A Minimal Solution to Radial Distortion Autocalibration

Zuzana Kukelova, *Member*, *IEEE*, and Tomas Pajdla, *Member*, *IEEE*

Abstract—Simultaneous estimation of radial distortion, epipolar geometry, and relative camera pose can be formulated as a minimal problem and solved from a minimal number of image points. Finding the solution to this problem leads to solving a system of algebraic equations. In this paper, we provide two different solutions to the problem of estimating radial distortion and epipolar geometry from eight point correspondences in two images. Unlike previous algorithms which were able to solve the problem from nine correspondences only, we enforce the determinant of the fundamental matrix be zero. This leads to a system of eight quadratic and one cubic equation in nine variables. We first simplify this system by eliminating six of these variables and then solve the system by two alternative techniques. The first one is based on the Gröbner basis method and the second one on the polynomial eigenvalue computation. We demonstrate that our solutions are efficient, robust, and practical by experiments on synthetic and real data.

Index Terms—Minimal problems, radial distortion, Gröbner bases, polynomial eigenvalue problems.

1 Introduction

Estimating camera motion and internal calibration parameters from sequences of images is an important problem with a broad range of applications [22]. It is one of the oldest computer vision problems and even though much has already been solved, some questions still remain open. For instance, a number of techniques for modeling and estimating projection models of wide angle lenses [16], [36], [21], [50], [51], [20] appeared recently. Often, in this case, the projection is modeled as the perspective projection followed by radial "distortion" in the image plane. In fact, not only wide angle lenses but almost all consumer camera lenses suffer from some radial distortion.

It was observed that ignoring the distortion, even for standard cameras, may lead to significant errors in 3D reconstruction [16], metric measurements from images, or when calibrating the cameras.

Many methods for autocalibration of radial distortion appeared recently. These methods estimate radial distortion parameters either from images of straight lines in the real world [4], [13], [25], [48], known calibration patterns [54], [56], [3], or from 2D image correspondences [2], [16], [33], [42], [49], [55], [36]. These 2D-2D methods are especially popular because they don't require knowledge of any scene structure a priori and can be smartly used in many applications like 3D reconstruction, structure from motion, and image registration. Most of these methods use iterative techniques such as Levenberg-Marquardt to solve for radial distortion parameters [14], [41], [42], [55]. Therefore, they have classical problems of iterative nonlinear optimization algorithms, i.e., the quality of the result is primarily determined by the point from which the optimization starts. Another problem of existing methods for estimating radial distortion parameters from image point correspondences is that they mostly require too many image correspondences, 9 in [16], [33], 15 in [2], 36 in [11], or even more. In most of these cases, the number of correspondences used is much higher than the minimal number of point correspondences needed to solve the particular problem.

Using a higher number of points may be problematic in some applications where we don't have enough point correspondences, the correspondences are affected by noise, or they contain mismatches. Moreover, using a smaller number of correspondences considerably reduces the number of samples and, therefore, also running time in RANSAC [15] estimation, which is often used in applications like 3D reconstruction and structure from motion [22], [52].

Whereas for many problems of estimating epipolar geometry of perspective cameras there exist minimal solutions [37], [45], [46] (solutions from a minimal number of point correspondence), for similar problems of estimating epipolar geometry and radial distortion parameters for cameras with radial distortion such solutions were missing until now. This was caused by the fact that these "minimal radial distortion problems" result in more difficult systems of polynomial equations. Therefore, the existing methods mainly focused on proposing new models for cameras with radial distortion and on choosing simpler systems which could be solved more easily from a higher number of correspondences.

In [16], Fitzgibbon proposed a nonminimal solution to the problem of simultaneous estimation of the epipolar geometry and one-parameter division model from point correspondences. The proposed division model leads to solving a system of algebraic equations, which is especially nice because the algebraic constraints of the epipolar geometry, $\det(\mathbf{F}) = 0$ [22], can be "naturally" added to the constraints

• *The authors are with the Center for Machine Perception, Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Karlovo namesti 13 121-35 Praha 2, Czech Republic. E-mail: {kukelova, pajdla}@cmp.felk.cvut.cz.*

Manuscript received 28 Jan. 2010; revised 22 Jan. 2011; accepted 17 Mar. 2011; published online 27 Apr. 2011.
Recommended for acceptance by A. Fitzgibbon.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2010-01-0066.
Digital Object Identifier no. 10.1109/TPAMI.2011.86.

arising from correspondences to reduce the number of points needed for estimating the distortion and the fundamental matrix. However, the algebraic constraints on the fundamental matrix haven't been used in [16] and therefore nine point correspondences were necessary. Thanks to neglecting this constraint, the resulting equations could be easily formulated as a quadratic eigenvalue problem (QEP) and solved using standard solvers. Micusik and Pajdla [36] also neglected the constraints when formulating the estimation of paracatadioptric camera model from image matches as a quartic eigenvalue problem. The work [40] extended Fitzgibbon's method for any number of views and any number of point correspondences using generalized quadratic eigenvalue problem for rectangular matrices, again without using the constraints on $\mathtt{F}$.

Li and Hartley [33] treated the original Fitzgibbon's problem as a system of algebraic equations and used the hidden variable technique [12] to solve them. No algebraic constraint on the fundamental matrix has been used. The resulting technique solves exactly the same problem as [16] but in a different way. Our experiments have shown that the quality of the result in [33] was comparable to [16], but the technique [33] was considerably slower than the original technique [16] because, in [33], Maple was used to evaluate large polynomial determinants. Work [33] mentioned the possibility of using the algebraic constraint $\det(\mathtt{F}) = 0$ to solve for a two parametric model from the same number of points, but it did not use it to really solve the problem.

The first solutions to the "minimal radial distortion problems" were proposed only recently. In [27], a solution to the problem of estimating epipolar geometry and radial distortion appeared for uncalibrated camera and, in [28], [9], and [31], for calibrated camera and two cameras with different radial distortions in each camera.

The problem of estimating epipolar geometry and one-parameter radial distortion for two uncalibrated cameras with different radial distortions presented in papers [28], [9], [31] is, in fact, a generalization of the problem for one uncalibrated camera with constant radial distortion considered in [27] and also in this paper. However, the problem for two different cameras results in a more complicated solver and in more solutions (24 versus 16) and requires one more point correspondence for computation. Therefore, when we know that both our images were taken with the same camera, it is better to incorporate this constraint and use a more efficient solver for a camera with constant radial distortion. This will reduce the minimal number of points necessary to solve the problem and in this way help to handle situations with a larger portion of mismatches. Moreover, a solver using this constant radial distortion constraint will bring more stable results than the solver for cameras with different radial distortions.

In this paper, we solve the minimal problem of simultaneous estimation of epipolar geometry and one-parameter radial distortion model from point correspondences for uncalibrated camera with a constant radial distortion. This problem is, among these three minimal problems, the most practical and thus deserves its own exposition.

Our solution is based on the one-parameter division model introduced by Fitzgibbon [16]. This model is able to handle almost all consumer camera lenses and fish-eye lenses up to 140 degree field of view and behaves as well as or even better than the standard one-parameter polynomial model.

We formulate the problem of estimating the radial distortion from image matches as a system of algebraic equations and, by using the constraint $\det(\mathtt{F}) = 0$, we get a minimal solution to the autocalibration of radial distortion from eight point correspondences in two views. This brings three benefits. The main benefit from using this singularity constraint is in reducing the minimal number of point correspondences necessary to compute radial distortion and camera pose with precision sufficient to reject mismatches. Using the constraint reduces the number of samples in RANSAC and therefore helps to solve situations with a larger portion of mismatches and difficult situations where only a small number of features was detected. Once the wrong data are rejected, least-squares methods on all remaining data can be used to get higher accuracy. The second benefit is that we directly obtain a valid fundamental matrix. Finally, the solver is more stable than previously known nonminimal 9-point algorithms [16], [33].

This minimal problem of estimating a fundamental matrix and one-parameter division model from eight point correspondences was first solved in [27], where a solution based on the Gröbner basis method leading to three Gauss-Jordan (G-J) eliminations has been suggested. Here, we present a new solution to this minimal problem based on the recent developments in Gröbner basis computation [8], [30]. This solution results in only one G-J elimination, which was observed to be numerically more stable than several G-J eliminations in this case. Moreover, we provide a new solution to this problem based on a quartic eigenvalue problem which is analogical to the Fitzgibbon's 9-point solution [16]. This paper extends the work in [27] and demonstrates that the new solutions are better than previous solutions in simulated and real experiments.

Our work adds a new minimal problem solution to the family of previously developed minimal problems [15], [18], [37], [45], [34], [29], [46], [32], [47], [19], [5], [27], [28], [9], [10], [23], [7], [38]. All these problems result in nontrivial systems of polynomial equations. Solving systems of polynomial equations is a challenging problem and there exist no practical, robust, and numerically stable algorithms in general. Instead, special purpose algorithms need to be developed for specific applications. The state-of-the-art method for solving systems of polynomial equations is the Gröbner basis method [12], which was used to solve almost all of the previously mentioned minimal problems. Another interesting method for solving systems of equations is the polynomial eigenvalue method based on multipolynomial resultants recently used in [29].

Next, we provide the problem formulation and compare the two approaches [12], [29] to its solution.

In this paper, we use the notation and definitions from [12], where the necessary concepts from polynomial algebra and algebraic geometry are explained.

2 Problem Formulation

We want to correct radial lens distortion and estimate the epipolar geometry for uncalibrated camera using the

minimal number of image point correspondences in two views. We assume the one-parameter division distortion model [16], which is given by the formula

$$\mathbf{p}_u \sim \mathbf{p}_d / \left(1 + \lambda r_d^2\right), \tag{1}$$

where λ is the distortion parameter, $\mathbf{p}_u = (x_u, y_u, 1)^\top$, respectively, $\mathbf{p}_d = (x_d, y_d, 1)^\top$, are the corresponding undistorted, respectively, distorted, image points, and r_d is the radius of $\mathbf{p}_d$ w.r.t. the distortion center. We assume that the distortion center is in the center of the image and that pixels are square, i.e., $r_d^2 = x_d^2 + y_d^2$. We show in experiments that these assumptions work well even when they are not completely satisfied. This was also shown in [16].

Undistorted image points $\mathbf{p}_u$ and $\mathbf{p}'_u$, which are projections of 3D point $\mathbf{P}$, are geometrically constrained by the epipolar geometry constraint

$$\mathbf{p}'^\top_u \mathtt{F}\, \mathbf{p}_u = 0, \tag{2}$$

where $\mathtt{F}$ is a 3×3 rank-2 fundamental matrix

$$\mathtt{F} = \begin{pmatrix} f_{1,1} & f_{1,2} & f_{1,3} \\ f_{2,1} & f_{2,2} & f_{2,3} \\ f_{3,1} & f_{3,2} & f_{3,3} \end{pmatrix}. \tag{3}$$

It is well known that for the standard uncalibrated case without considering radial distortion, seven point correspondences are sufficient and necessary to estimate the epipolar geometry. We have one more parameter, the radial distortion parameter λ. Therefore, we will need eight point correspondences to estimate λ and the epipolar geometry simultaneously. To get this "8-point algorithm," we have to use the singularity of the fundamental matrix $\mathtt{F}$. We obtain nine equations in 10 unknowns by taking equations from the epipolar constraint for eight point correspondences:

$$\mathbf{p}^\top_{u_i}(\lambda) \mathtt{F}\, \mathbf{p}'_{u_i}(\lambda) = 0, \quad i = 1, \ldots, 8, \tag{4}$$

and the singularity of $\mathtt{F}$:

$$\det(\mathtt{F}) = 0, \tag{5}$$

where $\mathbf{p}'_u(\lambda), \mathbf{p}_u(\lambda)$ represent homogeneous coordinates of a pair of undistorted image points.

These are the basic polynomial equations defining our problem. The complexity of solving polynomial equations depends mostly on the complexity of polynomials (degree, number of variables, form, etc.). It is usually better to have the degrees, as well as the number of variables, low. Therefore, in the first step, we simplify the original set of equations by eliminating some variables. Unfortunately, reducing the number of variables often leads to higher degrees.

Next, we present an elimination scheme which, for this problem, results in a suitable system of three polynomial equations in three variables. The presented methods for solving systems of polynomial equations would result in huge and unusable solvers without this elimination.

2.1 Eliminating Variables—Three Equations in Three Unknowns

The elimination procedure starts with the original set of equations (4) and (5). Assuming, e.g., $f_{3,3} \neq 0$, we can set $f_{3,3} = 1$. Then, the epipolar constraint (4) gives eight equations with 15 monomials $(f_{1,3}\lambda, f_{2,3}\lambda, f_{3,1}\lambda, f_{3,2}\lambda, \lambda^2, f_{1,1}, f_{1,2}, f_{1,3}, f_{2,1}, f_{2,2}, f_{2,3}, f_{3,1}, f_{3,2}, \lambda, 1)$ and nine variables $(f_{1,1}, f_{1,2}, f_{1,3}, f_{2,1}, f_{2,2}, f_{2,3}, f_{3,1}, f_{3,2}, \lambda)$.

Among these nine variables, we have four variables which appear in one monomial only $(f_{1,1}, f_{1,2}, f_{2,1}, f_{2,2})$ and four variables which appear in two monomials $(f_{1,3}, f_{2,3}, f_{3,1}, f_{3,2})$. Since we have eight equations from which each one contains all 15 monomials, we can eliminate six variables, four variables which appear in one monomial only and two variables from the variables which appear in two monomials. We have selected $f_{1,3}$ and $f_{2,3}$.

We reorder monomials contained in eight equations such that monomials containing $f_{1,1}$, $f_{1,2}$, $f_{2,1}$, $f_{2,2}$, $f_{1,3}$, and $f_{2,3}$ are at the beginning. The reordered monomial vector will be $X = (f_{1,1}, f_{1,2}, f_{2,1}, f_{2,2}, f_{1,3}\lambda, f_{1,3}, f_{2,3}\lambda, f_{2,3}, f_{3,1}\lambda, f_{3,2}\lambda, \lambda^2, f_{3,1}, f_{3,2}, \lambda, 1)^\top$.

Then, eight equations from the epipolar constraint can be written in a matrix form

$$\mathtt{M}\, X = 0, \tag{6}$$

where $\mathtt{M}$ is the coefficient matrix and X is this reordered monomial vector. After computing G-J elimination $\mathtt{M}'$ of $\mathtt{M}$, we obtain eight equations $\mathtt{M}'\, X = 0$ of the form

$$\begin{pmatrix} 1 & & & & & & & & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ & 1 & & & & & & & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ & & 1 & & & & & & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ & & & 1 & & & & & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ & & & & 1 & & & & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ & & & & & 1 & & & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ & & & & & & 1 & & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ & & & & & & & 1 & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \end{pmatrix} \begin{pmatrix} f_{1,1} \\ f_{1,2} \\ f_{2,1} \\ f_{2,2} \\ f_{1,3}\lambda \\ f_{1,3} \\ f_{2,3}\lambda \\ f_{2,3} \\ f_{3,1}\lambda \\ f_{3,2}\lambda \\ \lambda^2 \\ f_{3,1} \\ f_{3,2} \\ \lambda \\ 1 \end{pmatrix} = 0. \tag{7}$$

Denoting polynomials corresponding to the rows of this matrix equation (7) as p_i, we can write

$$p_i = LT(p_i) + g_i(f_{3,1}, f_{3,2}, \lambda) = 0, \tag{8}$$

where $LT(p_i)$ is the leading term of the polynomial p_i [12], which is, in our case, $LT(p_i) = f_{1,1}, f_{1,2}, f_{2,1}, f_{2,2}, f_{1,3}\lambda, f_{1,3}, f_{2,3}\lambda$ and $f_{2,3}$ for $i = 1, 2, 3, 4, 5, 6, 7, 8$. Polynomials $g_i(f_{3,1}, f_{3,2}, \lambda)$ are second order polynomials in three variables $f_{3,1}, f_{3,2}, \lambda$. Next, we express six variables, $f_{1,1}, f_{1,2}, f_{1,3}, f_{2,1}, f_{2,2}, f_{2,3}$ as the following functions of the remaining three variables $f_{3,1}, f_{3,2}, \lambda$:

$$f_{1,1} = -g_1(f_{3,1}, f_{3,2}, \lambda), \tag{9}$$

$$f_{1,2} = -g_2(f_{3,1}, f_{3,2}, \lambda), \tag{10}$$

$$f_{1,3} = -g_6(f_{3,1}, f_{3,2}, \lambda), \tag{11}$$

$$f_{2,1} = -g_3(f_{3,1}, f_{3,2}, \lambda), \tag{12}$$

$$f_{2,2} = -g_4(f_{3,1}, f_{3,2}, \lambda), \tag{13}$$

$$f_{2,3} = -g_8(f_{3,1}, f_{3,2}, \lambda). \tag{14}$$

We can substitute the expressions (9)-(14) into the remaining two equations p_5 and p_7 from the epipolar constraint (8) and also into the singularity constraint (5) for $\mathtt{F}$. In this way, we obtain three polynomial equations in three unknowns (two third order polynomials and one fifth order polynomial):

$$\lambda(-g_6(f_{3,1}, f_{3,2}, \lambda)) + g_5(f_{3,1}, f_{3,2}, \lambda) = 0, \tag{15}$$

$$\lambda(-g_8(f_{3,1}, f_{3,2}, \lambda)) + g_7(f_{3,1}, f_{3,2}, \lambda) = 0, \tag{16}$$

$$\det \begin{pmatrix} -g_1 & -g_2 & -g_6 \\ -g_3 & -g_4 & -g_8 \\ f_{3,1} & f_{3,2} & 1 \end{pmatrix} = 0. \tag{17}$$

We will use these three equations to solve our problem using two different methods described next.

3 Gröbner Basis Method

The Gröbner basis method for solving systems of polynomial equations became popular in computer vision since it helps to find fast and numerically stable solutions to difficult problems.

Imagine that we want to solve a system of m polynomial equations,

$$f_1(\mathbf{x}) = 0, \ldots, f_m(\mathbf{x}) = 0, \tag{18}$$

in n unknowns $\mathbf{x} = (x_1, \ldots, x_n)$ and that this system has a finite number of solutions.

These polynomials define *ideal* I, which is a set of all polynomials that can be generated as polynomial combinations of the initial polynomials $f_1, \ldots, f_m$:

$$I = \{\Sigma_{i=1}^{m} f_i \; h_i \mid h_i \in \mathbb{C}[x_1, \ldots, x_n]\}, \tag{19}$$

where h_i are arbitrary polynomials from $\mathbb{C}[x_1, \ldots, x_n]$.

In general, an ideal can be generated by many different sets of generators which all share the same solutions. There is a special set of generators though, the reduced Gröbner basis $G = \{g_1, \ldots, g_l\}$ w.r.t. the lexicographic ordering, which generates the ideal I but is easy (often trivial) to solve [12]. Computing this basis and "reading off" the solutions from it is one standard method for solving systems of polynomial equations.

Unfortunately, for most computer vision problems, this "Gröbner basis method w.r.t. the lexicographic ordering" is not feasible. This is because, in general, computation of Gröbner basis is an EXPSPACE-complete problem, meaning that it may take a very long time to compute this basis and huge space may be necessary for storing intermediate results [26].

Therefore, for some problems, a Gröbner basis G under another ordering, e.g., the graded reverse lexicographic ordering, which is often easier to compute, is constructed. Then, the properties of the *quotient ring* $A = \mathbb{C}[x_1, \ldots, x_n]/I$, i.e., the set of equivalence classes represented by remainders modulo I, can be used to get the solutions.

In this quotient ring A, the multiplication matrix $\mathtt{M}_p$ [43], also called the "action" matrix can be constructed. This action matrix is the matrix of the linear operator $T_p : A \to A$ of the multiplication by a suitably chosen polynomial p w.r.t. the basis $B = \{\mathbf{x}^\alpha | \overline{\mathbf{x}^\alpha}^G = \mathbf{x}^\alpha\}$ of A. Here, $\mathbf{x}^\alpha$ represents a monomial $\mathbf{x}^\alpha = x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}$ and $\overline{\mathbf{x}^\alpha}^G$ is the remainder of $\mathbf{x}^\alpha$ on the division by G. The solutions to the system of equations (18) can then be read off directly from the eigenvalues and the eigenvectors of the action matrix [12]. Therefore, this action matrix can be viewed as a generalization of the companion matrix used in solving one polynomial equation in one unknown [12].

Unfortunately, this method based on the computation of the Gröbner basis G w.r.t. the graded reverse lexicographic ordering and construction of the action matrix may take a very long time in general. However, in some cases, when we know the structure of the problem, Gröbner bases and action matrices can be obtained with low cost and efficiently, and this is the case here.

3.1 Standard Method for Constructing the Action Matrix

The standard method for constructing action matrices starts with constructing the monomial basis $B = \{\mathbf{x}^\alpha | \overline{\mathbf{x}^\alpha}^G = \mathbf{x}^\alpha\}$ of the quotient ring A w.r.t. some monomial ordering [12]. Let this monomial basis B consist of N monomials, i.e., $B = \{\mathbf{x}^{\alpha(1)}, \ldots, \mathbf{x}^{\alpha(N)}\}$. Here, the monomial $\mathbf{x}^{\alpha(i)} = x_1^{\alpha_1(i)} x_2^{\alpha_2(i)} \ldots x_n^{\alpha_n(i)}$ for $i = 1, \ldots, N$ and $\alpha(i) = (\alpha_1(i), \ldots, \alpha_n(i))$ is the vector of exponents of the ith monomial, $\alpha_j(i) \in \mathbb{N}$.

With basis $B = \{\mathbf{x}^{\alpha(1)}, \ldots, \mathbf{x}^{\alpha(N)}\}$, the remainder of an arbitrary polynomial on the division by the Gröbner basis G can be expressed as a linear combination of the basic monomials $\mathbf{x}^{\alpha(i)}$, $i = 1, \ldots, N$.

To be precise, it is not the monomials $\mathbf{x}^{\alpha(i)}$, $i = 1, \ldots, N$, but their cosets $[\mathbf{x}^{\alpha(i)}] = \mathbf{x}^{\alpha(i)} + I = \{\mathbf{x}^{\alpha(i)} + h : h \in I\}$ which form the basis of A [12]. However, it is common to identify a remainder with its coset in A and we will do this here, too.

The next step of constructing the action matrix $\mathtt{M}_p$ of the linear operator $T_p : A \to A$ of the multiplication by some polynomial p w.r.t. the monomial basis B requires computing $T_p(\mathbf{x}^{\alpha(i)}) = \overline{p\,\mathbf{x}^{\alpha(i)}}^G$ for all basic monomials $\mathbf{x}^{\alpha(i)} \in B = \{\mathbf{x}^{\alpha(1)}, \ldots, \mathbf{x}^{\alpha(N)}\}$ [12], i.e., computing the remainder of $p\,\mathbf{x}^{\alpha(i)}$ on the division by G.

The standard approach to computing $T_p(\mathbf{x}^{\alpha(i)}) = \overline{p\,\mathbf{x}^{\alpha(i)}}^G$ is to construct a complete Gröbner basis and then use it to get $\overline{p\,\mathbf{x}^{\alpha(i)}}^G$. Yet, to compute $\mathtt{M}_p$, we do not always need a complete Gröbner basis, as we will show in the next section.

3.2 Efficient Method for Constructing the Action Matrix

Here, we propose a method for constructing the action matrix which requires only the basis B of the quotient ring A or even only knowing the number of solutions to the problem. This method needs to generate polynomials from the ideal I with leading monomials from the set $p \cdot B \setminus B$ and the remaining monomials from B, where p is the monomial for which we are

creating the action matrix $\mathtt{M}_p$ and $\setminus$ stands for the set difference. Constructing these polynomials may, in general, be as difficult as generating a Gröbner basis, but for some systems, it requires to generate fewer polynomials than necessary for constructing a complete Gröbner basis; see Appendix B, which can be found in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2011.86.

We assume that the monomial basis B of the algebra A is known or can be computed for the class of problems we are solving in advance and that the action matrix $\mathtt{M}_p$ is constructed for multiplication by some monomial p which can be arbitrary.

Many minimal problems in computer vision, including the one presented here, have the convenient property that the monomials which appear in the set of initial generators F are always same, irrespective of the concrete coefficients arising from nondegenerate image measurements. Therefore, the leading monomials of the corresponding Gröbner basis, and thus the monomials in the basis B, are generally the same and they can be found once in advance. To do so, we use the approach originally suggested in [46], [44], and [45] for computing Gröbner bases, but we retrieve the basis B and polynomials required for constructing the action matrix instead.

Having B, the action matrix can be computed as follows: if for some monomial $\mathbf{x}^{\alpha(i)} \in B$ and the chosen monomial p, the monomial $p\,\mathbf{x}^{\alpha(i)} \in B$, then $T_p(\mathbf{x}^{\alpha(i)}) = \overline{p\,\mathbf{x}^{\alpha(i)}}^G = p\,\mathbf{x}^{\alpha(i)}$ and we are done. For all other monomials $\mathbf{x}^{\alpha(i)} \in B$, for which $p\,\mathbf{x}^{\alpha(i)} \notin B$, consider polynomials $q_i = p\,\mathbf{x}^{\alpha(i)} + h_i$ with $h_i = \sum_{j=1}^{N} c_j^i \mathbf{x}^{\alpha(j)}$, $c_j^i \in \mathbb{C}$ and $\mathbf{x}^{\alpha(j)} \in B$. Then, it holds:

Theorem 1. *For each $p\,\mathbf{x}^{\alpha(i)} \notin B$, there exists exactly one polynomial q_i of the form $q_i = p\,\mathbf{x}^{\alpha(i)} + \sum_{j=1}^{N} c_j^i \mathbf{x}^{\alpha(j)}$ in the ideal I and it holds*

$$T_p(\mathbf{x}^{\alpha(i)}) = \overline{p\mathbf{x}^{\alpha(i)}}^G = -\sum_{j=1}^{N} c_j^i \mathbf{x}^{\alpha(j)} = -h_i.$$

See Appendix A, which can be found in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2011.86, for the proof.

This theorem shows that remainders $T_p(\mathbf{x}^{\alpha(i)}) = \overline{p\,\mathbf{x}^{\alpha(i)}}^G$, and therefore also the action matrix $\mathtt{M}_p$, can be constructed here without explicitly constructing the Gröbner basis. All we need is to construct polynomials q_i from the ideal I with leading monomials from the set $p \cdot B \setminus B$ and the remaining monomials from B. Then, the ith row of the action matrix $\mathtt{M}_p$ for $p\,\mathbf{x}^{\alpha(i)} \notin B$ consists of the coefficients of q_i, which correspond to the monomials from B but with opposite signs.

Since polynomials q_i are from the ideal I, we can generate them as algebraic combinations of the initial generators $F = \{f_1, \ldots, f_m\}$ of the ideal I. This can be done using several methods. One possible way is to start with F and then systematically generate new polynomials from I by multiplying already generated polynomials by individual variables and reducing them each time by the Gauss-Jordan elimination. This method was, for example, used in [27] and resulted in several G-J eliminations. Another possible way is to generate all new polynomials up to a necessary degree in one step by multiplying polynomials from F with selected monomials and then reduce all generated polynomials at once using one G-J elimination. This method was used in [8] and we have observed that for systems with fewer variables, like in our case, this method was numerically more stable. Therefore, we use the latter method to construct the action matrix for our problem.

To clarify the presented method for creating the action matrix, we demonstrate it on a simple example in Appendix B, which can be found in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2011.86.

4 Gröbner Basis Solver

The Gröbner basis solver of our problem starts with three equations (15)-(17) in three unknowns, as described in Section 2. To create the action matrix, we use the method described in Section 3.3. This method assumes that the basis B of A is known. Next, we describe how we can find this basis.

4.1 Computing B and the Number of Solutions

To compute B, we solve our problem in a randomly chosen finite prime field $\mathbb{Z}_p$ ($\mathbb{Z}/\langle p \rangle$) with $p = 30{,}097$, where exact arithmetic can be used and numbers can be represented in a simple and efficient way. It speeds up computations and minimizes memory requirements.

We use algebraic geometry software Macaulay 2, which can compute in finite fields, to solve the polynomial equations for many random coefficients from $\mathbb{Z}_p$, to compute the number of solutions, the Gröbner basis, and especially the basis B. If the basis B remains stable for many different random coefficients, it is generically equivalent to the basis of the original system of polynomial equations [53].

We can use the Gröbner basis and the basis B computed for random coefficients from $\mathbb{Z}_p$ thanks to the fact that in our class of problems, the way of computing the Gröbner basis is always the same and, for particular data, these Gröbner bases differ only in coefficients. This holds for B, which consists of the same monomials, as well. Also, the way of obtaining polynomials that are necessary to create the action matrix is always the same and, for general data, the generated polynomials again differ only in their coefficients.

To create the action matrix, we use the graded reverse lexicographic ordering $f_{3,1} > f_{3,2} > \lambda$. With this ordering, we get the basis $B = (f_{3,1}^3, f_{3,1}^2 f_{3,2}, f_{3,1} f_{3,2}^2, f_{3,2}^3, f_{3,2}^2 \lambda, \lambda^3, f_{3,1}^2, f_{3,1} f_{3,2}, f_{3,2}^2, f_{3,1}\lambda, f_{3,2}\lambda, \lambda^2, f_{3,1}, f_{3,2}, \lambda, 1)$ of the quotient ring $A = \mathbb{C}[f_{3,1}, f_{3,2}, \lambda]/I$. This way, we have found that our problem has 16 solutions.

4.2 Constructing the Action Matrix

Once we know the basis B, the action matrix for floating-point coefficients can be created. For creating the action matrix, we have tested all three unknowns $f_{3,1}$, $f_{3,2}$, and λ. We have found that the best choice for the "action variable," which leads to the simplest solution, is λ. Therefore, we construct the action matrix $\mathtt{M}_\lambda$.

The method described in Section 3.2 calls for generating polynomials q_i with leading monomials from the set $\lambda \cdot B \setminus B$ and the remaining monomials from B. As described in

Section 3.2, this can be done using several methods. One possible way is to start with F and then systematically generate new polynomials from I by multiplying already generated polynomials by individual variables and reducing them each time by the Gauss-Jordan elimination. This method was used to solve this minimal problem in [27] and results in three G-J eliminations of 8×22, 11×30, and 36×50 matrices.

Here, we use a different approach in which we generate all necessary polynomials q_i in one step by multiplying the initial three polynomial equations with selected monomials and reducing all generated polynomials at once using one G-J elimination.

The set of selected monomials which should be used to multiply initial polynomial equations can be found once in advance. To do this, we again use Macaulay 2. We have found that obtaining all necessary polynomials for creating the action matrix calls for generating all monomial multiples of the initial three polynomial equations up to the total degree 6. This means that we need to multiply our two third degree polynomial equations (15) and (16) with all monomials up to degree 3 and our fifth degree polynomial equation (17) from the singularity constraint with all first degree monomials.

In this way, we generate 41 new polynomials which, together with the initial three polynomial equations, form a system of 44 polynomials in 84 monomials. Since these polynomials were generated systematically, they also contain polynomials that do not affect the resulting action matrix. We remove all these unnecessary polynomials by the following procedure, originally introduced in [5].

In this procedure, we rewrite our 44 polynomials in the matrix form $\mathtt{MX} = 0$, where $\mathtt{M}$ is the coefficient matrix and $\mathtt{X}$ is the vector of all 84 monomials. We know that after G-J elimination of this matrix $\mathtt{M}$, we obtain all polynomials q_i that we need for constructing the action matrix $\mathtt{M}_\lambda$. Since we know how these necessary polynomials q_i should look, i.e., which monomials they contain, we can systematically reduce the number of generated polynomials in the following way:

For all rows in $\mathtt{M}$, starting with the last row $\mathtt{r}$ (i.e., with the polynomial of highest degree), do:

1. Perform G-J elimination on the matrix $\mathtt{M}$ without the row $\mathtt{r}$.
2. If the eliminated matrix contains all necessary polynomials, $\mathtt{M} := \mathtt{M} \setminus r$.

For this problem, running the above removing procedure results in 32 polynomial equations in 84 monomials. After rewriting these 32 polynomials in the matrix form and performing the G-J elimination of the corresponding coefficient matrix $\mathtt{M}$, we obtain all polynomials which we need for the construction of the action matrix $\mathtt{M}_\lambda$.

Before this G-J elimination, we can also remove columns of the matrix $\mathtt{M}$ which correspond to the monomials which do not have impact on our solution, i.e., which do not appear in the action matrix $\mathtt{M}_\lambda$. In this way, we obtain 32×48 coefficient matrix $\mathtt{M}$.

4.3 The Online Gröbner Basis Solver

Note that all the processing until now (i.e., computation in Macaulay 2, finding the basis B, and finding the polynomials form the ideal which should be generated to obtain all necessary polynomials for constructing the action matrix) needs to be done only once when studying this problem and building its online solver. The online solver then does only one G-J elimination of the 32×48 coefficient matrix $\mathtt{M}$. This matrix contains coefficients which arise from concrete image measurements. After G-J elimination of $\mathtt{M}$, the action matrix $\mathtt{M}_\lambda$ can be created from rows which correspond to the polynomials with leading monomials from the set $\lambda \cdot B \setminus B$. The eigenvectors of this action matrix $\mathtt{M}_\lambda$ give solutions for $f_{3,1}, f_{3,2}, \lambda$. Backsubstituting these solutions to (9)-(14) gives solutions also for $f_{1,1}, f_{1,2}, f_{1,3}, f_{2,1}, f_{2,2}, f_{2,3}$. In this way, we obtain 16 (complex) solutions. Generally, less than 10 solutions are real.

5 Polynomial Eigenvalue Problems (PEP)

The second solution to our problem is based on polynomial eigenvalue problems.

Polynomial eigenvalue problems are problems of the form

$$\mathtt{A}(\alpha)\mathbf{v} = 0, \tag{20}$$

where $\mathtt{A}(\alpha)$ is a matrix polynomial defined as

$$\mathtt{A}(\alpha) \equiv \alpha^l \mathtt{C}_l + \alpha^{l-1}\mathtt{C}_{l-1} + \cdots + \alpha\mathtt{C}_1 + \mathtt{C}_0, \tag{21}$$

in which the $\mathtt{C}_j$ are square $n \times n$ coefficient matrices.

An important class of polynomial eigenvalue problems are quadratic eigenvalue problems, where the matrix polynomial $\mathtt{A}(\alpha)$ has degree 2. These problems have the form

$$(\alpha^2\mathtt{C}_2 + \alpha\mathtt{C}_1 + \mathtt{C}_0)\mathbf{v} = 0, \tag{22}$$

where $\mathtt{C}_2$, $\mathtt{C}_1$, and $\mathtt{C}_0$ are given matrices of size $n \times n$, $\mathbf{v}$ is the eigenvector, and α corresponding eigenvalue.

QEP (22) can be easily transformed to the following generalized eigenvalue problem (GEP) [1]:

$$\mathtt{A}\,\mathbf{y} = \alpha \mathtt{B}\,\mathbf{y}, \tag{23}$$

where

$$A = \begin{pmatrix} 0 & I \\ -\mathtt{C}_0 & -\mathtt{C}_1 \end{pmatrix}, \mathtt{B} = \begin{pmatrix} \mathtt{I} & 0 \\ 0 & \mathtt{C}_2 \end{pmatrix}, \mathbf{y} = \begin{pmatrix} \mathbf{v} \\ \alpha\mathbf{v} \end{pmatrix}. \tag{24}$$

It can easily be seen that this GEP (23) and therefore also the QEP (22) have $2n$ eigenvalues.

Higher order polynomial eigenvalue problems can be transformed to the generalized eigenvalue problems (23) in a similar way as QEPs [1] with

$$\mathtt{A} = \begin{pmatrix} 0 & \mathtt{I} & 0 & \dots & 0 \\ 0 & 0 & \mathtt{I} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ -\mathtt{C}_0 & -\mathtt{C}_1 & -\mathtt{C}_2 & \dots & -\mathtt{C}_{l-1} \end{pmatrix},$$

$$\mathtt{B} = \begin{pmatrix} \mathtt{I} & & & \\ & \dots & & \\ & & \mathtt{I} & \\ & & & \mathtt{C}_l \end{pmatrix}, \ \mathbf{y} = \begin{pmatrix} \mathbf{v} \\ \alpha\mathbf{v} \\ \dots \\ \alpha^{l-1}\mathbf{v} \end{pmatrix}. \tag{25}$$

Generalized eigenvalue problems (23) are well-studied problems and there exist efficient numerical algorithms for solving them [1]. For example, MATLAB provides the

polyeig function for solving polynomial eigenvalue problems (20) of arbitrary degree (including the transformation to the generalized eigenvalue problems). Moreover, these generalized eigenvalue problems can be further transformed to standard eigenvalue problems [1] and solved using standard eigenvalue algorithms which are available in almost all mathematical softwares and libraries.

For higher order PEPs, one has to work with larger matrices with $n\,l$ eigenvalues. Therefore, for larger values of n and l, problems with the convergence of the techniques for solving these problems sometimes appear [1].

6 Polynomial Eigenvalue Solver

The polynomial eigenvalue solution to the problem of simultaneous estimation of epipolar geometry and one-parameter division model from eight point correspondences starts with three equations (15)-(17) in three unknowns $f_{3,1}, f_{3,2}, \lambda$ and 39 monomials as described in Section 2.

Equations (15) and (16) are of degree 3 and (17) is of degree 5. If we take unknown radial distortion parameter λ to play the role of α in (20), we can rewrite these three equations as

$$(\lambda^4 \mathtt{C}_3 + \lambda^3 \mathtt{C}_3 + \lambda^2 \mathtt{C}_2 + \lambda\, \mathtt{C}_1 + \mathtt{C}_0)\mathbf{v} = 0, \qquad (26)$$

where $\mathbf{v} = (f_{3,1}^3,\ f_{3,1}^2 f_{3,2},\ f_{3,1} f_{3,2}^2,\ f_{3,2}^3, f_{3,1}^2,\ f_{3,1} f_{3,2},\ f_{3,2}^2,\ f_{3,1},\ f_{3,2}, 1)^\top$ is a 10×1 vector of monomials and $\mathtt{C}_4$, $\mathtt{C}_3$, $\mathtt{C}_2$, $\mathtt{C}_1$, and $\mathtt{C}_0$ are 3×10 coefficient matrices.

Polynomial eigenvalue formulation (20) requires having square coefficient matrices $\mathtt{C}_\mathrm{j}$. Unfortunately, in this case, in contrast to [11], [29], and [5], we do not have square coefficient matrices. We have only three equations and 10 monomials in the vector $\mathbf{v}$.

One way in which we can get square coefficient matrices $\mathtt{C}_\mathrm{j}$ is to generate new equations, monomial multiples of our initial three equations (15)-(17), in such a way that we do not produce new monomials, except for monomials that are in these three equations (15)-(17). However, this is usually not so easy to do.

In our case, two from three equations (15) and (16) are of degree 3 and contain unknowns $f_{3,1}$ and $f_{3,2}$ in degree 1 only. The third equation (17) from the determinant is of degree 5 and contains unknowns $f_{3,1}$ and $f_{3,2}$ up to degree 3. Therefore, we can multiply our two third degree equations (15) and (16) with all monomials containing $f_{3,1}$ and $f_{3,2}$ up to degree 2 $(f_{3,1}, f_{3,2}, f_{3,1}^2, f_{3,1} f_{3,2}, f_{3,2}^2)$. In this way, we can generate 10 new equations, monomial multiples of our initial equations, which have the same solutions as our initial equations and, what is important, which also have only monomials from $\mathbf{v}$.

To obtain square coefficient matrices $\mathtt{C}_\mathrm{j}$ in (26), we need seven new equations. Since we have 10 suitable equations, we can select seven from them. We can do this such that the resulting system has a small condition number. After rewriting our three equations together with new seven equations to the form (26), we obtain square 10×10 coefficient matrices $\mathtt{C}_\mathrm{j}$.

Since $\mathtt{C}_4$, $\mathtt{C}_3$, $\mathtt{C}_2$, $\mathtt{C}_1$, and $\mathtt{C}_0$ are known square matrices, the formulation (26) is a PEP of degree 4. Such a problem can be easily solved using standard efficient algorithms, for example, by MATLAB function `polyeig`.

After solving the PEP (26), we obtain 40 eigenvalues, solutions for λ and 40 corresponding eigenvectors $\mathbf{v}$ from which we extract solutions for $f_{3,1}$ and $f_{3,2}$. To do this, we normalize solutions for eigenvectors $\mathbf{v}$ to have the last coordinate 1 and extract values from $\mathbf{v}$ that correspond to $f_{3,1}$ and $f_{3,2}$.

Between these 40 eigenvalues, there are several "parasitic" zero eigenvalues. Eleven from these zero eigenvalues correspond to zero columns of the coefficient matrices $\mathtt{C}_j$ and can be easily removed before computing the eigenvalue problem. This can be done by removing the "zero" columns and corresponding rows of matrices `A` and `B` (25). This means that we reduce the original 40×40 eigenvalue problem to a 29×29 eigenvalue problem.

However, we still obtain more solutions than the number of solutions to the original system of polynomial equations which is 16. This is because this polynomial eigenvalue method solves a relaxed version of the proposed problem. This relaxed version does not take into account monomial dependencies induced by the problem, e.g., $\mathbf{v}(1) = \mathbf{v}(8)^3$. Therefore, the solution contains $\mathbf{v}$s that automatically (within limits of numerical accuracy) satisfy these constraints and additional general eigenvectors $\mathbf{v}$s that do not satisfy them. However, such $\mathbf{v}$s can be eliminated by verifying the monomial dependences or by substituting the solutions into the initial polynomial equations (15)-(17).

Solutions satisfying the monomial constraints on $\mathbf{v}$ will be obtained for exact as well as noisy data. Since we are solving a minimal problem, noisy data can be viewed as perfect input for a different camera configuration. Hence, we again obtain a solution satisfying the monomial constraints on the vector $\mathbf{v}$.

After extracting the solutions for $f_{3,1}$ and $f_{3,2}$, we use (9)-(14) to get solutions for the fundamental matrix `F`.

7 Experiments

We have tested our algorithms on both synthetic data (with various levels of noise, outliers, radial distortions, and radial distortion center shifts) and on real data sets and compared them with the minimal Gröbner basis algorithm presented in [27] and with the nonminimal 9-point algorithms for correcting radial distortion [16], [33].

From both our solvers, we get several (complex) roots. In the case of the Gröbner basis solver, we get 16 roots and, in the case of the polynomial eigenvalue solver (polyeig solver), we get 29 roots from which maximally 16 are satisfying all monomial constraints.

In general, more than one and less than 10 roots are real. If there is more than one real root, we need to select the best root, the root which is consistent with most measurements. To do so, we treat the real roots of the 16, in general complex, roots obtained by solving the equations for one input as real roots from different inputs and use RANSAC [15] or kernel voting [33], [27] for several inputs to select the best root among all computed roots. The kernel voting is done by a Gaussian kernel with fixed variance and the estimate of λ is found as the position of the highest peak. See [33], [27] for more on kernel voting for this problem.

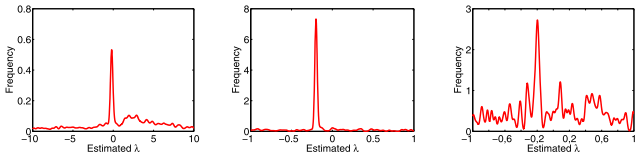

Fig. 1. Distribution of real roots using kernel voting for 1,000 point matches, 200 estimations, and $\lambda_{true} = -0.2$. (Left) Roots within the range $[-10, 10]$ and no noise contamination. (Center) Roots within the range $[-1, 1]$ and no noise. (Right) Roots within the range $[-1, 1]$, 80 percent of inliers and $0.5px$ noise.

Fig. 1 shows distributions of real roots for a typical situation constructed using the kernel voting. In this case, 200 estimates were computed for the ground truth $\lambda = -0.2$. It can be seen that it is suitable to use kernel voting and that it makes sense to select the best root by casting votes from all computed roots and estimate λ as the position of the highest peak.

Fig. 1 shows that it is meaningful to vote for λs either 1) within the range where most of the computed roots fall (in our case $[-10, 10]$), Fig. 1 (left), or 2) within the smallest range in which we are sure that the ground truth lies (in our case $[-1, 1]$), Fig. 1 (middle). Moreover, from Fig. 1 (right), it can be seen that this standard kernel voting method is robust to noise and small number of outliers (up to 30 percent). For a higher number of outliers, more advanced kernel voting techniques, e.g., the weighted kernel voting method which takes into account the number of inliers and combines RANSAC with kernel voting [6], can be used.

7.1 Synthetic Data Sets

We initially studied our algorithms on synthetically generated ground truth 3D scenes. These scenes were generated using the following procedure:

1. Generate a 3D scene consisting of N $(= 1{,}000)$ random points within a 3D cube.
2. Project M percent of the points on image planes of the two displaced cameras. These are matches. Generate $(100 - M)$ percent random points in both images. These are mismatches. Altogether, they become undistorted correspondences.
3. Apply the radial distortion to the undistorted correspondences to generate noiseless distorted points.
4. Add Gaussian noise of standard deviation σ to the distorted points assuming a $1{,}000 \times 1{,}000$ pixel image.

In the first synthetic experiment, we have studied how well the results from our solvers fit the distorted data. To do this, we have created four synthetic scenes using different radial distortion models. The first two scenes were created using the division model for which the solvers were intended and the remaining two scenes were created using the three parameter distortion model:

$$\mathbf{p}_d \sim \mathbf{p}_u\big(1 + \lambda_1 r_u + \lambda_2 r_u^2 + \lambda_3 r_u^3\big). \tag{27}$$

The numbers of mismatches in all these scenes were 400, which is 40 percent, and the noise level was 0.5 pixels.

Fig. 2 shows the mean value of the number of inliers over 1,000 runs of RANSAC as a function of the number of samples of the RANSAC. The top row shows the results for

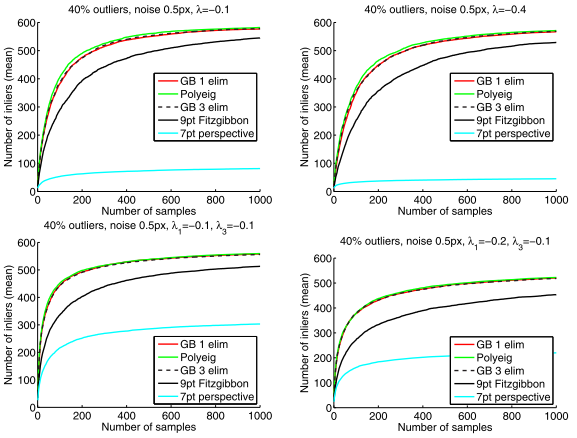

Fig. 2. The mean value of the number of inliers over 1,000 runs of RANSAC as a function of the number of samples of the RANSAC. The top row shows the results for the division model with parameters $\lambda = -0.1$ (left) and $\lambda = -0.4$ (right) and the bottom row the results for the model (27) with $\lambda_1 = -0.1$ and $\lambda_3 = -0.1$ (left) and $\lambda_1 = -0.2$ and $\lambda_3 = -0.1$ (right).

the division model with ground truth $\lambda = -0.1$ (left) and $\lambda = -0.4$ (right) and the bottom row the results for the model (27) with ground truth $\lambda_1 = -0.1$ and $\lambda_3 = -0.1$ (left), respectively, with $\lambda_1 = -0.2$ and $\lambda_3 = -0.1$ (right). In this experiment, we have compared both proposed minimal solvers, the Gröbner basis solver (red) and the polyeig solver (green), with the minimal Gröbner basis solver proposed in [27] (blue-dashed), the nonminimal solver [16] (black), and the standard perspective 7-point algorithm [22] (cyan). It can be seen from Fig. 2 that all minimal solvers give very similar and good results and outperform the nonminimal solver [16] (black) and, what is also expected, the standard perspective 7-point algorithm, which does not take into account radial distortion at all.

In our next synthetic experiment, we have studied the numerical stability of both proposed solvers and compared them with the nonminimal 9-point solver [16] and with the minimal Gröbner basis solver proposed in [27]. We have focused on radial distortion parameter estimation because once we have a good radial distortion estimate, the fundamental matrix is usually good or can be recomputed using well-known algorithms for perspective cameras [22], [37], [45].

Fig. 3 (left) shows the log_{10} relative error of the radial distortion parameter λ obtained by selecting the real root closest to the ground truth value $\lambda_{true} = -0.2$ for a noise-free synthetic scene. It can be seen that the new proposed polyeig solver (green) provides the most accurate results and that one Gauss-Jordan elimination (red) little bit helps in precision over three Gauss-Jordan eliminations [27] (blue). For data with noise, the results of all solvers are almost the same, see Fig. 3 (right).

In the next two synthetic experiments, we have studied the robustness of our algorithms to outliers and to increasing levels of Gaussian noise added to the distorted points. The ground truth radial distortion λ_{true} was -0.2 and the level of noise varied from 0 to 2 pixels. We have used two strategies for selecting the best λ among all generated radial distortion parameters, the RANSAC strategy, and the kernel voting.

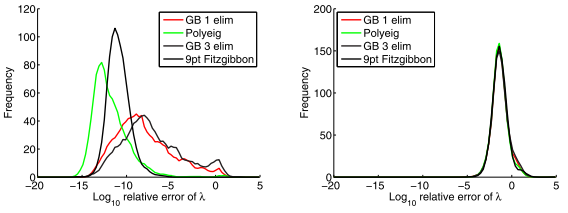

Fig. 3. Log_{10} relative error of the radial distortion parameter λ obtained by selecting the real root closest to the ground truth value $\lambda_{true} = -0.2$ for (left) noise-free synthetic scene and (right) noise $0.5px$.

Fig. 4 shows λs computed by the four studied algorithms as a function of noise. In this case, 500 λs were estimated using RANSAC for each noise level and 80 percent (left) and 60 percent (right) of inliers. The results are presented by the Matlab function *boxplot*, which shows values 25 to 75 percent quantile as a box with horizontal line at median. The crosses show data beyond 1.5 times the interquartile range.

The median values (from -0.1968 to -0.2036) for all 8-point algorithms as well as the 9-point algorithm (black) are very close to the ground truth value $\lambda_{true} = -0.2$ for all noise levels. The variances of the 9-point algorithm [16] (black) are little bit larger than the variances of the 8-point algorithms; however, this is not significant. More important is the fact that for larger noise levels, the 9-point algorithm sometimes failed and delivered result far from the ground truth value inside the RANSAC loop. For the noise level 2 pixels, 5 percent of the results of the 9-point algorithm were far from the ground truth value.

We have performed the same experiment using the kernel voting strategy for selecting the best λ. In this case, the resulting λ was estimated using the following procedure:

1. Repeat K times. (We use $K = 100$ here, but in many cases, K from 30 to 50 is sufficient.)
   a. Randomly choose eight point (nine point) correspondences from given N correspondences.
   b. Find all λs of the minimal (nonminimal) solution to the autocalibration of radial distortion.
   c. Select the real λs in the feasible interval, e.g., $-1 < \lambda < 1$ and the corresponding Fs.
2. Use kernel voting on all selected λs to estimate the best radial distortion parameter.

Fig. 5 shows estimated λs using this procedure as a function of noise. Five hundred λs were estimated from 100 ($K = 100$) 8 and 9-tuples of correspondences randomly drawn for each noise level and 100 percent (left) and 80 percent (right) of inliers. Here, the results are worse for

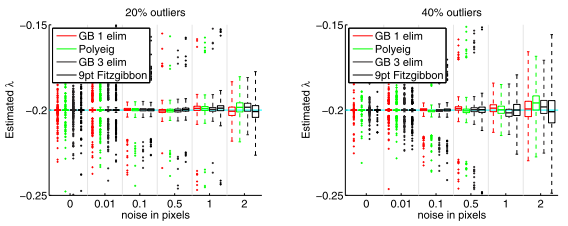

Fig. 4. RANSAC: Estimated λ as a function of noise, ground truth $\lambda_{true} = -0.2$ and (left) $inliers = 80\%$, (right) $inliers = 60\%$. Boxes contain values from 25 to 75 percent quantile.

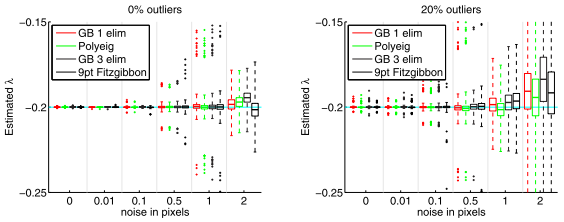

Fig. 5. KERNEL VOTING: Estimated λ as a function of noise, ground truth $\lambda_{true} = -0.2$, and (left) $inliers = 100\%$, (right) $inliers = 80\%$. Boxes contain values from 25 to 75 percent quantile.

larger noise levels than in the RANSAC experiment (Fig. 4); however, for smaller outlier contamination and lower noise levels, the results are again very precise.

Since in our algorithms we assume that the distortion center is in the center of the image, we have studied in the final synthetic experiment the robustness of our algorithms to a shift of the radial distortion center. The ground truth radial distortion λ_{true} was -0.1, the number of outliers 40 percent, and the shift of the radial distortion center from the image center varied from 0 to 30 percent of the image size (for $1{,}000 \times 1{,}000$ px image from 0 to 300 px).

Fig. 6 shows the mean value of the number of inliers over 1,000 runs of RANSAC as a function of the number of samples of the RANSAC. The top left figure shows the result for the radial distortion center shifted by 1 percent of the image size from the image center, the top right figure for the 5 percent shift, the bottom left for the 10 percent shift, and, finally, the bottom right figure shows the result for the 30 percent shift.

We again show results for both proposed minimal solvers, the Gröbner basis solver (red) and the polyeig solver (green), the minimal Gröbner basis solver proposed in [27] (blue-dashed), and the nonminimal solver [16] (black). It can be seen from Fig. 6 that all minimal solvers give very similar and good results for all shifts of the radial distortion center and outperform the nonminimal solver [16] (Black). For all radial distortion shifts, except for the 30 percent shift, all minimal solvers were able to find

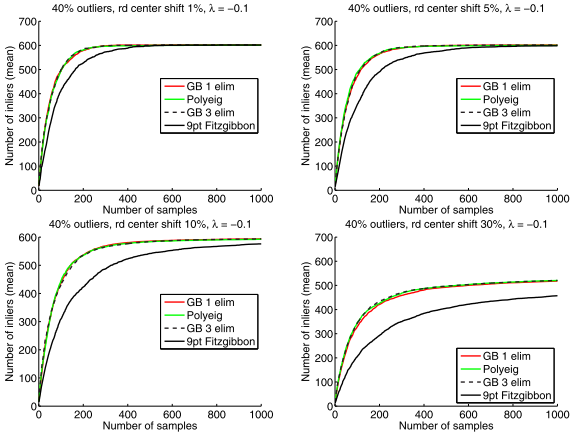

Fig. 6. The mean value of the number of inliers over 1,000 runs of RANSAC as a function of the number of samples of the RANSAC for $\lambda_{true} = -0.1$, $outliers = 40\%$ and the radial distortion shift 1 percent of the image size (top left), 5 percent of the image size (top right), 10 percent of the image size (bottom left), and 30 percent of the image size (bottom right).

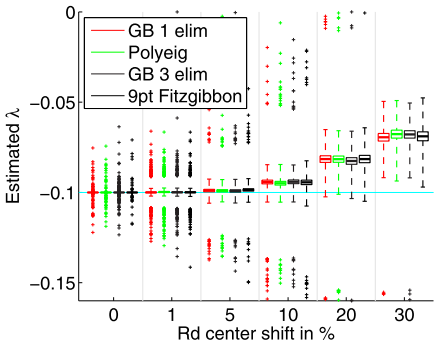

Fig. 7. Estimated λs as a function of the radial distortion center shift, ground truth $\lambda_{true} = -0.1$, $outliers = 40\%$. Boxes contain values from 25 to 75 percent quantile.

the radial distortion parameter and the epipolar geometry consistent with all 600 inliers within 200 cycles of RANSAC. For the 30 percent shift, the results were also satisfactory.

Fig. 7 shows λs computed by the four studied algorithms as a function of the radial distortion center shift. One thousand λs were estimated using RANSAC for each radial distortion center shift from 0 to 30 percent of the image size and 40 percent of outliers. The results are presented by the Matlab function *boxplot*. The median values of estimated λs are quite close to the ground truth value $\lambda_{true} = -0.1$ for all radial distortion shifts.

These RANSAC experiments (Figs. 6 and 7) show that even large shifts of the radial distortion center from the image center do not make problems in finding quite precise distortion parameters and good inliers sets.

7.2 Tests on Real Images

We have tested both our algorithms on several sequences of real images. The first image sequence was sequence with relatively large distortions. These images were obtained as 75 percent cutouts from 180 degree angle of view fish-eye images. We use cutouts since the one-parameter division model (1) is not able to handle 180 degree images. Tentative point correspondences were found by the wide baseline matching algorithm [35]. They contained correct as well as incorrect matches. Note that for such distorted images, the number of incorrect matches is relatively large (usually between 30 and 70 percent). For all images, we approximated the center of radial distortion with the image center.

In our first experiment with this image sequence, we have studied the stability of the estimated radial distortion parameter. To do this, we have estimated λs for the image sequence taken by a single camera. Fig. 8 shows λs estimated for each image pair using our Gröbner basis algorithm (left), our polyeig algorithm (middle), and the nonminimal 9-point algorithm [16] (right). In this case, 100 λs were estimated using RANSAC and the results are presented using the Matlab function *boxplot*.

It can be seen that both minimal solvers give very similar and relatively stable results. The median values (red lines in the figure) for these solvers varied from -0.3809 to -0.2935. The variances of the 9-point solver, Fig. 8 (right), were considerably larger than the variances of both minimal solvers, Fig. 8 (left and middle). The mean range of upper and lower quartile (blue box in Fig. 8) of estimated λs was, for the nonminimal 9-point algorithm, 0.1021, for our polyeig algorithm, 0.0521, and for the Gröbner basis algorithm, 0.0481, which is approximately two times lower than for the nonminimal case. One should also mention that the number of outliers in this image sequence was not negligible. It varied from 35 to 60 percent.

Fig. 9 (top) shows examples of images from processed image sequence. Corrected images (bottom) were obtained using median values from λs computed using our polyeig algorithm together with RANSAC (Fig. 8 (middle)). It can be seen that straight lines in the real world are really straight on corrected images.

In the second experiment, we have tested the kernel voting strategy for selecting the best λ on the same image sequence as it was used in the first experiment. Fig. 10 shows results of the kernel voting made on λs obtained from 500 runs of our polyeig algorithm for image pairs 4-5, 5-6, 8-9, and 11-12. Red vertical lines show estimated λs using the kernel voting method and dashed green lines show λs computed as median values from 100 results of RANSAC (Fig. 8 (Middle)). In both cases, our polyeig algorithm was used to estimate λ.

It can be seen that both methods, the kernel voting and RANSAC, behave similarly when estimating radial distortion parameter. On image pairs 4-5 and 5-6, where the RANSAC method returns very stable results, the kernel voting strategy produces histograms with significant peaks. In this case, results from RANSAC (green dashed line) are very close to the results from the kernel voting (red line). On the other hand, on image pairs 8-9 and 11-12 where the variations of results from RANSAC are larger, the kernel

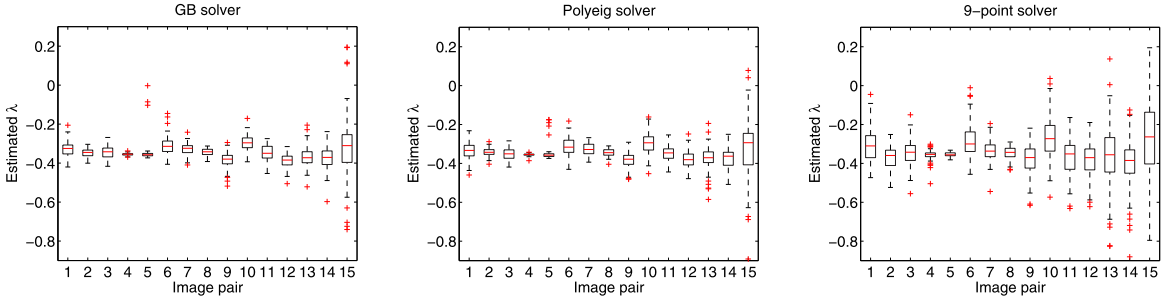

Fig. 8. One hundred λs estimated using RANSAC for the real image sequence consisting of 16 images taken by one camera. Blue boxes contain values from 25 to 75 percent quantile: (left) the Gröbner basis algorithm, (middle) the polyeig algorithm, (right) the nonminimal 9-point algorithm [16].

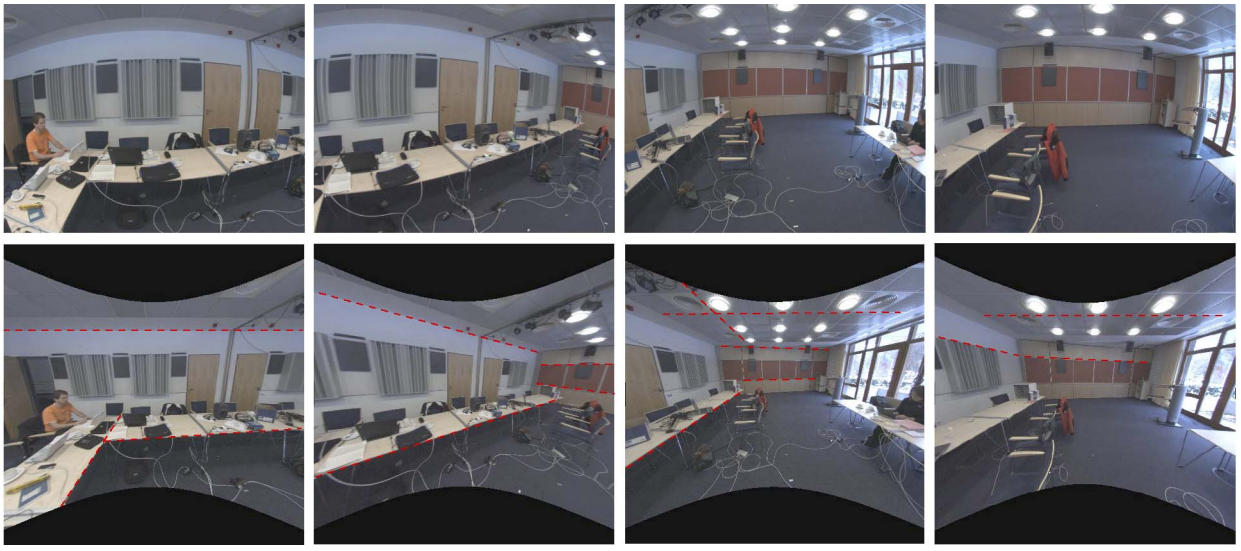

Fig. 9. Examples of images from processed image sequence. (Top) Input image with significant radial distortion. (Bottom) Corrected images using median values from Fig. 8.

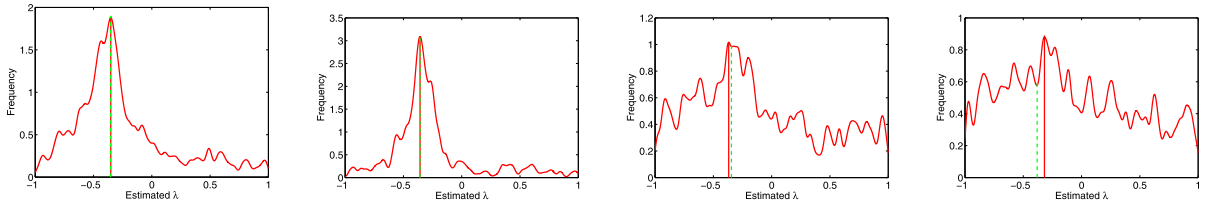

Fig. 10. Result of the kernel voting strategy on image pairs 4-5, 5-6, 8-9, and 12-13 from image sequence from Fig. 8. Red vertical lines show estimated λs using the kernel voting method and dashed green lines show λs computed as median values from 100 results of RANSAC (Fig. 8 (middle)). In both cases, our polyeig algorithm was used to estimate λ.

Fig. 11. (Left) Example of input image, (middle) Corrected image, (right) 100 λs estimated using RANSAC for each from 10 image pairs taken by one camera. Blue boxes contain values from 25 to 75 percent quantile.

voting strategy doesn't produce so significant peaks. Still, the results of the kernel voting corresponding to the highest peak (red line) are close to the results from RANSAC (green dashed line). However, we suggest using RANSAC or a combination of RANSAC with weighted kernel voting [6] instead of the standard kernel voting in the cases with higher noise level and outlier contamination, as it was in this case.

Finally, we have performed the same experiments on several image pairs taken by standard consumer camera at 28 mm focal length. Fig. 11 (right) shows λs estimated for each image pair using our Gröbner basis algorithm. Again, 100 λs were estimated using RANSAC and the results are presented using the Matlab function *boxplot*. Here, the results were even more stable than the results of the same experiment for the fish-eye camera (Fig. 8). The mean values varied from -0.0243 to -0.0476. These values of λ are quite small, what corresponds to quite small radial distortion of the input images as can be seen in the example of input image in Fig. 11 (left). However, this radial distortion was still large enough to create problems in 3D reconstruction and, without its correction, standard SfM pipelines were not able to give satisfactory results. The example of corrected image can be seen in Fig. 11 (middle).

7.3 Time Complexity

Our current MATLAB implementation of the Gröbner basis solver runs about 1.2 ms on an AMD Athlon 64 X2 Dual, 4800+, 2.51 GHz. The running time of the polyeig solver is about 1.75 ms. For comparison, our MATLAB implementation of Fitzgibbon's nonminimal algorithm runs about 1.68 ms and the original implementation of Li's algorithm [33] based on MATLAB Symbolic-Math Toolbox runs about 860 ms.

8 Conclusion

We have presented two robust and efficient solutions to the minimal problem for the autocalibration of radial distortion from eight point correspondences. Both solutions have been obtained by a specialization of general techniques for solving systems of polynomial equations. The first solution is based on the Gröbner basis method and the second on the polynomial eigenvalue technique for solving systems of algebraic equations. Our algorithms provide singular fundamental matrices, reduce the number of samples in RANSAC, and are more stable than previously known 9-point algorithms [16], [33]. We demonstrate the quality of both our solvers by experiments on synthetic and real data. Since the Gröbner basis solver solves smaller eigenvalue problem and delivers less solutions (16 versus 29), it is faster and more efficient. The polynomial eigenvalue solver is, on the other hand, easy to understand and implement and also a little bit more stable than the Gröbner basis solver. However, both solvers perform well on synthetic and real data.

Acknowledgments

This work has been supported by EC project FP7-SPACE-241523 PRoViScout and by the Czech Government under the research program MSM6840770038.

References

[1] Z. Bai, J. Demmel, J. Dongorra, A. Ruhe, and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems.* SIAM, 2000.

[2] J. Barreto and K. Daniilidis, "Fundamental Matrix for Cameras with Radial Distortion," *Proc. IEEE Int'l Conf. Computer Vision,* Oct. 2005.

[3] J. Barreto, J. Roquette, P. Sturm, and F. Fonseca, "Automatic Camera Calibration Applied to Medical Endoscopy," *Proc. British Machine Vision Conf.,* 2009.

[4] C. Bräuer-Burchardt and K. Voss, "A New Algorithm to Correct Fish-Eye and Strong Wide-Angle-Lens-Distortion from Single Images," *Proc. Int'l Conf. Image Processing,* pp. 225-228, 2001.

[5] M. Bujnak, Z. Kukelova, and T. Pajdla, "A General Solution to the P4P Problem for Camera with Unknown Focal Length," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.

[6] M. Bujnak, Z. Kukelova, and T. Pajdla, "Robust Focal Length Estimation by Voting in Multiview Scenes," *Proc. Asian Conf. Computer Vision,* 2009.

[7] M. Bujnak, Z. Kukelova, and T. Pajdla, "New Efficient Solution to the Absolute Pose Problem for Camera with Unknown Focal Length and Radial Distortion," *Proc. Asian Conf. Computer Vision,* 2010.

[8] M. Byröd, K. Josephson, and K. Åström, "Improving Numerical Accuracy of Gröbner Basis Polynomial Equation Solver," *Proc. IEEE Int'l Conf. Computer Vision,* 2007.

[9] M. Byröd, Z. Kukelova, K. Josephson, T. Pajdla, and K. Åström, "Fast and Robust Numerical Solutions to Minimal Problems for Cameras with Radial Distortion," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.

[10] M. Byröd, M. Brown, and K. Åström, "Minimal Solutions for Panoramic Stitching with Radial Distortion," *Proc. 20th British Machine Vision Conf.,* 2009.

[11] D. Claus and A. Fitzgibbon, "A Rational Function Lens Distortion Model for General Cameras," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 213-219, 2005.

[12] D. Cox, J. Little, and D. O'Shea, *Using Algebraic Geometry,* second ed. Springer Verlag, 2005.

[13] D. Devernay and O. Faugeras, "Straight Lines Have to Be Straight," *Machine Vision and Applications* , vol. 13, no. 1, pp. 14-24, 2001.

[14] F. Du and J.M. Brady, "Self-Calibration of the Intrinsic Parameters of Cameras for Active Vision Systems," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 477-482, 1993.

[15] M.A. Fischler and R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. ACM,* vol. 24, no. 6, pp. 381-395, 1981.

[16] A. Fitzgibbon, "Simultaneous Linear Estimation of Multiple View Geometry and Lens Distortion," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* pp. 125-132, 2001.

[17] J.-C. Faugère, "A New Efficient Algorithm for Computing Gröbner Bases (f_4)," *J. Pure and Applied Algebra,* vol. 139, nos. 1-3, pp. 61-88, 1999.

[18] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete Solution Classification for the Perspective-Three-Point Problem," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 8, pp. 930-943, Aug. 2003.

[19] C. Geyer and H. Stewenius, "A Nine-Point Algorithm for Estimating Para-Catadioptric Fundamental Matrices," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2007.

[20] D.B. Gennery, "Generalized Camera Calibration Including Fish-Eye Lenses," *Int'l J. Computer Vision* vol. 68, no. 3, pp. 239-266, 2006.

[21] R. Hartley and S. Kang, "Parameter-Free Radial Distortion Correction with Centre of Distortion Estimation," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1834-1841, 2005.

[22] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge Univ. Press, 2003.

[23] K. Josephson, M. Byröd, and K. Åström, "Pose Estimation with Radial Distortion and Unknown Focal Length," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2009.

[24] S. Kang, "Catadioptric Self-Calibration," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2000.

[25] S. Kang, "Radial Distortion Snakes," *Proc. IAPR Workshop Machine Vision Applications,* pp. 603-606, 2000.

[26] K. Kuehnle and E. Mayr, "Exponential Space Computation of Groebner Bases," *Proc. Int'l Symp. Symbolic and Algebraic Computation,* 1996.

[27] Z. Kukelova and T. Pajdla, "A Minimal Solution to the Autocalibration of Radial Distortion," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2007.

[28] Z. Kukelova and T. Pajdla, "Two Minimal Problems for Cameras with Radial Distortion," *Proc. Workshop Omnidirectional Vision, Camera Networks and Non-Classical Cameras,* 2007.

[29] Z. Kukelova, M. Bujnak, and T. Pajdla, "Polynomial Eigenvalue Solutions to the 5-Pt and 6-Pt Relative Pose Problems," *Proc. British Machine Vision Conf.,* 2008.

[30] Z. Kukelova, M. Bujnak, and T. Pajdla, "Automatic Generator of Minimal Problem Solvers," *Proc. European Conf. Computer Vision,* Oct. 2008.

[31] 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,* vol. 114, no. 2, pp. 234-244, Feb. 2010.

[32] H. Li, "A Simple Solution to the Six-Point Two-View Focal-Length Problem," *Proc. European Conf. Computer Vision,* pp. 200-213, 2006.

[33] H. Li and R. Hartley, "A Non-Iterative Method for Correcting Lens Distortion from Nine-Point Correspondences," *Proc. Workshop Omnidirectional Vision, Camera Networks and Non-Classical Cameras,* 2005.

[34] H. Li and R. Hartley, "Five-Point Motion Estimation Made Easy," *Proc. Int'l Conf. Pattern Recognition,* pp. 630-633, 2006.

[35] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust Wide-Baseline Stereo from Maximally Stable Extremal Regions," *Image and Vision Computing,* vol. 22, no. 10, pp. 761-767, 2004.

[36] B. Micusik and T. Pajdla, "Structure from Motion with Wide Circular Field of View Cameras," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 7, pp. 1135-1149, July 2006.

[37] D. Nister, "An Efficient Solution to the Five-Point Relative Pose," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 26, no. 6, pp. 756-770, June 2004.

[38] S. Ramalingam and P. Sturm, "Minimal Solutions for Generic Imaging Models," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2008.

[39] *Manual of Photogrammetry,* C. Slama, ed., fourth ed. Am. Soc. of Photogrammetry, 1980.

[40] R. Steele and C. Jaynes, "Overconstrained Linear Estimation of Radial Distortion and Multi-View Geometry," *Proc. European Conf. Computer Vision,* 2006.
[41] G.P. Stein, "Accurate Internal Camera Calibration Using Rotation, with Analysis of Sources of Error," *Proc. IEEE Int'l Conf. Computer Vision,* 1995.
[42] G. Stein, "Lens Distortion Calibration Using Point Correspondences," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 600-602, 1997.
[43] H.J. Stetter, *Numerical Polynomial Algebra.* SIAM, 2004.
[44] H. Stewenius, "Gröbner Basis Methods for Minimal Problems in Computer Vision," PhD thesis, Lund Univ., 2005.
[45] H. Stewenius, C. Engels, and D. Nister, "Recent Developments on Direct Relative Orientation," *ISPRS J. Photogrammetry and Remote Sensing,* vol. 60, pp. 284-294, 2006.
[46] H. Stewenius, D. Nister, F. Kahl, and F. Schaffalitzky, "A Minimal Solution for Relative Pose with Unknown Focal Length," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* pp. 789-794, 2005.
[47] H. Stewenius, D. Nister, M. Oskarsson, and K. Astrom, "Solutions to Minimal Generalized Relative Pose Problems," *Proc. Workshop Omnidirectional Vision, Camera Networks and Non-Classical Cameras,* 2005.
[48] R. Strand and E. Hayman, "Correcting Radial Distortion by Circle Fitting," *Proc. British Machine Vision Conf.,* 2005.
[49] J.P. Tardif, P. Sturm, and S. Roy, "Plane-Based Self-Calibration of Radial Distortion," *Proc. 11th IEEE Int'l Conf. Computer Vision,* Oct. 2007.
[50] S. Thirthala and M. Pollefeys, "Multi-View Geometry of 1D Radial Cameras and Its Application to Omnidirectional Camera Calibration," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1539-1546, 2005.
[51] S. Thirthala and M. Pollefeys, "The Radial Trifocal Tensor: A Tool for Calibrating the Radial Distortion of Wide-Angle Cameras," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* pp. 321-328, 2005.
[52] P.H.S. Torr and D.W. Murray, "The Development and Comparison of Robust Methods for Estimating the Fundamental Matrix," *Int'l J. Computer Vision,* vol. 24, no. 3, pp. 271-300, 1997.
[53] C. Traverso, "Gröbner Trace Algorithms," *Proc. Int'l Symp. Symbolic and Algebraic Computation,* pp. 125-138, 1988.
[54] R. Tsai, "A Versatile Camera Calibration Technique for High-accuracy 3D Machine Vision Metrology using Off-the-Shelf TV Cameras and Lenses," *IEEE J. Robotics and Automation,* vol. 3, no. 4, pp. 323-344, Aug. 1987.
[55] Z. Zhang, "On the Epipolar Geometry between Two Images with Lens Distortion," *Proc. Int'l Conf. Pattern Recognition,* 1996.
[56] Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 11, pp. 1330-1334, Nov. 2000.

Zuzana Kukelova received the master's degree in mathematics from the Faculty of Mathematics, Physics and Informatics, Comenius University in Bratislava in 2005. She is working toward the PhD degree at the Center for Machine Perception, which is a part of the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague. Since her master's studies, she has been interested in algebraic geometry and computer vision, where she focuses on minimal problems and methods for solving systems of polynomial equations. She is a member of the IEEE.

Tomas Pajdla received the MSc and PhD degrees from the Czech Technical University in Prague. He works in geometry and algebra of computer vision and robotics with emphasis on nonclassical cameras, 3D reconstruction, and industrial vision. He contributed to introducing epipolar geometry of panoramic cameras, non-central camera models generated by linear mappings, generalized epipolar geometries, and to developing solvers for minimal problems in structure from motion. He coauthored works awarded the best paper prizes at OAGM '98 and BMVC '02. He is a member of the IEEE.