat{\lambda}_{3,j}\hat{w}^h \quad i \quad \{1,2,3\},\ \ j \quad \{1,...,4\} \tag{5.55}$$

Then, we find the relative pose $(\hat{R}_{1,2}, \hat{t}_{1,2})$ according to Section 4.7.1. The relative poses $(\hat{R}_{1,3}, \hat{t}_{1,3})$ and $(\hat{R}_{2,3}, \hat{t}_{2,3})$ are obtained analogously.

5.7.2 Passing From the Solution of the Aligned Problem

Now, we are going to describe how the relative poses of the aligned problem $(\hat{x}, \hat{y}, \hat{w})$ are transformed to the relative poses of the original problem (x, y, w).

We know the relative poses $(\hat{R}_{1,2}, \hat{t}_{1,2})$, $(\hat{R}_{1,3}, \hat{t}_{1,3})$ and $(\hat{R}_{2,3}, \hat{t}_{2,3})$ of the aligned problem, the rotation matrices R_x^I, R_y^I, R_w^I transforming the original points to the invariantized points and the permutation $s \quad Sym(\{1,2,3\})$ of the views during the invariantization. The task is to find the relative poses $(R_{1,2}, t_{1,2})$, $(R_{1,3}, t_{1,3})$, $(R_{2,3}, t_{2,3})$.

In this section, we utilize a permutation $s^{-1} \quad Sym(\{1,2,3\})$, which is an inverse to the permutation s. First, we are going to show how the relative pose $(R_{1,2}, t_{1,2})$ is obtained. If the index $s^{-1}(1)$ is smaller than $s^{-1}(2)$, then the relative pose $(R_{1,2}, t_{1,2})$ is obtained as:

$$\begin{aligned} R_{1,2} &= (R_y^I)^T \hat{R}_{s^{-1}(1),s^{-1}(2)} R_x^I \\ t \ &= (R_y^I)^T \hat{t}_{s^{-1}(1),s^{-1}(2)} \end{aligned} \tag{5.56}$$

If, on the other hand, the index $s^{-1}(1)$ is greater than $s^{-1}(2)$, then the relative pose $(R_{1,2}, t_{1,2})$ is obtained as:

$$\begin{aligned} R_{1,2} &= (R_x^I)^T \hat{R}^T_{s^{-1}(1),s^{-1}(2)} R_y^I \\ t \ &= -(R_x^I)^T \hat{R}^T_{s^{-1}(1),s^{-1}(2)} \hat{t}_{s^{-1}(1),s^{-1}(2)} \end{aligned} \tag{5.57}$$

The relative poses $(R_{1,3}, t_{1,3})$, $(R_{2,3}, t_{2,3})$ are computed analogously.

Chapter 6

Experiments

6.1 Evaluation metrics

6.1.1 Distance between two relative poses

Now, we are going to describe the distance between two relative poses. The distance is taken from [Mys20]. Let us have two relative poses $(R_1.t_1)$, (R_2, t_2). We want to find a distance $dist$ R which would measure the dissimilarity of the poses. The rotation distance $dist_R$ is an angle of a rotation matrix $R_1 R_2^T$. We obtain the rotation distance as follows:

$$\begin{aligned} R_{diff} &= R_1 R_2^T \\ dist_R &= \arccos\left(\frac{trace(R_{diff}) - 1}{2}\right) \end{aligned} \tag{6.1}$$

The translation distance $dist_T$ is the angle between two translation vectors. It is obtained as follows:

$$dist_T = \arccos\left(\frac{t_1^T t_2}{t_1 \quad t_2}\right) \tag{6.2}$$

Then, the distance $dist$ between two poses is obtained as the maximum of these two distances:

$$dist = \max\{dist_R, dist_T\} \tag{6.3}$$

6.1.2 Mean Average Accuracy

Now, we are going to describe the Mean Average Accuracy score, which is used for the evaluation of the Five-Point Solver in the benchmark described in Section 6.3. The input is a sequence of N errors (6.3) $dist_i, i \quad \{1, ..., N\}$. The task is to compute one aggregated score from the set of errors. The **larger** the mAA is, the **more acurate** the results are. The computation of mAA is shown in Algorithm

Algorithm 18: Mean Average Accuracy, (mAA)

input : A sequence $dist_i, i \quad \{1, ..., N\}$ of N errors (in degrees).
output : Mean Average Accuracy α
$\alpha := 0$;
for $i \quad \{1, ..., N\}$ **do**
  $\alpha := \alpha + \max(0, \text{ceil}(\frac{10 - dist_i}{10})$;
end
$\alpha := \frac{1}{N}\alpha$;

6.1.3 Sampson Error

Now, we are going to describe the Sampson Error. This error is used in the RANSAC scheme to approximate the reprojection error without having to actually reproject the points. Let us have a Fundamental matrix $F \quad \mathsf{R}^{3,3}$ and two tentatively matched points x, y in two views, which are related by the Fundamental matrix F. Let $x \quad \mathsf{R}^3$ be the homogeneous representation of a point in the first view, and $y \quad \mathsf{R}^3$ be the homogeneous representation of a point in the second view. The task is to compute the Sampson error of the match x, y with respect to Fundamental matrix F.

Let us define a matrix S as follows:

$$S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{6.4}$$

Then, the Sampson error is computed as:

$$\frac{y^T F x}{\sqrt{\ SFx\ ^2 + \ SF^T y\ ^2}} \tag{6.5}$$

6.2 Training and evaluation of the Five-Point problem

In this section, we are going to describe the results obtained during the training and evaluation of the Five-Point solver. The training of the classifier consists of the generation of the set of anchors according to 4.6.1 and the training of the classifier according to 4.6.4. In the evaluation part, we describe the success rate of the solver.

6.2.1 Generation of the anchors

We have generated the anchors according to Section 4.6.1 from a set of 10000 points generated according to Section 2.1.3. A problem **is covered** by a set of anchors, if there is an anchor in the set, from which the problem is correctly tracked. In Table 6.1 we show the dependence between the number of considered anchors m and the portion of problems covered by the anchors. For further experiments, we have considered 26 anchors.

# Anchors	1	8	26	72	130	465
Problems covered	14.0 %	51.2 %	75.4 %	90.0 %	95.0 %	100 %

Table 6.1: Percentage of problems from the generated dataset which are covered by different numbers of anchors.

We would like to know how do these anchors generalize to different sets of problem-solution pairs. Therefore, we have generated another set of points according to Section 2.1.3. The set consists of 37700 problems. In Table 6.2 we show how many problems are covered by anchor sets of different sizes. We can see from the table that the anchors generalize well to different problems.

# Anchors	1	8	26	72	130	465
Covered	12.86%	50.30%	73.81%	87.36%	91.67%	95.91%

Table 6.2: Percentage of problems from a different dataset which are covered by anchor sets of various sizes.

6.2.2 Training of the classifier

For training of the classifier, we have generated 1015870 training problems and 32589 validation problems according to Section 2.1.3. Then, we have transformed the problems into the training and testing data according to Section 4.6.3 and we have trained the classifier using this data according to Section 4.6.4. We have trained the neural network using SGD optimizer with learning rate 0.001, momentum 0.9 and batch size 32. We have trained the network for 110 epochs and, eventually, we have selected the intermediate result which classifies correctly the highest number of problems from the validation dataset. The final neural network is able to classify correctly 6958 out of 32589 validation problems.

6.2.3 Evaluation of the solver

We have trained the solver described in Chapter 5 according to Sections 6.2.1 and 6.2.2. Then, we have generated a set of 195464 testing problems according to Section 2.1.3 from the ETH dataset "Facade". We have used the trained solver to solve every problem from the testing dataset. Then, we have measured the average running time of the solver and the number of successfully solved problems. The number of problems successfully solved with the solver is 43050, which is about 22% of all problems.

6.3 Benchmark of the Five-Point Solver

Now, we are going to describe how the Five-Point solver (Chapter 4) is evaluated. We use a benchmark described in [Mys20] for the evaluation of our solver. The input to the benchmark is a sequence of camera pairs. For every camera pair, we know a set of tentative matches between the cameras, the camera calibration matrix, and the ground truth relative pose between the cameras. For every camera pair, we want to compute the relative pose from the tentative matches and to evaluate the distance between the computed relative pose and the ground truth pose. Let us introduce the notation which we will use in this section.

$T = \{(x_i^U, y_i^U, K_{1i}, K_{2i}, R_i, t_i)\}_{i\ [1,N]}$ The sequence of N testing camera pairs, each of which consists of M_i tentative correspondences.

$x_i^\epsilon, y_i^\epsilon$ R^{2,M_i} are tentatively matched uncalibrated points, K_{1i}, K_{2i} $\mathsf{R}^{3,3}$ are the calibration cameras, and R_i, t_i is the ground truth relative pose.

x_i^C, y_i^C R^{2,n_i} Calibrated matched points obtained from the uncalibrated points x_i^U, y_i^U

(R_i, t_i) Relative pose obtained from the matches x_i^C, y_i^C using a RANSAC procedure.

$dist_i$ Distance (6.3) between the obtained relative pose (R_i, t_i) and the ground truth pose (R_i, t_i) from T.

τ_i R Time needed for the RANSAC procedure to computate (R_i, t_i).

α R mAA score (Section 6.1.2) obtained from the sequence of errors $dist_i$, $i \in \{1, ..., N\}$

μ N The number of samples considered in the RANSAC loop.

θ R The threshold used in the RANSAC loop. If the Sampson error measured on the uncalibrated pair of matched points is $\leq \tau$, the match is considered an inlier.

$x_{i,j}^C$ $\mathsf{R}^{2,5}$ A sample of five points from x_i^C obtained in the j-th step of RANSAC.

$y_{i,j}^C$ $\mathsf{R}^{2,5}$ A sample of five points from y_i^C obtained in the j-th step of RANSAC. The points are matched with points in $y_{i,j}^C$.

$E_{i,j,k}$ $\mathsf{R}^{3,3}$ k-th essential matrix obtained in the j-th step of the RANSAC loop solving the i-th problem from T.

$F_{i,j,k}$ $\mathsf{R}^{3,3}$ Fundamental matrix obtained from the essential matrix $E_{i,j,k}$. There holds $F_{i,j,k} = K_2^{-T} E_{i,j,k} K_1^{-T}$.

$\mathsf{E}_{i,j}$ Set of n essential matrices $E_{i,j,k}$ obtained by the Five-Point solver. For our solver $n \in \{0, 1\}$, for the Nistér solver $n \leq 10$.

In the benchmark, the tentative correspondences are calibrated using K_{1i}, K_{2i} as follows:

$$\begin{bmatrix} x_i^C \\ \vec{1}^T \end{bmatrix} = K_{1i}^{-1} \begin{bmatrix} x_i^U \\ \vec{1}^T \end{bmatrix}$$
$$\begin{bmatrix} y_i^C \\ \vec{1}^T \end{bmatrix} = K_{2i}^{-1} \begin{bmatrix} y_i^U \\ \vec{1}^T \end{bmatrix} \tag{6.6}$$

Where $\vec{1}^T$ is a row vector of ones. Then, the relative pose between the cameras is obtained using the RANSAC scheme. Then, the quality of the obtained

relative pose is evaluated using the distance (6.3). Finally, the Mean Average Accuracy (mAA) score is obtained according to Section 6.1.2.

In order to evaluate our solver, we plug it into the RANSAC scheme instead of the Nistér solver. The generic benchmark is described in Algorithm 19. The original RANSAC Scheme using the Nistér solver is described in Algorithm 20. The RANSAC scheme using our solver is described in Algorithm 21. The results of the benchmark are shown in Section 6.3.1.

Algorithm 19: Benchmark

input : Sequence $T = \{(x_i^U, y_i^U, K_{1i}, K_{2i}, R_i, t_i)\}$ of N testing camera pairs, the number of samples μ, threshold θ.

output : Sequence $dist_i$ of distances (6.1.1) between the obtained poses, sequence of processing times τ_i and the mAA score α

for $i \in \{1, ..., N\}$ **do**

  $x_i^C, y_i^C :=$ Calibrate the points in x_i^U, y_i^U according to (6.6);

  $E_i :=$ Essential matrix computed from x_i^C, y_i^C using the RANSAC procedure with parameters μ, θ;

  $\tau_i :=$ Time spent on the RANSAC procedure;

  $(R_i, t_i) :=$ Relative pose consistent with E_i which reconstructs all points in front of both cameras;

  $dist_i :=$ Distance (6.3) between (R_i, t_i) and (R_i, t_i);

end

$\alpha :=$ mAA score obtained from sequence $dist_i$ (Section 6.1.2);

In the RANSAC procedure using the Nistér solver, μ iterations are performed. In every iteration $j \in \{1, ..., \mu\}$, five matched points $x_{i,j}^C, y_{i,j}^C$ are sampled from the calibrated points x_i^C, y_i^C. Then, we obtain a set of $n \leq 10$ essential matrices $\mathsf{E}_{i,j} = E_{i,j,k}, k \in \{1, ..., n\}$ using the Nistér solver (Section 2.1.1). We build the fundamental matrix $F_{i,j,k}$ from the essential matrix $E_{i,j,k}$ as:

$$F_{i,j,k} = K_{2i}^{-T} E_{i,j,k} K_{1i}^{-T} \tag{6.7}$$

After that, we compute the Sampson error (6.5) for every pair of matched uncalibrated points from x_i^U, y_i^U. The match is considered as the inlier, if the Sampson error on the point is smaller than or equal to threshold θ. At the end, we output the essential matrix $E_{i,j,k}$ which has the highest number of inliers. The RANSAC procedure for the Nistér solver is described in Algorithm 20.

For the evaluation of our solver, we replace the Nistér solver with it. In this case, the set $\mathsf{E}_{i,j}$ of the essential matrices obtained by the solver contains at most one essential matrix $E_{i,j}$ or is empty, if the homotopy continuation has failed. We do not have to decompose the final essential matrix into rotation and translation, as these are obtained directly by the solver. RANSAC procedure for our solver is described in Alg. 21.

Algorithm 20: RANSAC Nistér

input : Tentatively matched calibrated points x_i^C, y_i^C, Camera calibration matrices K_{1i}, K_{2i}, Tentatively matched uncalibrated points x_i^U, y_i^U, the number of samples μ, threshold θ.

output : E_i the essential matrix relating the points in x_i^C, y_i^C

$score_{best} := 0$;

for $j \quad \{1, ..., \mu\}$ **do**

  $x_{i,j}^C, y_{i,j}^C :=$ Sample five points from x_i^C, y_i^C;

  $\mathsf{E}_{i,j} :=$ Find a set of Essential matrices consistent with $x_{i,j}^C, y_{i,j}^C$ using Nistér algorithm (Section 2.1.1);

  for $E_{i,j,k} \quad \mathsf{E}_{i,j}$ **do**

    $F_{i,j,k} := K_{2i}^{-T} E_{i,j,k} K_{1i}^{-T}$;

    $score :=$ Number of matches from x_i^U, y_i^U whose Sampson error (6.5) with Fundamental matrix $F_{i,j,k}$ is $\leq \theta$;

    if $score < score_{best}$ **then**

      $E_i := E_{i,j,k}$, $score_{best} := score$;

    end

  end

end

Algorithm 21: RANSAC Homotopy

input : Tentatively matched calibrated points x_i^C, y_i^C, Camera calibration matrices K_{1i}, K_{2i}, Tentatively matched uncalibrated points x_i^U, y_i^U, the number of samples μ, threshold θ.

output : E_i the essential matrix relating the points in x_i^C, y_i^C

$score_{best} := 0$;

for $j \quad \{1, ..., \mu\}$ **do**

  $x_{i,j}^C, y_{i,j}^C :=$ Sample five points from x_i^C, y_i^C;

  $E_{i,j} :=$ Find an Essential matrix consistent with $x_{i,j}^C, y_{i,j}^C$ using our homotopy continuation solver (Section 4);

  if $E_{i,j} =$ **then**

    $F_{i,j} := K_{2i}^{-T} E_{i,j} K_{1i}^{-T}$;

    $score :=$ Number of matches from x_i^U, y_i^U whose Sampson error (6.5) with Fundamental matrix $F_{i,j}$ is $\leq \theta$;

    if $score < score_{best}$ **then**

      $E_i := E_{i,j}$, $score_{best} := score$;

    end

  end

end

6.3.1 Evaluation of the benchmark

We have evaluated the benchmark described in Algorithm 19 on the data from [Mys20]. We have evaluated the benchmark for both the Nistér algorithm and our solver and we have compared the resulting errors, mAA and running times.

First, we are going to describe the data. The benchmark contains two validation datasets "St. Peter's Square" and "Sacre Coeur" and 11 test datasets, which are described in Table 6.3. Most of the datasets consist of 4950 camera pairs.

ID	Name	# Camera Pairs	Type
V01	St. Peter's Square	4950	Validation
V02	Sacre Coeur	4950	Validation
T01	British Museum	4950	Test
T02	Florence Cathedral Side	4950	Test
T03	Lincoln Memorial Statue	4950	Test
T04	London Bridge	4950	Test
T05	Milan Cathedral	4950	Test
T06	Mount Rushmore	4950	Test
T07	Piazza San Marco	4950	Test
T08	Reichstag	2701	Test
T08	Sagrada Familia	4950	Test
T10	St. Pauls Cathedral	4950	Test
T11	United States Capitol	4950	Test

Table 6.3: Overview of the datasets used in the benchmark

We have evaluated the benchmark on all datasets in Table 6.3 for both the Nistér algorithm and our solver. In the case of our solver, we distinguish two cases. In the first case (LHC single), only one anchor is considered, i.e., the problems are always tracked from the same starting problem. In the second case (LHC NN), we consider 26 anchors, and the starting point is selected according to Section 4.6.4. We use parameters $\mu = 1000$, $\theta = 3$ for all three methods. The resulting mAA scores are shown in Table 6.4 and average running times in Table 6.5. More detailed evaluation of the rotation errors is shown in Figures 6.1, 6.2, 6.3. The evaluation of the translation errors is shown in Figures 6.4, 6.5, 6.6. The Figures show that the Nistér solver is able to output poses with higher precision than the homotopy continuation solver, which could be expected, as the Nistér solver succeeds in almost all cases and outputs all solutions to the problem. The experiments also show that the Homotopy Continuation solver with the Neural Network is faster and its precision is not much worse than the precision of the Nistér solver.

The Homotopy Continuation solver may be generalized to different minimal problems, for which a symbolic solution is difficult to find.

ID	LHC single	LHC NN	Nistér
V01	0.420727	0.47703	0.547232
V02	0.567919	0.670525	0.760465
T01	0.490465	0.520828	0.527495
T02	0.631333	0.688909	0.712465
T03	0.506909	0.600384	0.709596
T04	0.224222	0.30099	0.354384
T05	0.628606	0.646505	0.656242
T06	0.294101	0.339556	0.372061
T07	0.321374	0.357051	0.377192
T08	0.612181	0.640133	0.717623
T09	0.554364	0.593253	0.644566
T10	0.556182	0.592889	0.658505
T11	0.10503	0.162162	0.277495

Table 6.4: Table of Mean Average Accuracy of the methods LHC single, LHC NN, and Nistér on every dataset from 6.3.

ID	LHC single	LHC NN	Nistér
V01	31.68 ms	31.56 ms	41.95 ms
V02	31.89 ms	32.84 ms	46.41 ms
T01	31.34 ms	33.32 ms	51.36 ms
T02	31.03 ms	34.89 ms	51.94 ms
T03	30.54 ms	32.61 ms	46.73 ms
T04	32.54 ms	32.62 ms	44.50 ms
T05	31.06 ms	34.22 ms	50.75 ms
T06	31.08 ms	31.76 ms	51.00 ms
T07	33.68 ms	34.40 ms	42.80 ms
T08	32.27 ms	33.69 ms	49.22 ms
T09	32.06 ms	36.98 ms	52.94 ms
T10	32.24 ms	35.52 ms	48.90 ms
T11	31.68 ms	31.56 ms	41.95 ms

Table 6.5: Table of Average running time of RANSAC using the methods LHC single, LHC NN, and Nistér on every dataset from 6.3.

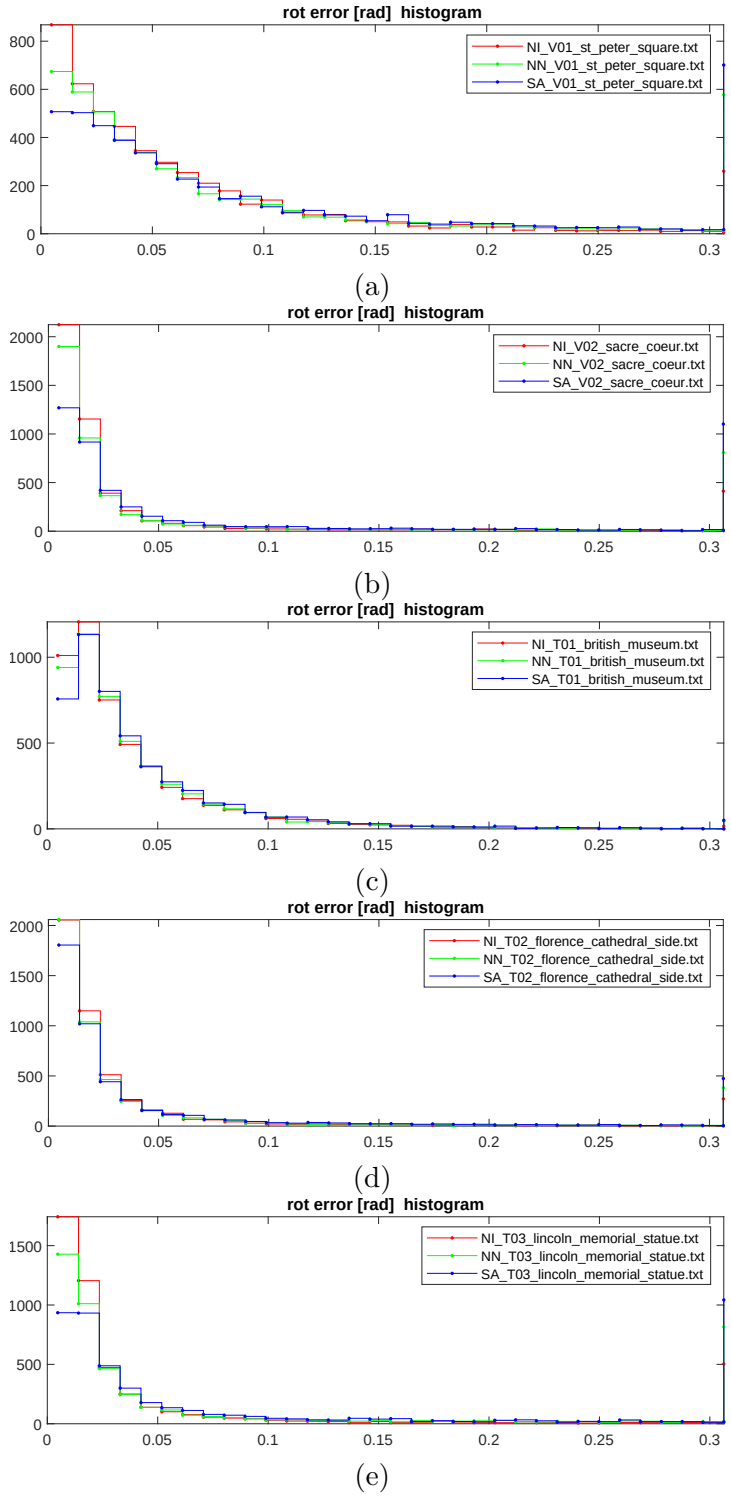


Figure 6.1: Histogram of the rotation errors of the problems in dataset V01 (a), V02 (b), T01 (c), T02 (d), T03 (e). The results of Nistér solver are red, the results of our solver with Neural network are green, and the results of our method with a single anchor are blue.

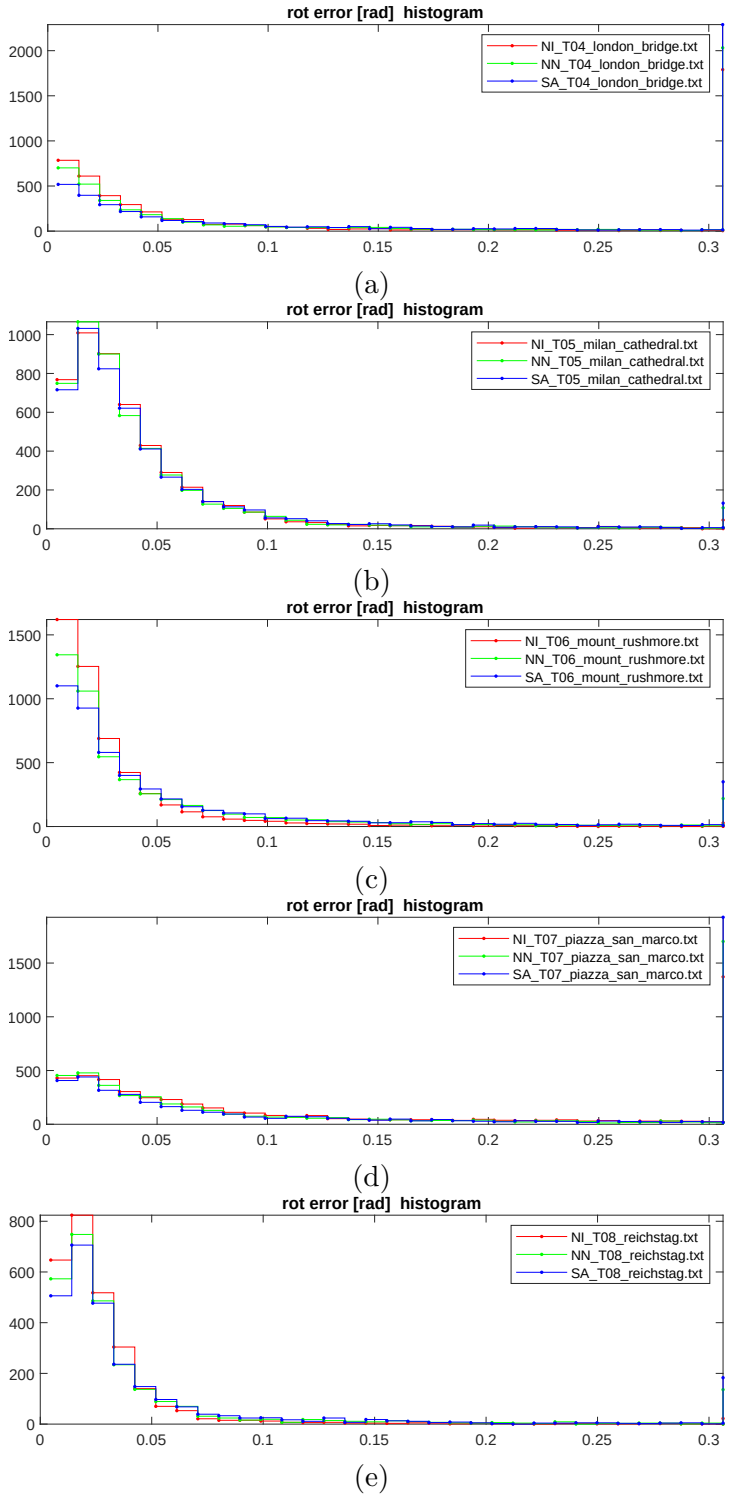


Figure 6.2: Histogram of the rotation errors of the problems in dataset T04 (a), T05 (b), T06 (c), T07 (d), T08 (e). The results of Nistér solver are red, the results of our solver with Neural network are green, and the results of our method with a single anchor are blue.

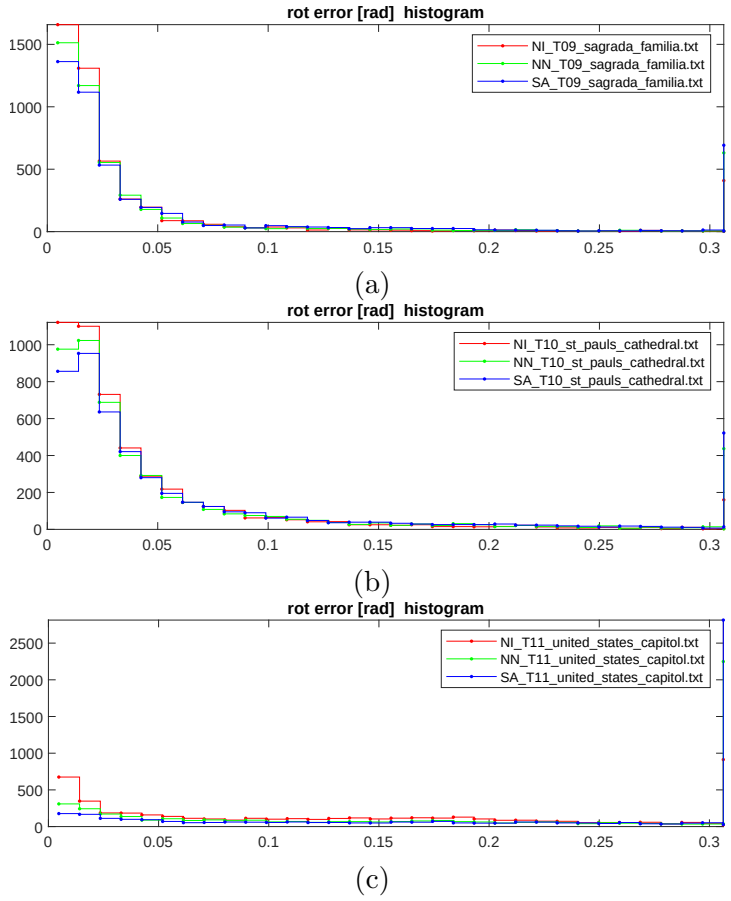


Figure 6.3: Histogram of the rotation errors of the problems in dataset T09 (a), T10 (b), T11 (c). The results of Nistér solver are red, the results of our solver with Neural network are green, and the results of our method with a single anchor are blue.

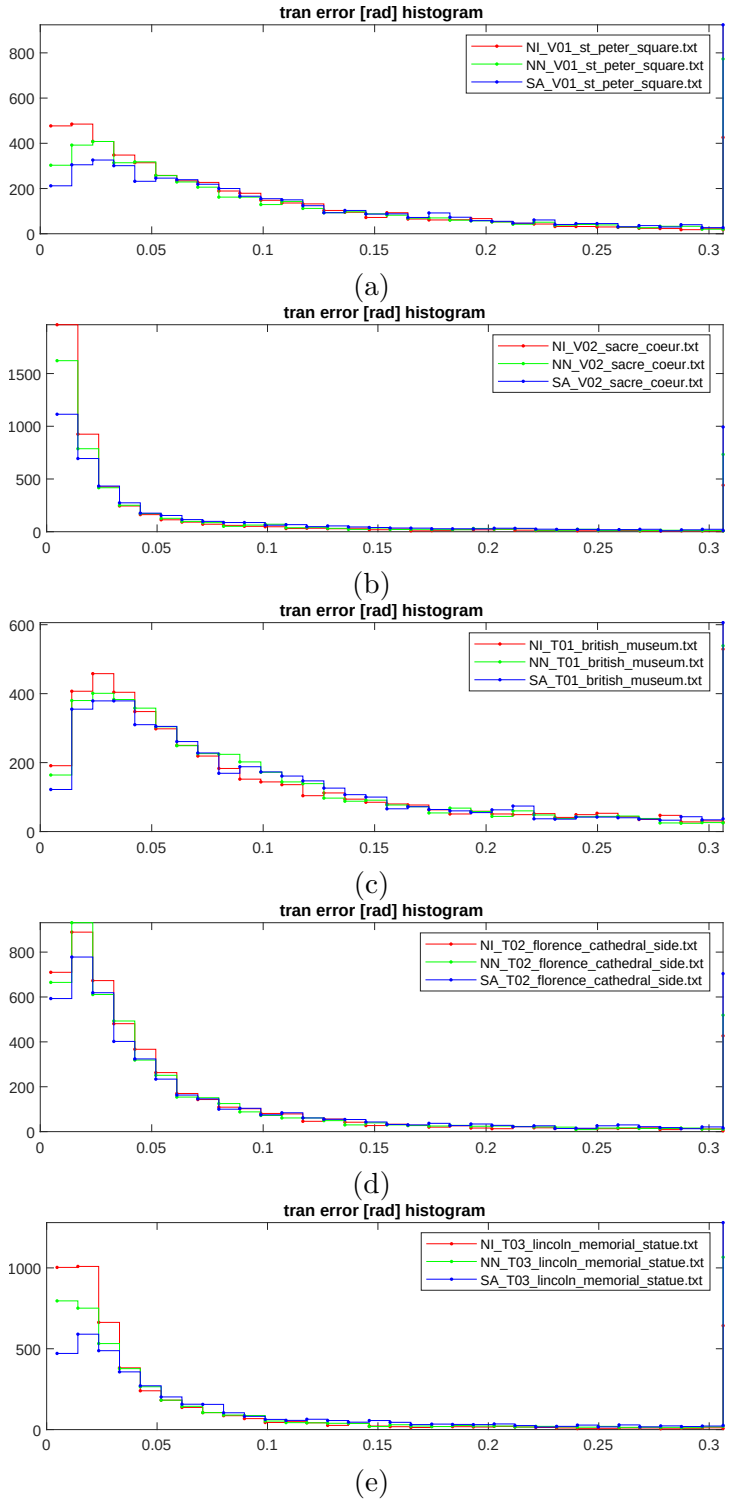


Figure 6.4: Histogram of the translation errors of the problems in dataset V01 (a), V02 (b), T01 (c), T02 (d), T03 (e). The results of Nistér solver are red, the results of our solver with Neural network are green, and the results of our method with a single anchor are blue.

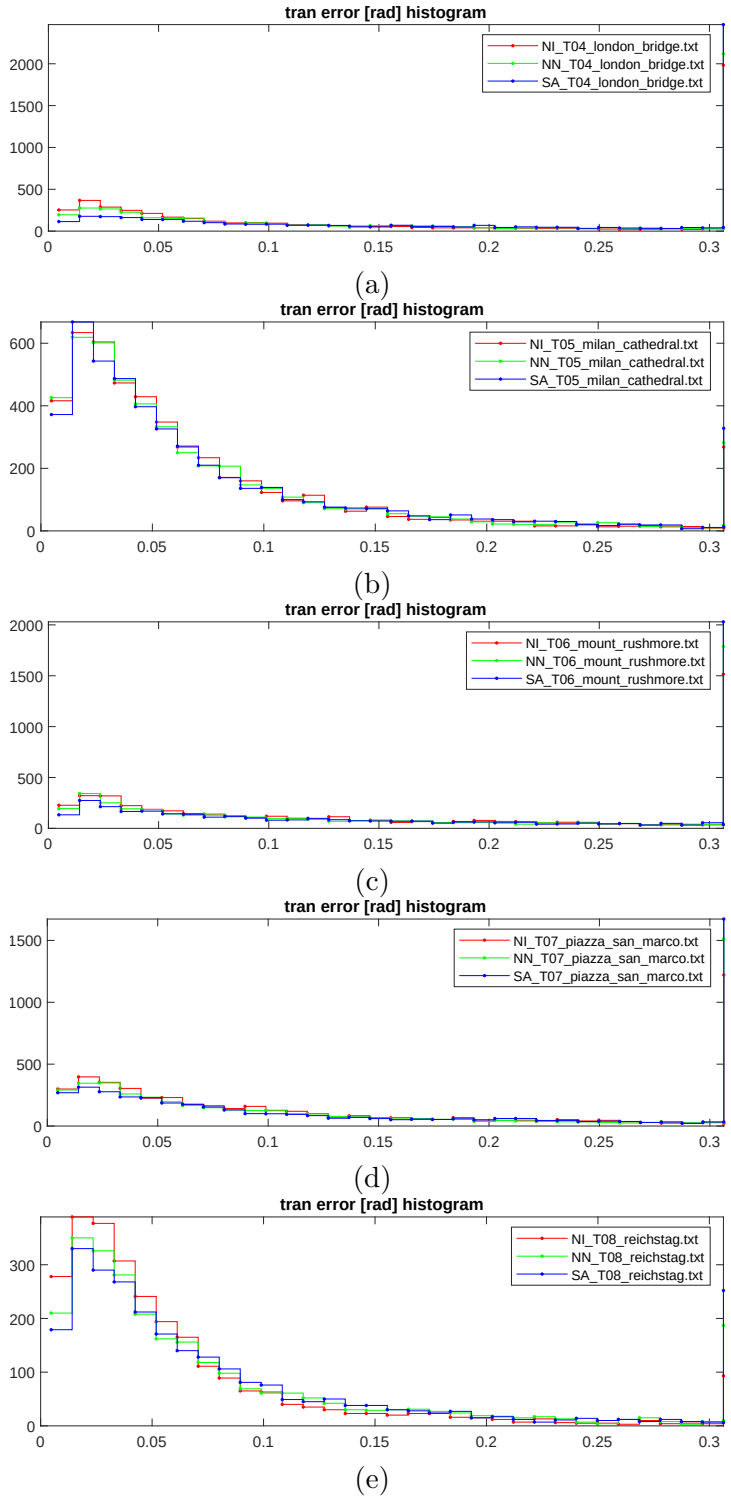


Figure 6.5: Histogram of the translation errors of the problems in dataset T04 (a), T05 (b), T06 (c), T07 (d), T08 (e). The results of Nistér solver are red, the results of our solver with Neural network are green, and the results of our method with a single anchor are blue.

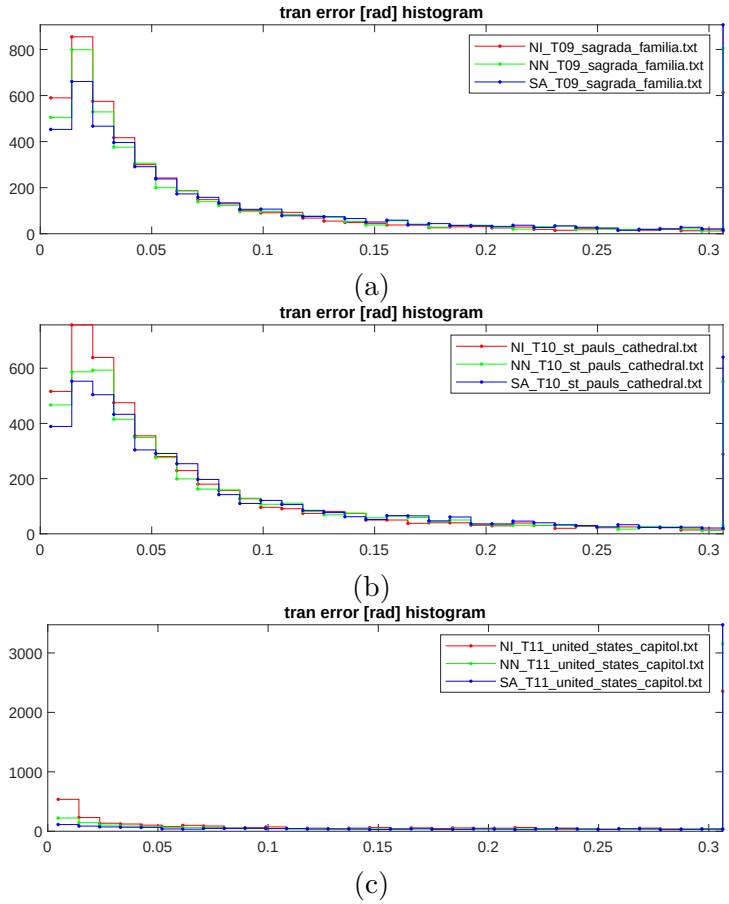


Figure 6.6: Histogram of the translation errors of the problems in dataset T09 (a), T10 (b), T11 (c). The results of Nistér solver are red, the results of our solver with Neural network are green, and the results of our method with a single anchor are blue.

6.4 Training and evaluation of the Four-Point problem

In this section, we are going to describe the results obtained during the training and evaluation of the Four-Point solver. The training of the classifier consists of the generation of the set of anchors according to 5.6.1 and the training of the classifier according to 5.6.4. In the evaluation part, we describe the success rate of the solver, as well as its running time.

6.4.1 Generation of the anchors

We have generated the anchors according to Section 5.6.1 from a set of 10773 points generated according to Section 2.2.3. We say that a problem **is covered** by a set of anchors, if there exists an anchor in the set, from which the problem is correctly tracked. In Table 6.6 we show the dependence between the number of considered anchors m and the portion of problems covered by the anchors. For further experiments, we have considered 62 anchors.

# Anchors	1	17	43	62	93	134	400
Problems covered	5.4 %	50 %	75 %	83 %	90 %	95 %	100 %

Table 6.6: Percentage of problems from the generated dataset which are covered by different numbers of anchors.

We would like to know how do these anchors generalize to real problems. Therefore, we have also generated two sets of points according to Section 2.2.4. The first set (Real) consists of the points generated with measurement errors just according to Section 2.2.4. The second set (Exact) consists of the same points reprojected to the ground truth cameras, therefore, the measurement error is zero. Both sets consist of 6054 problems. In Table 6.7 we show how many problems have been covered by anchor sets of different sizes. In most of experiments, the resulting pose is considered correct if the distance from the ground truth is $\leq 5^\circ$. Here, we have considered multiple different thresholds for the result to be considered correct: 1°, 5°, 10°, 20°.

# Anchors	1	17	43	62	93	134	400
Real 1°	0.23%	3.35%	5.80%	6.99%	8.37%	9.60%	13.58%
Real 5°	2.15%	23.13%	35.15%	40.27%	44.91%	48.94%	58.46%
Real 10°	5.32%	42.24%	58.80%	64.06%	69.24%	73.06%	82.04%
Real 20°	11.78%	66.52%	83.02%	87.03%	90.01%	92.17%	95.99%
Exact 1°	2.08%	24.33%	41.91%	50.20%	58.61%	64.90%	80.46%
Exact 5°	4.87%	44.91%	66.91%	74.64%	81.95%	86.64%	95.18%
Exact 10°	7.66%	58.03%	79.86%	86.42%	91.15%	94.20%	98.46%
Exact 20°	13.68%	74.81%	91.67%	95.00%	97.44%	98.45%	99.39%

Table 6.7: Percentage of problems from the real datasets which are covered by different numbers of anchors.

We can see from Table 6.7, that the anchors generalize well in the case of the exact problems. In the case of the real problems, the small change of projected points may cause a significant change of depths in the obtained solution, which may then cause a significant change of the relative pose.

Therefore, the results on the real dataset have bigger errors than the results on the exact dataset. Still, the number of problems covered by the anchors with the tolerance of 10° and 20° is similar to the coverage of the original dataset in Table 6.6. We believe that with errors of this magnitude we are able to obtain a better solution using local optimization.

6.4.2 Training of the classifier

For training of the classifier, we have generated 772054 training problems and 32028 validation problems according to Section 2.2.4. Then, we have transformed the problems into training and testing data according to Section 5.6.3 and we have trained the classifier using this data according to Section 5.6.4. We have trained the neural network using SGD optimizer with learning rate 0.001, momentum 0.9 and batch size 32. We have trained the network for 40 epochs and, eventually, we have selected the intermediate result which classifies correctly the highest number of problems from the validation dataset. The final neural network is able to classify correctly 3019 out of 32028 validation problems.

6.4.3 Evaluation of the solver

We have trained the solver described in Chapter 5 according to Sections 6.4.1 and 6.4.2. Then, we have generated a set of 16752 testing problems according to Section 2.2.4 from the ETH dataset "Facade". We have used the solver to solve every problem from the testing dataset. Then, we have measured the average running time of the solver and the percentage of successfully solved problems. We have considered different variants of the solver: NN1 is the solver described in Section 5, NN2 selects for each input problem two anchors with the highest score given by the neural network and tracks each problem from two anchors, NN4 tracks each problem from four top anchors. Like in Section 6.4.1, we have considered multiple thresholds for accepting the result: 5°, 10°, 20°. The results are given in Table 6.8.

Method	5°	10°	20°	Time
NN1	13.34%	17.57%	22.92%	56.7μs
NN2	19.77%	25.88%	33.83%	82.8μs
NN4	27.50%	36.26%	47.29%	139.2μs

Table 6.8: Evaluation of the trained Four-Point solver.

Table 6.8 shows that the success rate of the solver is about 13% with threshold of 5° and a single track for every problem. If the results are evaluated with a more relaxed threshold or multiple tracks are performed for each problem, we can achieve the same or even better success rate than in the case of the Five-Point problem.

Chapter 7

Conclusion

In the thesis, we have studied the problem of estimation of the relative pose from five points in two views, and the problem of estimation of the relative pose from four points in three views. We have designed generators of problem-solution pairs for the problems. Then, we have proposed two solvers for the considered problems based on Homotopy Continuation. We have performed several experiments to evaluate the proposed solvers.

Instead of tracking all solutions to the problem, we track only one solution for every input problem from one starting problem, which has been selected using a neural network. This, together with the tracking in the real domain only and with an efficient evaluation of the linear equations arising in the Homotopy Continuation, allows us to achieve the high speed of the solvers. Therefore, the solvers are eligible for use in the RANSAC scheme, which has been shown in the experiments.

In the future, we would like to design better classifiers, which would allow us to select the starting solutions with a higher success rate.

Bibliography

[AL87] N. Ayache and L. Lustman, *Fast and reliable passive trinocular stereo vision*, 1st International Conference on Computer Vision (1987), 422–427.

[AO14] Chris Aholt and Luke Oeding, *The ideal of the trifocal variety*, Math. Comput. **83** (2014), no. 289, 2553–2574.

[AT10] Alberto Alzati and Alfonso Tortora, *A geometric approach to the trifocal tensor*, J. Math. Imaging Vis. **38** (2010), no. 3, 159–170.

[BF81] Robert C. Bolles and Martin A. Fischler, *A ransac-based approach to model fitting and its application to finding cylinders in range data*, Proceedings of the 7th International Joint Conference on Artificial Intelligence, IJCAI '81, Vancouver, BC, Canada, August 24-28, 1981 (Patrick J. Hayes, ed.), William Kaufmann, 1981, pp. 637–643.

[BHM+20] Edgar A. Bernal, Jonathan D. Hauenstein, Dhagash Mehta, Margaret H. Regan, and Tingting Tang, *Machine learning the real discriminant locus*, CoRR **abs/2006.14078** (2020).

[BHN94] Alfred M. Bruckstein, Robert J. Holt, and Arun N. Netravali, *How to catch a crook*, J. Vis. Commun. Image Represent. **5** (1994), no. 3, 273–281.

[BHSW] Daniel J. Bates, Jonathan D. Hauenstein, Andrew J. Sommese, and Charles W. Wampler, *Bertini: Software for numerical algebraic geometry*, Available at bertini.nd.edu with permanent doi: dx.doi.org/10.7274/R0H41PB5.

[Bli96] J. Blinn, *Consider the lowly 2 x 2 matrix*, IEEE Computer Graphics and Applications **16** (1996), no. 2, 82–88.

[BSHW13] Daniel J. Bates, Andrew J. Sommese, Jonathan D. Hauenstein, and Charles W. Wampler, *Numerically solving polynomial systems with bertini*, Software, environments, tools, vol. 25, SIAM, 2013.

[CG00] Roberto Cipolla and Peter J. Giblin, *Visual motion of curves and surfaces*, Cambridge University Press, 2000.

[CLL14] Tianran Chen, T. Lee, and T. Li, *Hom4ps-3: A parallel numerical solver for systems of polynomial equations based on polyhedral homotopy continuation methods*, ICMS, 2014.

[CLO07] David A. Cox, John Little, and Donal O'Shea, *Ideals, varieties, and algorithms: An introduction to computational algebraic geometry and commutative algebra, 3/e (undergraduate texts in mathematics)*, Springer-Verlag, Berlin, Heidelberg, 2007.

[DHJ+16] Timothy Duff, Cvetelina Hill, Anders Nedergaard Jensen, Kisun Lee, Anton Leykin, and Jeff Sommars, *Solving polynomial systems via homotopy continuation and monodromy*, CoRR **abs/1609.08722** (2016).

[Die19] Marc Diesse, *On local real algebraic geometry and applications to kinematics*, CoRR **abs/1907.12134** (2019).

[DKLP19] Timothy Duff, Kathlén Kohn, Anton Leykin, and Tomás Pajdla, *PLMP - point-line minimal problems in complete multi-view visibility*, 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, IEEE, 2019, pp. 1675–1684.

[DKLP20] ———, *Pl$_1$p - point-line minimal problems under partial visibility in three views*, Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXVI (Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, eds.), Lecture Notes in Computer Science, vol. 12371, Springer, 2020, pp. 175–192.

[EJ10] Ady Ecker and Allan D. Jepson, *Polynomial shape from shading*, The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010, IEEE Computer Society, 2010, pp. 145–152.

[Fab10] Ricardo Fabbri, *Multiview di erential geometry in application to computer vision*, Ph.D. thesis, Division Of Engineering, Brown University, Providence, 2010.

[FDF+20] Ricardo Fabbri, Timothy Duff, Hongyi Fan, Margaret H. Regan, David da Costa de Pinho, Elias P. Tsigaridas, Charles W. Wampler, Jonathan D. Hauenstein, Peter J. Giblin, Benjamin B. Kimia, Anton Leykin, and Tomás Pajdla, *TRPLP - trifocal relative pose from lines at points*, 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, IEEE, 2020, pp. 12070–12080.

[FKG12] Ricardo Fabbri, Benjamin B. Kimia, and Peter J. Giblin, *Camera pose estimation using first-order curve di erential geometry*, Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part IV (Andrew W. Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, eds.), Lecture Notes in Computer Science, vol. 7575, Springer, 2012, pp. 231–244.

[FLM92] Olivier D. Faugeras, Quang-Tuan Luong, and Stephen J. Maybank, *Camera self-calibration: Theory and experiments*, Computer Vision - ECCV'92, Second European Conference on Computer Vision, Santa Margherita Ligure, Italy, May 19-22, 1992, Proceedings (Giulio Sandini, ed.), Lecture Notes in Computer Science, vol. 588, Springer, 1992, pp. 321–334.

[HC62] John R. Hurley and Raymond B. Cattell, *The procrustes program: Producing direct rotation to test a hypothesized factor structure*, Behavioral Science **7** (1962), no. 2, 258–262.

[HN94] Robert J. Holt and Arun N. Netravali, *Motion and structure from line correspondences: Some further results*, Int. J. Imaging Syst. Technol. **5** (1994), no. 1, 52–61.

[HNH90] R. J. Holt, A. Netravali, and T. Huang, *Experience in using homotopy methods to solve motion estimation problems*, Other Conferences, 1990.

[HR18] Jonathan D. Hauenstein and Margaret H. Regan, *Adaptive strategies for solving parameterized systems using homotopy continuation*, Appl. Math. Comput. **332** (2018), 19–34.

[HR20] _____, *Real monodromy action*, Appl. Math. Comput. **373** (2020), 124983.

[JOÅ02] Björn Johansson, Magnus Oskarsson, and Kalle Åström, *Structure and motion estimation from complex features in three views*, ICVGIP 2002, Proceedings of the Third Indian Conference on Computer Vision, Graphics & Image Processing, Ahmadabad, India, December 16-18, 2002 (Subhasis Chaudhuri, Andrew Zisserman, Anil K. Jain, and Kantilal L. Majumder, eds.), Allied Publishers Private Limited, 2002.

[KGB19] Yoni Kasten, Meirav Galun, and Ronen Basri, *Resultant based incremental recovery of camera pose from pairwise matches*, IEEE Winter Conference on Applications of Computer Vision, WACV 2019, Waikoloa Village, HI, USA, January 7-11, 2019, IEEE, 2019, pp. 1080–1088.

[Kil17] Joe Kileel, *Minimal problems for the calibrated trifocal variety*, SIAM J. Appl. Algebra Geom. **1** (2017), no. 1, 575–598.

[KOÅ14] Yubin Kuang, Magnus Oskarsson, and Karl Åström, *Revisiting trifocal tensor estimation using lines*, Pattern Recognition (ICPR), 2014 22nd International Conference on, IEEE - Institute of Electrical and Electronics Engineers Inc., 2014, 22nd International Conference on Pattern Recognition (ICPR 2014) ; Conference date: 24-08-2014 Through 28-08-2014, pp. 2419–2423 (English).

[Kor20] Viktor Korotynskiy, *Using monodromy to simplify polynomial systems*, Master's thesis, Czech Technical University in Prague, 2020.

[KP89] D.J. Kriegman and J. Ponce, *Computing exact aspect graphs of curved objects: solids of revolution*, [1989] Proceedings. Workshop on Interpretation of 3D Scenes, 1989, pp. 116–122.

[KP91] David J. Kriegman and Jean Ponce, *A new curve tracing algorithm and some applications*, p. 267–270, Academic Press Professional, Inc., USA, 1991.

[KP92] David J. Kriegman and Jean Ponce, *Geometric modeling for computer vision*, Curves and Surfaces in Computer Vision and Graphics II (Martine J. Silbermann and Hemant D. Tagare, eds.), vol. 1610, International Society for Optics and Photonics, SPIE, 1992, pp. 250 – 260.

[LÅO17] Viktor Larsson, Kalle Åström, and Magnus Oskarsson, *E cient solvers for minimal problems by syzygy-based reduction*, 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, IEEE Computer Society, 2017, pp. 2383–2392.

[Ley09] Anton Leykin, *Numerical algebraic geometry for macaulay2*, CoRR **abs/0911.1783** (2009).

[LOÅ+18] Viktor Larsson, Magnus Oskarsson, Kalle Åström, Alge Wallis, Zuzana Kukelova, and Tomás Pajdla, *Beyond grobner bases: Basis selection for minimal solvers*, 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, IEEE Computer Society, 2018, pp. 3945–3954.

[LTD15] Spyridon Leonardos, Roberto Tron, and Kostas Daniilidis, *A metric parametrization for trifocal tensors with non-colinear pinholes*, IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015, IEEE Computer Society, 2015, pp. 259–267.

[Luo92] Q.-T. Luong, *Matrice fondamentale et calibration visuellesur l'environnement-vers une plus grande autonomie des systemes robotiques.*, Ph.D. thesis, Universitè de Paris-Sud, Centre d'Orsay, 1992.

[Mar17] Evgeniy V. Martyushev, *On some properties of calibrated trifocal tensors*, J. Math. Imaging Vis. **58** (2017), no. 2, 321–332.

[Mat16] J. Mathews, *Multi-focal tensors as invariant di erential forms.*, arXiv e-prints (2016), 1610.0429.

[MF92] Stephen J. Maybank and Olivier D. Faugeras, *A theory of self-calibration of a moving camera*, Int. J. Comput. Vis. **8** (1992), no. 2, 123–151.

[Mor09] A. Morgan, *Solving polynomial systems using continuation for engineering and scientific problems*, Classics in Applied Mathematics, Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 2009.

[Mys20] Dmytro Myshkin, *Benchmarking robust estimation methods*.

[NDP94] P. K. Nanda, Uday B. Desai, and P. G. Poonacha, *A homotopy continuation method for parameter estimation in MRF models and image restoration*, 1994 IEEE International Symposium on Circuits and Systems, ISCAS 1994, London, England, UK, May 30 - June 2, 1994, IEEE, 1994, pp. 273–276.

[Nis04] David Nistér, *An e cient solution to the five-point relative pose problem*, IEEE Trans. Pattern Anal. Mach. Intell. **26** (2004), no. 6, 756–777.

[NS06] David Nistér and Frederik Schaffalitzky, *Four points in two or three calibrated views: Theory and practice*, Int. J. Comput. Vis. **67** (2006), no. 2, 211–231.

[Oed15] Luke Oeding, *The quadrifocal variety*, CoRR **abs/1501.01266** (2015).

[OZÅ04] Magnus Oskarsson, Andrew Zisserman, and Kalle Åström, *Minimal projective reconstruction for combinations of points and lines in three views*, Image Vis. Comput. **22** (2004), no. 10, 777–785.

[Paj21] Tomáš Pajdla, *Elements of geometry for robotics*.

[Pet99] Sylvain Petitjean, *Algebraic geometry and computer vision: Polynomial systems, real and complex roots*, J. Math. Imaging Vis. **10** (1999), no. 3, 191–220.

[PG99] Marc Pollefeys and Luc Van Gool, *Stratified self-calibration with the modulus constraint*, IEEE Trans. Pattern Anal. Mach. Intell. **21** (1999), no. 8, 707–724.

[Pol] Marc Pollefeys, *Vnl realnpoly: A solver to compute all the roots of a system of n polynomials in n variables through continuation*, Available at `https://github.com/vxl/vxl/blob/master/core/vnl/algo/vnl_rnpoly_solve.h`.

[QN17] Ashraf Qadir and Jeremiah Neubert, *A line-point unified solution to relative camera pose estimation*, CoRR **abs/1710.06495** (2017).

[QTAM01] Long Quan, Bill Triggs, Marc-André Ameller, and Bernard Mourrain, *Uniqueness of minimal euclidean reconstruction from 4 points*.

[QTM06] Long Quan, Bill Triggs, and Bernard Mourrain, *Some results on minimal euclidean reconstruction from four points*, J. Math. Imaging Vis. **24** (2006), no. 3, 341–348.

[RF91] Luc Robert and Olivier D. Faugeras, *Curve-based stereo: figural continuity and curvature*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 1991, 3-6 June, 1991, Lahaina, Maui, Hawaii, USA, IEEE, 1991, pp. 57–62.

[RJH97] Arun N. Netravali Robert J. Holt, *Number of solutions for motion and structure from multiple frame correspondence*, Int. J. Comput. Vis. **23** (1997), no. 1, 5–15.

[Rod15] Volker Rodehorst, *Evaluation of the metric trifocal tensor for relative three-view orientation*, Aug 2015.

[Sal13] Mathieu Salzmann, *Continuous inference in graphical models with polynomial energies*, 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013, IEEE Computer Society, 2013, pp. 1744–1751.

[SI05] Andrew J. Sommese and Charles W. Wampler II, *The numerical solution of systems of polynomials - arising in engineering and science*, World Scientific, 2005.

[SZ00] Cordelia Schmid and Andrew Zisserman, *The geometry and matching of lines and curves over multiple views*, Int. J. Comput. Vis. **40** (2000), no. 3, 199–233.

[Ver99] Jan Verschelde, *Algorithm 795: Phcpack: a general-purpose solver for polynomial systems by homotopy continuation*, ACM Trans. Math. Softw. **25** (1999), no. 2, 251–276.

[VLZ19] Alexander Vakhitov, Victor S. Lempitsky, and Yinqiang Zheng, *Stereo relative pose from line and point feature triplets*, CoRR **abs/1907.00276** (2019).

[ZKHM20] Ji Zhao, Laurent Kneip, Yijia He, and Jiayi Ma, *Minimal case relative pose computation using ray-point-ray features*, IEEE Trans. Pattern Anal. Mach. Intell. **42** (2020), no. 5, 1176–1190.