Efficient Intersection of Three Quadrics and Applications in Computer Vision

Zuzana Kukelova[1] Jan Heller[2*] Andrew Fitzgibbon[1]

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

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

Abstract

In this paper, we present a new algorithm for finding all intersections of three quadrics. The proposed method is algebraic in nature and it is considerably more efficient than the Gröbner basis and resultant-based solutions previously used in computer vision applications. We identify several computer vision problems that are formulated and solved as systems of three quadratic equations and for which our algorithm readily delivers considerably faster results. Also, we propose new formulations of three important vision problems: absolute camera pose with unknown focal length, generalized pose-and-scale, and hand-eye calibration with known translation. These new formulations allow our algorithm to significantly outperform the state-of-the-art in speed.

1. Introduction

Many computer vision and robotics problems can be formulated as problems of finding solutions to systems of polynomial equations. Examples include minimal problems of estimating relative and absolute camera poses [26, 34, 3], problems of calibrating radial distortion from point correspondences [7, 19], as well as minimization problems with polynomial cost functions [35, 6, 22]. Fast solvers of minimal computer vision problems are often used inside RANSAC-style loops [12] and, consequently, are parts of large systems like 3D reconstruction and recognition pipelines. Maximizing the efficiency of the solvers for these problems is of high importance.

A popular approach to dealing with polynomial problems is to design efficient specific solvers of the polynomial systems for given problem classes. Such a solver can be used to solve instances of a specific problem class only, however, it can do it much more efficiently than any general solver could. The design of these specific solvers is usually based on the Gröbner basis method or on the resultant-based method [9]. Several tools to aid the design of the specific solvers based on the Gröbner basis method exist, *e.g.* the automatic generator by Kukelova *et al.* [20].

However, not every problem needs or admits a specifically designed solver. In fact, many vision problems lead, if appropriately formulated, to the same polynomial problem: the problem of solving three quadratic equations in three unknowns. In this paper, we propose a new algorithm for solving this problem. Our solution is algebraic in nature and is more efficient than current solutions, which are based on Gröbner bases or resultants. For the sake of brevity, we will denote the problem of solving three quadrics in three unknowns as 3Q3 and the proposed solver as the efficient 3Q3 solver—E3Q3—throughout the rest of the paper.

In §2, we discuss the development of 3Q3, and in §3, we formally introduce the 3Q3 problem and present the E3Q3 solver. Further, we provide a discussion of the problem degeneracies (§3.1) as well as its computational complexity (§3.2). In §4 we discuss several computer vision problems that have been previously formulated and solved as 3Q3 and where our algorithm readily delivers considerably more efficient solutions. Finally, we present novel 3Q3 formulations for three important computer vision problems: absolute camera pose with unknown focal length (§5.1), generalized pose-and-scale (§5.2), hand-eye calibration with known translation (§5.3). Using the E3Q3 solver, we provide new efficient solutions that are significantly faster than the state-of-the-art algorithm for these problems. The speedups gained by these new solutions are summarized in §6.

2. The 3Q3 problem

The problem of finding the intersection of quadrics has been studied in mathematics [31, 18], computational physics, computer graphics, robotics, kinematics [32], and computer vision. Early algorithms for this special class of polyno-

*The author was supported by Czech Ministry of Education under Project RVO37000.

mial system came from computer-aided design. In 1978, Levin [24] described an efficient parameterization of the intersection curve of two quadrics. Xu *et al*. [38] later used it to solve 3Q3, but numerical difficulties instead required them to use the algorithm proposed in [11] which runs in rational numbers, making the final 3Q3 solver impractically slow.

Chionh *et al*. [8] presented a method for solving the 3Q3 problem based on Macaulay's multivariate resultant. The proposed method needs to perform more than 10^6 operations as well as to compute the greatest common divisor, which may be numerically unstable.

Roth [32] describes a general 3Q3 solver using an elimination method, which is based on the resultant method [9]. Its solution computes the determinant of a 6×6 polynomial matrix. This determinant is a degree 8 polynomial in one variable and its computation using standard symbolic methods costs more than 100K operations.

Ramalingam *et al*. [30] created a general 3Q3 solver as a component of a 3D-3D registration problem. An automatic solver generator [20] was used to obtain a Gröbner basis solver, which needs to perform Gauss-Jordan (G-J) elimination of a 26×34 matrix and to compute eigenvalues of an 8×8 matrix.

These general solvers are all slower than our new algorithm, and do not handle, except for [38], all degeneracies and symmetries in the solution.

3. E3Q3: Efficient intersection of 3 quadrics

Now, let us formalize the 3Q3 problem. Let x, y, z be the problem unknowns, c_{ij}, $i = 1, 2, 3$, $j = 1, \ldots, 10$, the problem coefficients and (with $\mathbf{c}_i = [c_{i1}, c_{i2}, \ldots, c_{i10}]$),

$$q_i = \mathbf{c}_i \cdot [x^2, y^2, z^2, xy, xz, yz, x, y, z, 1] \tag{1}$$

the three polynomials of degree 2. The problem of 3Q3 is to find the intersections of q_i, *i.e*., to solve the system

$$q_i = 0, \;\; i = 1, 2, 3. \tag{2}$$

To solve Eqs. 2, let us start by 'hiding' the unknown x in the coefficient field, *i.e*., by considering x to be a coefficient for a moment. This leaves us with six monomials $[y^2, z^2, yz, y, z, 1]$ in unknowns y, z in every equation. By splitting these monomials into two sets $\{y^2, z^2, yz\}$, $\{y, z, 1\}$ and by rearranging them to the left- and right-hand sides, Eqs. 2 can be rewritten as a matrix equation:

$$\mathtt{A} \begin{bmatrix} y^2 \\ z^2 \\ yz \end{bmatrix} = \begin{bmatrix} p_{11}(x) & p_{12}(x) & p_{13}(x) \\ p_{21}(x) & p_{22}(x) & p_{23}(x) \\ p_{31}(x) & p_{32}(x) & p_{33}(x) \end{bmatrix} \begin{bmatrix} y \\ z \\ 1 \end{bmatrix}, \tag{3}$$

where $\mathtt{A}$ is a coefficient matrix

$$\mathtt{A} = \begin{bmatrix} c_{12} & c_{13} & c_{16} \\ c_{22} & c_{23} & c_{26} \\ c_{32} & c_{33} & c_{36} \end{bmatrix}, \tag{4}$$

and $p_{11}(x), \ldots, p_{33}(x)$ are polynomials in x. Note that $p_{13}(x), p_{23}(x), p_{33}(x)$ are quadratic polynomials in x and the remaining polynomials in Eq. 3 are linear in x.

Next, let us assume that the matrix $\mathtt{A}$ has full rank. Of course, this may not always be the case, however, we will defer the discussion on the rank of $\mathtt{A}$ until §3.1. For now, we can multiply Eq. 3 by $\mathtt{A}^{-1}$, resulting in

$$\begin{bmatrix} y^2 \\ z^2 \\ yz \end{bmatrix} = \begin{bmatrix} p'_{11}(x) & p'_{12}(x) & p'_{13}(x) \\ p'_{21}(x) & p'_{22}(x) & p'_{23}(x) \\ p'_{31}(x) & p'_{32}(x) & p'_{33}(x) \end{bmatrix} \begin{bmatrix} y \\ z \\ 1 \end{bmatrix}, \tag{5}$$

where $p'_{11}(x), \ldots, p'_{33}(x)$ are linear combinations of polynomials $p_{11}(x), \ldots, p_{33}(x)$. Again, $p'_{13}(x), p'_{23}(x), p'_{33}(x)$ are quadratic polynomials in x and the remaining polynomials in Eq. 5 are linear in x. The reason why we manipulated Eqs. 2 into Eq. 5 is to express monomials y^2, z^2, and yz as polynomial functions in y, z, and 1.

Now, let us introduce three trivial identities involving the left-hand side monomials y^2, z^2, and yz from Eq. 5

$$\left(y^2\right) z = (yz)\, y, \tag{6}$$

$$(yz)\, z = \left(z^2\right) y, \tag{7}$$

$$(yz)\,(yz) = \left(y^2\right)\left(z^2\right). \tag{8}$$

Expanding these identities using expressions for y^2, z^2, and yz from Eq. 5 yields three equations:

$$\begin{aligned} &(p'_{11}(x)y + p'_{12}(x)z + p'_{13}(x))z \\ &\qquad = (p'_{31}(x)y + p'_{32}(x)z + p'_{33}(x))y, \end{aligned} \tag{9}$$

$$\begin{aligned} &(p'_{31}(x)y + p'_{32}(x)z + p'_{33}(x))z \\ &\qquad = (p'_{21}(x)y + p'_{22}(x)z + p'_{23}(x))y, \end{aligned} \tag{10}$$

$$\begin{aligned} &(p'_{31}(x)y + p'_{32}(x)z + p'_{33}(x)) \\ &\cdot\, (p'_{31}(x)y + p'_{32}(x)z + p'_{33}(x)) \\ &\qquad = (p'_{11}(x)y + p'_{12}(x)z + p'_{13}(x)) \\ &\qquad\quad \cdot\, (p'_{21}(x)y + p'_{22}(x)z + p'_{23}(x)). \end{aligned} \tag{11}$$

Since Eqs. 9, 10, and 11 again contain monomials y^2, z^2 and yz, we substitute expressions for y^2, z^2 and yz from Eq. 5 into Eqs. 9-11 once more. This double substitution transforms the identities from Eqs. 6-8 into the following matrix equation

$$\begin{bmatrix} s^{[2]}_{11}(x) & s^{[2]}_{12}(x) & s^{[3]}_{13}(x) \\ s^{[2]}_{21}(x) & s^{[2]}_{22}(x) & s^{[3]}_{23}(x) \\ s^{[3]}_{31}(x) & s^{[3]}_{32}(x) & s^{[4]}_{33}(x) \end{bmatrix} \begin{bmatrix} y \\ z \\ 1 \end{bmatrix} = \mathtt{M}(x) \begin{bmatrix} y \\ z \\ 1 \end{bmatrix} = \mathbf{0}, \tag{12}$$

where the upper index $[\cdot]$ denotes the maximum possible degree of the respective polynomial $s_{ij}(x)$.

As we know from elementary linear algebra, matrix Eq. 12 has a non-trivial solution if and only if the determinant of its matrix $\mathtt{M}(x)$ is zero. It can be easily inferred

from the degrees of $s_{ij}(x)$ that $\det(\mathtt{M}(x))$ is an up to degree 8 polynomial in x.

The solutions to the unknown x from the original Eq. 2 can now be obtained by finding the roots of the up to degree 8 polynomial $\det(\mathtt{M}(x))$. For example, these can be computed as the eigenvalues of its companion matrix [17] or more efficiently using Sturm sequences [39] in some feasible interval. Solutions to the unknowns y and z can be obtained from $\mathtt{M}(x)$ after substituting the particular solutions for x into this matrix and solving the resulting system of two linear equations. Alternatively, to increase stability, the solutions can be obtained from the eigenvectors of $\mathtt{M}(x)$. To elaborate, let x^* be a particular solution to $\det(\mathtt{M}(x)) = 0$, let $\mathtt{M}(x^*) = \mathtt{USV}^\top$ be SVD of the numerical 3×3 matrix $\mathtt{M}(x^*)$ and let $\mathbf{v} = [v_1, v_2, v_3]^\top$ be the column of matrix $\mathtt{V}$ that corresponds to the smallest singular value. Then, the particular solutions y^* and z^* correspond to v_1/v_3 and v_2/v_3, respectively.

3.1. Problem degeneracies

The E3Q3 solver presented in §3 assumes that the matrix $\mathtt{A}$ from Eq. 4 and the polynomial matrix $\mathtt{M}(x)$ from Eq. 12 are regular. Quadrics with coefficients that do not lead to regular matrices $\mathtt{A}$ and $\mathtt{M}(x)$ cannot be solved by this general E3Q3 solver, nor by any of the previously proposed algebraic solvers for the 3Q3 problem: the Gröbner basis solver [20] assumes that the 26×34 elimination matrix is regular and the resultant-based solver [32] assumes regularity of the 6×6 polynomial matrix. This means that these solvers do not work for 'degenerate' inputs, *e.g.*, when one or more of the input quadrics are in fact planes or curves or when certain coefficients vanish.

In the case of the formulation in §3 however, it is quite easy to recognize the degenerate configurations based on the structure of the matrix $\mathtt{A}$. By using G-J elimination and, if necessary, by interchanging of the matrix rows, matrix $\mathtt{A}$ can be reduced into one of the following configurations:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & \bullet \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \\ \begin{bmatrix} 1 & \bullet & \bullet \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & \bullet & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & \bullet \\ 0 & 1 & \bullet \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{13}$$

where '$\bullet$' stands for an arbitrary coefficient from $\mathbb{R}$. All other configurations can be transformed into one of these 8 configurations [38] either by interchanging variables y and z or by assuming '$\bullet$' equal to 0.

A particular solver can be derived for every configuration from Eq. 13, with the last configuration corresponding to the non-degenerate case derived above. The derivations of the remaining configurations follow the same idea: three equations are derived that are linear in y, z and yield a determinant-zero problem like Eq. 12. The full treatment of the degeneracies of E3Q3 can be found in Appendix §A.1.

3.2. Computational complexity

Let us now analyze the computational complexity of the E3Q3 solver and compare it to the complexity of several pre-existing methods for solving 3Q3.

First, the E3Q3 solver needs to invert a numerical 3×3 matrix. Next, by comparing and multiplying the coefficients of quadratic and linear polynomials, the solver creates the 3×3 polynomial matrix $\mathtt{M}(x)$ with univariate polynomials of degrees 2, 3, and 4 as coefficients. The determinant of $\mathtt{M}(x)$ directly provides the coefficients of a degree 8 polynomial in one variable. To compute the coefficients of this polynomial, 678 additions and 1133 multiplications need to be performed. Altogether, E3Q3 performs 1811 operations to obtain a degree 8 univariate polynomial.

In comparison, the Gröbner basis solver [20, 30] performs one G-J elimination of a 26×34 matrix. This matrix already contains several zero elements, so the elimination needs fewer operations than G-J elimination of a full matrix of the same size. On average, the G-J elimination step performs $\sim$4955 additions and $\sim$5511 multiplications, depending on the coefficients of the input quadrics. Altogether, the G-J elimination step costs $\sim$10466 operations. This is almost 6 times more than the number of operations performed by E3Q3. After the G-J step, the Gröbner basis solver needs to compute eigenvalues of a 8×8 multiplication matrix. Bujnak *et al.* [5] showed that it is more efficient to extract its univariate characteristic polynomial and compute the roots using Sturm sequences than to compute its eigenvalues and eigenvectors. The Danilevskii method [10] for computing the characteristic polynomial of a 8×8 matrix costs $\sim$1380 operations. Altogether, the Gröbner basis solver performs $\sim$11846 operations to obtain a degree 8 univariate polynomial. This is 6.5 times more than the number of operations performed by the E3Q3 solver. Once the degree 8 polynomial is computed, both E3Q3 and the Gröbner basis solver use the same Sturm sequences approach to recover its roots.

The following table summarizes computational costs as the number of operations performed by the respective solvers to obtain the final univariate polynomial.

	Resultant [13, 32, 8]	GB [30, 20]	E3Q3
# of ops	>100000	$\sim$11846	1811

4. 3Q3 in computer vision

The problem of finding the intersection of three quadrics has already appeared in several computer vision problems, from the venerable problem of estimating the absolute pose of a calibrated camera (P3P) [12] to the recently published generalized pose [29] and multi-camera problems [23].

In the case of the P3P problem, the three quadratic equations are derived from the law of cosines. As such, the quadratic equations have a specific structure and contain only monomials of degree 2. The final system has only four solutions and can be easily solved by simple polynomial manipulations [12].

Guo [13] used the 3Q3 formulation to solve a variant of the P3P problem for an uncalibrated camera; however the problem of estimating absolute pose of a camera with unknown focal length and aspect ratio (P4P) was solved using Roth's method [32], which is quite computationally expensive.

4.1. Generic sources of 3Q3

In this section, we describe in more detail several generic sources of 3Q3 that quite often appear in computer vision and robotics problems.

4.1.1 Euclidean distances (ED)

One of the sources of quadratic equations are Euclidean distances, which are ubiquitous in computer vision and robotics problems.

4.1.2 Rotations (R)

A part of many computer vision and robotics problems is the problem of estimation of unknown rotations, and these are yet another source of quadratic equations. These equations can be the result of the constraints on the orthogonality and the unit norm of rows and columns of rotation matrices. Also, rotations can be parametrized using Cayley transformation [16]. In this parametrization, a vector $\mathbf{x} = [x, y, z]^\top \in \mathbb{R}^3$ represents the following rotation matrix:

$$\mathtt{R}(\mathbf{x}) = \frac{\mathtt{R}'(\mathbf{x})}{k} = \frac{1}{k}\begin{bmatrix} 1+x^2-y^2-z^2 & 2\,x\,y-2\,z & 2\,y+2\,x\,z \\ 2\,z+2\,x\,y & 1-x^2+y^2-z^2 & 2\,y\,z-2\,x \\ 2\,x\,z-2\,y & 2\,x+2\,y\,z & 1-x^2-y^2+z^2 \end{bmatrix}, \tag{14}$$

where $k = 1 + x^2 + y^2 + z^2$. The parametrization from Eq. 14 produces a rotation matrix corresponding to the quaternion $w + ix + jy + kz$ normalized such that $w = 1$. This rotation matrix directly contains quadratic monomials in three variables. Similarly, quaternion parameterizations are quadratic in the parameters.

4.1.3 Nullspace vectors (NV)

Using null space vectors of some coefficient matrix, one can often reduce the number of variables in an input system by reparameterizing the monomial vectors and by incorporating dependencies between monomials in these vectors.

4.2. Generalized pose estimation problem (ED, NV)

The generalized pose problem generates 3Q3 both through its use of Euclidean distances, and through nullspace vectors. The goal is to estimate the unknown pose of a calibrated generalized camera w.r.t. known 3D points. Because the minimal number of 2D-to-3D point correspondences necessary to solve this problem is three, the minimal generalized pose estimation problem (gP3P) can be thought of as the problem of searching for possible poses of the camera such that the three given rays meet the three known world points.

Several approaches to formulating the gP3P problem as a 3Q3 system have been proposed in the past [27, 29, 25]. Ramalingam *et al.* [29] presented a formulation based on Euclidean distances. Let us now review this formulation.

First, let us represent the known i^{th} ray in the camera coordinate system using a point of origin $\mathbf{A}_i$ and a unit direction vector $\mathbf{v}_i$. Now, an arbitrary point $\mathbf{X}_i$ coinciding with the i^{th} ray can be parametrized in the camera coordinate system using scalar λ_i as $\mathbf{X}_i^C = \mathbf{A}_i + \lambda_i \mathbf{v}_i$, where the superscript C stands for the camera coordinate system.

Now, the goal is to find $\mathbf{X}_i^C$, parametrized by λ_i, $i = 1, 2, 3$ that coincide with the known 3D points $\mathbf{X}^W$, $i = 1, 2, 3$ in the world coordinate system. Since the camera motion is a rigid transformation, the mutual distances d_{ij} between the coordinates in the world coordinate system and the camera coordinate system are preserved, *i.e.*,

$$\begin{aligned} d_{ij}^2 = \left\|\mathbf{X}_i^W - \mathbf{X}_j^W\right\|^2 &= \left\|\mathbf{X}_i^C - \mathbf{X}_j^C\right\|^2 \\ &= \left\|\mathbf{A}_i + \lambda_i \mathbf{v}_i - \mathbf{A}_j - \lambda_j \mathbf{v}_j\right\|^2, \end{aligned} \tag{15}$$

for $(i, j) \in \{(1, 2), (1, 3), (2, 3)\}$. This leads to three quadratic equations in three unknowns λ_1, λ_2 and λ_3. In [29], the authors suggested to use Maple and the resultant-based method to solve the resulting system.

Miraldo and Araujo [25] proposed a 3Q3 formulation for the same problem. In this case, the three quadratic equations were obtained as linear combination of four null-space vectors of homography parametrization and by incorporating this parametrization into the Euclidean distance constraints. The authors provided a polynomial eigenvalue solution to this 3Q3 formulation.

It is worth noting that the first formulation of the generalized absolute pose problem is due to Nistér [27]. His solution also leads to a 3Q3 system, however, it is formulated using the geometric primitives inherent to the problem. Because of the geometrical nature of this specific solution, it cannot be easily extended into a general 3Q3 solver.

4.3. Pose from one point and two lines (NV, R)

The aim of this problem is to recover the pose, *i.e.*, the unknown rotation and translation, of a calibrated camera given one point and two lines in the world coordinate system and their projections in the image.

Ramalingam *et al.* [28] formulated this problem using collinearity and coplanarity constraints. Two collinearity

equations arising from the one point correspondence and four coplanarity equations arising from the two line correspondences can be stacked and written in a matrix form as $\mathtt{A}\mathbf{X} = \mathbf{0}$, where $\mathbf{A}$ is a 6×10 coefficient matrix of rank 6 and $\mathbf{X} = [r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, t_1, t_2, t_3, 1]^\top$ is a 10×1 vector containing the unknown elements of the rotation matrix $\mathtt{R} = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3] = [r_{ij}]_{i,j=1}^{3}$ and the translation vector $\mathbf{t} = [t_1, t_2, t_3]^\top$. Since the matrix $\mathtt{A}$ has rank 6, the solutions to the unknown vector $\mathbf{X}$ can be obtained by reparameterizing it as a linear combination of the four vectors spanning the null-space of $\mathtt{A}$, *i.e.*,

$$\mathbf{X} = x_1\mathbf{n}_1 + x_2\mathbf{n}_2 + x_3\mathbf{n}_3 + \mathbf{n}_4, \tag{16}$$

where $\mathbf{n}_i \in \mathbb{R}^{10},\ i = 1, \ldots, 4$ are the null-space vectors and x_1, x_2, x_3 are the new unknowns.

To solve for the unknowns x_1, x_2, x_3, the orthogonality and the unit norm constraints involving the rotation variables from the first two columns, $\mathbf{r}_1, \mathbf{r}_2$ can be used:

$$\mathbf{r}_1^\top\mathbf{r}_1 = 1, \quad \mathbf{r}_2^\top\mathbf{r}_2 = 1, \quad \mathbf{r}_1^\top\mathbf{r}_2 = 0. \tag{17}$$

Ramalingam *et al*. [28] solved the final 3Q3 system in Eqs. 17 using the Gröbner basis method [20].

Note that in this particular problem formulation the constraints on the null space vectors comes from the properties of the rotation matrices. However, the constraints may have many different sources and one may use them to set up dependencies between the elements of the unknown vector $\mathbf{X}$ (*e.g.*, $X_1 = xy = x \cdot y = X_5 \cdot X_6$).

4.4. Registration of three 3D points and three 3D lines (NV, R)

Ramalingam *et al*. [30] presented a 3Q3 formulation of the problem of 3D-3D registration of three points and three lines. This is equivalent to the generalized camera pose problem. The 3Q3 system was obtained by parametrizing the rotation as a linear combination of four null-space vectors and by using quadratic constraints on the elements of the rotation matrix.

5. New 3Q3 formulations

In this section, three existing vision/robotics problems are reformulated as 3Q3, yielding improvements in speed, accuracy, and stability through the use of E3Q3.

5.1. P4Pf problem

The first problem that we will formulate as 3Q3 is the problem of estimating the camera position, orientation, and the focal length from four 2D-to-3D correspondences, also known as the P4Pf problem. This problem is a part of several structure-from-motion and localization applications and was recently solved by Bujnak *et al*. [3] using the Gröbner basis method. By using ratios of distances, the P4Pf problem can be formulated as a system of five equations of degree 3 in four unknowns. Their solver performs G-J elimination of a 154×180 matrix and computes eigenvalues of a 10×10 matrix. Later work [2] proposed a more efficient solver, consisting of G-J elimination of a 78×88 and eigenvalue computations for a 10×10 matrix. Here, we will show that for non-planar points, P4Pf can be formulated as 3Q3 and efficiently solved using E3Q3.

In the following, we will assume the standard pinhole camera model [15]. In this model, homogeneous coordinates of a scene point $\mathbf{X}$ and the corresponding image projection $\mathbf{u}$ are linearly connected as $\alpha\,\mathbf{u} = \mathtt{P}\,\mathbf{X}$, where $\alpha \in \mathbb{R}$ is a scalar and $\mathtt{P} = [p_{ij}]_{i,j=1}^{3,4} \in \mathbb{R}^{3\times4}$ is the camera projection matrix. The projection matrix $\mathtt{P}$ can be further decomposed as $\mathtt{P} = \mathtt{K}\,[\mathtt{R}\,|\,\mathbf{t}]$, where $\mathtt{R} = [r_{ij}]_{i,j=1}^{3} \in SO(3)$ is a rotation matrix, $\mathbf{t} = [t_1, t_2, t_3]^\top \in \mathbb{R}^3$ is a translation vector, and $\mathtt{K} \in \mathbb{R}^{3\times3}$ is a so-called calibration matrix. If we assume that the only unknown camera internal parameter is the focal length f, the calibration matrix takes on a particularly simple form $\mathtt{K} = \mathrm{diag}[f, f, 1]$. And since the projection matrix is given only up to scale anyway, we can parametrize $\mathtt{K}$ equivalently as $\mathtt{K} = \mathrm{diag}[1, 1, w]$, where $w = \frac{1}{f}$. Using this parametrization, the projection matrix $\mathtt{P}$ can be written as

$$\mathtt{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ wr_{31} & wr_{32} & wr_{33} & wt_3 \end{bmatrix}. \tag{18}$$

Now, we can give the the formal definition of the P4Pf problem: given four 3D scene points $\mathbf{X}_i = [x_i, y_i, z_i, 1]^\top$, $i = 1, \ldots, 4$, and four corresponding image points $\mathbf{u}_i = [u_i, v_i, 1]^\top$, $i = 1, \ldots, 4$, the task is to recover the unknown rotation $\mathtt{R}$, translation $\mathbf{t}$, and focal length $f = \frac{1}{w}$.

First, let us eliminate the α_i's by multiplying the projection equation $\mathtt{P}\mathbf{X}_i = \alpha_i\mathbf{u}_i$ from the left by the skew symmetric matrix $[\mathbf{u}_i]_\times$ to obtain the following matrix equation

$$\begin{bmatrix} 0 & -1 & v_i \\ 1 & 0 & -u_i \\ -v_i & u_i & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ wr_{31} & wr_{32} & wr_{33} & wt_3 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} = \mathbf{0}. \tag{19}$$

Eq. 19 encodes three polynomial equations, two of which are linearly independent. This is caused by the fact that the skew symmetric matrix $[\mathbf{u}_i]_\times$ has rank two. The equation corresponding to the third row of the matrix Eq. 19 can be rewritten in the elements of the projection matrix $\mathtt{P}$ as

$$\begin{aligned} &- v_i\,(p_{11}\,x_i + p_{12}\,y_i + p_{13}\,z_i + p_{14}) \\ &\quad + u_i\,(p_{21}\,x_i + p_{22}\,y_i + p_{23}\,z_i + p_{24}) = 0. \end{aligned} \tag{20}$$

Eq. 20 is a homogeneous linear equation in 8 unknowns p_{11}, p_{12}, p_{13}, p_{14}, p_{21}, p_{22}, p_{23}, and p_{24}. Since we have four

2D-3D point correspondences, we can construct four specific variants of Eq. 20. The four equations can be stacked into a matrix form as $\mathtt{A}\,\mathbf{v} = 0$, where $\mathtt{A} \in \mathbb{R}^{4\times 8}$ is a coefficient matrix and $\mathbf{v} = [p_{11}, p_{12}, p_{13}, p_{14}, p_{21}, p_{22}, p_{23}, p_{24}]^\top$ is the vector of unknowns. Assuming that $\mathtt{A}$ has full rank, we can parametrize the eight unknowns in $\mathbf{v}$ as a linear combination of four null space basis vectors $\mathbf{n}_i$ of $\mathtt{A}$:

$$\mathbf{v} = \textstyle\sum_{i=1}^{4} \gamma_i\, \mathbf{n}_i, \tag{21}$$

where γ_i, $i = 1, \ldots, 4$ are new unknowns. Since $\mathtt{P}$ can be recovered up to scale only, one of the new unknowns can be set to one, *e.g.*, $\gamma_4 = 1$. Eq. 21 gives us a new parametrization of the first two rows of the projection matrix $\mathtt{P}$ in three unknowns γ_1, γ_2, and γ_3. To get a detailed derivation of the third row of $\mathtt{P}$ using three unknowns γ_1, γ_2, and γ_3 as

$$[p_{31}, p_{32}, p_{33}, p_{34}]^\top = \mathtt{D}\,[\gamma_1, \gamma_2, \gamma_3, 1]^\top, \tag{22}$$

using a coefficient matrix $\mathtt{D} \in \mathbb{R}^{4\times 4}$, see Appendix §A.4.

Now, let $\mathtt{S}$ denote the left 3×3 submatrix of $\mathtt{P}$, $\mathtt{S} = \mathtt{KR}$. We will exploit four constraints on the elements of $\mathtt{S}$ that come from the fact that $\mathtt{R}$ is a rotation matrix. The first three constraints capture the fact that the rows of $\mathtt{S}$ are perpendicular to each other. The fourth constraint asserts the fact that the first and the second row of $\mathtt{S}$ have the same norm. These constraints, combined with the parametrization of $\mathtt{P}$ from Eqs. 21 and 22, form a system of four quadratic equations in three unknowns γ_1, γ_2, and γ_3:

$$\begin{aligned} p_{11}p_{21} + p_{12}p_{22} + p_{13}p_{23} &= 0, \\ p_{31}p_{11} + p_{32}p_{12} + p_{33}p_{13} &= 0, \\ p_{31}p_{21} + p_{32}p_{22} + p_{33}p_{23} &= 0, \\ p_{11}^2 + p_{12}^2 + p_{13}^2 - p_{21}^2 - p_{22}^2 - p_{23}^2 &= 0. \end{aligned} \tag{23}$$

The system of Eqs. 23 is clearly an over-determined problem. However, by taking any three of the four equations we can determine $\mathtt{S}$ by a 3Q3 system solvable by E3Q3.

Next, we will use the fact that the squared norm of the first row of $\mathtt{S}$ multiplied by w^2 is equal to the squared norm of the third row:

$$w^2\, p_{11}^2 + w^2\, p_{12}^2 + w^2\, p_{13}^2 - p_{31}^2 - p_{32}^2 - p_{33}^2 = 0. \tag{24}$$

Eq. 24 is a quadratic equation in $w = \frac{1}{f}$, positive roots of which provide the solutions for the focal length f.

Finally, the last column of $\mathtt{P}$, *i.e.*, the translation vector $\mathbf{t}$, can be determined by stacking Eq. 19 for the four 2D-3D correspondences and by solving the resulting overdetermined linear system.

Note that the above approach works for non-planar 3D points only. The solution for coplanar points is even simpler and can be solved linearly—in fact, this case is equivalent to homography estimation. The two cases can be joined into one general solver, akin to the solver presented in [4].

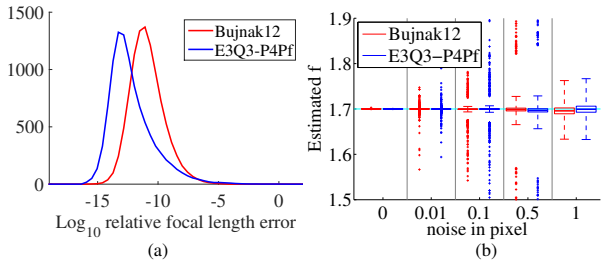

Figure 1: **P4Pf problem:** (a) Log_{10} of the relative error of the focal length. (b) Error of the focal length estimates in the presence of noise with $f_{gt} = 1.7$; E3Q3-P4Pf solver (blue), Bujnak12 [2] (red).

Experiments. We tested the presented E3Q3-based algorithm for the P4Pf problem (E3Q3-P4Pf) on synthetic data with varying focal lengths and levels of noise and compared its stability and precision to the state-of-the-art (SOTA) P4Pf solver Bujnak12 [2].

First, we studied the behavior of the new E3Q3-P4Pf solver on noise-free data to check its numerical stability and compared it to the results of Bujnak12. In this experiment, we generated 10000 synthetic scenes with 3D points distributed at random in a $[-10, 10]^3$ cube. Each 3D point was projected by a camera with random but feasible orientation and position and with random focal length $f_{gt} \in [0.5, 5]$. Figure 1(a) shows $\log_{10}$ of the relative error of the focal length f obtained by selecting the real root closest to the ground truth value f_{gt}. The E3Q3-P4Pf solver (blue) is slightly more stable than Bujnak12 (red). Still, both solvers provide very stable results without larger errors.

In the second experiment, we studied the accuracy of the E3Q3-P4Pf solver in the presence of image noise. Figure 1(b) shows the results for 3D scenes with different levels of noise added to the image projections. In this case, the ground truth focal length was set to $f_{gt} = 1.7$. For each noise level, ranging from 0 to 1 pixels, 1000 estimates for random scenes and random feasible camera positions were made. The results are represented using MATLAB `boxplot` function, which shows values 25% to 75% quantile as a box with a horizontal line at median, in Figure 1(b). The crosses show data beyond 1.5 times the interquartile range. Figures show that both solvers give very similar and accurate estimates of the focal length. Similar results were obtained for the estimated rotations and translations as well.

Since both E3Q3-P4Pf and Bujnak12 [2] solvers provide very similar results on simulated data, we haven't performed any real data experiments. For a detailed analysis of the usefulness of P4Pf solvers in real applications, see [3, 2].

5.2. Generalized pose-and-scale problem

The second problem that we will formulate as 3Q3 is the absolute pose problem for a generalized camera with unknown internal scale (gsP4P). This problem has applica-

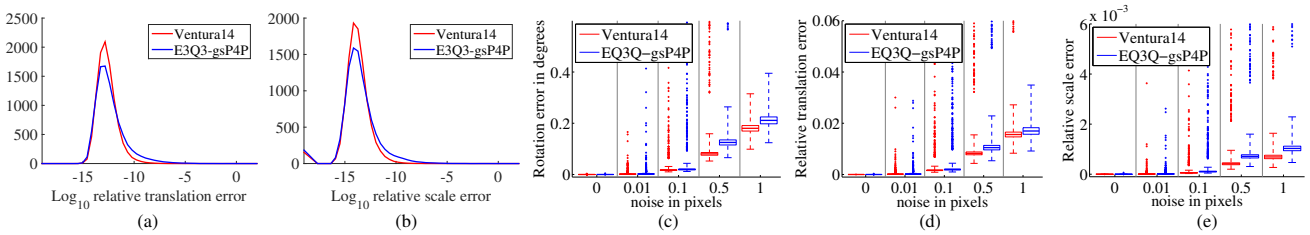

Figure 2: **gsP4P problem:** Log_{10} of (a) the relative translation and (b) relative scale error for noise free data. (c) Error of rotation, (d) relative error of translation and (e) relative error of scale estimates in the presence of noise for the new E3Q3-gsP4P solver (blue) and the state-of-the-art Ventura14 solver [37] (red).

tions in structure-from-motion (SfM) alignment or loop closure integration [37]. The problem can be thought of as the problem of estimating the position and orientation of a set of perspective cameras as well as the scale of the translation between them, with respect to a set of known 3D points. The generalized pose-and-scale problem was solved by Ventura *et al*. [37] as a minimal problem from four correspondences using the Gröbner basis method. They formulated the problem as a system of ten quadratic polynomial equations in five unknowns by exploiting properties of rotation matrices. The final Gröbner basis solver was obtained using the automatic generator [20]. It needs to perform SVD of a 8×13 matrix, G-J elimination or LU decomposition of a 48×56 matrix, and to compute eigenvalues of a 8×8 matrix. In the next, we will show that gsP4P can be solved by the much simpler E3Q3 solver once it is formulated as 3Q3 using Cayley's parametrization of rotations.

Let us denote the set of known 3D points as $\mathbf{Q}_i \in \mathbb{R}^3$. The corresponding projections in the generalized camera are described by rays with starting points $\mathbf{P}_i \in \mathbb{R}^3$ and unit direction vectors $\mathbf{d}_i \in \mathbb{R}^3$. The problem is to estimate the unknown rotation $\mathtt{R} \in SO(3)$, translation $\mathbf{t} \in \mathbb{R}^3$ and scale $s \in \mathbb{R}$ so that the rays hit the 3D points

$$\mathtt{R}\mathbf{Q}_i + \mathbf{t} = s\mathbf{P}_i + \alpha_i \mathbf{d}_i, \; i = 1, \ldots, 4, \tag{25}$$

where α_i's are unknown scales that stretch the direction vectors up to the 3D points.

First, let's start by eliminating the α_i's by multiplying Eqs 25 from the left by the skew symmetric matrices $[\mathbf{d}_i]_\times$ to obtain the following matrix equations

$$[\mathbf{d}_i]_\times \left(\mathtt{R}\mathbf{Q}_i + \mathbf{t} - s\mathbf{P}_i\right) = \mathbf{0}, \; i = 1, \ldots, 4. \tag{26}$$

Note that only 8 of the 12 Eqs. 26 are linearly independent. Next, we use Cayley parametrization of rotations [16] to transform Eqs. 26 to a 3Q3 system. Using the Cayley parametrization $\mathtt{R}(\mathbf{x}) = \frac{1}{k}\mathtt{R}'(\mathbf{x})$, see Eq. 14, Eqs. 26 can be rewritten as

$$[\mathbf{d}_i]_\times \left(\tfrac{1}{k}\mathtt{R}'(\mathbf{x})\mathbf{Q}_i + \mathbf{t} - s\mathbf{P}_i\right) = \mathbf{0}, \; i = 1, \ldots, 4. \tag{27}$$

Now, let us multiply Eqs. 27 by the denominator k to transform them into polynomials. To simplify the system, let's

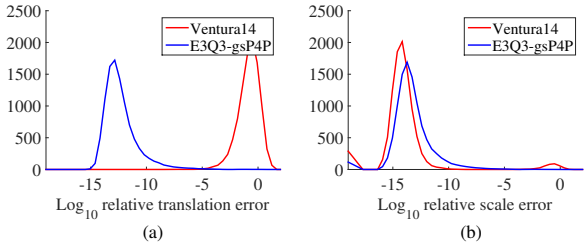

Figure 3: **gsP4P problem, planar scene:** (a) Log_{10} of the relative translation error and (b) relative scale error for planar scenes and noise free correspondences.

replace vector $k\mathbf{t}$ by vector $\hat{\mathbf{t}}$ of three new unknowns and scalar ks by a new scalar unknown $\hat{s}$. This leads to eight linearly independent equations in seven unknowns x, y, z, $\hat{\mathbf{t}}$, and $\hat{s}$:

$$[\mathbf{d}_i]_\times \left(\mathtt{R}'(\mathbf{x})\mathbf{Q}_i + \hat{\mathbf{t}} - \hat{s}\mathbf{P}_i\right) = \mathbf{0}, \; i = 1, \ldots, 4. \tag{28}$$

Since Eqs. 28 depend on $\hat{\mathbf{t}}_\mathtt{x}$ and $\hat{s}$ linearly, four of the equations can be used to eliminate these variables from the remaining four equations, *e.g*., using G-J elimination. After the elimination, we'll end up with a system of four quadratic equations in three unknowns x, y and z. This system is analogous to the final system of the P4Pf-E3Q3 solver. It is an over-determined system. Again, by taking any three of the four equations we get a 3Q3 system and we can solve for the rotation $\mathtt{R}(\mathbf{x})$ using the E3Q3 solver. In the final step, we resubstitute the obtained solutions for $\mathtt{R}(\mathbf{x})$ into Eqs. 27 and determine the unknown translation $\mathbf{t}$ and scale s by solving this linear system of equations.

Experiments. As in the case of the P4Pf problem, we tested our new E3Q3-gsP4P formulation on synthetic data only. Here, we compared the numerical stability and precision to the state-of-the-art gsP4P solver Ventura14 [37].

In the numerical stability experiment, we generated 10K scenes with four 3D points distributed at random in a $[-10, 10]^3$ cube or on a plane, depending on the configuration. For each scene, we generated four cameras with random but feasible positions and orientations and fixed focal lengths. To simulate a generalized camera, we used the generated 3D points as the anchor points $\mathbf{Q}'_i$, the centres of the four cameras as the starting points of rays $\mathbf{P}'_i$, and we

set the rays directions as $\mathbf{d}_i = (\mathbf{Q}'_i - \mathbf{P}'_i) \,/\, \|\mathbf{Q}'_i - \mathbf{P}'_i\|$. Finally, we applied a random feasible transformation to generate the input 3D points $\mathbf{Q}_i = \mathtt{R}_{gt}^\top(\mathbf{Q}'_i - \mathbf{t}_{gt})$ and the starting points $\mathbf{P}_i = \frac{\mathbf{P}'_i}{s_{gt}}$ from Eq. 25.

Figures 2(a–b) show $\log_{10}$ of the relative translation error and the relative scale error obtained by selecting the real roots closest to the ground truth values $\mathbf{t}_{gt}$ and s_{gt} for a general 3D scene. Plots in Figure 3 show the same errors for the planar scene. Errors for the rotation are not displayed due to lack of space. We can see that both solvers, the new E3Q3-gsP4P solver (blue) and the state-of-the-art Ventura14 [37] (red), have similar numerical stability. Moreover, the E3Q3-gsP4P solver works also for planar scenes for which the Ventura14 solver returns only the correct scale (Fig. 3(b)), not the correct rotation and translation (Fig. 3(a)).

In the next experiment, we have studied the accuracy of our new E3Q3-gsP4P solver in the presence of noise added to image rays. The noise was generated as noise in image points of the original pinhole cameras, assuming 1000 px×1000 px image. Figure 1(b) shows the results for general 3D scene with different levels of noise added to the image projections. In this case, the rotation error was measured as the rotation angle in the angle-axis representation of the relative rotation $\mathtt{R}\mathtt{R}_{gt}^{-1}$ For each noise level, from 0 to 1 pixel, 1000 estimates for random scenes and random feasible camera positions and transformations of generalized camera were made. The results are plotted using `boxplot` function in Figure 2(c–d). We can see that the Ventura14 solver (red) returns slightly more accurate estimates for larger noise levels, however, the results of both solvers are very precise even for the noise level of 1 px.

5.3. Hand-eye calibration

The last problem for which we propose a 3Q3 formulation is the problem of hand-eye calibration (HEC) with known translation. The HEC problem [33, 36] appeared for the first time in the connection with cameras mounted on robotic systems. Since then, it arose in many other fields ranging from medical applications to automotive industry. The HEC task is to find a rigid transformation `X` from the coordinate system connected with the robot's gripper to the coordinate system of a rigidly attached camera.

We consider a variation of the HEC problem where the rotation of the gripper w.r.t. the robot global coordinate system is not known, however its translation can be measured. This variation was recently solved using the Gröbner basis method by Kukelova *et al*. [21], who formulated the problem using quaternions as a system of seven equations in seven unknowns with 16 solutions. The final Gröbner basis solver needs to perform G-J elimination of a 187×203 matrix and to compute eigenvalues for a 16×16 matrix. In Appendix §A.3, we will show that this variation of the HEC problem can be formulated as a much simpler 3Q3 system, again using Cayley's parametrization of rotations and that this new E3Q3 solution is a slightly more stable and noise resistant than [21].

6. Solver speedups

Here, we present the speedups provided by the new E3Q3-based solvers from the previous sections over the SOTA solvers [2, 37, 21]. Unfortunately, we do not have C++ implementations of the SOTA solvers, thus we compared only the times of the major steps performed by each solver using C++ implementations based on Eigen linear algebra library [1]. The timings, averaged over 10K trials on a 3.5 GHz i7 based desktop, are reported in the following table:

Method	SVD	G-J	Eig	E3Q3	Total	Speedup
Bujnak12	(3×3) 1	(78×88) 164	(10×10) 17	-	185.0	1.0
E3Q3-P4Pf	(4×8) 3.9	(4×8) 0.3	-	2.9	7.1	**26.0**
Ventura14	(8×13) 17	(48×56) 42	(8×8) 13	-	72.0	1.0
E3Q3-gsP4P	-	(8×14) 0.7	-	3.2	3.9	**18.5**
Kukelova12	-	(187×203) 1756	(16×16) 56	-	1812.0	1.0
E3Q3-HEC	-	(6×13) 0.6	-	3.6	4.2	**431.7**

All timings are reported in μs. The bracketed dimensions represent the sizes of the respective input matrices. The E3Q3 solver uses Hartley's Sturm sequences implementation [14], with relative error set to 10^{-14}. The different timings of the E3Q3 solver are due to the different average number of solutions. Note that the eigenvalue computation step (Eig) could be speeded by up to the factor of four by using the method proposed in [5]. Still, this would not significantly affect the overall run times.

7. Conclusion

The problem of finding the intersections of three quadrics appears in many places in computer vision and robotics, often through one of the generic sources: Euclidean distances, rotation parameterization, or nullspace vectors. The main contribution of the paper is the new solution to the 3Q3 problem called E3Q3. This new method is simpler, more efficient, and significantly faster than the previously proposed solutions. The E3Q3 solver can handle degenerate systems as well as systems with symmetric solutions and solutions with multiplicity, see §A.2.

Further, we have described several recent works where the original problem led to a 3Q3 formulation and where our new solver can readily deliver faster solutions. Last but not least, we have proposed new formulations to three important computer vision problems. These new algorithms, while being comparably stable and noise-resistant, significantly outperform the state-of-the-art solutions in terms of speed.

References

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

[2] M. Bujnak. *Algebraic solutions to absolute pose problems*. PhD thesis, Czech Technical University in Prague, September 2012. 5, 6, 8

[3] M. Bujnak, Z. Kukelova, and T. Pajdla. A general solution to the P4P problem for camera with unknown focal length. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008. 1, 5, 6

[4] M. Bujnak, Z. Kukelova, and T. Pajdla. New efficient solution to the absolute pose problem for camera with unknown focal length and radial distortion. In *Computer Vision ACCV 2010*, volume 6492, pages 11–24. 2011. 6

[5] M. Bujnak, Z. Kukelova, and T. Pajdla. Making minimal solvers fast. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1506–1513, June 2012. 3, 8

[6] M. Byröd, K. Josephson, and K. Åström. A column-pivoting based strategy for monomial ordering in numerical Gröbner basis calculations. In *Computer Vision ECCV 2008*, volume 5305, pages 130–143. Springer Berlin Heidelberg, 2008. 1

[7] M. Byröd, Z. Kukelova, K. Josephson, T. Pajdla, and K. Åström. Fast and robust numerical solutions to minimal problems for cameras with radial distortion. In *Computer Vision and Image Understanding*, 2010. 1

[8] E.-W. Chionh, R. N. Goldman, and J. R. Miller. Using multivariate resultants to find the intersection of three quadric surfaces. *ACM Trans. Graph.*, 10(4):378–400, 1991. 2, 3

[9] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Springer, 2nd edition, 2005. 1, 2

[10] A. Danilevskii. The numerical solution of the secular equation (in Russian). In *Mat. Sbornik*, pages 169–171. 1937. 3

[11] L. Dupont, D. Lazard, S. Lazard, and S. Petitjean. Near-Optimal Parameterization of the Intersection of Quadrics. In *19th Symposium on Computational Geometry*, 2003. 2

[12] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981. 1, 3, 4

[13] Y. Guo. A novel solution to the P4P problem for an uncalibrated camera. *Journal of Mathematical Imaging and Vision*, 45(2):186–198, 2013. 3, 4

[14] R. Hartley and H. Li. An efficient hidden variable approach to minimal-case camera motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2303–2314, 2012. 8

[15] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge U. Press, 2^{nd} edition, 2004. 5

[16] M. Hazewinkel. *Encyclopaedia of mathematics*. Kluwer Academic Publishers, Dordrecht Boston, 1997. 4, 7

[17] R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge university press, 2012. 3

[18] M. Kettner. *Algorithmic and topological aspects of semi-algebraic sets defined by quadratic polynomials*. PhD thesis, School of Mathematics, Georgia Tech, 2007. 1

[19] Y. Kuang, J. E. Solem, F. Kahl, and K. Åström. Minimal solvers for relative pose with a single unknown radial distortion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 1

[20] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. In *ECCV, Part III*, volume 5304 of *Lecture Notes in Computer Science*, 2008. 1, 2, 3, 5, 7

[21] Z. Kukelova, J. Heller, and T. Pajdla. Hand-eye calibration without hand orientation measurement using minimal solution. In *ACCV 2012, 11th Asian Conference on Computer Vision*, pages 576–589, November 2013. 8

[22] Z. Kukelova, T. Pajdla, and M. Bujnak. Fast and stable algebraic solution to L_2 three-view triangulation. In *International Conference on 3D Vision*, pages 326–333, 2013. 1

[23] G. Lee, B. Li, M. Pollefeys, and F. Fraundorfer. Minimal solutions for pose estimation of a multi-camera system. In *International Symposium on Robotics Research*, 2013. 3

[24] J. Z. Levin. Mathematical models for determining the intersections of quadric surfaces. *Computer Graphics and Image Processing*, 11(1):73–87, 1979. 2

[25] P. Miraldo and H. Araujo. Direct solution to the minimal generalized pose. *Transactions on Cybernetics*, 2015. 4

[26] D. Nister. An efficient solution to the five-point relative pose problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):756–770, June 2004. 1

[27] D. Nister. A minimal solution to the generalised 3-point pose problem. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004*, volume 1, pages 560–567, June 2004. 4

[28] S. Ramalingam, S. Bouaziz, and P. Sturm. Pose estimation using both points and lines for geo-localization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4716–4723, May 2011. 4, 5

[29] S. Ramalingam, S. K. Lodha, and P. Sturm. A generic structure-from-motion framework. *Comput. Vis. Image Underst.*, 103(3):218–228, Sept. 2006. 3, 4

[30] S. Ramalingam, Y. Taguchi, T. K. Marks, and O. Tuzel. P2Pi: A minimal solution for registration of 3D points to 3D planes. In *ECCV*, pages 436–449, 2010. 2, 3, 5

[31] M. Reid. *The complete intersection of two or more quadrics*. PhD thesis, Department of Pure Mathematics, Cambridge University, 1972. 1

[32] B. Roth. Computations in kinematics. In J. Angeles, G. Hommel, and P. Kovcs, editors, *Computational Kinematics*, volume 28 of *Solid Mechanics and Its Applications*, pages 3–14. Springer Netherlands, 1993. 1, 2, 3, 4

[33] Y. Shiu and S. Ahmad. Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form AX=XB. *Robotics and Automation, IEEE Transactions on*, 5(1):16–29, Feb 1989. 8

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

[35] H. Stewénius, F. Schaffalitzky, and D. Nistér. How hard is three-view triangulation really? In *International Conference on Computer Vision*, volume 1, pages 686–693, Oct. 2005. 1

[36] R. Tsai and R. Lenz. A new technique for fully autonomous and efficient 3D robotics hand/eye calibration. *IEEE Transactions on Robotics and Automation*, 5:345 –358, 1989. 8

[37] J. Ventura, C. Arth, G. Reitmayr, and D. Schmalstieg. A minimal solution to the generalized pose-and-scale problem. In *CVPR*, June 2014. 7, 8

[38] Z.-G. Xu, X. Wang, X.-D. Chen, and J.-G. Sun. A robust algorithm for finding the real intersections of three quadric surfaces. *Computer Aided Geometric Design*, 22(6):515 – 530, 2005. 2, 3

[39] C.-K. Yap. *Fundamental problems of algorithmic algebra*, volume 49. Oxford University Press Oxford, 2000. 3