QuEst: A Quaternion-Based Approach for Camera Motion Estimation from Minimal Feature Points

Kaveh Fathian, J. Pablo Ramirez-Paredes, Emily A. Doucette, J. Willard Curtis, Nicholas R. Gans.

***Abstract*—In this paper, we consider the problem of recovering the rotation and translation changes of a moving camera from captured images. This problem is traditionally solved using the epipolar constraint, where the rotation and translation changes are recovered from the essential matrix. We propose a new formulation based on quaternion representation of rotation. Using this formulation, we recover the rotation and translation changes separately without resorting to an essential matrix. We compare the estimation accuracy and execution time of the proposed method with several state-of-the-art algorithms using both synthetic and actual image sequences. On average, our approach shows better accuracy in the presence of noise, but it has a larger execution time. For the benefit of community, we have made the implementation of our algorithm available online and free.**

***Index Terms*—Computer Vision for Automation, Computer Vision for Transportation, Computer Vision for Other Robotic Applications**

Supplementary Material

The implementation of the algorithm and code for generating benchmark results is available at https://goo.gl/QH5qhw.

I. Introduction

MANY applications in computer vision and robotics require measurements of the rotation and translation (i.e., *pose*) changes of a moving object. In photogrammetry, for example, by knowing the pose changes of the camera, a 3D model of a scene can be constructed from a set of 2D images [1]. In robotics, camera motion estimated from images can be used for navigation [2] or fused with other sensor measurements (e.g., IMU and GPS) to increase the reliability and accuracy [3]. Camera motion estimation has applications in simultaneous localization and mapping (SLAM) [4], autonomous vehicles [5], and augmented reality [6].

Camera motion estimation techniques are often based on image features (e.g., edges, corners, etc.) that can be detected and matched in two or more images [7]. Fig. 1 shows an example where feature points are detected and matched as indicated by yellow lines in two images. Many existing methods [8]–[14] use the coordinates of feature points to construct an essential matrix, from which relative rotation and translation of the camera can be recovered.

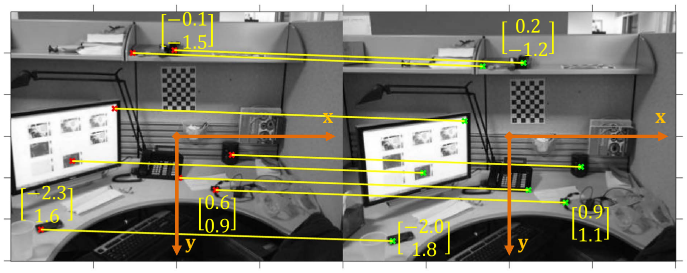
$\begin{bmatrix}-0.1\\-1.5\end{bmatrix}$
$\begin{bmatrix}0.2\\-1.2\end{bmatrix}$
x
x
$\begin{bmatrix}-2.3\\1.6\end{bmatrix}$
$\begin{bmatrix}0.6\\0.9\end{bmatrix}$
y
$\begin{bmatrix}-2.0\\1.8\end{bmatrix}$
y
$\begin{bmatrix}0.9\\1.1\end{bmatrix}$

Fig. 1. Pairs of matched feature points across two images shown side by side (image courtesy of MathWorks).

It is known that a minimum of five matched feature points in general positions in the space are needed to estimate the relative pose [9]. Algorithms that are based on five points must solve a nonlinear problem (i.e., a system of polynomial equations), whereas algorithms based on more than five points can recover the pose from a linear formulation. Consequently, former algorithms are in general slower than the latter. Unfortunately, algorithms that are based on more than five points often suffer from degeneracies and cannot recover the relative pose in many important scenarios. For example, the 8-point algorithm [8] fails when points are coplanar or when the camera only had a rotational motion. On the other hand, a five point algorithm can recover the correct solution in such cases due to exploiting all available geometric constraints of the problem.

In this work, we present a novel mathematical formulation of the camera motion estimation using quaternions and propose the Quaternion Estimation (QuEst) algorithm to recover the rotation and translation (up to a scale factor) from five or more feature points. We compare the performance of QuEst with existing algorithms in the presence of noise in feature point coordinates and for various numbers of feature points. The performance is further vetted by using real-world image datasets that come with the ground truth camera pose information.

The main contributions and benefits of this work can be summarized as follows.

- The performance of several state-of-the-art pose estimation algorithms are thoroughly compared. The estimation accuracy of QuEst is better than the best existing algo-

Manuscript received September 19, 2017; Revised November 3, 2017; Accepted December 20, 2017. This paper was recommended for publication by Editor Francois Chaumette upon evaluation of the Associate Editor and reviewers' comments. This work was supported in part by the OSD Sponsored Autonomy Research Pilot Initiative, RW Autonomy Initiative, and U.S. Air Force Research Laboratory under Grant FA8651-17-1-0001.

K. Fathian and N. R. Gans are with the Department of Electrical Engineering, University of Texas at Dallas, Richardson, TX, 75080 USA. E-mail: {kaveh.fathian, ngans}@utdallas.edu.

JP Ramirez-Paredes is with the Department of Electronics Engineering, University of Guanajuato, Salamanca, Gto, 36885, Mexico. E-mail: jpi.ramirez@ugto.mx.

E. A. Doucette and J. W. Curtis are with the Air Force Research Laboratory, Munitions Directorate, Eglin AFB, FL, 32542, USA. E-mail: {emily.doucette, jess.curtis}@us.af.mil.

Digital Object Identifier (DOI): see top of this page.

rithms in all of our comparisons.

- QuEst is based on five points and recovers the pose correctly when points are on critical surfaces (e.g., coplanar points. See p.169 in [15] for more details). QuEst can provide a reliable pose estimate for SLAM applications that require an accurate pose estimate to initialize the bundle adjustment [4].
- QuEst recovers translation and depths of the feature points simultaneously and up to a common scale factor. Due to this common scale factor, the magnitude of the recovered translation goes to zero as the distance between the camera views vanishes. This property is desirable in applications such as visual servoing [16]. Note that translation recovered from an essential matrix can be rescaled to have this property, however it requires an additional step.
- The implementation of QuEst is freely available online.

The rest of the paper is organized as follows. We briefly review related approaches in Section II, before we formulate the pose estimation problem using quaternions in Section III. We present the QuEst algorithm in Section IV and evaluate its performance under noise in Section V. In Section IV, we further vet the performance of QuEst using the real-world image datasets.

II. Related Work

The 8-point algorithm [8], [9] is one of the first practical camera motion estimation algorithms that used the epipolar constraint, embodied in the essential matrix, to recover the pose from a set of linear equations that are generated from eight or more feature points. Philip [17] showed that when six or more general points are available, the pose estimation problem is linear and produces a unique solution.

Work by [9], [10], [18]–[20] showed that the relative pose can be estimated from a minimum of five feature points, where in this case up to ten solutions candidates are acquired. Nister [11] formulated the 5-point pose estimation problem as a system of polynomial equations in four unknown variables. A practical algorithm was then proposed to reduce this system to a tenth degree polynomial in one variable, and from its roots the solutions of the original system are acquired. Li and Hartley [12] produced a version of Nister's algorithm that is more numerically stable and has faster execution time. The algorithm proposed by Stewenius et al. [13] used Gröbner bases to solve the polynomial system. Their algorithm has better numerical stability and recovers the pose correctly when the relative translation between the camera views is zero. Kukelova et al. [14] used the polynomial eigenvalue method to find the solutions.

What these previous work have in common is the use of the Essential matrix in the problem formulation. In recent years, new algorithms have been developed that estimate the rotation independently of the translation, i.e., without resorting to the Essential matrix (which entangles the rotation and translation). Kalantari et al. [21] used Gröbner bases to solve a formulation of the 5-point problem with independent rotation and translation estimation. Their method is based on finding the bases online, which is inefficient since the path for finding the bases remains constant for a specific polynomial system. Kneip et al. [22] proposed a more efficient algorithm based on Gröbner bases. However the algorithm is sensitive in the case of small translation.

We initially used an optimization-based method to solve the quaternion formulation of the problem [23]. This initial work was reliant on an accurate initial estimate for the relative pose. We then proposed the QuEst algorithm to solve the problem from six or more feature points [24]. In this work we extend QuEst to solve the problem for the minimal case of five points. QuEst is based on formulating an eigenvalue problem, which does not require an initial guess for the solution. We compare the performance of QuEst with all aforementioned algorithms except for Kalantari et al. [21], for which we could not find an open-source implementation. To avoid possible implementation mistakes that could misrepresent a work, we have only used algorithms that have an open-source implementation provided by their authors.

Lastly, we should mention that in the special case where feature points are coplanar, methods based on the homography matrix can be used to estimate the pose [10], [25]. Since homography does not work for general point configurations it is not used in our comparisons.

III. Camera Motion Estimation

We introduce the notation and assumptions, followed by formulating the pose estimation problem in quaternions.

A. *Assumptions and Notation*

Throughout the paper, we assume that the camera calibration matrix is known; this matrix can be easily found through the existing camera calibration routines [26].

Scalars are represented by lower case (e.g., s), vectors by lowercase and bold (e.g., $\mathbf{v}$), and matrices by upper case and bold letters (e.g., $\mathbf{M}$). All vectors are column vectors. The Moore-Penrose pseudoinverse of matrix $\mathbf{M}$ is shown by $\mathbf{M}^{\dagger}$. Binomial coefficients are denoted by $\binom{n}{k} := \frac{n!}{k!(n-k)!}$. By norm of a vector, we imply the l^2-norm. Degree 4 *monomials* in variables w, x, y, z are single term polynomials $w^a x^b y^c z^d$, such that $a+b+c+d = 4$, for $a,b,c,d \in \{0,1,2,3,4\}$.

B. *Problem Formulation*

Consider images of a scene taken by a camera at two views (see Fig. 1). Let $\mathbf{R} \in \mathrm{SO}(3)$ and $\mathbf{t} \in \mathbb{R}^3$ respectively represent the relative rotation and translation of the camera frame between the views. Assume that feature points are detected and matched, and their x-y coordinates in the image plane are extracted from the images. In practice, the coordinates are first obtained in pixels and should be mapped via the camera calibration matrix to Cartesian coordinates on the image plane.

For each pair of matched feature points, the rigid motion constraint

$$u\mathbf{R}\mathbf{m} + \mathbf{t} \;=\; v\mathbf{n} \tag{1}$$

must hold, in which $\mathbf{m}, \mathbf{n} \in \mathbb{R}^3$ are homogeneous coordinates (i.e., a 1 is appended to the x-y coordinates) of the feature

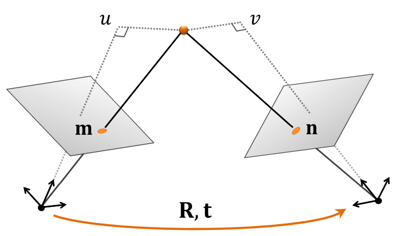

Fig. 2. Projection of a 3D point onto the image plane at two views.

point in two images. Scalars u and v represent depths of the 3D point at each view, and as shown in Fig. 2, are the distance from the 3D point to the origin along the z-axis of the camera coordinate frame. Point coordinates $\mathbf{m}$ and $\mathbf{n}$ are known from the images, and the unknowns in (1) are u, v, $\mathbf{R}$, and $\mathbf{t}$, which need to be recovered. Example 1 in the Appendix demonstrates (1) written for the matched feature points in Fig. 1.

We need to point out that in the pose estimation problem, translation and depths of the points can only be recovered *up to a scale factor*. This can be seen from (1), where any constant multiplied into both hand sides can be absorbed into the unknown variables u, v, and $\mathbf{t}$.

Equation (1) uses the matrix representation of rotation, in which $\mathbf{R} \in \mathrm{SO}(3)$ is a 3×3 orthogonal matrix with determinant one. Representing the rotation in the matrix form with $\mathrm{SO}(3)$ constraints leads to a very nonlinear problem that is challenging to solve. Instead, we use quaternions, which represent a rotation by four elements $w, x, y, z \in \mathbb{R}$ under the constraint that $w^2 + x^2 + y^2 + z^2 = 1$. Although (1) can be formulated directly in quaternions, for simplicity and to avoid introducing the quaternion algebra we only mention what is essential to solve the problem here: if $w, x, y, z \in \mathbb{R}$ are elements of a rotation quaternion, the associated rotation matrix is given by

$$\mathbf{R} = \begin{bmatrix} w^2+x^2-y^2-z^2 & 2(xy-wz) & 2(xz+wy) \\ 2(xy+wz) & w^2-x^2+y^2-z^2 & 2(yz-wx) \\ 2(xz-wy) & 2(yz+wx) & w^2-x^2-y^2+z^2 \end{bmatrix}. \tag{2}$$

Quaternions provide a singularity free representation of rotation, and by restricting the first element to nonnegative numbers (i.e., $w \geq 0$), there is a one to one and onto correspondence between rotation matrices and quaternions.

To recover the pose, we first eliminate the unknowns u, v and $\mathbf{t}$, and derive a system of equations in terms of the quaternion elements. From solving this system, all rotation solution candidates are found. Subsequently, the translation and depths of the points are recovered. By taking (1) for two different pairs of feature points and subtracting the equations, $\mathbf{t}$ can be eliminated. Repeating this using a third feature point pair and one of the previous pairs results in two equations with $\mathbf{t}$ removed. Depths can be removed by rewriting the resulting two equations in the matrix-vector form and noting that the depths form a null vector for the matrix. The determinant of this matrix, which consists of only the point coordinates and rotation unknowns, must equal zero. This gives a polynomial equation of degree four in quaternion variables w, x, y, z, with coefficients that are in terms of the feature point coordinates. Example 2 in the Appendix explicates this procedure. Note that there are 35 degree four terms in quaternion variables (called monomials). In general, the number of monomials of degree m in n variables is $\binom{m+n-1}{n-1}$.

Since any three feature points give a polynomial equation, from n points $\binom{n}{3}$ equations are generated. These equations are used in the next section to recover the pose.

IV. The QuEst Algorithm

In what follows we first show how rotation can be recovered for five (or more) matched feature points. Given a rotation solution candidate, it is then shown how the associated translation vector and depths can be recovered.

A. *Recovering Relative Rotation*

Consider $\binom{5}{3} = 10$ degree four equations that are generated from five feature points. Multiplying these equations by w, x, y, and z results in 40 degree five equations. The multiplication creates more equations, which allow us to solve the polynomial system. This new system of equations can be stacked into the matrix-vector form $\mathbf{A}\,\mathbf{x} = \mathbf{0}$, where matrix $\mathbf{A} \in \mathbb{R}^{40\times 56}$ consists of the coefficients, and vector $\mathbf{x} \in \mathbb{R}^{56}$ consists of all degree five monomials (the number of degree five monomials in w, x, y, z is $\binom{8}{3} = 56$). Example 3 in the Appendix demonstrates what we have mentioned above.

To solve the new system of equations, we split $\mathbf{x} \in \mathbb{R}^{56}$ into two vectors $\mathbf{x}_1 \in \mathbb{R}^{35}$ and $\mathbf{x}_2 \in \mathbb{R}^{21}$, where $\mathbf{x}_1$ is the vector of all monomials that contain a power of w (e.g., $w^5, w^4x, w^4y, \ldots, wy^4, wz^4$), and $\mathbf{x}_2$ consists of the rest of the monomials (e.g., $x^5, x^4y, x^3y^2, \ldots, yz^4, z^5$). Let $\mathbf{A} = [\mathbf{A}_1\ \mathbf{A}_2]$, where $\mathbf{A}_1 \in \mathbb{R}^{40\times 35}$ consists of the columns of $\mathbf{A}$ that are associated with $\mathbf{x}_1$, and $\mathbf{A}_2 \in \mathbb{R}^{40\times 21}$ is the columns that are associated with $\mathbf{x}_2$. The system $\mathbf{A}\mathbf{x} = \mathbf{0}$ can be equivalently written as

$$\mathbf{A}_1\,\mathbf{x}_1 + \mathbf{A}_2\,\mathbf{x}_2 = \mathbf{0}. \tag{3}$$

By multiplying the pseudoinverse of $\mathbf{A}_2$ in (3) we obtain

$$\mathbf{x}_2 = -\mathbf{A}_2^{\dagger}\mathbf{A}_1\,\mathbf{x}_1. \tag{4}$$

Vector $\mathbf{x}_1$ consists of monomials that have at least one power of w. Thus, by factoring out w and representing the remaining vector by $\mathbf{v} \in \mathbb{R}^{35}$, i.e., $\mathbf{v} = \frac{1}{w}\mathbf{x}_1$, (4) can be written as

$$\mathbf{x}_2 = w\,\bar{\mathbf{B}}\,\mathbf{v}, \tag{5}$$

where $\bar{\mathbf{B}} := -\mathbf{A}_2^{\dagger}\mathbf{A}_1 \in \mathbb{R}^{21\times 35}$.

Equation (5) allows us to construct a square matrix $\mathbf{B} \in \mathbb{R}^{35\times 35}$ and build an eigenvalue problem of the form $\lambda\mathbf{v} = \mathbf{B}\mathbf{v}$. Indeed, let us choose $\lambda = \frac{x}{w}$, and consider the eigenvalue problem

$$x\,\mathbf{v} = w\,\mathbf{B}\,\mathbf{v}. \tag{6}$$

The entries of vector $x\mathbf{v}$ either belong to $\mathbf{x}_2$ or $\mathbf{x}_1$. For entries that belong to $\mathbf{x}_2$, the associated rows of $\mathbf{B}$ are chosen from the corresponding rows of $\bar{\mathbf{B}}$. For entries that belong to $\mathbf{x}_1$, rows of $\mathbf{B}$ are chosen as an appropriate natural basis row vector, i.e., a vector with exactly one entry of 1 and zeros elsewhere. Example 4 in the Appendix explicates this procedure.

Once the eigenvalue problem (6) is constructed, 35 solution candidates for $\mathbf{v}$ are derived by computing the eigenvectors of $\mathbf{B}$. Variables w, x, y, z are then computed from the entries of $\mathbf{v}$ (e.g., fourth root of the w^4 entry in $\mathbf{v}$ gives w). Lastly, the recovered solutions are normalized to meet the unit norm constraint $w^2+x^2+y^2+z^2=1$. Since the 5-point problem has up to 20 solutions (including solutions with negative depths), 15 of the recovered solutions do not satisfy the original degree four system and can be easily discarded by plugging the solutions in the original system. Note that solutions corresponding to points behind the camera can be discarded after the depths are recovered. This leaves only up to 10 possible solutions.

Note that multiplying the original equations by w,x,y,z makes $\mathbf{A}_2$ a tall matrix for which the pseudoinverse multiplication $\mathbf{A}_2^\dagger \mathbf{A}_2$ gives identity. The reader can verify that the approach above could not have been used with the original degree four system. In the face of inaccuracies or noise in the feature point coordinates, the pseudoinverse operation $\mathbf{A}_2^\dagger \mathbf{A}_1$ in (4) relates $\mathbf{x}_2$ and $\mathbf{x}_1$ in the least square sense. When more than five feature points are available, matrices $\mathbf{A}_1,\mathbf{A}_2$ become taller. Consequently, noise and inaccuracies are suppressed further through the pseudoinverse operation, and the estimation accuracy of rotation improves. Finally, by choosing $\mathbf{x}_1$ or λ differently (e.g., $\lambda=\frac{y}{w}$) it is possible to derive different eigenvalue problems, however the pose solutions remain the same.

B. Recovering Relative Translation and Depths

Once quaternion elements w,x,y,z are recovered, the corresponding rotation matrix $\mathbf{R}$ is given by (2). Having $\mathbf{R}$, the rigid motion constraint $u\mathbf{R}\mathbf{m}+\mathbf{t}=v\mathbf{n}$ can be written for all matched feature points and stacked into the matrix-vector form

$$\underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{R}\mathbf{m}_1 & -\mathbf{n}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{R}\mathbf{m}_2 & -\mathbf{n}_2 & & \mathbf{0} & \mathbf{0} \\ \vdots & & \vdots & & & \ddots & & \vdots \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{R}\mathbf{m}_k & -\mathbf{n}_k \end{bmatrix}}_{\mathbf{C}} \underbrace{\begin{bmatrix} \mathbf{t} \\ u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \\ u_k \\ v_k \end{bmatrix}}_{\mathbf{y}} = \mathbf{0} \tag{7}$$

where $\mathbf{I}\in\mathbb{R}^{3\times 3}$ is the identity matrix, k is the number feature points, $\mathbf{C}\in\mathbb{R}^{3k\times 2k+3}$, and $\mathbf{y}\in\mathbb{R}^{2k+3}$. Equation (7) implies that $\mathbf{y}$ is in the null space of $\mathbf{C}$. Thus, $\mathbf{y}$ is found by calculating the rightmost singular vector of $\mathbf{C}$ (i.e., eigenvector of $\mathbf{C}^\top\mathbf{C}$ corresponding to the zero eigenvalue). Notice that $\mathbf{y}$ consists of the translation vector and feature point depths. Therefore, these parameters are recovered simultaneously and with a common scale factor.

V. PERFORMANCE BENCHMARKS

In this section, we benchmark QuEst and several existing algorithms using synthetic images in a Monte Carlo framework. To this end, a mixed set of general and coplanar 3D points are randomly generated with uniform distribution respectively inside a rectangular parallelepiped and on a plane in front of the camera at the initial location. The camera is then moved to a second location by a random translation and rotation quaternion, with uniform distribution within a bounded box of $\mathbb{R}^3$ and on the 3-sphere, respectively. Feature points are generated by projecting the 3D points on the image planes according to the pinhole camera model. We assume that each image is 1024×768 pixels, and the calibration matrix is set as

$$\mathbf{K}=\begin{bmatrix} 1060 & 0 & 514 \\ 0 & 1060 & 384 \\ 0 & 0 & 1 \end{bmatrix}. \tag{8}$$

For the comparison, we have used the 8-point [8], 7-point, and 6-point [15] algorithms, together with the 5-point algorithms by Nister [11], Li [12], Stewenius [13], Kneip [22], and Kukelova [14]. To avoid any implementation mistakes that could misrepresent a work, we have only used open-source implementations of the aforementioned algorithms provided by the authors themselves [27]–[30], and we did not include the work by Kalantari et al. [21] for which we could not find an implementation.

A. Noise Benchmark

To evaluate the performance under noise, Gaussian noise with zero mean and standard deviation ranging from 0 to 3 pixels is added to all image coordinates. The noise standard deviation is increased by 0.1 pixel increments, and for each noise increment, 100 random experiments are compared. The solution candidate that is closest to the ground truth is chosen as the estimated pose for each algorithm. To pick this solution, the metric

$$\rho(\mathbf{q},\mathbf{q}^*)=\frac{1}{\pi}\arccos(\mathrm{dot}(\mathbf{q},\mathbf{q}^*))\in[0,1] \tag{9}$$

is used to define the rotation error [31], where $\mathbf{q}=[w\ x\ y\ z]^\top$ is the rotation quaternion estimated from the noisy images, and $\mathbf{q}^*$ is the ground truth rotation. Similarly, the estimation error for translation is defined by replacing quaternion vectors by the unit norm translation vectors in (9). Note that the recovered translation is normalized because the translation can only be recovered up to a scale factor, and this scale factor may vary for each algorithm.

Fig. 3 compares the mean rotation and translation estimation errors at different noise standard deviations between QuEst and algorithms that are based on more than five feature points. The number of points used in each algorithm is equal to the minimum number that the algorithm needs to return a pose solution, e.g., QuEst is provided with 5 points and the 8-point algorithm is provided with 3 additional points. Both rotation and translation estimation accuracy of QuEst outperform these algorithms. In Fig. 4, a similar comparison is performed between QuEst and 5-point algorithms, where all algorithms are provided with the same set of points. Kneip's algorithm has a higher rotation error compared to other algorithms. Stewenius's algorithm and QuEst show the best performance, and their associated lines in the figure roughly overlap.

We have to point out that the same set of random experiments were used to generate the results in Figs. 3 and 4. Thus, the lines associated to QuEst are identical in both figures. This allows the reader to compare the performance of other 5-point algorithms with algorithms that need more points.

B. Number of Points Benchmark

The number of feature points that can be detected and matched between two images are often more than the minimum number required by an algorithm. Therefore, in practice,

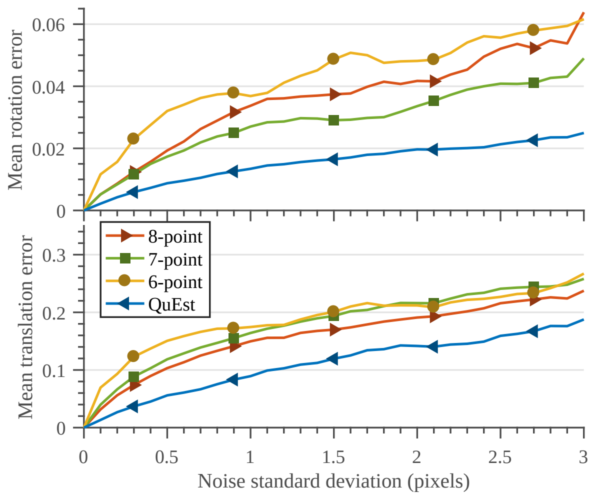

Fig. 3. Estimation error of QuEst and algorithms that need more feature points vs. Gaussian noise on feature point coordinates. Each algorithms is provided with the minimum required number of points.

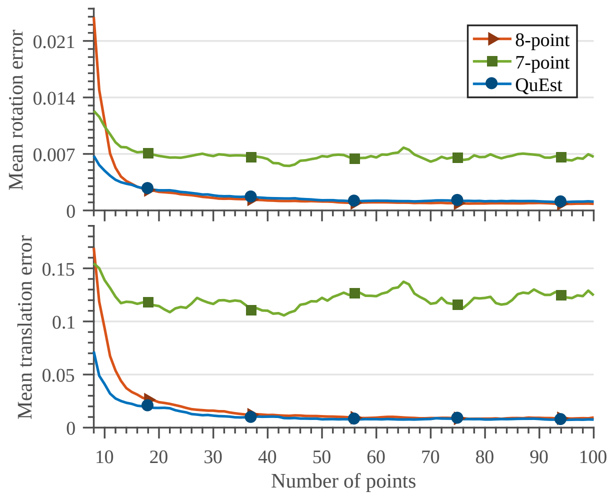

Fig. 5. Estimation error for fixed value of noise vs. number of provided matched feature points. The results for the 6-point algorithm has a large error that is outside of the figure limits and is not included.

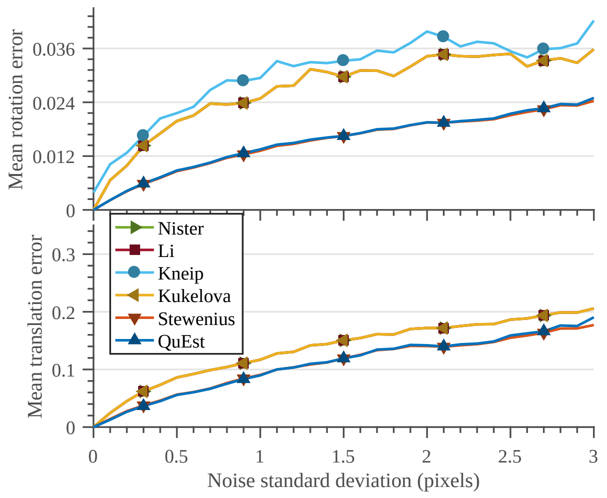

Fig. 4. Estimation error of five point algorithms vs. Gaussian noise on feature point coordinates. All algorithms are provided with five feature points. The pose recovered by Nister's, Li's, and Kukelova's algorithms are nearly identical and therefore their associated lines overlap. Kneip's algorithm only returns the rotation and translation error is not compared for this algorithm. Stewenius's algorithm and QuEst show the best performance and their associated lines almost overlap.

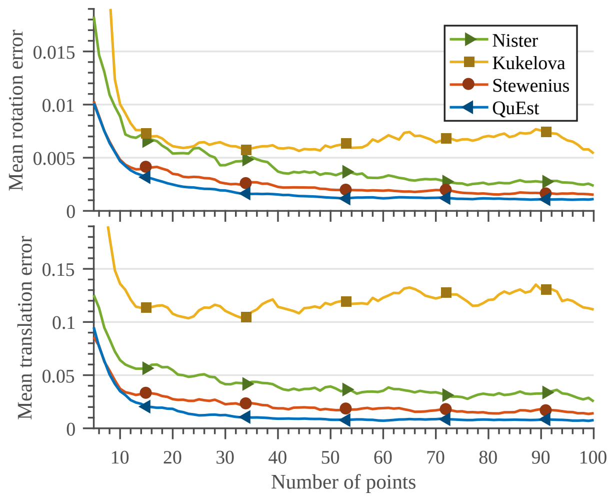

Fig. 6. Estimation error for fixed value of noise vs. number of provided matched feature points. The results for Kneip's algorithm has a large error that is outside of the figure limits and is not included. Li's implementation does not accept more than five feature points and is not compared.

pose estimation algorithms often solve an overdetermined problem with more than the minimum number of points to reduce the effects of noise and false matches. Outlier rejection methods such as Random Sample Consensus (RANSAC) solve an overdetermined problem with a determined set of inliers. As removal of all outliers can never be guaranteed, the estimated pose should satisfy the rigid motion constraint for all points as closely as possible.

To test the performance when more than the minimum number of feature points are provided, 100 experiments are conducted at each integer increment of the number of feature points. The number of points ranges from the minimum necessary to 100, and Gaussian noise with standard deviation of 0.75 pixels is added to the pixel coordinates to model the image pixelization noise and inaccuracies in feature point detection and matching.

Figure 5 shows the comparison results between QuEst and algorithms that need more than five points. As can be seen from the figure, when 15 or more feature points are used, the rotation estimate from QuEst and the 8-point algorithm become close and show the best performance. Their translation estimates become close for 35 or more feature points. Figure 6 shows similar comparison between the 5-point algorithms. The pose estimate from QuEst and Stewenius's algorithm are closely similar up to the 10 points, after which QuEst outperforms the Stewenius's algorithm.

One can compare the results in Figs. 5 and 6 for all algorithms, since the same set of random experiments were used. Thus, the results for QuEst are identical in both figures (note that the minimum number of points in Fig. 5 is eight whereas in Fig. 6 is five). These figures suggest that, in general, estimation error decreases as more points are provided to the algorithms.

TABLE I. Average execution time of algorithms when provided with minimum required number of points.

Algorithm	Average execution time (ms)
8-point	0.0428
7-point	**0.0282**
6-point	0.0829
5-point (Nister)	0.2559
5-point (Li)	**0.0287**
5-point (Kneip)	0.9298
5-point (Kukelova)	0.3727
5-point (Stewenius)	0.1046
5-point (QuEst)	0.8445

C. *Time Benchmark*

Table I lists the average execution time of all algorithms in milliseconds when minimum required number of points is used. To generate the results, 1000 random experiments with various noise magnitude are used. All implementations are executed as Mex files in Matlab and tested on the same platform with Intel's 4th Gen i-7 CPU. In general, algorithms that are based on more than five points have smaller execution time due to the linear formulation. In comparison, 5-point algorithms have a larger execution time due to their nonlinear nature, with the exception of Li's algorithm. Among the 5-point algorithms, Li's algorithm is the fastest, followed by the Stewenius' method. QuEst has the second longest execution time, however, we should emphasize that all algorithms are fast enough to be used in conjunction with RANSAC in real-time applications.

VI. real-world Performance

To further assess the performance of each algorithm, images from four real-world datasets are used to estimate the pose and compare the results with the ground truth that is provided by their authors. Datasets used for the comparison are ICL [32], KITTI [33], NAIST [34], and TUM [35]. Rather than processing every image in the dataset/video stream, we skip a fixed number of images (between 1 to 5 images depending on the dataset) before performing feature point detection and matching using SURF features. In this way, we increase the parallax and get a more reliable translation estimate for algorithms that are sensitive to parallax, e.g., the 8-point algorithm. Note that our choice for skipping frames does not affect the comparison results for 5-point algorithms. Feature points are preprocessed using RANSAC to reject incorrect matches, however they occasionally contain outliers. We provide all algorithms with the same set of feature points.

The ICL-NUIM dataset consists of computer-generated renderings of 3D scenes. These 640×480 pixel artificial images are accompanied by exact ground truth information, and their main purpose is to benchmark 3D reconstruction algorithms. The KITTI and TUM datasets were created to evaluate SLAM algorithms. The KITTI dataset consists of outdoor stereo 1226×370 pixel image sequences taken from a moving vehicle. The images are accompanied by LIDAR, IMU and GPS measurements, so full pose information is provided in the left camera frame. The TUM dataset is comprised of indoor 640×480 pixel images as well as point depth information from a RGB-D camera. The RGB-D camera poses were recorded by using an optical tracking system, with millimeter-level accuracy. The NAIST campus sequences are part of an Augmented Reality benchmark called TrakMark. The ground truth files for these sequences were created by solving a PnP problem from known 3D world point coordinates, acquired using high precision surveying equipment. The 1280×720 pixel images come from a handheld camera, with the operator walking during some sequences and running for others.

The comparison results are shown in Table II, where for each dataset the rotation and translation errors of all image sequences are computed and averaged. To reflect statistics about the errors, the median and 0.25 and 0.75 quantiles of the averaged errors are given in the table. The sequences labeled as TUM were challenging for every algorithm, since the camera is handheld and has erratic motion at times. The NAIST sequences, while generated by a handheld sensor, do not contain motions as abrupt as those in TUM. The performance of all algorithms on the NAIST sequences is better than for the TUM sequences. Most algorithms perform well on the ICL benchmark, due to its lack of image noise. The translation error for all algorithms is lowest for the KITTI sequences. This can be explained by the fairly large translation of the camera between consecutive frames, since the camera is mounted on a car and most of the time is moving with a high velocity along its optical axis.

As can be seen from Table II, QuEst respectively has from 3% to 36% and from 28% to 52% improved median accuracy for rotation and translation estimates over the Stewenius algorithm, which has the second best accuracy among the 5-point algorithms. Also, on average, Quest outperforms the 8-point algorithm in both rotation and translation estimate accuracy by as much as 54% lower median error.

VII. Conclusions

By using the quaternion representation of rotation, we formulated the camera pose estimation problem and presented the QuEst algorithm to recover the relative pose between two camera views. QuEst decouples the estimation of rotation and translation and can be used with a minimum of five feature points. Using both synthetic and real-world images, we compared the performance of several existing algorithms. In particular, we compared the accuracy of estimated pose for various noise level and number of feature points. On average, QuEst was shown to have the best accuracy among the compared algorithms.

VIII. Acknowledgments

We would like to thank David Liu, George Chen, Eric Luo, and Abhishek Nagar at Samsung Research America, and the anonymous reviewers for their valuable feedback that considerably improved this work.

Appendix

Example 1. Consider the pictures shown in Fig. 1, where feature points are matched, and their coordinates on the image

TABLE II. Median (Med), 0.25 quantile (Q1), and 0.75 quantile (Q3) for the rotation and translation estimation error associated to six algorithms tested on image datasets KITTI, TUM, ICL, and NAIST. Kneip's algorithm does not return a translation estimate, and is not included for translation error comparison. Li's implementation does not accept more than five feature points and is not compared.

		Rotation error × 1000						**Translation error × 10**				
		8-pt	**Nister**	**Kneip**	**Kukel.**	**Stew.**	**QuEst**	**8-pt**	**Nister**	**Kukel.**	**Stew.**	**QuEst**
KITTI	Med	0.4697	0.3197	0.898	0.3197	0.2933	**0.2155**	0.0669	2.4635	2.4635	0.084	**0.0596**
	Q1	0.2378	0.1801	0.4321	0.1801	0.1724	**0.1318**	0.0383	1.1119	1.1119	0.0443	**0.0358**
	Q3	0.9129	0.647	1.906	0.647	0.4913	**0.3452**	0.1256	4.09	4.09	0.1899	**0.1063**
TUM	Med	1.8289	1.3069	2.0601	1.3069	1.0415	**1.0099**	2.3068	2.9565	2.9565	2.1682	**1.455**
	Q1	1.0999	0.8096	1.2689	0.8096	0.6559	**0.6371**	1.2453	1.5268	1.5268	1.0206	**0.7945**
	Q3	3.012	2.1642	3.3846	2.1642	1.6797	**1.5916**	3.4994	4.083	4.083	3.4727	**2.5377**
ICL	Med	1.0092	0.8155	1.1879	0.8155	0.6545	**0.6202**	2.2591	2.5963	2.5963	2.0429	**1.4507**
	Q1	0.6093	0.5135	0.7424	0.5135	0.4049	**0.3937**	1.3115	1.3794	1.3794	1.0151	**0.7085**
	Q3	1.6024	1.2928	1.9744	1.2928	0.9601	**0.9356**	3.3328	3.7566	3.7566	3.043	**2.3593**
NAIST	Med	0.8191	1.0008	0.8487	1.0008	0.748	**0.4759**	0.2827	2.3717	2.3717	0.5727	**0.27**
	Q1	0.4885	0.4454	0.5012	0.4454	0.3941	**0.2607**	0.1773	1.0819	1.0819	0.2681	**0.1555**
	Q3	1.3768	2.0418	1.6832	2.0418	1.2932	**0.8338**	**0.4212**	4.0515	4.0515	1.4121	0.4614

plane are determined. For the matched feature points with coordinates $(-0.1, -1.5)$ in the left image and $(0.2, -1.2)$ in the right image, from (1) we get

$$u_1\,\mathbf{R}\begin{bmatrix}-0.1\\-1.5\\1\end{bmatrix}+\mathbf{t} \;=\; v_1\begin{bmatrix}0.2\\-1.2\\1\end{bmatrix}, \tag{10}$$

where scalar u_1 and v_1 are the depths of the 3D point at each view. Similarly, for two other matched pairs we can write

$$u_2\,\mathbf{R}\begin{bmatrix}-2.3\\1.6\\1\end{bmatrix}+\mathbf{t} \;=\; v_2\begin{bmatrix}-2.0\\1.8\\1\end{bmatrix}, \tag{11}$$

$$u_3\,\mathbf{R}\begin{bmatrix}0.6\\0.9\\1\end{bmatrix}+\mathbf{t} \;=\; v_3\begin{bmatrix}0.9\\1.1\\1\end{bmatrix}, \tag{12}$$

where subscripts are used to distinguish the depths of the points (scalars u and v). Notice that rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ are the same in all equations.

Example 2. By subtracting (11) and (12) from (10), respectively, and bringing terms to the left hand side we get

$$u_1\,\mathbf{R}\begin{bmatrix}-0.1\\-1.5\\1\end{bmatrix}-v_1\begin{bmatrix}0.2\\-1.2\\1\end{bmatrix}-u_2\,\mathbf{R}\begin{bmatrix}-2.3\\1.6\\1\end{bmatrix}+v_2\begin{bmatrix}-2.0\\1.8\\1\end{bmatrix}=\mathbf{0}, \tag{13}$$

$$u_1\,\mathbf{R}\begin{bmatrix}-0.1\\-1.5\\1\end{bmatrix}-v_1\begin{bmatrix}0.2\\-1.2\\1\end{bmatrix}-u_3\,\mathbf{R}\begin{bmatrix}0.6\\0.9\\1\end{bmatrix}+v_3\begin{bmatrix}0.9\\1.1\\1\end{bmatrix}=\mathbf{0}, \tag{14}$$

in which the unknown translation $\mathbf{t}$ has been eliminated. We can represent (13) and (14) in the matrix-vector form

$$\underbrace{\begin{bmatrix}\mathbf{R}\begin{bmatrix}-0.1\\-1.5\\1\end{bmatrix} & \begin{bmatrix}-0.2\\1.2\\-1\end{bmatrix} & \mathbf{R}\begin{bmatrix}2.3\\-1.6\\-1\end{bmatrix} & \begin{bmatrix}-2.0\\1.8\\1\end{bmatrix} & \mathbf{0} & \mathbf{0}\\ \mathbf{R}\begin{bmatrix}-0.1\\-1.5\\1\end{bmatrix} & \begin{bmatrix}-0.2\\1.2\\-1\end{bmatrix} & \mathbf{0} & \mathbf{0} & \mathbf{R}\begin{bmatrix}-0.6\\-0.9\\-1\end{bmatrix} & \begin{bmatrix}0.9\\1.1\\1\end{bmatrix}\end{bmatrix}}_{\mathbf{M}}\begin{bmatrix}u_1\\v_1\\u_2\\v_2\\u_3\\v_3\end{bmatrix}=\mathbf{0}, \tag{15}$$

which implies that matrix $\mathbf{M}\in\mathbb{R}^{6\times6}$ has a null vector. Therefore, its determinant must be zero. Calculating determinant of $\mathbf{M}$ with $\mathbf{R}$ given in the parametric form (2) gives

$$\begin{aligned}&-0.1\,w^4+2.4\,w^3x+35.6\,w^2x^2-11.4\,wx^3+0.3\,x^4+\dots\\&-19.8\,y^2z^2-168\,wz^3+9.4\,xz^3-17.8\,yz^3+0.3\,z^4=0.\end{aligned} \tag{16}$$

Equation (16) consists of degree four monomials in w, x, y, z (i.e., terms such as $w^4, w^3x, w^2x^2, \dots$), with coefficients that depend on the feature point coordinates. Note that since $\mathbf{M}$ is a 6×6 matrix, one may expect its determinant to have degree six monomials, however, due to the special structure of $\mathbf{M}$, it is always possible to factor out $w^2+x^2+y^2+z^2$ from the determinant expression. The constraint $w^2+x^2+y^2+z^2=1$ gives the degree four polynomial equation in (16).

In practice, we do not calculate the determinant of $\mathbf{M}$ to find (16). Explicit formulas for the polynomial coefficients can be derived by calculating the determinant using symbolic expressions for the feature point coordinates. Substituting the point coordinates in these formulas give the coefficients and hence the equations. Due to the space limitation, and since the code that generates the coefficients from point coordinates is provided online, we do not give the explicit formulas here.

Example 3. Consider equation (16) in Example 2. Multiplying w, x, y, and z, respectively, results in four equations

$$\begin{aligned}-0.1\,w^5+2.4\,w^4x+35.6\,w^3x^2+\dots+0.3wz^4&=0\\-0.1\,w^4x+2.4\,w^3x^2+35.6\,w^2x^3+\dots+0.3xz^4&=0\\-0.1\,w^4y+2.4\,w^3xy+35.6\,w^2x^2y+\dots+0.3yz^4&=0\\-0.1\,w^4z+2.4\,w^3xz+35.6\,w^2x^2z+\dots+0.3\,z^5&=0.\end{aligned}$$

Hence, from 10 equations that are generated from five feature points, 40 degree five equations are derived after the multiplications. These equations can be stacked into the matrix-vector form

$$\underbrace{\begin{bmatrix}-0.1 & 2.4 & 35.6 & \cdots & 0\\ 0 & -0.1 & 2.4 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & \cdots & 0.3\end{bmatrix}}_{\mathbf{A}}\underbrace{\begin{bmatrix}w^5\\w^4x\\w^3x^2\\ \vdots\\ z^5\end{bmatrix}}_{\mathbf{x}}=\mathbf{0} \tag{17}$$

where matrix $\mathbf{A}\in\mathbb{R}^{10\times56}$ consists of the coefficients, and vector $\mathbf{x}\in\mathbb{R}^{56}$ consists of all degree five monomials. Our goal is to find all w, x, y, z, for which (17) is satisfied.

Example 4. Suppose entries of $\mathbf{x}_1$ and $\mathbf{x}_2$ are arranged as

$$\mathbf{x}_1 = \begin{bmatrix} w^3x^2 \\ w^2x^3 \\ wx^4 \\ \vdots \\ wz^4 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} z^5 \\ xz^4 \\ yz^4 \\ \vdots \\ x^5 \end{bmatrix}, \quad \mathbf{v} := \frac{1}{w}\mathbf{x}_1 = \begin{bmatrix} w^2x^2 \\ wx^3 \\ x^4 \\ \vdots \\ z^4 \end{bmatrix}, \tag{18}$$

and assume that from the feature point coordinates we have derived (5) as

$$\underbrace{\begin{bmatrix} z^5 \\ xz^4 \\ yz^4 \\ \vdots \\ x^5 \end{bmatrix}}_{\mathbf{x}_2} = \underbrace{\begin{bmatrix} -0.1 & 5.2 & 2.9 & \cdots & 1.4 \\ 0.1 & -6.7 & -4.4 & \cdots & -1.5 \\ 0.0 & -4.7 & -1.1 & \cdots & -1.1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -0.1 & -0.4 & 6.2 & \cdots & 4.6 \end{bmatrix}}_{\bar{\mathbf{B}}} \underbrace{\begin{bmatrix} w^3x^2 \\ w^2x^3 \\ wx^4 \\ \vdots \\ wz^4 \end{bmatrix}}_{w\mathbf{v}}. \tag{19}$$

From (19) we can construct the eigenvalue problem (6) as

$$\underbrace{\begin{bmatrix} w^2x^3 \\ wx^4 \\ x^5 \\ \vdots \\ xz^4 \end{bmatrix}}_{x\mathbf{v}} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ -0.1 & -0.4 & 6.2 & \cdots & 4.6 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0.1 & -6.7 & -4.4 & \cdots & -1.5 \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} w^3x^2 \\ w^2x^3 \\ wx^4 \\ \vdots \\ wz^4 \end{bmatrix}}_{w\mathbf{v}}, \tag{20}$$

where the first two entries of $x\mathbf{v}$ belong to $\mathbf{x}_1 = w\mathbf{v}$, and hence their associated rows in $\mathbf{B}$ consist of zeros except for a single one entry. The third and last entries of $x\mathbf{v}$ belong to $\mathbf{x}_2$, and their associated rows come from $\bar{\mathbf{B}}$ in (19).

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building rome in a day," in *IEEE International Conference on Computer Vision*, Sept 2009, pp. 72–79.

[2] N. Gans, G. Hu, and W. E. Dixon, "Image based state estimation," in *Encyclopedia of Complexity and Systems Science*. Springer, 2009, pp. 4751–4776.

[3] D. Tick, A. C. Satici, J. Shen, and N. Gans, "Tracking control of mobile robots localized via chained fusion of discrete and continuous epipolar geometry, IMU and odometry," *IEEE transactions on cybernetics*, vol. 43, no. 4, pp. 1237–1250, 2013.

[4] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[5] J. Schneider, C. Eling, L. Klingbeil, H. Kuhlmann, C. Stachniss, *et al.*, "Fast and effective online pose estimation and mapping for UAVs," in *International Conference on Robotics and Automation*. IEEE, 2016, pp. 4784–4791.

[6] E. Marchand, H. Uchiyama, and F. Spindler, "Pose estimation for augmented reality: a hands-on survey," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 12, pp. 2633–2651, 2016.

[7] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[8] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, no. 5828, pp. 133–135, 1981.

[9] M. Demazure, "Sur deux problemes de reconstruction," INRIA, Tech. Rep. RR-0882, July 1988.

[10] O. Faugeras, *Three-dimensional computer vision: a geometric viewpoint*. MIT Press, 1993.

[11] D. Nister, "An efficient solution to the five-point relative pose problem," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. II–195–202 vol.2.

[12] H. Li and R. Hartley, "Five-point motion estimation made easy," in *IEEE International Conference on Pattern Recognition (ICPR)*, vol. 1, 2006, pp. 630–633.

[13] D. H. Stewénius, C. Engels, and D. D. Nistér, "Recent developments on direct relative orientation," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 60, no. 4, pp. 284–294, 2006.

[14] Z. Kukelova, M. Bujnak, and T. Pajdla, "Polynomial eigenvalue solutions to the 5-pt and 6-pt relative pose problems." in *BMVC*, vol. 2, 2008, p. 2008.

[15] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An Invitation to 3-D Vision*. Springer, 2003.

[16] F. Chaumette and S. Hutchinson, "Visual servo control. I. basic approaches," *Robotics & Automation Magazine, IEEE*, vol. 13, no. 4, pp. 82–90, 2006.

[17] J. Philip, "A non-iterative algorithm for determining all essential matrices corresponding to five point pairs," *The Photogrammetric Record*, vol. 15, no. 88, pp. 589–599, 1996.

[18] O. D. Faugeras and S. Maybank, "Motion from point matches: multiplicity of solutions," in *Workshop on Visual Motion*, 1989, pp. 248–255.

[19] S. Maybank, *Theory of Reconstruction from Image Motion*. Springer Berlin Heidelberg, 1993.

[20] D. Hook and P. McAree, *Using Sturm sequences to bracket real roots of polynomial equations*. Academic Press, 1990.

[21] M. Kalantari, F. Jung, J.-P. Guedon, and N. Paparoditis, "The five points pose problem: A new and accurate solution adapted to any geometric configuration," in *Advances in image and video technology*. Springer, 2009, pp. 215–226.

[22] L. Kneip, R. Siegwart, and M. Pollefeys, *Finding the exact rotation between two images independently of the translation*. Springer, 2012.

[23] K. Fathian and N. R. Gans, "A new approach for solving the five-point relative pose problem for vision-based estimation and control," in *American Control Conference*. IEEE, 2014, pp. 103–109.

[24] K. Fathian, J. P. Ramirez-Paredes, E. A. Doucette, J. W. Curtis, and N. R. Gans, "Quaternion based camera pose estimation from matched feature points," *arXiv preprint*, 2017.

[25] E. Malis and F. Chaumette, "2 1/2 D visual servoing with respect to unknown objects through a new estimation scheme of camera displacement," *International Journal of Computer Vision*, vol. 37, no. 1, pp. 79–97, 2000.

[26] J.-Y. Bouguet, "Camera calibration toolbox," https://www.vision.caltech.edu/bouguetj/calib_doc/, 2015.

[27] H. Stewenius, "Eight-point, seven-point, and six-point solvers," http://www.vis.uky.edu/~stewe/FIVEPOINT, 2006.

[28] H. Li, "Five- and six-point essential matrix estimation," http://users.cecs.anu.edu.au/~hongdong/Publications.html, 2012.

[29] Z. Kukelova, "Minimal problems in computer vision," http://cmp.felk.cvut.cz/mini/, 2012.

[30] L. Kneip and P. Furgale, "OpenGV: A unified and generalized approach to real-time calibrated geometric vision," in *International Conference on Robotics and Automation*. IEEE, 2014, pp. 1–8.

[31] D. Q. Huynh, "Metrics for 3D rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[32] A. Handa, T. Whelan, J. McDonald, and A. Davison, "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM," in *IEEE Intl. Conf. on Robotics and Automation*, May 2014.

[33] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition*, 2012.

[34] H. Tamura and H. Kato, "Proposal of international voluntary activities on establishing benchmark test schemes for AR/MR geometric registration and tracking methods," in *International Symposium on Mixed and Augmented Reality*. IEEE, 2009, pp. 233–236.

[35] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *International Conference on Intelligent Robot Systems*, Oct. 2012.