Efficient Solution to the Epipolar Geometry for Radially Distorted Cameras

Zuzana Kukelova[1] Jan Heller[2] Martin Bujnak[3] Andrew Fitzgibbon[1] Tomas Pajdla[2]

[1]Microsoft Research Ltd, 21 Station Road, Cambridge CB1 2FB, UK
{zukuke,awf}@microsoft.com

[2]Czech Technical University in Prague, 166 27 Praha 6, Technická 2, Czech Republic
{hellej1,pajdla}@cmp.felk.cvut.cz

[3]Capturing Reality s.r.o., Syslia 46, 821 05, Bratislava, Slovakia
martin@capturingreality.com

Abstract

The estimation of the epipolar geometry of two cameras from image matches is a fundamental problem of computer vision with many applications. While the closely related problem of estimating relative pose of two different uncalibrated cameras with radial distortion is of particular importance, none of the previously published methods is suitable for practical applications. These solutions are either numerically unstable, sensitive to noise, based on a large number of point correspondences, or simply too slow for real-time applications. In this paper, we present a new efficient solution to this problem that uses 10 image correspondences. By manipulating ten input polynomial equations, we derive a degree 10 polynomial equation in one variable. The solutions to this equation are efficiently found using the Sturm sequences method. In the experiments, we show that the proposed solution is stable, noise resistant, and fast, and as such efficiently usable in a practical Structure-from-Motion pipeline.

1. Introduction

In computer vision, estimation of the uncalibrated relative pose of two cameras from image correspondences—also known as epipolar geometry of fundamental matrix estimation—is a fundamental problem with many applications. It is one of the core components of all Structure-from-Motion (SfM) and 3D reconstruction pipelines [23]. In these applications, stochastic RANSAC-style algorithms [8, 6] are often employed to find the "correct" epipolar geometries of camera pairs, *i.e.*, geometries deemed highly probable, and at the same time to label the correspondences as inliers or outliers (mismatches) of the model.This approach, however, relies on the correctness of the underlying model of the problem.

By far the most common camera model used for estimating the epipolar geometry between two cameras is the standard pinhole camera model [10]. The 5-point relative pose solver [20, 21] for completely calibrated cameras, the 6-point focal length solver [22] for cameras with unknown focal length, or the well known 7-point or 8-point solvers for uncalibrated cameras [10] are frequently used algorithms based on the pinhole camera.

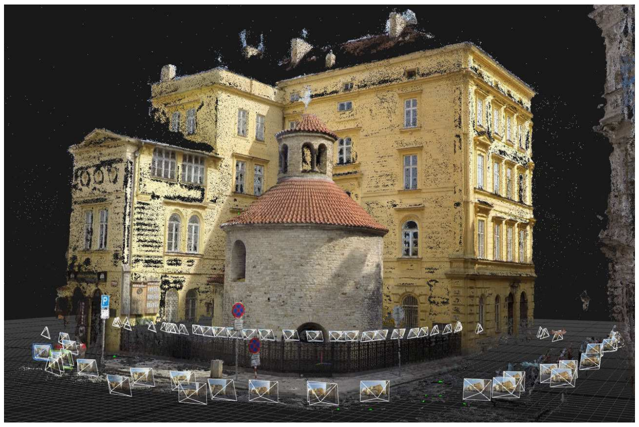
Figure 1: **3D Reconstruction** of a rotunda captured by a GoPro Hero4 camera with varying field-of-view settings. The proposed F10 solver was used as a part of the SfM pipeline.

Since all of the algorithms above assume perspective projections, they do not provide accurate estimates for image correspondences corrupted by radial distortion. This poses a serious drawback, since virtually all projections in real cameras involve some amount of radial distortion. Moreover, the importance of radial distortion modeling increases with the ever growing popularity of wide field of view cameras such as GoPro Hero, cheap smartphone cameras, or superzoom cameras. Fitzgibbon [9] showed that ignoring the radial distortion—even for standard consumer cameras—may lead to significant errors in camera calibration or 3D reconstruction.

One way to deal with radial distortion is to simply ignore it in the image matching phase and to model it only in the final optimization step, *i.e.*, in the bundle adjustment [23] phase. However, this approach is not very robust, since it requires a set of correct image matches that need to cover a significant area in both images and at the same time it needs to include the significantly distorted points lying further from the image center. To obtain correspondence sets that fit the pinhole model and contain enough radially distorted matches at the same time, the epipolar constraints have to be loosened, *e.g.*, by setting higher RANSAC thresholds. This has the unfortunate, yet expected, effect of increasing the number of mismatches, *i.e.*, true outliers wrongly labeled as inliers.

It may be very difficult to find such sets of correct matches by assuming the pinhole camera model only. And in many situations—and especially for cameras with larger radial distortions—this may be outright impossible. It is thus necessary to model the distortion when searching for initial geometry model and inlier correspondences in the RANSAC loop [8, 6]. This helps not only to discover "better quality" consensus sets, but it also provides better models for which the subsequent bundle adjustment is more likely to converge to the correct distortion parameters and correct geometries. Radial distortion modeling has proved to be a mathematically challenging task and algorithms for estimation of epipolar geometry, homography, or absolute camera pose together with radial distortion have been proposed only recently [9, 2, 12, 4, 5, 13, 15].

Fitzgibbon [9] proposed an algorithm for fundamental matrix estimation for cameras with identical radial distortions, under the one-parameter *division model* for modeling radial distortion. Since the singularity constraint on the fundamental matrix is not used in this algorithm, nine point correspondences, instead of the minimal number of eight, are needed to solve this problem. On the other hand, neglecting the singularity constraint means that the problem can be easily formulated as a quadratic eigenvalue problem, which, after being solved using standard eigenvalue solvers, yields up to 10 real solutions.

The first minimal solution to the problem of epipolar geometry estimation under the assumption of the one parameter division model for an uncalibrated camera was proposed by Kukelova *et al.* [17]. This method uses the minimal number of eight point correspondences and the Gröbner basis method [7] to solve the resulting system of polynomial equations. An improved version of this Gröbner basis solver was proposed again by Kukelova in [14]. This time, the solver was generated using an automatic generator of Gröbner basis solvers [14], performing Gauss-Jordan (G-J) elimination of a 32×48 matrix and eigendecomposition of a 16×16 matrix to yield up to 16 solutions.

Recently, several minimal solvers for estimating relative pose between a radially distorted camera and a camera with no or with known radial distortion were proposed by Kuang *et al.* [13]. All of the proposed solvers are based on the Gröbner basis method and need 8 points in the uncalibrated camera scenario, 7 points in the calibrated camera with unknown focal length scenario, and 6 points in the calibrated cameras with known focal length scenario.

In [11], Jiang *et al.* solved the minimal problem of essential matrix estimation for two cameras with constant but unknown focal length and radial distortion. This problem leads to a quite complicated system of polynomial equations and a large solver that needs to perform LU decomposition of a 886×1011 matrix and to compute eigenvalues of a 68×68 matrix. Unfortunately, this makes the solver too computationally expensive for any practical application.

All of the previously listed algorithms estimate one radial distortion parameter only. This parameter is either shared by both cameras, *i.e.*, both cameras exhibit the same amount of radial distortion, or the parameter describes radial distortion of one of the two cameras while the the second camera has no or known radial distortion. This makes these algorithms unusable in situations where the relative pose of two completely uncalibrated cameras with different amounts of radial distortions is to be estimated. This is a practical problem with many applications, *e.g.*, 3D reconstruction from images downloaded from Internet. For the sake of brevity, we will refer to this problem as $\mathrm{F}\lambda_1\lambda_2$ through the rest of this paper.

The problem of fundamental matrix estimation under the assumption of different radial distortions, $\mathrm{F}\lambda_1\lambda_2$, was studied for the first time by Barreto *et al.* [2], where a non-minimal algorithm (F15) based on 15 point correspondences was given. The first work to study the minimal 9-point formulation of $\mathrm{F}\lambda_1\lambda_2$ problem was presented in [18]. The authors found the problem to be numerically very challenging and provided the solution to this problem in exact rational arithmetic only.

The first numerically stable solution to the minimal formulation of the $\mathrm{F}\lambda_1\lambda_2$ problem was proposed in [16, 5] (F9), where the authors used special techniques, based on basis selection and SVD and LU decompositions, to improve the numerical stability of the Gröbner basis solvers involved. Unfortunately, these techniques in combination with the complicated input equations lead to a quite large and slow solver. The solver needs to performs LU decomposition of a 393×389 matrix, SVD decomposition of a 69×69 matrix and to compute eigenvalues of a 24×24 matrix.

In [14], Kukelova *et al.* proposed an alternative Gröbner basis solution ($\mathrm{F9_A}$) to minimal $\mathrm{F}\lambda_1\lambda_2$ formulation. This algorithm uses the automatic generator of Gröbner basis solvers and leads to a slightly smaller solver that performs G-J elimination of a 179×203 matrix and eigendecomposition of a 24×24 matrix. Still, the final solver is too slow for any real-time application. Moreover, since this solution doesn't take any additional steps to improve the numerical stability of the final Gröbner basis solver, it is slightly less numerically stable than F9.

Kukelova *et al.* [5] also suggested a non-minimal solution from 12 point correspondences (F12). This solution is based on the generalized eigenvalue problem, yields up to four real solutions, and is quite fast. However, this algorithm is very noise sensitive and the authors concluded that inside a RANSAC loop, it has worse performance than the minimal 9-point algorithm F9$_A$.

While the F$\lambda_1\lambda_2$ problem is of particular importance, none of the previously published methods is suitable for practical applications. The solvers are either numerically unstable (F9$_A$ [14]), sensitive to noise (F12 [5], F15 [2]), based on a large number of point correspondences (F12 [5], F15 [2]), or simply too slow for real-time applications (F9 [16, 5], F9$_A$ [14]).

In this paper, we present a new solution to the F$\lambda_1\lambda_2$ problem. Our motivation is to propose an algorithm that uses a small number of point correspondences while being sufficiently stable, noise-resistant, and fast for practical real-time applications.

Next, in section §2 we formally introduce the F$\lambda_1\lambda_2$ problem and in section §3, we present our new solution. Finally, in section §4, we thoroughly compare the proposed solvers to the state-of-the-art solutions in terms of numerical stability (§4.1), sensitivity to noise (§4.1), and speed (§4.3).

2. Problem Formulation

Let us consider the epipolar constraint for a pair of image correspondences

$$\mathbf{x}_{u_i}^\top(\lambda_1)\,\mathtt{F}\,\mathbf{x}'_{u_i}(\lambda_2) = 0. \tag{1}$$

In equation (1), $\mathtt{F} = [f_{ij}]_{i,j=1}^3 \in \mathbb{R}^{3\times 3}$ is the fundamental matrix and $\mathbf{x}_{u_i}(\lambda_1), \mathbf{x}'_{u_i}(\lambda_2) \in \mathbb{R}^3$ are the homogeneous coordinates of corresponding ideally projected image points, *i.e.*, points not corrupted by radial distortion [10]. In the next, we will use the one-parameter division model for radial distortion modeling [9]. This is an undistortion model that can handle even quite pronounced radial distortions:

$$\mathbf{x}_{u_i}(\lambda) = \left[x_{d_i}, y_{d_i}, 1 + \lambda(x_{d_i}^2 + y_{d_i}^2)\right]^\top, \tag{2}$$

where $\mathbf{x}_{d_i} = [x_{d_i}, y_{d_i}, 1]^\top$ are the homogeneous coordinates of the measured (and radially distorted) image points and $\lambda \in \mathbb{R}$ is the distortion parameter. Note that this model assumes the centre of distortion to be in the center of the coordinate system. Now, we can formally introduce the F$\lambda_1\lambda_2$ problem as the problem of recovering the unknown fundamental matrix $\mathtt{F}$ and radial distortion parameters λ_1, λ_2, given a set of image correspondences $\mathbf{x}_{d_i} \leftrightarrow \mathbf{x}'_{d_i}$.

As discussed in section §1, the previous attempts to solve F$\lambda_1\lambda_2$ from the minimal number of 9 point correspondences led to a complicated system of polynomial equations and large and relatively slow Gröbner basis solvers [16, 5, 14]. In this work, we have decided to use one more point correspondence that enabled us to develop an algorithm for the F$\lambda_1\lambda_2$ problem that is both stable and fast. We will call this new solver F10 through the rest of the paper. By using one more than the minimal number of points necessary to solve the F$\lambda_1\lambda_2$ problem, we follow the idea proposed by Kukelova *et al.* [15]. There, the authors showed that for practical applications, it is sometimes more effective to use more than the minimal number of correspondences to get a simpler system of polynomial equations that can be solved faster, while maintaining numerical stability.

3. F10: efficient solution to the F$\lambda_1\lambda_2$ problem

In this section, we introduce our new solution to the F$\lambda_1\lambda_2$ problem: the F10 solver. As we will present the algorithm, we will arrive to four different variants of this solution, in order of increasing computational efficiency: F10$_{u3}$, F10$_e$, F10$_s$, and finally F10. This family of of 10-point solvers demonstrates several polynomial solution strategies. It also shows how—given an equivalent problem formulation—different polynomial manipulations lead to different results.

Let us start with the epipolar constraint as expressed by equation (1). The 10 point correspondences $\mathbf{x}_{d_i} \leftrightarrow \mathbf{x}'_{d_i}$ give us 10 equations (1) in 16 monomials $\mathbf{X} = [f_{11}, f_{12}, f_{21}, f_{22}, \lambda_1 f_{31}, f_{31}, \lambda_1 f_{32}, f_{32}, \lambda_2 f_{13}, f_{13}, \lambda_2 f_{23}, f_{23}, f_{33}, \lambda_1 f_{33}, \lambda_2 f_{33}, \lambda_1\lambda_2 f_{33}]^\top$. Now, let us stack these 10 equations into a matrix form

$$\mathtt{M}\mathbf{X} = \mathbf{0}, \tag{3}$$

where $\mathtt{M}$ is a 10×16 coefficient matrix. After performing G-J elimination of the coefficient matrix $\mathtt{M}$, we obtain 10 homogeneous equations of the form

$$p_i \equiv m_i + q_i(f_{23}, f_{33}, \lambda_1, \lambda_2) = 0, i = 1, \ldots, 10, \tag{4}$$

where m_i stands for monomials $f_{11}, f_{12}, f_{21}, f_{22}, \lambda_1 f_{31}, f_{31}, \lambda_1 f_{32}, f_{32}, \lambda_2 f_{13}$ and f_{13}, respectively, and $q_i(f_{23}, f_{33}, \lambda_1, \lambda_2)$ are degree 2 polynomials in four variables $f_{23}, f_{33}, \lambda_1$ and λ_2. Now, let us exploit several dependencies revealed by the G-J elimination. We can easily se that $m_5 = \lambda_1 m_6, m_7 = \lambda_1 m_8$, and $m_9 = \lambda_2 m_{10}$. Using these equalities, we can combine the respective rows p_i to get a system of three homogeneous equations in four unknowns only:

$$-\lambda_1 q_6(f_{23}, f_{33}, \lambda_1, \lambda_2) + q_5(f_{23}, f_{33}, \lambda_1, \lambda_2) = 0, \tag{5}$$

$$-\lambda_1 q_8(f_{23}, f_{33}, \lambda_1, \lambda_2) + q_7(f_{23}, f_{33}, \lambda_1, \lambda_2) = 0, \tag{6}$$

$$-\lambda_2 q_{10}(f_{23}, f_{33}, \lambda_1, \lambda_2) + q_9(f_{23}, f_{33}, \lambda_1, \lambda_2) = 0. \tag{7}$$

Since the fundamental matrix can be determined up to scale only, we can fix one of its entries (and assume it is nonzero at the same time). Here, we will fix $f_{33} = 1$. This is the same constraint that is assumed in the minimal solvers F9 and F9$_{\mathrm{A}}$. This assumption brings the number of unknowns in equations (5)–(7) down to three and, as a system of three equations in three unknown, these equations can be solved using the Gröbner basis method. The resulting Gröbner basis solver needs to compute G-J elimination of a 17×27 matrix and to compute eigenvalues of a 10×10 matrix, yielding the expected number of 10 solutions. Once the unknowns f_{23}, λ_1, and λ_2 are recovered, the rest of the unknowns can be determined by simple back-substitution into equations (4). We will call this variant of the solution the F10$_{u3}$ solver.

Even though the F10$_{u3}$ solver is already quite simple, we can take things even further by eliminating f_{23} and f_{33} from equations (5)–(7) and by solving a smaller system. First, let us rewrite equations (5)–(7) into the following matrix equation

$$\begin{bmatrix} s_{11}^{[3]}(\lambda_1,\lambda_2) & s_{12}^{[2]}(\lambda_1,\lambda_2) \\ s_{21}^{[3]}(\lambda_1,\lambda_2) & s_{22}^{[2]}(\lambda_1,\lambda_2) \\ s_{31}^{[3]}(\lambda_1,\lambda_2) & s_{32}^{[2]}(\lambda_1,\lambda_2) \end{bmatrix} \begin{bmatrix} f_{23} \\ f_{33} \end{bmatrix} = \mathtt{M}_2(\lambda_1,\lambda_2) \begin{bmatrix} f_{23} \\ f_{33} \end{bmatrix} = \mathbf{0}, \quad (8)$$

where the upper index $[\cdot]$ denotes the degree of the respective polynomial $s_{ij}(\lambda_1, \lambda_2)$ in λ_1 and λ_2. Assuming $f_{33} = 1$, we know from elementary algebra that the matrix equation (8) has a non-trivial solution if and only if all 2×2 subdeterminants of the coefficient matrix $\mathtt{M}_2(\lambda_1, \lambda_2)$ are equal to zero. These constraints lead to three degree 5 polynomial equations in two unknowns λ_1 and λ_2:

$$s_{11}^{[3]}(\lambda_1,\lambda_2)s_{22}^{[2]}(\lambda_1,\lambda_2) - s_{21}^{[3]}(\lambda_1,\lambda_2)s_{12}^{[2]}(\lambda_1,\lambda_2) = 0, \quad (9)$$
$$s_{11}^{[3]}(\lambda_1,\lambda_2)s_{32}^{[2]}(\lambda_1,\lambda_2) - s_{31}^{[3]}(\lambda_1,\lambda_2)s_{12}^{[2]}(\lambda_1,\lambda_2) = 0, \quad (10)$$
$$s_{21}^{[3]}(\lambda_1,\lambda_2)s_{32}^{[2]}(\lambda_1,\lambda_2) - s_{31}^{[3]}(\lambda_1,\lambda_2)s_{22}^{[2]}(\lambda_1,\lambda_2) = 0, \quad (11)$$

Since we now have three equations in two unknowns, we can solve any two out of these three polynomial equations (9)–(11), *e.g.*, by using the Sylvester resultant method [7]. However, ignoring the one remaining constraint leads to 13 solutions, which is three more than the correct number of 10 solutions. To get rid of these three parasitic solutions, we need to consider all three equations (9)–(11).

The system of equations (9)–(11) is still quite a simple system that can be solved "by hand", *e.g.*, using the Gröbner basis method. Here, we will consider a Gröbner basis solver created using automatic solver generator of Kukelova *et al.* [14]. This solver performs G-J elimination of a small 10×20 matrix and computes eigenvalues and eigenvectors of a 10×10 matrix. After recovering solutions for λ_1 and λ_2 using the Gröbner basis solver, we can back-substitute them into equation (8) to recover f_{23}. Finally, analogously to the F10$_{u3}$ solution, we back-substitute λ_1, λ_2, and f_{23} into equations (4) to linearly determine the rest of the unknowns. In the next, we will call this solution variant the F10$_e$ solver. Notice that the size of the underlying Gröbner basis solver is equivalent to the size of the Gröbner basis solver for the well known 5-point relative pose problem for perspective calibrated cameras [21].

The most computationally expensive part of the F10$_e$ solver is the eigendecomposition of the 10×10 multiplication matrix. As proposed by Bujnak *et al.* [3], this eigendecomposition can be replaced by computation of the characteristic polynomial of this matrix and by finding the roots of this single variable polynomial, *e.g.*, in λ_1, of degree 10 using the efficient Sturm sequences method [24]. Moreover, since from the physical properties of the camera lenses we know the feasible interval for λ_1, we can naturally speed up the Sturm sequences method convergence and limit the number of solutions by restricting the search to this feasible interval only. Once we have the solutions for λ_1, we can rewrite equations (9)–(11) as the following system

$$\mathtt{M}_3(\lambda_1)[\lambda_2^3, \lambda_2^2, \lambda_2, 1]^\top = \mathbf{0}, \quad (12)$$

where $\mathtt{M}_3(\lambda_1)$ is a 3×4 coefficient matrix. Since $\mathtt{M}_3(\lambda_1)$ depends on the already computed λ_1 only, we can compute λ_2 as the linear solution to this system. We will call this solution variant the F10$_s$ solver.

We can also solve the system of equations (9)–(11) without the use of the Gröbner basis method entirely, by using a similar approach to that used for equation (8); this method is sometimes called the hidden variable trick. Let us consider the first subdeterminant of $\mathtt{M}_2$ from equation (8), *i.e.*, the subdeterminant corresponding to equations (5) and (6). When considered as a polynomial in λ_2—after "hiding" λ_1 in the coefficient field—this determinant is a quadratic polynomial. Let us multiply this polynomial by λ_2 and add it to equations (9)–(11). We get a system of four equations that can be rewritten as

$$\begin{bmatrix} 0 & r_{12}^{[3]}(\lambda_1) & r_{13}^{[3]}(\lambda_1) & r_{14}^{[3]}(\lambda_1) \\ r_{12}^{[3]}(\lambda_1) & r_{13}^{[3]}(\lambda_1) & r_{14}^{[3]}(\lambda_1) & 0 \\ r_{31}^{[2]}(\lambda_1) & r_{32}^{[2]}(\lambda_1) & r_{33}^{[2]}(\lambda_1) & r_{34}^{[2]}(\lambda_1) \\ r_{41}^{[2]}(\lambda_1) & r_{42}^{[2]}(\lambda_1) & r_{43}^{[2]}(\lambda_1) & r_{44}^{[2]}(\lambda_1) \end{bmatrix} \begin{bmatrix} \lambda_2^3 \\ \lambda_2^2 \\ \lambda_2 \\ 1 \end{bmatrix} = \mathtt{M}_4(\lambda_1) \begin{bmatrix} \lambda_2^3 \\ \lambda_2^2 \\ \lambda_2 \\ 1 \end{bmatrix} = \mathbf{0}, \quad (13)$$

where $r_{ij}^{[k]}(\lambda_1)$ are degree k polynomials in λ_1. Again, from the elementary algebra we know that the matrix equation (13) has a non-trivial solution if and only if the determinant of its 4×4 polynomial coefficient matrix $\mathtt{M}_4$ is equal to zero. This directly leads to a degree 10 polynomial equation in λ_1. Its solutions can be once again found using the Sturm sequences method. Unfortunately, a naive symbolic expansion of $\det(\mathtt{M}_4(\lambda_1))$ and subsequent coefficient extraction lead to a prohibitively large polynomials. However, we have found that if the determinant extraction is properly implemented, this final solver, F10, outperforms all previously presented variants.

4. Experiments

To precisely gauge the stability, precision, and speed of the proposed solvers, we performed extensive tests on synthetic as well as real world data with various amounts of radial distortion, noise, and outlier contamination. We compared the proposed solvers to a comprehensive set of related algorithms. See table 3 for an explanatory list of solvers considered in the experiments.

In our experiments, we observed that the proposed variants $F10_e$, $F10_s$, and F10 have virtually equivalent numerical stability and noise resistance. To make the figures more readable, we have decided not show these equivalent results and to represent this family of solvers by the most time-efficient solver only, the F10 solver.

4.1. Synthetic data

First, we studied the performance of the proposed solvers on synthetically generated 3D scenes with known ground-truth parameters. The scenes consisted of 3D points distributed at random in a 3D cube $[-10, 10]^3$. Each 3D point was projected by two cameras with random, yet still realistic, focal lengths. The orientations and positions of the cameras were selected at random so as to look at the scene from a random distance, varying from 15 to 35 from the center of the scene. Next, the image projections in both cameras were corrupted by different amounts of radial distortion, following the one-parameter division model [9]. Finally, the radially distorted image projections were corrupted by Gaussian noise with standard deviation σ, assuming a 1000×1000 px image.

Numerical stability. In the numerical stability experiment, we studied the behavior of the proposed solvers on noise-free data and compared the results to the numerical stability of the relevant $F\lambda_1\lambda_2$ competitors F9 [5], $F9_A$ [14], F12 [5], and F15 [2]. In this experiment, we also included the $F10_{u3}$ solver.

We generated 10,000 scenes with 3D points distributed in a cube and cameras with random feasible poses. The radial distortion parameters λ_1 and λ_2 were drawn at random from the interval $[-0.8, 0]$ to investigate the behaviour of the algorithms for large as well as small amounts of radial distortion.

Figures 2(a–b) show the results of the new F10 solver (blue) compared to the relevant algorithms. The $\log_{10}$ of the relative error of the radial distortion parameter λ_1, figure 2(a), and λ_2, figure 2(b), were obtained by selecting the real root closest to the ground truth values. The results suggest that the numerical stability of the F10 solver

F9 [5]	Minimal 9pt solver for $F\lambda_1\lambda_2$ based on the Gröbner basis method and LU and SVD decompositions
$F9_A$ [14]	Minimal 9pt solver for $F\lambda_1\lambda_2$ generated using the automatic generator of Gröbner basis solvers
F12 [5]	12pt solver for $F\lambda_1\lambda_2$ based on the polynomial eigenvalue method
F15 [2]	Linear 15pt solver for $F\lambda_1\lambda_2$
$F8_\lambda$ [14]	Minimal 8pt solver for the $F\lambda$ problem based on the Gröbner basis method
F7 [10]	Standard 7pt algorithm for perspective cameras with no radial distortion
$F10_{u3}$	New 10pt solver for $F\lambda_1\lambda_2$ based on eqs. (5)–(7) and the Gröbner basis method
$F10_e$	New 10pt solver for $F\lambda_1\lambda_2$ based on eqs. (9)–(11) and the Gröbner basis method combined with eigendecomposition
$F10_s$	New 10pt solver for $F\lambda_1\lambda_2$ based on eqs. (9)–(11) and the Gröbner basis method combined with Danilevskii and Sturm sequences methods [3]
F10	New 10pt solver for $F\lambda_1\lambda_2$ based on the polynomial determinant equation (13)

Figure 3: **List of solvers** considered in the experiments.

(blue) is comparable to the numerical stability of the linear F15 solver (orange) and the F12 solver (green). The minimal solvers F9 (red) and $F9_A$ (magenta) have significantly worse numerical stability than the rest of the competition. This is caused by the fact that both F9 and $F9_A$ need to perform LU decompositions/G-J eliminations of quite large matrices. Moreover, since the $F9_A$ solver doesn't include any methods for numerical stability improvement as the F9 solver does, it also shows many failures. This is caused by the ill-conditioned matrix used in the G-J elimination step.

Note that the numerical precision of the F10 solver is directly proportional to the relative error settings of the Sturm sequence method. In this experiment, we set this relative error to 10^{-13}. Also, notice that the $F10_{u3}$ solver exhibits slightly worse numerical stability than the F10 solver. This suggests that simplifying equations (5)–(7) by eliminating f_{23} brings not only the advantage of a smaller and faster solver (G-J elimination of a 10×20 matrix vs. G-J elimination of a 17×27 matrix) but also improves the numerical stability of the solver.

Noise experiment. In this experiment, we studied the performance of the F10 solver in the presence of image noise.

Figures 2(c–d) show the estimated radial distortion parameters for cameras with different radial distortions. In this case, the ground truth radial distortion parameters were set to $\lambda_{1_{gt}} = -0.2$ and $\lambda_{2_{gt}} = -0.4$. We compared the proposed F10 solver to the same set of $F\lambda_1\lambda_2$ solvers as in the stability experiment, with the exception of the $F10_{u3}$ solver, which solves a problem algebraically equivalent to the problem F10 solves, albeit less efficiently. Results in figures 2(c–d) are depicted using MATLAB function `boxplot` which shows 25% to 75% quantile values as

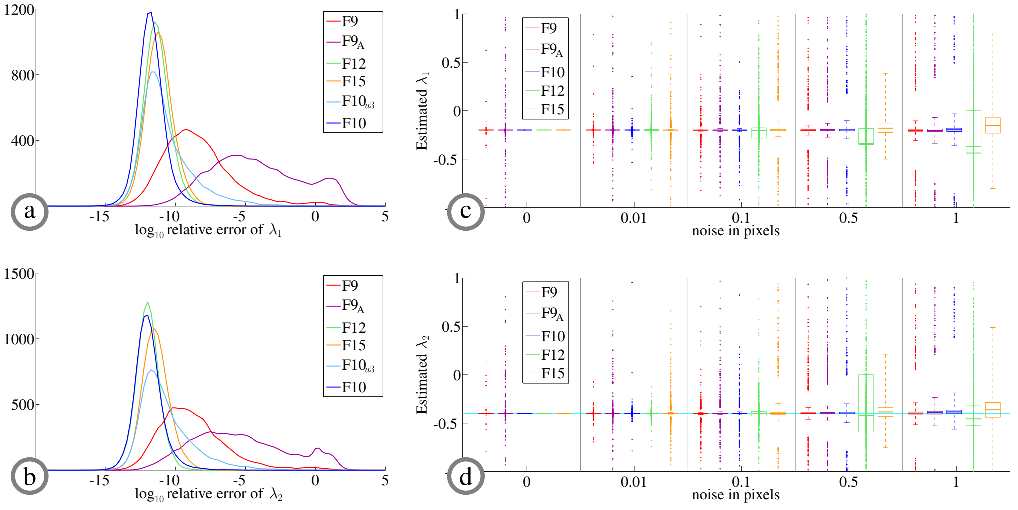

Figure 2: **Synthetic experiments.** (a–b) Stability experiment: histograms of $\log_{10}$ relative errors of the estimated λ_1 and λ_2. (c–d) Noise experiment: comparison of λ_1 and λ_2 estimated by different F$\lambda_1\lambda_2$ solvers for varying levels of noise. The ground truth values of radial distortion were set to $\lambda_{1_{gt}} = -0.2$ and $\lambda_{2_{gt}} = -0.4$.

boxes with a horizontal line for median. The crosses show data beyond 1.5 times the inter-quartile range.

The performance of the F10 solver (blue) is quite similar to the performance of the minimal solvers F9 (red) and F9$_A$ (magenta) in the presence of noise. The remaining two non-minimal solvers F12 (green) and F15 (orange) are very sensitive to noise and thus not very useful for any practical application. This noise sensitivity is most probably caused by the fact that in these solvers, both the entries of the fundamental matrix and the radial distortion parameters can be computed from different elements of null space/eigen- vectors, *i.e.*, there is more than one way to compute the unknowns. However, since neither F12 nor F15 consider the monomial dependencies among the entries of these vectors, they may produce completely different results, depending on the entries of null space/eigen- vectors used. Moreover, F12 and F15 solvers do not minimize any meaningful error.

Planar scene. Scene planarity remains to be an issue for the F$\lambda_1\lambda_2$ solvers. All considered F$\lambda_1\lambda_2$ solvers, F9, F9$_A$, F10, F12 and F15, fail to recover the correct geometry for planar scenes. However, they are able to recover the radial distortion parameters with satisfactory precision, see figure 4. A notable exception is the noiseless case, where F9 and F9$_A$ completely fail, whereas F10, F12 and F15 are still able to recover the correct radial distortion parameters. To simplify figure 4, we have decided not show the results of the F12 and F15 solvers. For planar scenes, as in the case of non-planar scenes depicted in figure 2, the F12 and F15 solvers completely fail for any reasonable amount on noise.

Even though the behavior of the F10 solver for planar scenes is problematic from the theoretical point of view, we believe that it is of little practical consequence. Indeed, if the planarity of a scene is to be detected, one can use λ_1 and λ_2 recovered by F10 to undistort the correspondences and proceed with homography estimation. If, on the other hand, the scene planarity is known *a priori*, one could opt for homography estimation directly.

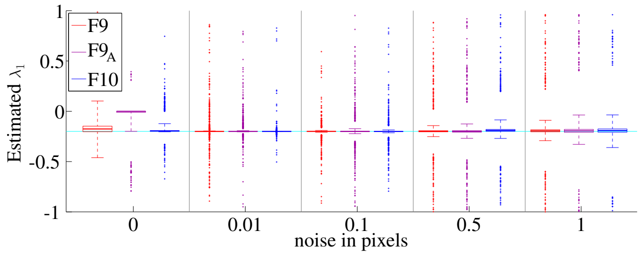

Figure 4: **Planar scene.** Comparison of λ_1 estimated by different F$\lambda_1\lambda_2$ solvers for a planar scene and varying levels of noise. The ground truth value of radial distortion were set to $\lambda_{1_{gt}} = -0.2$ and $\lambda_{2_{gt}} = -0.4$.

Feasible solutions. From the physical properties of camera lenses, we can quite accurately infer the feasible interval for the values of λ_1 and λ_2. This fact is important for RANSAC-style algorithms, where one can quickly discard models that do not fall into this interval, without the need to perform the inlier test. In this experiments, we considered scenes analogous to those in the stability experiment. We scaled the camera measurements to the interval $[-1, 1]^2$ and set the feasible interval to $\lambda_{1,2} \in [-10, 2]$, which covers a large range of physically plausible distortions. The average number of feasible solutions attained over 10K experiments was 1.99 (F9), 3.29 (F9$_A$) and 1.54 (F10).

4.2. Real data

Radial distortion estimation. In this experiment, we show that in some cases it is indeed necessary to consider radial distortion already while searching for the initial geometry model in RANSAC. Figures 2(a–b) show a stereo pair of a city scene taken by a GoPro Hero4 camera from two different positions and with different field of view (FOV) settings. The different FOV settings (medium and wide, respectively) effectively translate into different amounts of radial distortion. For the purposes of this experiment, we rescaled the images to 1900×1425 px. First, we

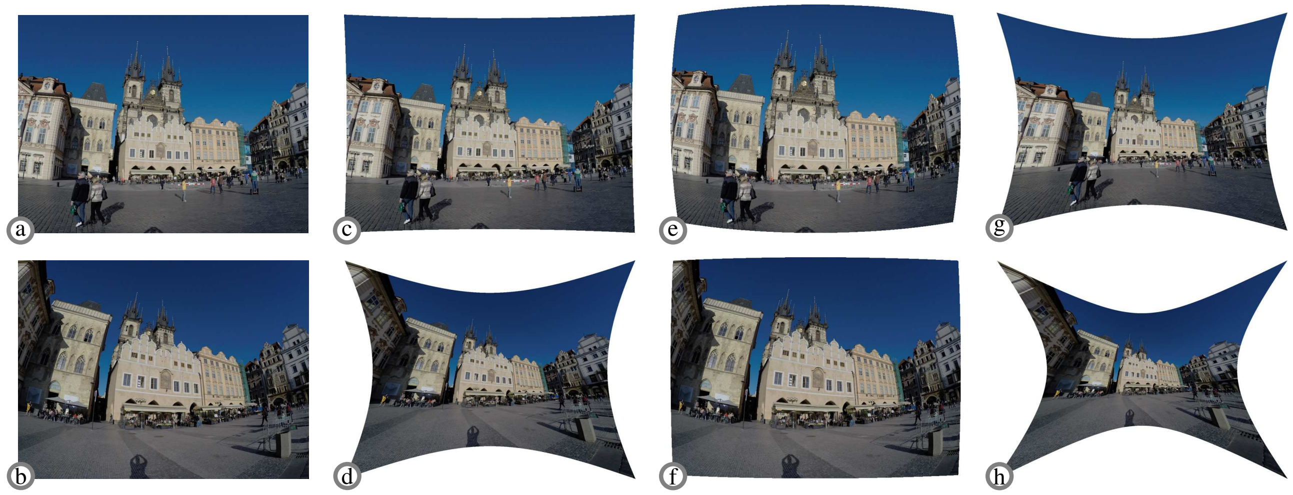

Figure 5: **Radial distortion estimation.** (a–b) A stereo pair of a city scene taken by a GoPro Hero4 camera with different FOV settings. The pair undistorted using λ_1, λ_2 obtained by nonlinear optimization of a model initialized by (c–d) F7, (e–f) $\mathrm{F8}_\lambda$, and (g–h) F10.

detected SIFT features [19] and found 1005 tentative correspondences covering significant area of the images. Next, we run 300 RANSAC cycles with the threshold set to 3 px for three algorithms F7 [10], $\mathrm{F8}_\lambda$ [14], and F10 to recover the fundamental matrix $\mathtt{F}$ and, where applicable, the radial distortion parameters λ_1, λ_2. From the set of 1005 tentative correspondences, F7, $\mathrm{F8}_\lambda$, and F10 recovered models with 50.1%, 53% and 79% inliers, respectively. Finally, for all three models we performed joint nonlinear estimation of $\mathtt{F}$, λ_1, and λ_2. For the F7 model, we initialized $\lambda_1 = \lambda_2 = 0$ and for $\mathrm{F8}_\lambda$ as $\lambda_1 = \lambda_2 = \lambda$. Figures 2(c–h) show the initial image pair undistorted using the different amounts or radial distortion recovered in the nonlinear optimization step for F7 (c–d), $\mathrm{F8}_\lambda$ (e–f), and F10 (g–h). As we can see, the only solver able to recover the proper radial distortion parameters was in this case the F10 solver.

RANSAC. To better understand the behaviour of the new F10 solver in RANSAC, we once again used the city scene stereo pair from Figures 2(a–b). Figure 6 shows the mean number of gained inliers by different solvers as a function of RANSAC cycles over 300 runs for 20% outlier contamination. Again, the RANSAC threshold was set to 3 px. First, we can notice a remarkably bad performance of the F12 and F15 solvers. This is partly by the high noise sensitivity of these solvers and partly by the high number of involved correspondences, which lowers the probability of selecting a true inlier "minimal" set. Next, can see that even though the $\mathrm{F8}_\lambda$ solver is able to recover one radial distortion parameter, the fact that this particular image pair was taken with two quite different distortion settings, the solver performs almost on par with the distortion oblivious F7 solver. Finally, we can see the new F10 slightly outperformed by the F9 and $\mathrm{F9}_\mathrm{A}$ solvers, as F10 needs one more point for

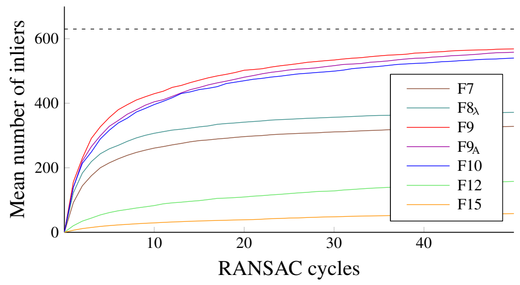

Figure 6: **RANSAC.** The mean number of gained inliers by different solvers as a function of RANSAC cycles over 300 runs for 20% outlier contamination. The dotted line represents the number of true inliers.

model recovery. In practice however, this loss is offset by F10's significant speed gain.

3D Reconstruction Finally, to test the suitability of the F10 solver for practical applications, we deployed it as a part of a larger SfM pipeline. We experimented with a dataset of 66 images of a rotunda taken by a GoPro Hero4 camera, again, combining medium and wide FOV settings. Figure 1 shows an overview of the final reconstruction of the dataset. Even though no prior information about the camera focal length or radial distortion was used, the F10 solver helped to recover the correct scene geometry as well as radial distortion, see Figure 7 for details.

4.3. Speedup

As we have seen in section §4.1, the performance of the solvers F12 and F15 in the presence of noise makes them quite impractical. This is why in this experiment we compared the F10 family of solvers to the F9 and $\mathrm{F9}_\mathrm{A}$ solvers

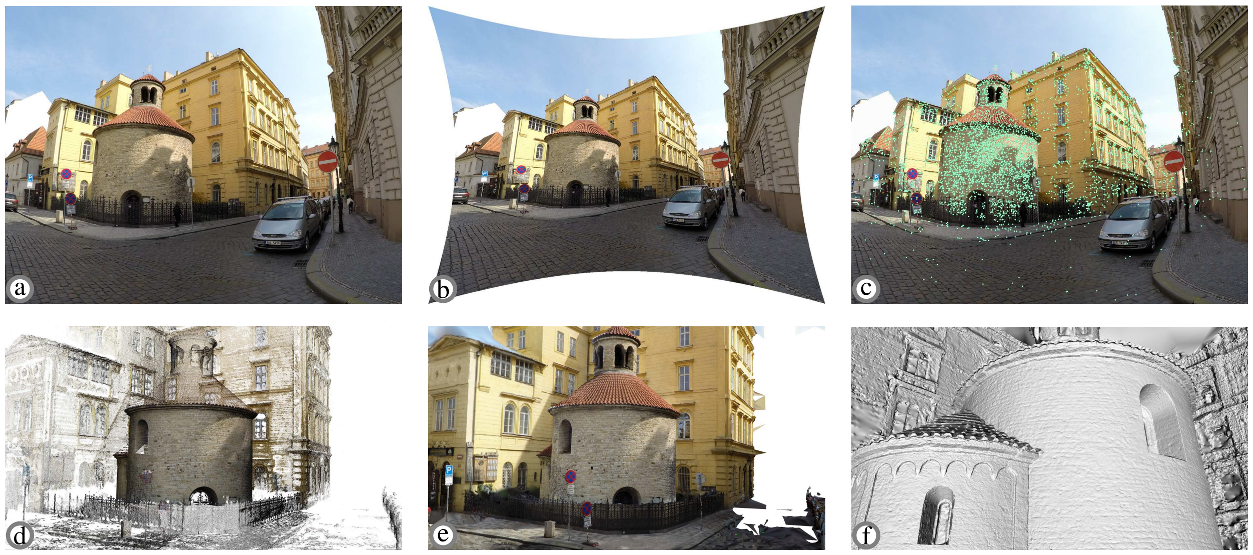

Figure 7: **3D reconstruction.**(a) An input image from a rotunda dataset. (b) The image undistorted using λ_1 obtained by nonlinear optimization of a model initialized by F10. (c) Inlier set of F10 as recovered by RANSAC. (d) 3D point cloud of the final reconstruction. (e) Textured 3D mesh. (f) Mesh detail.

only. Unfortunately, we have C++ implementations of neither F9 nor F9$_\text{A}$, thus we compared only times of the major computationally involved steps performed by each solver. Here, we used the C++ implementations of these steps based on linear algebra library Eigen [1]. The timings, averaged over 10K trials on a 3.5 GHz i7 based desktop, are reported in the following table:

	SVD	G-J	LU	Eig	λ_1-poly	Total	Speedup
F9	(69×69) 2606.1	(9×16) 1.0	(393×389) 2617.3	(24×24) 112.1	–	5336.5	1.0
F9$_\text{A}$	–	(9×16)+ (179×203) 930.7	–	(24×24) 121.1	–	1042.8	**5.1**
F10$_e$	–	(10×16)+ (10×20) 1.76	–	(10×10) 15.34	–	17.57	**303.7**
F10$_s$	–	(10×16)+ (10×20) 1.76	–	–	3.42	5.89	**906.0**
F10	–	(10×16) 1.13	–	–	2.46	4.34	**1229.6**

All timings are reported in μs and the speedups are measured with respect to F9. The bracketed dimensions represent the sizes of the respective input matrices. The column labeled "λ_1-poly" represents the times spent by the extraction of the univariate polynomials in λ_1 (from the multiplication matrix of the Gröbner basis solver in case of F10$_s$ or from the polynomial matrix from equation (13) in case of F10) followed by the root evaluation using the Sturm sequences method. Notice that the times of F10 solvers do not sum up to the values in column "Total". In this case, the values under "Total" represent time taken by *all* the operations performed by the solvers, including fillings of matrices, back-substitutions, and other overhead.

5. Conclusion

Epipolar geometry estimation in the presence of radial distortion is a challenging problem, as attested by the veritable zoo of solvers proposed in the past. In this work, we presented a new solution to the most general variation of this problem: two uncalibrated cameras with different amounts of radial distortion. Our algorithm uses a small number of 10 point correspondences and it is both numerically stable and noise-resistant, while still being fast for practical or even real-time applications. For the sake of completeness, we have exploited several possible solution strategies in the course of derivation of the algorithm. We have arrived to four solvers: F10$_{u3}$, F10$_e$, F10$_s$, and F10. As we have observed in our experiments, the last three variants, F10$_e$, F10$_s$, and F10 are virtually equivalent from the noise resistance and numerical stability point of view. Based on our implementation of these solvers in C++, we have concluded F10 to be the fastest. Yet, the experimental timings strongly suggest that all of the proposed solvers are fast enough to be used even in real-time applications, by far outperforming the previously proposed algorithms. It is thus up to the user to decide whether to opt for the "simplest" variant F10$_e$, or whether the application at hand merits a bit more involved implementation of the F10 solver.

Acknowledgment

The authors were supported by EC project FP7-SPACE-2012-312377 PRoViDE.

References

[1] Eigen: C++ template library for linear algebra. `eigen.tuxfamily.org`. 8

[2] J. Barreto and K. Daniilidis. Fundamental matrix for cameras with radial distortion. In *Tenth IEEE International Conference on Computer Vision*, volume 1, pages 625–632 Vol. 1, Oct 2005. 2, 3, 5

[3] M. Bujnak, Z. Kukelova, and T. Pajdla. Making minimal solvers fast. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1506–1513, June 2012. 4, 5

[4] M. Byröd, M. Brown, and K. Åström. Minimal solutions for panoramic stitching with radial distortion. In *BMVC'09*, 2009. 2

[5] M. Byröd, Z. Kukelova, K. Josephson, T. Pajdla, and K. Åström. Fast and robust numerical solutions to minimal problems for cameras with radial distortion. In *CVPR'08*, 2008. 2, 3, 5

[6] O. Chum, J. Matas, and J. Kittler. Locally optimized ransac. In *Pattern Recognition*, pages 236–243. Springer Berlin Heidelberg, 2003. 1, 2

[7] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Springer, 2nd edition, 2005. 2, 4

[8] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981. 1, 2

[9] A. W. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. In *CVPR'01*, volume 1, page 125, Los Alamitos, CA, USA, 2001. IEEE Computer Society. 1, 2, 3, 5

[10] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004. 1, 3, 5, 7

[11] F. Jiang, Y. Kuang, J. E. Solem, and K. Åström. A minimal solution to relative pose with unknown focal length and radial distortion. In *The 12th Asian Conference on Computer Vision*, 2014. 2

[12] H. Jin. A three-point minimal solution for panoramic stitching with lens distortion. In *CVPR'08*, 2008. 2

[13] Y. Kuang, J. E. Solem, F. Kahl, and K. Åström. Minimal solvers for relative pose with a single unknown radial distortion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 2

[14] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. In *ECCV'08, Part III*, volume 5304 of *Lecture Notes in Computer Science*, 2008. 2, 3, 4, 5, 7

[15] Z. Kukelova, M. Bujnak, and T. Pajdla. Real-time solution to the absolute pose problem with unknown radial distortion and focal length. In *2013 IEEE International Conference on Computer Vision (ICCV 2013)*, pages 2816–2823, Los Alamitos, US, December 2013. IEEE Computer Society, IEEE Computer Society Press. 2, 3

[16] Z. Kukelova, M. Byröd, K. Josephson, T. Pajdla, and K. Åström. Fast and robust numerical solutions to minimal problems for cameras with radial distortion. *Computer Vision and Image Understanding*, 114(2):234–244, February 2010. 2, 3

[17] Z. Kukelova and T. Pajdla. A minimal solution to the autocalibration of radial distortion. In *Proceedings of the Computer Vision and Pattern Recognition conference (CVPR)*, page 7, Los Alamitos, USA, June 2007. IEEE Computer Society, IEEE Computer Society Press. 2

[18] Z. Kukelova and T. Pajdla. Two minimal problems for cameras with radial distortion. In *OMNIVIS 2007: Proceedings of The Seventh Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras*, page 8, Los Alamitos, USA, October 2007. IEEE Computer Society, IEEE Computer Society Press. 2

[19] D. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150–1157 vol.2, 1999. 7

[20] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(6):756–777, June 2004. 1

[21] H. Stewénius, C. Engels, and D. Nistér. Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60:284–294, June 2006. 1, 4

[22] H. Stewénius, F. Kahl, D. Nistér, and F. Schaffalitzky. A minimal solution for relative pose with unknown focal length. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 789–794, San-Diego, USA, June 2005. Chapter 8 of my thesis. 1

[23] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, pages 298–372. Springer-Verlag, 2000. 1, 2

[24] C.-K. Yap. *Fundamental problems of algorithmic algebra*, volume 49. Oxford University Press Oxford, 2000. 4