Using Sparse Elimination for Solving Minimal Problems in Computer Vision

Janne Heikkilä
Center for Machine Vision and Signal Analysis
University of Oulu, Finland
janne.heikkila@oulu.fi

Abstract

Finding a closed form solution to a system of polynomial equations is a common problem in computer vision as well as in many other areas of engineering and science. Gröbner basis techniques are often employed to provide the solution, but implementing an efficient Gröbner basis solver to a given problem requires strong expertise in algebraic geometry. One can also convert the equations to a polynomial eigenvalue problem (PEP) and solve it using linear algebra, which is a more accessible approach for those who are not so familiar with algebraic geometry. In previous works PEP has been successfully applied for solving some relative pose problems in computer vision, but its wider exploitation is limited by the problem of finding a compact monomial basis. In this paper, we propose a new algorithm for selecting the basis that is in general more compact than the basis obtained with a state-of-the-art algorithm making PEP a more viable option for solving polynomial equations. Another contribution is that we present two minimal problems for camera self-calibration based on homography, and demonstrate experimentally using synthetic and real data that our algorithm can provide a numerically stable solution to the camera focal length from two homographies of unknown planar scene.

1. Introduction

Many camera pose estimation and calibration problems boil down to solving a system of polynomial equations. These are often so-called minimal problems, where the camera parameters are computed from a minimal number of constraints so that there are essentially as many unknowns as equations, but the relationship between the unknown variables and the measurements follows a polynomial model that makes the dependence nonlinear and difficult to solve by means of linear algebra. Such minimal problems include for example, the classical P3P (Perspective-Three-Point) problem for a calibrated camera where an image of three points with known distances is sufficient to compute the camera pose, but it requires solving a system of three quadratic equations in three variables [9]. Another classical example is the five-point problem that allows finding the relative pose between two views from an unknown scene using five point correspondences. Nistér [17] converted the resulting system of polynomial equations to a tenth degree univariate polynomial that can be efficiently solved using standard numerical techniques. The relative pose problem has been modified in various studies to incorporate also unknown camera parameters that enable the use of uncalibrated cameras and makes it a self-calibration problem. To mention few of them, Stewenius et al. [20] used six point correspondences to solve the relative pose together with the focal length. Fitzgibbon [8] augmented the fundamental matrix estimation to include one term of radial lens distortion, and solved them from 9 point correspondences. Kukelova and Pajdla [15] used an additional constraint to solve the same problem from 8 point correspondences, and Jiang et al. [11] added still one constraint and they were able to solve the problem from 7 point correspondences. A comprehensive list of minimal problems in computer vision and related papers can be found in [18].

Planar objects are commonly used for estimating the camera pose and intrinsic parameters. Well-known Zhang's calibration method [23] provides a closed form solution to the calibration problem from images of a known planar target. Also, OpenCV and Matlab include tools to perform calibration with a similar setup. Despite of the extensive number of minimal problems introduced in recent years, it is surprising that homography has not been much considered in this context, and there are only few related works. Minimal solutions to panorama stitching in [1] and [2] assume that the camera centers coincide which reduces the motion to pure rotation. Methods for decomposing a homography into rotation, translation, and surface normal parameters have been proposed e.g. in [7] and [24]. Saurer et al. [19] consider a minimal solution to a 3-point plus a common direction relative pose problem using homography. Recently, Kukelova et al. [14] have presented two algorithms for estimating the homography between two cameras with different radial distortions. However, none of these works address the problem of solving the camera focal length from images of an unknown planar target, which is the homography-based minimal problem presented in this paper.

The most common approach for solving minimal problems in computer vision and corresponding systems of polynomial equations is to use Gröbner basis techniques. One drawback of this approach is that when the polynomial degrees are high, it often suffers from numerical inaccuracies. To address this problem, for example, Byröd et al. [3] have proposed a generalization of the Gröbner basis method for improving the numerical stability. Another limitation of this approach is that implementing a Gröbner basis solver for a given problem requires expertise in algebraic geometry because the solver needs to be handcrafted in practice to make it efficient. Because of the complicated theory this approach is often beyond the reach of non-experts. An alternative approach that is also commonly used for solving polynomial equations is multipolynomial resultant, which provides an efficient tool for eliminating variables from multiple equations, and solving the remaining variable as a root of a univariate polynomial [4]. However, there are some limitations that make resultants less useful for engineering applications. For classical multipolynomial resultants such as the Macaulay resultant, most of the polynomial coefficients need to be non-zero, the roots distinct and there should be no solutions at infinity. This problem can often be avoided by using a sparse resultant [6] that works also for polynomials with several zero coefficients. Another disadvantage is that after elimination the remaining univariate polynomial is a determinant of a matrix that often has high dimensions. Because a determinant of an $N \times N$ matrix has $N!$ terms, finding the roots of the remaining polynomial can easily become computationally infeasible or unstable.

In addition to the Gröbner basis techniques and resultants, systems of polynomial equations can be often solved using eigenvalues and eigenvectors. One approach is to convert the classical multipolynomial resultant to a standard eigenvalue problem [5],[4] which however works only with dense polynomials. In [8] the minimal problem of computing the radial distortion coefficient was expressed as a quadratic polynomial eigenvalue problem and later it was extended in [16] to include an additional constraint. A resultant-based algorithm for transforming a system of polynomial equations to a polynomial eigenvalue problem (PEP) was proposed in [13] that enabled solving several minimal relative pose problems using linear algebra. However, this algorithm has an inherent property of leading to unnecessarily high dimensional vector spaces and spurious roots that make the algorithm numerically unstable when solving sparse systems of polynomials with high degrees.

To overcome the problems related to the algorithm presented in [13], we propose a new algorithm based on *sparse elimination theory* that provides more stable solutions to sparse systems that are the most typical cases in practical applications. In addition, we demonstrate the applicablility of our algorithm in two new minimal problems that have higher number of solutions than typical relative pose problems previously presented in the literature.

2. Polynomial eigenvalue problems

Polynomial eigenvalue problem (PEP) is an extension of the standard eigenvalue problem $(\mathbf{C} - \lambda\mathbf{I})\mathbf{v} = \mathbf{0}$ to a system of polynomials represented with the matrix equation:

$$(\mathbf{C}_0 + \mathbf{C}_1\lambda + \mathbf{C}_2\lambda^2 + \cdots + \mathbf{C}_l\lambda^l)\mathbf{v} = \mathbf{0}, \qquad (1)$$

where l is the highest degree of the polynomials in the variable λ that we want to solve, $\mathbf{v}$ is a vector of monomials in other variables than λ, and $\mathbf{C}_0, \ldots, \mathbf{C}_l$ are $m \times m$ square matrices that contain the coefficients of the polynomials. This equation can be converted to a generalized eigenvalue problem

$$\mathbf{A}\mathbf{u} = \lambda\mathbf{B}\mathbf{u}, \qquad (2)$$

where

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\mathbf{C}_0 & -\mathbf{C}_1 & -\mathbf{C}_2 & \cdots & -\mathbf{C}_{l-1} \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{C}_l \end{bmatrix}, \mathbf{u} = \begin{bmatrix} \mathbf{v} \\ \lambda\mathbf{v} \\ \vdots \\ \lambda^{l-1}\mathbf{v} \end{bmatrix}.$$

Since most of the mathematical software libraries and packages can solve this problem it becomes easy to find all the roots for λ. The eigenvector $\mathbf{u}$ contains the solutions of the monomials that exist in the polynomials, and therefore, one can extract the roots of the remaining variables by computing suitable ratios between individual elements of $\mathbf{u}$.

The most difficult part of converting the system of polynomials to PEP is to determine the monomials in $\mathbf{v}$ and consequently the matrices $\mathbf{C}_0, \ldots, \mathbf{C}_l$. Given n polynomials one can easily construct $n \times m$ matrices and the corresponding $\mathbf{v}$ satisfying (1), but usually $n < m$ which results in an underdetermined system of linear equations that cannot be solved. The trick emplyed in [13] is to generate new equations by multiplying the initial equations with monomials produced by their algorithm. Some of these new equations may be linearly independent from the initial equations, which then enables constructing a fully determined system. Notice that this procedure also increases the number of monomials in $\mathbf{v}$ and hence the dimensions of the coefficient matrices. In [13] they used the classical Macaulay resultant formulation for creating the set of basis monomials in $\mathbf{v}$. Because the Macaulay resultant is designed for dense homogeneous polynomials, it is not guaranteed to produce a basis that is linearly independent. Therefore, they proposed a small modification to the resultant-based approach that gives a higher number of polynomial equations that increases the chances to get a linearly independent set of basis monomials. However, for larger systems of polynomials the basis generated by this method becomes huge, because the Macaulay resultant is based on the assumption that the number of the solutions obtained is maximal for the given degrees of the polynomials. According to Bézout's theorem [4] the maximum number of solutions is $d_1 \cdot d_2 \cdots d_n$, where d_i is the degree of the polynomial f_i. The set of basis monomials contain all the monomials with total degree of $d = \sum(d_i - 1) + 1$. One can easily see that the number of monomials increases exponentially. Therefore, this approach is feasible only for small systems of equations and low polynomial degrees. Next, we present a method based on sparse elimination that exploits the sparsity of the general polynomials, and produces smaller monomial bases and coefficient matrices enabling solutions to problems that have been previously intractable.

3. Determining basis monomials

Most of the polynomial equations encountered in computer vision are sparse, and therefore classical multivariate resultants are not well-suited for generating the basis monomials. Sparse elimination theory [21],[5] has been developed to deal with general polynomials that have many zero coefficients. The benefit of the sparsity is that the resultants obtained have much smaller dimensions than the classical resultants. Therefore, instead of selecting all the monomials of a certain total degree we can get a significantly smaller set of monomials by using the tools provided by the sparse elimination theory.

Let $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ be a set of unknown variables that we want to solve from n multivariate polynomials

$$f_1(\mathbf{x}) = f_2(\mathbf{x}) = \cdots = f_n(\mathbf{x}) = 0 \tag{3}$$

defined by

$$f_i(\mathbf{x}) = \sum_{j=1}^{s_i} c_{ij}\mathbf{x}^{\mathbf{a}_{ij}}, \tag{4}$$

where $\mathbf{x}^{\mathbf{a}_{ij}} = x_1^{\alpha_{ij1}} x_2^{\alpha_{ij2}} \cdots x_n^{\alpha_{ijn}}$ are the monomials corresponding to the non-zero coefficients c_{ij}. Let $\mathcal{A}_i = \{\mathbf{a}_{i1}, \ldots, \mathbf{a}_{is_i}\} \subset \mathbb{Z}_+^n$ denote the exponent vectors of all the monomials in f_i that is also called the support of f_i. Next we introduce few concepts from algebraic geometry [4] that are needed to formulate our method.

Definition 1: The *Newton polytope* of f_i is the convex hull of the support $\mathcal{A}_i$ denoted by $P_i = \mathrm{Conv}(\mathcal{A}_i) \subset \mathbb{R}^n$. The volume of P_i is denoted by $\mathrm{Vol}_n(P_i)$.

Notice that in the low dimensional cases when $n = 1, 2$ or 3, the Newton polytope represents a line, polygon or polyhedron, respectively. Clearly, the way how the volume $\mathrm{Vol}_n(P_i)$ is computed depends on n. For example, $\mathrm{Vol}_1(P_i)$ is the length of the line, and $\mathrm{Vol}_2(P_i)$ is the area of the polygon.

Definition 2: The *Minkowski sum* of two convex polytopes P_i and P_j is the convex polytope

$$P_{ij} = P_i + P_j = \{p_i + p_j | p_i \in P_i, p_j \in P_j\} \subset \mathbb{R}^n.$$

Using the Minkowski sum (also known as dilation) we can aggregate the Newton polytopes of individual polynomials f_i to form combined supports. It is also needed for defining the mixed volume.

Definition 3: Given convex polytopes $P_1, \ldots, P_n \subset \mathbb{R}^n$ there is a real-valued function called *mixed volume* that can be computed as

$$MV_n(P_1, \ldots, P_n) = \sum_{k=1}^{n} (-1)^{n-k} \sum_{\substack{I \subset \{1,\ldots,n\} \\ |I|=k}} \mathrm{Vol}_n\left(\sum_{i \in I} P_i\right). \tag{5}$$

In high-dimensional cases computing the mixed volume using (5) can be time consuming. There are faster algorithms that use a so called *mixed subdivision* of the Minkowski sum, and one can also find their software implementations from the Internet, but in the cases discussed in this paper $n \leq 4$ and using (5) is still tractable. The following theorem is the reason why we introduced the mixed volume.

Theorem 1 (Bernstein's Theorem): Given the polynomials $f_1, \ldots f_n$ over $\mathbb{C}$ with finitely many common zeroes in $(\mathbb{C}^*)^n$, where $\mathbb{C}^* = \mathbb{C} \setminus \{0\}$, let P_i be the Newton polytope of f_i in $\mathbb{R}^n$. Then the number of solutions of the f_i in $(\mathbb{C}^*)^n$ is bounded above by the mixed volume $MV_n(P_1, \ldots, P_n)$. For generic choices of the coefficients c_{ij} the number of common solutions is exactly $MV_n(P_1, \ldots, P_n)$.

The proof of the theorem can be found from [4]. Bernstein's theorem is an important result of the sparse elimination theory that gives us a tool for calculating the maximum number of the roots in advance without knowing the numerical values of the coefficients. What we need is only the exponent vectors $\mathbf{a}_{ij}$ of the monomials, i.e., supports $\mathcal{A}_i$. This also determines the minimum size of the monomial basis as we will see later.

Next we discuss about finding the basis monomials for the polynomial eigenvalue problem, i.e., the elements of $\mathbf{v}$ in (1). Sparse elimination provides the tools for constructing sparse resultants that generalize the classical multivariate resultant. While the degree of the classical multivariate resultant is determined by Bézout's theorem (i.e. $d_1 \cdot d_2 \cdots d_n$), the degree of the sparse resultant comes from Bernstein's theorem which is the mixed volume. These two types of resultants coincide only when all Newton polytopes are n-simplices scaled by the total degree of the respective polynomials [4]. Otherwise the degree of the sparse resultant is smaller, which also means that the matrix constructed from the coefficients c_{ij} has smaller dimensions. Furthermore, it is necessary that the matrix has full rank and its determinant vanishes only when the equations have a common solution. It often happens that the multivariate resultant is rank-deficient if the polynomials have zero coefficients, and thus it fails to provide a solution. In order to have a full rank, it becomes necessary to select the basis monomials of the sparse resultant carefully using, e.g., the Lift-Prune algorithm proposed by Emiris & Canny [6]. The main disadvantage of the resultant-based approach for solving the polynomial equations is that it requires computing the determinant of a matrix which often has high dimensions. Because the determinant of an $N \times N$ matrix has $N!$ terms, solving the unknowns from the resultant often becomes computationally infeasible even for relatively small problems. For example, if the dimension of the coefficient matrix for the sparse resultant is 10×10, the resultant is a factor of an expression that has more than 3.6 million terms. In such cases finding the solution via PEP is much more efficient.

A sparse resultant to a system of n equations is computed in $n-1$ variables, which means that one of the variables of our original problem (3) needs to be hidden to the coefficient field. The resultant obtained is then a univariate polynomial of the hidden variable which can be solved by finding the roots of this polynomial. Without loss of generality we can decide to solve the first variable x_1 that is then treated as a coefficient in the polynomials

$$f_i'(\tilde{\mathbf{x}}) = \sum_{j=1}^{s_i'} c_{ij}' \tilde{\mathbf{x}}^{\mathbf{a}_{ij}'}, \tag{6}$$

where $c_{ij}' = \sum_{\alpha_{ij1}} c_{ij} x_1^{\alpha_{ij1}}$, $\tilde{\mathbf{x}} = \{x_2, \ldots, x_n\}$ and $\mathbf{a}_{ij}' = (\alpha_{ij2}, \ldots, \alpha_{ijn}) \in \mathcal{A}_i'$ are $n-1$ dimensional support vectors with $|\mathcal{A}_i'| = s_i'$. The first step is to create the Newton polytopes $P_1', \ldots, P_n' \subset \mathbb{R}^{n-1}$ corresponding to the modified system (6), and compute the Minkowski sum $P = P_1' + \cdots + P_n'$. The set of basis monomials $\mathcal{S}$ for the sparse resultant is then obtained from

$$\mathcal{S} = \mathbb{Z}^{n-1} \cap (P + \mathbf{d}), \tag{7}$$

where $\mathbb{Z}^{n-1}$ defines a square lattice with integer points, and $\mathbf{d} \in \mathbb{R}^{n-1}$ is a small translation vector that displaces P slightly so that the lattice points lie in the interiors of the convex polytope [6, 4]. In practice, the elements of $\mathbf{d}$ can be randomly selected from $\{-\epsilon, 0, \epsilon\}$ where $\epsilon \in \mathbb{Q}$ is a small rational number.

Sparse resultants need to be of full rank in order to have a non-zero determinant. In our case, the only strict requirement is that $\mathbf{C}_0, \ldots, \mathbf{C}_l$ in (1) must be square matrices. Notice that this is a looser condition than in [13] where they assumed that either $\mathbf{C}_l$ or $\mathbf{C}_0$ must be of full rank and invertible. Here this not necessary, but in some cases rank-deficiency may lead to numerical instability with the eigenvalue solver. Because the PEP in (1) is defined for one unknown variable λ which is then computed as an eigenvalue of (2) we need to choose this variable from $x_1, \ldots, x_n$. This is exactly the same situation as with the sparse resultant, and therefore, we decide again without loss of generality that $\lambda \equiv x_1$, and we hide x_1 to the coefficient field which then results in the modified system (6).

Due to the relaxed requirements, we can try to find a smaller set of basis monomials than (7) defined for the sparse resultant. The lower bound is determined by Bernstein's theorem which gives the maximum number of the common roots for the polynomials denoted by $r \equiv MV_n(P_1, \ldots, P_n)$. It should be noticed that the mixed volume is computed for the original system (3). The eigenvector $\mathbf{u}$ in (2) has the same dimension as the maximum number of unique eigenvalues i.e. possible roots of the system. The length of $\mathbf{u}$ is clearly lm which gives us the bound

$$m \geq \frac{r}{l}. \tag{8}$$

Hence, it is sufficient to find a set of support vectors $\mathcal{B}$ for the basis monomials where $|\mathcal{B}| \geq r/l$.

Algorithm 1 summarizes the procedure for constructing $\mathcal{B}$ based on the previous discussion. It generates several putative sets of support vectors for the basis and selects the smallest set among these candidates. It also returns a set $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_n\}$ where $\mathcal{T}_i \neq \emptyset$ are subsets of vectors that can be used to construct the coefficient matrices $\mathbf{C}_0, \ldots, \mathbf{C}_l$. These vectors are first converted to n sets of monomials $\mathcal{M}_i = \{\tilde{\mathbf{x}}^{\mathbf{t}} | \ \forall \mathbf{t} \in \mathcal{T}_i\}$, and the monomials are multiplied with the original equations f_i which then results in n sets of new equations $\mathcal{E}_i = \{\tilde{\mathbf{x}}^{\mathbf{t}} f_i(\mathbf{x}) | \ \forall \tilde{\mathbf{x}}^{\mathbf{t}} \in \mathcal{M}_i\}$. These equations are converted to a matrix form (1), which then directly gives us the coefficient matrices $\mathbf{C}_0, \ldots, \mathbf{C}_l$. The total number of new equations $\sum_i |\mathcal{E}_i|$ is greater or equal to the number of the basis monomials, which means that the coefficient matrices have at least as many rows as columns. If there are more rows than columns, one can choose m rows that minimize the condition number, and discard the remaining rows. It may also happen that the most compact basis does not work, and in that case one could try the next candidate produced by the algorithm.

There are often monomials (or vectors) in $\mathcal{B}$ that do not contribute to the equations. Such monomials may cause instability to the eigenvalue solver, and they need to be removed. In [13] they call them "parasitic" zero eigenvalues, and they propose to convert the generalized eigenvalue problem to a standard eigenvalue problem so that one can easily identify and remove these monomials as they correspond to zero columns of the matrix to be decomposed. The drawback is that either $\mathbf{C}_0$ or $\mathbf{C}_l$ need to be of full rank, which then causes extra constraints to selection of the basis monomials. Hence, we propose here a simple strategy for finding these zero monomials. First, we need to specialize the coefficient matrices with some random numerical val-

Algorithm 1 Generate basis monomials

```
Input: $\mathcal{A}'_1, \dots, \mathcal{A}'_n, r, l$
Output: $\mathcal{B}, \mathcal{T}$
 1: Create Newton polytopes $P'_i \leftarrow \mathrm{Conv}(\mathcal{A}'_i) \subset \mathbb{R}^{n-1}$ for
    $i = 1, \dots, n$, and a unit $(n-1)$-simplex $P'_0 \subset \mathbb{R}^{n-1}$.
 2: Create a list of index sets:
    $K \leftarrow [\{k_0, \dots, k_i\} \mid \forall i = 0, \dots, n ; k_0, \dots, k_i \in \{0, \dots, n\} ; k_{j+1} > k_j]$.
 3: Create a list of displacement vectors:
    $\Delta \leftarrow [(\delta_1, \dots, \delta_{n-1}) \mid \forall \delta_1, \dots, \delta_{n-1} \in \{-\epsilon, 0, \epsilon\}]$.
 4: Initialize $\mathcal{B} \leftarrow \emptyset$, $\mathcal{T} \leftarrow \emptyset$ and $N \leftarrow \infty$.
 5: for $I$ in $K$ do
 6:     Compute Minkowski sum $Q \leftarrow \sum_{k \in I} P'_k$.
 7:     for $\mathbf{d}$ in $\Delta$ do
 8:         Create a putative basis $B \leftarrow \mathbb{Z}^{n-1} \cap (Q + \mathbf{d})$.
 9:         if $|B| \geq \frac{r}{l}$ AND $|B| < N$ then
10:             Find the sets of vectors:
                    $T_i \leftarrow \{\mathbf{t} | \mathbf{t} \in \mathbb{Z}^{n-1}_+, \mathcal{A}'_i + \mathbf{t} \subset B\}$
                    for $i = 1, \dots, n$.
11:             if $\sum_i |T_i| \geq |B|$ AND $\min(|T_i|) > 0$ then
12:                 $\mathcal{B} \leftarrow B, \mathcal{T} \leftarrow \{T_i\}_{i=1,\dots,n}$, and
                    $N \leftarrow |B|$.
13:             end if
14:         end if
15:     end for
16: end for
```

ues as c_{ij}. Using these values we construct the matrices $\mathbf{A}$ and $\mathbf{B}$ in (2), and compute the singular value decomposition $\mathbf{B} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$. Next we perform a unitary transformation

$$\mathbf{A}' = \mathbf{U}^\top \mathbf{A} \mathbf{V}, \tag{9}$$

and find the zero columns of $\mathbf{A}'$. These columns correspond to the zero monomials and they can be removed from $\mathbf{A}$ and $\mathbf{B}$. The rows with the same indices are also removed so that the matrices will remain square. This procedure might need to be repeated few times to find all zero monomials. However, one should notice that the elimination is performed offline when designing the solver, and there is no need to do it runtime once the zero polynomials have been identified.

4. Planar self-calibration

A standard approach for geometric camera calibration is to use a known checker board pattern printed on a planar surface. To demonstrate the applicability of our algorithm, we present two minimal problems for solving the camera focal length from two homographies corresponding to three images where the patterns are unknown, which makes this a self-calibration problem. We consider the following cases: 1) a constant focal length and 2) two different focal lengths. The resulting polynomials can be converted to PEPs using Algorithm 1 and solved efficiently, for example, with Matlab or some other software package or library capable of computing generalized eigenvalues.

Let two 3D vectors $\mathbf{a}$ and $\mathbf{b}$ span a plane so that they are both orthogonal and of equal length fulfilling the constraints

$$\mathbf{a}^\top \mathbf{b} = 0 \quad \text{and} \quad \mathbf{a}^\top \mathbf{a} - \mathbf{b}^\top \mathbf{b} = 0. \tag{10}$$

If $|\mathbf{a}| = |\mathbf{b}| = 1$ the normal vector of the plane is defined by $\mathbf{n} = \mathbf{a} \times \mathbf{b}$. In order to express the vectors $\mathbf{a}$ and $\mathbf{b}$ in terms of the normal vector $\mathbf{n}$ we can choose

$$\mathbf{a} = \mathbf{n} \times \mathbf{e} \quad \text{and} \quad \mathbf{b} = \mathbf{n} \times \mathbf{a}, \tag{11}$$

where $\mathbf{e}$ is a unit vector not parallel to $\mathbf{n}$. For simplicity, we select $\mathbf{e} = [1, 0, 0]^\top$. We use parametrization $\mathbf{n} = [n_x, n_y, 1]^\top / \sqrt{n_x^2 + n_y^2 + 1}$, and we can now express $\mathbf{a}$ and $\mathbf{b}$ in variables n_x and n_y. Further assuming that $\mathbf{a}$ and $\mathbf{b}$ are represented in the camera coordinate frame of the reference view we can convert them to the image coordinates using

$$\hat{\mathbf{a}}_0 = \mathbf{K}_0 \mathbf{a} \quad \text{and} \quad \hat{\mathbf{b}}_0 = \mathbf{K}_0 \mathbf{b}, \tag{12}$$

where $\mathbf{K}_0$ is the intrinsic camera matrix for the reference camera. Because one can often assume with reasonable accuracy that the principal point of the camera is in the center of the image, the pixel aspect ratio is 1, and lens distortion is negligible, we limit ourselves to the case where we have only one intrinsic parameter, the focal length λ_0, that leads to the camera matrix $\mathbf{K}_0 = \mathrm{diag}(\lambda_0, \lambda_0, 1)$.

The mapping from the reference image to the i^{th} image is described by the homography $\mathbf{H}_i$. After back-projecting to the 3D space the corresponding vectors are obtained from

$$\mathbf{a}_i = \mathbf{K}_i^{-1} \mathbf{H}_i \mathbf{K}_0 \mathbf{a} \quad \text{and} \quad \mathbf{b}_i = \mathbf{K}_i^{-1} \mathbf{H}_i \mathbf{K}_0 \mathbf{b}, \tag{13}$$

where $\mathbf{K}_i = \mathrm{diag}(\lambda_i, \lambda_i, 1)$ and λ_i is the focal length of the i^{th} camera. Because orthogonality and equality in the length should hold in each frame we have the following self-calibration constraints expressed by two polynomials

$$f_{1,i}(\lambda_0, \lambda_i, n_x, n_y) = \mathbf{a}_i^\top \mathbf{b}_i = 0 \tag{14}$$
$$f_{2,i}(\lambda_0, \lambda_i, n_x, n_y) = \mathbf{a}_i^\top \mathbf{a}_i - \mathbf{b}_i^\top \mathbf{b}_i = 0. \tag{15}$$

Herrera et al. [10] used similar constraints for planar self-calibration but they solved the camera parameters with non-linear minimization. The solver was initialized by assuming that the reference view is fronto-parallel when it becomes easy to compute an initial value for the focal length. In this paper, we do not make such assumptions, and no initialization is needed. There are now $3 + k$ unknowns $\lambda_0, \dots, \lambda_k$, n_x and n_y, and two equations which means that we cannot solve the problem from a single homography, but we need at least two homographies ($i = 1, 2$) that lead to four equations with four unknowns. This is true also in general, because one homography provides only 8 constraints to calibration. Five constraints are needed for the camera pose (3 for rotation and 2 for translation up to scale), the normal of the plane $\mathbf{n}$ requires two constraints and one

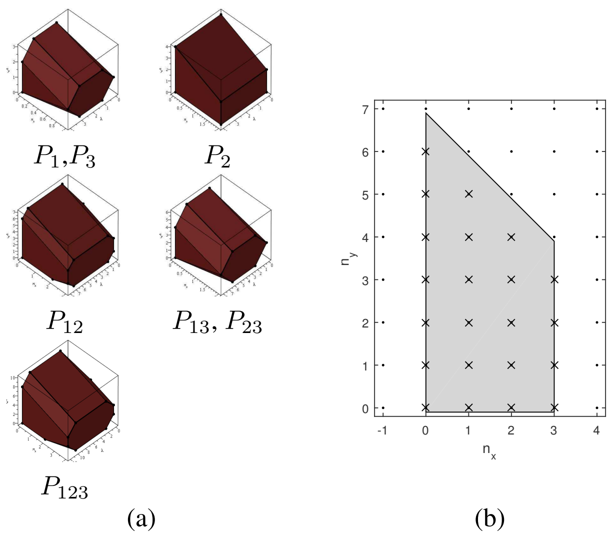


Figure 1. Components of the mixed volume in the case of equal focal lengths (a). The exponent vectors of the basis obtained as $\mathbb{Z}^2 \cap ((P_1' + P_2') + (0, -\epsilon))$ are shown with crosses inside the shaded convex polytope (b).

is needed for the perspective scaling factor. There are algorithms such as [24] available for decomposing a single homography to the pose parameters. In order to estimate intrinsic camera parameters we need more constraints.

In our case, the minimal problem consists of two homographies that enables us to consider two calibration problems. In the first problem, we assume that all cameras have the same focal length ($\lambda = \lambda_0 = \lambda_1 = \lambda_2$). Because there are now three unknowns only three constraints are needed. We can see that (14) has fewer terms than (15), and therefore, we select the equations $f_{1,1}$, $f_{2,1}$ and $f_{1,2}$. The corresponding numbers of monomials are 15, 28, and 15 with the total degrees of 6, 8, and 6, respectively. All these polynomials are sparse, and the total degrees are higher than in many of the minimal problems considered in computer vision. The degree of λ is $l = 4$ for all equations. The first thing to do is to compute the mixed volume for obtaining the number of common solutions to these equations. We use Maple's `PolyhedralSets` package to construct the convex polytopes from the Minkowski sums of the polynomial supports (notice that $P_i + P_j = \text{Conv}(\mathcal{A}_i) + \text{Conv}(\mathcal{A}_j) = \text{Conv}(\mathcal{A}_i + \mathcal{A}_j)$). The components of the mixed volume are illustrated in Fig. 1 (a), and after computing their volumes and substituting them into (5) we get $r = 22/3 + 68/3 + 22/3 - 117 - 176/3 - 117 + 976/3 = 70$ which is the upper bound for the number of solutions.

Next, we use Algorithm 1 to construct the monomial basis. The index list we get is $K = [\{0\}, \{1\}, \{2\}, \{3\}, \{0,1\}, \{0,2\}, \{0,3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{0,1,2\}, \{0,1,3\}, \{0,2,3\}, \{1,2,3\}, \{0,1,2,3\}]$ and the list of displacement vectors $\Delta = [(-\epsilon, -\epsilon), (-\epsilon, 0), (-\epsilon, \epsilon), (0, -\epsilon), (0, 0), (0, \epsilon), (\epsilon, -\epsilon), (\epsilon, 0), (\epsilon, \epsilon)]$ where we use $\epsilon = 1/10$. We obtain 135 putative bases B (62 valid), and from those we select the minimal one which has more than or equal to $\text{ceil}(70/4) = 18$ monomials and produces 18 or more equations. The basis we get is $\{1, n_y, n_y^2, n_y^3, n_y^4, n_y^5, n_y^6, n_x, n_xn_y, n_xn_y^2, n_xn_y^3, n_xn_y^4, n_xn_y^5, n_x^2, n_x^2n_y, n_x^2n_y^2, n_x^2n_y^3, n_x^2n_y^4, n_x^3, n_x^3n_y, n_x^3n_y^2, n_x^3n_y^3\}$. It consists of 22 monomials and it has been obtained with $Q = P_1' + P_2'$ and $\mathbf{d} = (0, -\epsilon)$. The exponent vectors $\mathcal{B}$ of this basis are illustrated in Fig. 1 (b). We multiply the original equations $f_{1,1}$, $f_{2,1}$ and $f_{1,2}$ with monomials $\mathcal{M}_{1,1} = \{1, n_x, n_y, n_x^2, n_y^2, n_y^3, n_xn_y, n_xn_y^2, n_x^2n_y\}$, $\mathcal{M}_{2,1} = \{1, n_x, n_y, n_y^2, n_xn_y\}$, $\mathcal{M}_{1,2} = \mathcal{M}_{1,1}$, and as a result we get 23 equations that can be expressed as linear combinations of the basis monomials. In order to get square coefficient matrices we simply discard the last equation. Then we need to identify the zero monomials by using the strategy described in the previous section. We find 6 zero monomials with two iterations, and remove the corresponding rows and columns from $\mathbf{A}$ and $\mathbf{B}$ which means that the final dimensions of these matrices are 82×82. Hence, besides the actual 70 roots we get 12 spurious roots that can be found, for example, by substituting the solutions obtained to the original equations. In practice most of the roots are complex-valued, and there are typically only few positive real-valued roots that we need to consider.

Our second self-calibration problem enabled by (14) and (15) has two unknown focal lengths λ_0 and $\lambda_1 = \lambda_2$. This corresponds to a situation where the first uncalibrated camera is used to capture the reference image and the second uncalibrated camera captures two target images. Interestingly, we cannot solve the focal length of the first camera λ_0 in this case, because the variety is no longer zero-dimensional, and a single reference image does not provide enough constraints for λ_0, but it can still be used to solve λ_1. Now we use all four constraints $f_{1,1}$, $f_{2,1}$, $f_{1,2}$ and $f_{2,2}$. The total degrees remain the same but because there are now 4 variables, the degree of λ_1 is two and hence, the PEP will have degree $l = 2$. Next we compute the mixed volume in 4-dimensional space. The volumes of the corresponding convex polytopes are $\text{Vol}(P_1) = \text{Vol}(P_3) = 14/3$, $\text{Vol}(P_2) = \text{Vol}(P_4) = 64/3$,$\text{Vol}(P_{12}) = \text{Vol}(P_{14}) = \text{Vol}(P_{23}) = \text{Vol}(P_{34}) = 204$,$\text{Vol}(P_{13}) = 224/3$,$\text{Vol}(P_{24}) = 1024/3$, $\text{Vol}(P_{123}) = \text{Vol}(P_{134}) = 800$,$\text{Vol}(P_{124}) = \text{Vol}(P_{124}) = \text{Vol}(P_{234}) = 1270$, and $\text{Vol}(P_{1234}) = 3264$. This gives us the mixed volume $r = 304$ which is the upper bound on the number of solutions. The minimum cardinality for the monomial basis is hence 152. Then we use Algorithm 1 to construct the basis. Because $P_3' = P_1'$ and $P_4' = P_2'$ we can use $k_0, \ldots, k_i \in \{0, 1, 2\}$ that results in an index list with $|K| = 17$. The displacement vector Δ contains 27 elements, and therefore we get 459 putative bases (175 valid). From those the minimal one with 191 monomials is obtained using $Q = P_2' + P_4'$ and $\Delta = [0, -\epsilon, \epsilon]$. The set $\mathcal{T}$ contains 196 elements, and we decide to discard the 5 last ones to get square coefficient matrices. Now it turns out that $\mathbf{C}_1$ becomes a zero matrix, and we can rewrite the PEP to $\mathbf{C}_0\mathbf{v} = -\lambda_1^2\mathbf{C}_2\mathbf{v}$, which is again a generalized eigenvalue problem. Instead of λ_1 we get solutions to λ_1^2 which then results in 191 positive solutions, and most of them are complex valued or at infinity. Finally, we observe that there are 15 zero columns common to $\mathbf{C}_0$ and $\mathbf{C}_1$. We can remove these columns and corresponding rows which reduces the number of solutions to 176 meaning that there are now 24 spurious roots in the solutions obtained.

5. Experiments

In this section, we show experimentally that the self-calibration methods presented in Section 4 give numerically stable results both with synthetic and real data. We compared our method to the modified resultant-based method [13] which generates $\binom{n+d-1}{d}$ basis monomials, where $d = \sum(d_i - 1) + 1$. Since $d = 8$ and $n = 3$ in the first calibration problem (equal focal length) we get 45 basis monomial which is more than twice the number of the monomials produced by our method. In total 57 equations can be constructed from the original ones by multiplying them with monomials of degree $d - d_i$. Despite of 12 extra equations we were not able to find a combination that would result in $\mathbf{C}_0$ with a full rank, and hence, it is not possible to invert $\mathbf{C}_0$ and convert the PEP to a standard eigenvalue problem. Instead we selected the equations randomly and used our SVD based elimination scheme to remove 11 zero monomials that helped to improve the stability of the method. The final number of solutions is therefore 169 which means that there are 99 spurious roots, while in our method the number of spurious roots is only 12. Both solvers were implemented with Matlab.

In the experiments with synthetic data we uniformly sampled 1000 random points on a plane with dimensions of 1000×1000 units. Three cameras were placed around 2000 units from the plane so that the cameras were roughly pointing at the center of the plane. The size of the images were 1000×1000 pixels. For the first calibration problem the focal length was set to $\lambda = \lambda_0 = \lambda_1 = \lambda_2 = 1200$, and we first tested the numerical stability of the methods by using noise-free data. We randomly picked 4 points from the dataset and computed the homographies. This was repeated $30,000$ times, and every time the closest root $\hat{\lambda}$ to the ground truth λ_{gt} was selected, and the relative error $|\hat{\lambda} - \lambda_{gt}|/\lambda_{gt}$ was computed. The distributions of the relative errors in the first calibration problem are shown in Fig. 2 (a) on a logarithmic scale for our method based on sparse elimination (SPE) and for the modified resultant-based method (MRE). As it can be observed our method is more precise which is explained by the smaller basis. We

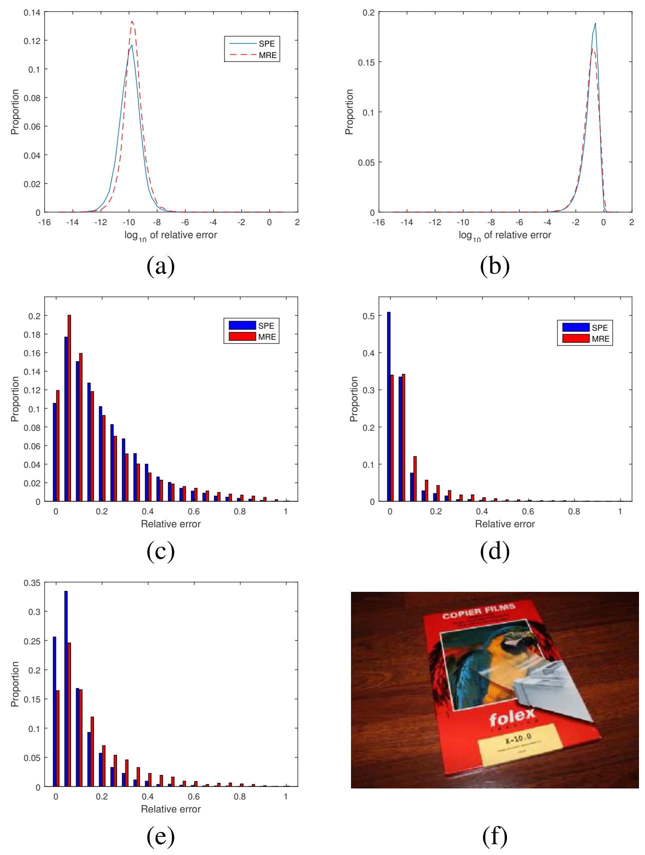
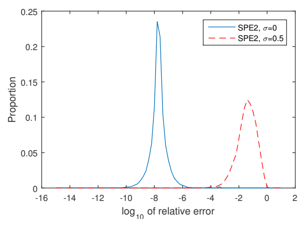


Figure 2. Relative errors for the first calibration problem (equal focal length): (a)-(c) synthetic data and (d)-(f) real images.

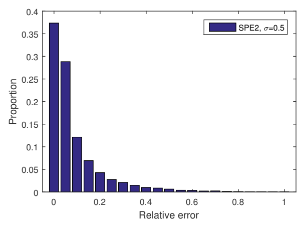
then added random noise with $\sigma = 0.5$ pixels to the image coordinates to simulate the quantization error. To make the comparison more fair, we have compensated for the large difference in the number of roots by doing the random sampling twice for SPE and selecting the relative error which is smaller. Furthermore, we rescaled the pixel values by $1/10$ to reduce the numerical values of the homographies. This was done, because the polynomial coefficients c_{ij} are themselves polynomials of the homographies, and high powers of large values tend to amplify the noise. Using smaller values regularizes the noise and reduces numerical problems. The corresponding relative errors are plotted on a logarithmic scale in Fig. 2 (b), and they are also presented as histograms on a linear scale in Fig. 2 (c). The precision is marginally better for SPE than for MRE (mean error 0.1935 vs. 0.1947), but the computational cost is much higher for MRE, because the eigenvalue decomposition has typically complexity of $O(n^3)$.

We also tested the methods using real images captured with a Canon EOS 600D camera. The camera was first calibrated with a checkerboard pattern and Matlab's `cameraCalibrator` tool to provide a ground truth value for the focal length. The lens was adjusted to $f = 18\,\text{mm}$ and the images were decimated by 4 to 1296×864 pixels. The calibration was performed from 10 images and the ground truth focal length obtained was 1108 pixels. In the first experiment, we used the same images of the checkerboard pattern to estimate the focal length with both solvers. In contrast to the Matlab's tool, these solvers do not exploit the prior information that the points form a known pattern. In addition, we only used three randomly selected images with a half of the points (27 out of 54) to compute the homographies. The random sampling for images and points is again repeated $30,000$ times to get statistically reliable results. The relative errors are presented as histograms in Fig. 2 (d). One can observe that the precision is now higher than with the simulated data ($\sigma = 0.5$). This is explained by the larger number of points that were used to compute the homographies. Four points, which is the minimum, is not clearly sufficient for obtaining reliable results, but because the homography can be easily estimated from more than 4 points, it is beneficial to use as many points as possible. The results also indicate that our approach can produce much higher precision than MRE. In the other experiment with real data, we used 14 images of a copier transparency film box captured with the same camera (an example is shown in Fig. 2 (f)). SURF keypoints and descriptors were used to find correspondences between the image pairs, and RANSAC was employed to compute the homographies. This procedure was repeated $30,000$ times for randomly selected image triplets. The distributions of the relative errors presented in Fig. 2 (e) indicate that the results with SPE are again better than with MRE although the precision for both methods is lower than for the calibration images, because the keypoint locations are more noisy than the corners of the checkerboard pattern.

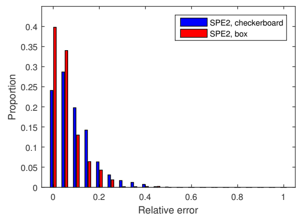
For the second calibration problem (unequal focal length) we set $\lambda_0 = 1000$ and $\lambda_1 = \lambda_2 = 1200$, and we repeated the same experiments as for the first problem with our method (SPE2). We did not implement the modified resultant-based method, because in this case $d = 17$ and $n = 4$ that would give us 1140 monomials which is almost 6 times more than with our method, and it is clear that the results would be inferior. The relative errors are plotted in Fig. 3 (a) both for the noiseless case and for $\sigma = 0.5$ using the simulated data. The latter one is also presented on a linear scale in Fig. 3 (b). One can observe that without noise the precision is lower than in the first problem which is due to the larger matrix size that causes higher numerical errors to the eigenvalues. However, in the case of noisy measurements the precision is actually higher than in the first problem. One possible reason for this is that in the second problem the solver uses 4 constraints in contrast to the 3 constraints of the first problem. Similar behavior can be seen in the case of real data shown in Fig. 3 (c) both for the checkerboard and the transparency film box images that also produce slightly smaller errors than in the first problem.



(a)



(b)



(c)

method	time [ms]
SPE	3.2
MRE	17.5
SPE2	19.5

(d)

Figure 3. Relative errors for the second calibration problem (unequal focal length): (a)-(b) synthetic data and (c) real images. Execution times for the solvers (d).

Execution times for the three solvers are given in Fig. 3 (d) that clearly demonstrates the advantage of our algorithm. It should be noticed that Matlab's `eig` function is used to compute the generalized eigenvalues. In practice, one can build a more efficient solver by exploiting the sparsity of the matrices. Finally, we also tried to compare our method with a Gröbner basis solver using the automatic generator [22] implemented based on [12], but the software failed to produce solvers to these calibration problems that have very high polynomial degrees, and hence, we were unable to perform the comparison with a reasonable effort.

6. Conclusions

In this paper, we have proposed a new algorithm for selecting the monomial basis for polynomial eigenvalue problems based on sparse elimination that has been previously used for constructing sparse resultants. Our approach has two important advantages over the sparse resultants: 1) the solution is provided by eigenvalues instead of the roots of a high-order determinant, and 2) the cofficient matrices do not need to be of full rank unlike sparse resultants that simplifies the algorithm and often leads to a more compact basis. In contrast to the modified resultant-based method [13] our algorithm can exploit the sparsity of the polynomials that is a common property in real-world problems. As a result, the monomial basis becomes smaller, and it is the same only in the limiting case where the polynomials are dense. We also presented two new minimal problems for camera self-calibration, and demonstrated that our algorithm can provide numerically more stable results than the modified resultant-based method.

References

[1] M. A. Brown, R. I. Hartley, and D. Nistér. Minimal solutions for panoramic stitching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 1

[2] M. Byröd, M. Brown, and K. Åström. Minimal solutions for panoramic stitching with radial distortion. In *British Machine Vision Conference (BMVC)*, pages 41.1–41.11. BMVA Press, 2009. doi:10.5244/C.23.41. 1

[3] M. Byröd, K. Josephson, and K. Åström. Fast and stable polynomial equation solving and its application to computer vision. *Int. J. Comput. Vision*, 84(3):237–256, Sept. 2009. 2

[4] D. A. Cox, J. B. Little, and D. O'Shea. *Using algebraic geometry*. Graduate texts in mathematics. Springer, New York, 1998. 2, 3, 4

[5] I. Emiris. On the complexity of sparse elimination. *Journal of Complexity*, 12(2):134–166, 1996. cited By 23. 2, 3

[6] I. Emiris and J. Canny. Efficient incremental algorithms for the sparse resultant and the mixed volume. *Journal of Symbolic Computation*, 20(2):117–149, 1995. 2, 4

[7] O. D. Faugeras and F. Lustman. Motion and Structure From Motion in a Piecewise Planar Environment. *Intern. J. of Pattern Recogn. and Artific. Intelige.*, 2(3):485–508, 1988. 1

[8] A. W. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. 1, 2

[9] R. M. Haralick, D. Lee, K. Ottenburg, and M. Nolle. Analysis and solutions of the three point perspective pose estimation problem. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 592–598, 1991. 1

[10] D. Herrera C., J. Kannala, and J. Heikkilä. Forget the checkerboard: practical self-calibration using a planar scene. In *IEEE Winter Conference on Applications of Computer Vision*, 2016. 5

[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 *Asian Conference on Computer Vision (ACCV)*, 2014. 1

[12] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. pages 302–315, 2008. 8

[13] Z. Kukelova, M. Bujnak, and T. Pajdla. Polynomial eigenvalue solutions to minimal problems in computer vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(7):1381–1393, 2012. 2, 4, 7, 8

[14] Z. Kukelova, J. Heller, M. Bujnak, and T. Pajdla. Radial distortion homography. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 639–647. IEEE, 2015. 1

[15] Z. Kukelova and T. Pajdla. A minimal solution to the autocalibration of radial distortion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 1

[16] Z. Kukelova and T. Pajdla. A minimal solution to radial distortion autocalibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(12):2410–2422, 2011. 2

[17] 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

[18] T. Pajdla and Z. Kukelova. Minimal problems in computer vision. http://cmp.felk.cvut.cz/mini/. 1

[19] O. Saurer, P. Vasseur, C. Demonceaux, and F. Fraundorfer. A homography formulation to the 3pt plus a common direction relative pose problem. In *Asian Conference on Computer Vision (ACCV)*, pages 288–301. Springer, 2014. 1

[20] H. Stewénius, D. Nistér, F. Kahl, and F. Schaffalitzky. A minimal solution for relative pose with unknown focal length. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 1

[21] B. Sturmfels. On the newton polytope of the resultant. *Journal of Algebraic Combinatorics*, 3(2):207–236, 1994. 3

[22] P. Trutman, Z. Kukelova, and T. Pajdla. Automatic generator of Groebner basis solvers. https://github.com/PavelTrutman/Automatic-Generator, 2014. 8

[23] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, Nov. 2000. 1

[24] Z. Zhang and A. R. Hanson. 3d reconstruction based on homography mapping. In *In ARPA Image Understanding Workshop*, pages 0249–6399, 1996. 1, 6