Quadratic Pose Estimation Problems: Globally Optimal Solutions, Solvability/Observability Analysis, and Uncertainty Description

Jin Wu, *Member, IEEE*, Yu Zheng, *Senior Member, IEEE*, Zhi Gao, *Senior Member, IEEE*, Yi Jiang, *Member, IEEE*, Xiangcheng Hu, Yilong Zhu, Jianhao Jiao, and Ming Liu, *Senior Member, IEEE*

***Abstract*—Pose estimation problems are fundamental in robotics. Most of these problems are challenging due to the nonconvex nature. This also sets up an obstacle for uncertainty description that is essential for pose integration and quality control. In this article, we show that a large class of related problems can be categorized as the quadratic pose estimation problems (QPEPs) and we propose a general quaternion-based mathematical model to unify these problems. To solve the nonconvex QPEPs, a Gröbner-basis method is investigated to derive their globally optimal and robust solutions. Furthermore, we develop the rules for characterizing the solvability and observability of these solutions. In addition, the uncertainty description, i.e., covariance matrix, as an important piece of information in robotic state estimation frameworks, is analyzed in detail. Theoretical results show that the covariance can be estimated via online optimization, in an efficient and unbiased manner. In this way, both the solution and covariance are guaranteed to be globally optimal. Through simulations and experiments, we show that the proposed QPEP-based solver is not only accurate, robust, and efficient but outperforms the representatives for covariance estimation. The designed algorithms are also assembled as a C++/MATLAB/Octave/ROS library, while these developed interfaces are built for main stream platforms and simultaneous localization and mapping schemes.**

***Index Terms*—Globally optimal solution, observability analysis, pose estimation, robotic optimization, uncertainty description.**

Manuscript received 14 October 2021; revised 19 December 2021; accepted 23 January 2022. Date of publication 30 March 2022; date of current version 4 October 2022. This work was supported by Shenzhen Science, Technology and Innovation Comission under Grant JCYJ20160401100022706, in part by General Research Fund of Research Grants Council Hong Kong under Grant 11210017, and in part by Early Career Scheme Project of Research Grants Council Hong Kong under Grant 21202816. This paper was recommended for publication by Associate Editor A. Kim and Editor W. Burgard upon evaluation of the reviewers' comments. *(Corresponding author: Ming Liu.)*

Jin Wu, Xiangcheng Hu, Yilong Zhu, Jianhao Jiao, and Ming Liu are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong (e-mail: jin_wu_uestc@hotmail.com; 2022087641@qq.com; yzhubr@connect.ust.hk; jiaojh1994@gmail.com; liu.ming.prc@gmail.com; eelium@ust.hk).

Yu Zheng is with the Tencent Robotics X, Shenzhen 518054, China (e-mail: petezheng@tencent.com).

Zhi Gao is with the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430072, China (e-mail: gaozhinus@gmail.com).

Yi Jiang is with the Department of Biomedical Engineering, College of Science and Engineering, City University of Hong Kong, Hong Kong (e-mail: jy369356904@163.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TRO.2022.3155880.

Digital Object Identifier 10.1109/TRO.2022.3155880

I. Introduction

ROBOTIC agents sense the world and their nearby environments by localization, mapping, and perception techniques. The pose information of a robot is always essential for self-awareness and motion control. The terminology *pose* here consists of attitude and position in a compact manner. To compute the egomotion and obtain accurate localization results, the robot needs to estimate its pose from cameras/IMUs/laser scanners and conduct calibration between these sensors. Many algorithms get involved in giving solutions to absolute camera pose estimation problems from perspective measurements, including point, line, and conics-based correspondences. Perspective geometry also helps attitude determination of spacecraft relative to planets, astroids and other spacecrafts. There are also many sensors other than cameras, e.g., inertial measurement unit (IMU), laser scanner, and global navigation satellite systems (GNSS) receiver. In robotics, pose estimation is not only obtained for self-navigation, but also occurs in kinematics of multiple robots like serial and parallel ones. As many joints and links of these robots may have been coupled, this also provides a background that the attitude and position may also be coupled. Actually, all pose estimation problems described previously are not simple, which means that accurate solutions of them will be challenging. This motivates us to propose a concept called the quadratic pose estimation problem (QPEP) that unifies all these seemingly very different problems. The QPEPs are defined as those ones whose optimization objectives are quadratic terms of the pose elements. Or equivalently, after some analytical manipulations, its Jacobians can be interpreted as those ones only in terms of quadratic terms only in the attitude part. Most of the aforementioned problems belong to the set of QPEPs, which are going to be solved in a generalized manner.

A. Major Challenges

1) Model Generalization: There are so many different robotic pose estimation problems in various scenes. Each of these problems has a different mathematical model and input data. Therefore, it is not easy to give a very explicit generalized mathematical model for all these problems. A model may work quite well for one problem and obtain quite bad results for another. Moreover, a generalized model should also consider those conditions when some problems degenerate.

2) Globally Optimality and Computational Efficiency: Solving QPEPs are sophisticated since these problems are generally nonconvex, which implies that any analytical solution will just be an approximation and finding globally optimal solution is actually not easy. Note that, although solving QPEPs in a globally optimal fashion is important, the solution must be robust and efficient when encountering degenerate cases. Otherwise the robots cannot effectively uses available measurements, which indirectly lowers the feasibility of the developed algorithm. That is to say, to fully address this problem, one must design a unified framework with accurate uncertainty description together with high robustness and low computation time, which is challenging for the community.

3) Solution Quality Control: Due to the difficulty in solving QPEPs globally optimally, the covariance information, being an important issue for quantifying uncertainty of one pose, has not been completely solved. The nonconvexity of QPEPs limits an accurate uncertainty propagation using classical tools like Lie algebra. Many globally optimal solutions have been proposed for some specific cases of QPEPs. However, they still do not build up a complete framework for quantifying the solvability and observability, which are also essential for system monitoring.

B. Contribution

To solve related QPEPs, this paper contributes in the following ways.

1) We would like to discuss something in common among QPEPs. Then, we solve all general QPEPs in a unified quaternion framework. The mathematical framework will be simple and elegant, leading to the convenience in solution.
2) We would like to solve the derived mathematical problem in a globally optimal manner. Besides, the algorithm is guaranteed to be robust and computationally efficient in current robotic computers. Solvability and observability rules are derived to quantify the globally optimal solutions.
3) Also, we would like to obtain uncertainty description based on the developed unified framework. We will also show that using quaternion, the error propagation is almost unbiased. The uncertainty description is more accurate than the current description methods.

The major discussed QPEPs are summarized in Table I.

By solving these QPEPs, we will also benefit as follows.

1) It is convenient to give unified globally optimal solutions to hybrid problems of many different QPEPs. The solution system can be compact, i.e., using only one single system, a large branch of essential problems can be solved accurately and efficiently.
2) It is also flexible to achieve unified covariance computation of related hybrid problems, e.g., propagating uncertainty from camera and industrial robot for a typical hand–eye calibration scenario; propagating uncertainty for interlaced pipelines of RGB-Depth (RGBD) scene reconstruction.

To verify that the proposed method is a generalized effective one, we will show performance evaluation in the experimental results. Experiments on PnP, hand–eye calibration, and a new hybrid RGBD mapping pipeline are conducted. We will show that the proposed method is generalized for all these problems and is accurate compared with state-of-the-art methods. It also builds up a general accurate and robust covariance estimation framework for the first time. In this way, both the solution and covariance can be guaranteed globally optimal.

TABLE I
RELATED PROBLEMS OF QPEP

Problems	Important References
Absolute camera pose from points (Perspective-n-Points, PnP)	[3, 4, 31]
Camera pose from point-to-line (PnL) and point-to-point/line (PnPL)	[11, 14]
Conics-based camera pose	[18]
Hand-eye calibration	[19, 20, 33]
Point-to-plane registration	[22, 23]
Multi-robot relative pose from range measurements	[28, 29]
Forward kinematics of parallel robots, e.g. Stewart platform	[37, 38]
Multi-GNSS attitude determination	[25, 26]

C. Outline

The rest of this article is organized as follows: To clearly state these problems, we first introduce some notations of this article in Section I-D. Section II contains the definition of QPEPs and a brief review of representative QPEPs and existing representatives. Section III-B consists of the proposed unified mathematical framework and globally optimal solution. We derive principal rules for solvability and observability analysis in Section III-C. Section III-D presents the uncertainty description of the proposed scheme. In Section IV, we conduct many robot experiments with ground truth to verify the efficiency of the proposed candidate. Finally, Section V concludes this article.

D. Notations

1) Linear Algebra: The n-dimensional real Euclidean vector space is $\mathbb{R}^n$. A real matrix with size of $m \times n$ belongs to $\mathbb{R}^{m\times n}$. For $\boldsymbol{X} \in \mathbb{R}^{m\times n}$, the symbol $(\boldsymbol{X})_{[i]}$ extracts the ith column of $\boldsymbol{X}$. The Frobenius norm of a matrix $\boldsymbol{X}$ is $\|\boldsymbol{X}\| = \sqrt{\mathrm{tr}(\boldsymbol{X}^\top \boldsymbol{X})}$ in which tr denotes the trace of the matrix and $(\cdot)^\top$ stands for the transpose. Vectorizing an arbitrary matrix $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ into n column vectors and listing them in a compact vector gives the vectorization $\mathrm{vec}(\boldsymbol{X}) = (\boldsymbol{x}_1^\top, \boldsymbol{x}_2^\top, \ldots, \boldsymbol{x}_n^\top)^\top$. The Kronecker product between two matrices is denoted as $\otimes$. For a matrix $\boldsymbol{X} \in \mathbb{R}^{n\times n}$, the matrix is positive semidefinite if all its eigenvalues are nonnegative, say $\boldsymbol{X} \succeq 0$, and is positive definite if all eigenvalues are positive, i.e., $\boldsymbol{X} \succ 0$. For real matrices, the Hermitian is its transpose, i.e., $\boldsymbol{X}^{\mathrm{H}} = \boldsymbol{X}^\top$. We use $\boldsymbol{I}$ and $\boldsymbol{0}$ as the identity and zero matrices of proper dimensions, respectively. All $n \times n$ real symmetric matrices such that $\boldsymbol{X} = \boldsymbol{X}^\top$ form the space of $\mathbb{S}^n$.

The vectorization of the upper triangular part of $\boldsymbol{X} \in \mathbb{S}^n$ in rows is given by $\mathrm{symvec}(\boldsymbol{X}) \in \mathbb{R}^{n(n+1)/2}$ and its inverse reads $\mathrm{symmat}[\mathrm{symvec}(\boldsymbol{X})] = \boldsymbol{X}$. We use the symbol mineig to denote the minimum eigenvalue of a certain matrix. We say $\boldsymbol{X} \succeq \upsilon$, if $\mathrm{mineig}(\boldsymbol{X}) \geq \upsilon$.

2) Lie Algebra: All 3-D rotation matrices form the special orthogonal group $\mathrm{SO}(3)$ such that $\boldsymbol{R} \in \mathrm{SO}(3) \subset \mathbb{R}^{3\times 3} \Rightarrow \boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{I}, \det(\boldsymbol{R}) = +1$. For any real 3-D vector $\boldsymbol{\xi} = (\xi_1, \xi_2, \xi_3)^\top \in \mathbb{R}^3$, its skew-symmetric matrix is

$$\boldsymbol{\xi}_\times = \begin{pmatrix} 0 & -\xi_3 & \xi_2 \\ * & 0 & -\xi_1 \\ * & * & 0 \end{pmatrix} \in \mathfrak{so}(3) \tag{1}$$

which belongs to the Lie algebra $\mathfrak{so}(3)$ and $*$ denotes the skew symmetry. The exponential map $\boldsymbol{R} = \exp(\boldsymbol{\xi}_\times) = \boldsymbol{I} + \boldsymbol{\xi}_\times + \boldsymbol{\xi}_\times^2/2! + \cdots$ for a rotation $\boldsymbol{R} \in \mathrm{SO}(3)$ characterizes a mapping from $\mathfrak{so}(3)$ to $\mathrm{SO}(3)$. The inverse mapping is the logarithm map such that $\boldsymbol{\xi}_\times = \log(\boldsymbol{R})$, in which $\boldsymbol{\xi} \in \mathbb{R}^3$ is the Lie algebra element that corresponds to $\boldsymbol{R}$. The wedge operation is defined as $\wedge$ being a mapping from $\mathfrak{so}(3)$ to $\mathbb{R}^3$ is denoted as $\boldsymbol{\xi}_\times^\wedge = \boldsymbol{\xi}$. We use $\boldsymbol{R} \in \mathrm{SO}(3)$ and $\boldsymbol{t} \in \mathbb{R}^3$ as rotation matrix and translation vector, respectively, in the entire article. When considering $\boldsymbol{R}$ and $\boldsymbol{t}$ simultaneously, the 3-D special Euclidean group is defined as $\mathrm{SE}(3)$, which forms a transformation $\boldsymbol{T}$ via $\boldsymbol{T} = \mathrm{SE}_3(\boldsymbol{R}, \boldsymbol{t}) = \left(\begin{smallmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{smallmatrix}\right) \in \mathrm{SE}(3) \subset \mathbb{R}^{4\times 4}$. To simplify the attitude representation, we will also use unit quaternion $\boldsymbol{q} = (q_0, q_1, q_2, q_3)^\top$ such that $\|\boldsymbol{q}\| = 1$, in some places. One rotation $\boldsymbol{R}$ is quadratically represented by its quaternion $\boldsymbol{q}$, and for one rotation $\boldsymbol{R}$, the $\boldsymbol{q}$ and its negative $-\boldsymbol{q}$ represent the same rotation.

3) Polynomials: For a vector variable $\boldsymbol{x}$, the term $\boldsymbol{x}^p$ forms a new vector containing a complete minimal set of p-order products of elements in $\boldsymbol{x}$, where $p \in \mathbb{Z}_+$ is a nonnegative integer. In this article, the ordering of the elements in $\boldsymbol{x}^p$ is determined by the Veronese map, such that $\boldsymbol{q}^p = \{q_i q_j \cdots q_k | i \leq j \leq \ldots \leq k, (i, j, \ldots, k \in \{0, 1, 2, 3\})\}$. A multivariate polynomial $f(\boldsymbol{x})$ is called p-order if its monomial has the largest total order of p. The gradient of multivariate polynomial $f(\boldsymbol{x})$ with respect to vector $\boldsymbol{y}$ is represented by $\nabla_{\boldsymbol{y}} f(\boldsymbol{x})$ and all $\boldsymbol{x}$ satisfying $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \boldsymbol{0}$ are called critical points, which include a complete set of points reaching local minima or maxima of $f(\boldsymbol{x})$.

4) Probability: We use $\Delta\boldsymbol{X}$ to represent a small difference of a matrix $\boldsymbol{X}$ and the expectation is denoted as $\langle\cdot\rangle$. For a vector signal $\boldsymbol{x}$ with noise term $\boldsymbol{\epsilon}$, its compound result is $\tilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\epsilon}$. The noise $\boldsymbol{\epsilon}$ has the mean (expectation) value of $\bar{\boldsymbol{\epsilon}}$ and the covariance of $\boldsymbol{\Sigma}_{\boldsymbol{x}}$. If $\boldsymbol{\epsilon}$ is subject to Gaussian distribution, it is denoted as $\boldsymbol{\epsilon} \sim \mathcal{N}(\bar{\boldsymbol{\epsilon}}, \boldsymbol{\Sigma}_{\boldsymbol{x}})$. Given two stochastic signals $\boldsymbol{x}$ and $\boldsymbol{y}$, their cross covariance is $\boldsymbol{\Sigma}_{\boldsymbol{x},\boldsymbol{y}}$. When $\boldsymbol{y}$ is the function of $\boldsymbol{x}$, such that $\boldsymbol{y} = f(\boldsymbol{x})$, the covariance propagation is

$$\boldsymbol{\Sigma}_{\boldsymbol{y}} \approx \boldsymbol{\mathcal{J}}_{\boldsymbol{x}}^{\boldsymbol{y}} \boldsymbol{\Sigma}_{\boldsymbol{x}} \left(\boldsymbol{\mathcal{J}}_{\boldsymbol{x}}^{\boldsymbol{y}}\right)^\top \tag{2}$$

where $\boldsymbol{\mathcal{J}}_{\boldsymbol{x}}^{\boldsymbol{y}}$ denotes the Jacobian matrix of $\boldsymbol{y}$ at $\bar{\boldsymbol{x}}$ with respect to $\boldsymbol{x}$. Note that the accuracy of such an approximation is largely determined by the nonlinearity of f and the nonlinearity description ability of $\boldsymbol{\mathcal{J}}_{\boldsymbol{x}}^{\boldsymbol{y}}$.

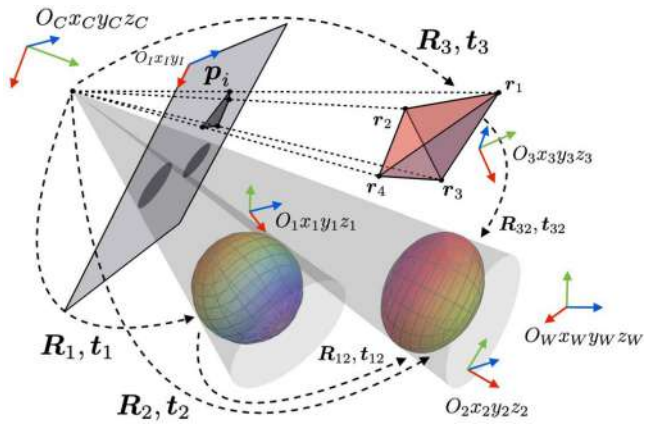


Fig. 1. Diagram of perspective points, lines, and conics.

II. QPEPs and Related Work

A. What are QPEPs?

Definition 1 (QPEPs): The definition of the *quadratic* pose estimation problems (QPEPs) is proposed for the first time in this article, that is: For an $\mathrm{SE}(3)$-differentiable optimization objective $\mathcal{L}(\boldsymbol{R}, \boldsymbol{t})$ for $\mathrm{SE}_3(\boldsymbol{R},\ \boldsymbol{t}) \in \mathrm{SE}(3)$ such that

$$\underset{\boldsymbol{R}\in\mathrm{SO}(3), \boldsymbol{t}\in\mathbb{R}^3}{\arg\min} \ \mathcal{L}(\boldsymbol{R}, \boldsymbol{t}) \tag{3}$$

where $\mathcal{L}$ denotes a *quadratic* objective if $\mathcal{L}$ can be represented in terms of quadratic products of elements in $\boldsymbol{R}$ and $\boldsymbol{t}$. More generally, $\mathcal{L}$ is also called a *quadratic* objective, if one can conduct limited algebraic manipulations to its Jacobian $\nabla_{\mathrm{vec}(\boldsymbol{R})}\mathcal{L}$ to further form an equation of critical points

$$\nabla_{\mathrm{vec}(\boldsymbol{R})}\mathcal{L} \ \Rightarrow \ \mathcal{F}\left[\mathrm{vec}(\boldsymbol{R})^2\right] = \boldsymbol{0} \tag{4}$$

where $\mathcal{F}$ is *only* in the quadratic form of $\mathrm{vec}(\boldsymbol{R})$. In a degenerate QPEP, there is no translation $\boldsymbol{t}$, namely, only the attitude part $\boldsymbol{R}$ is considered. QPEPs own the property of *additivity*, i.e., a hybrid problem of *ad hoc* QPEPs will also be a QPEP. This can be proven by the additivity of Jacobian of *ad hoc* loss functions. Solving QPEPs has been very important in robotics, since many applications involve them, as described in the next subsection.

B. Related Algorithms

1) Perspective Geometry and Calibration: Fig. 1 shows a diagram of perspective geometry of multiple features for absolute camera pose estimation. Multiple objects including two ellipsoids and a tetrahedron are imaged on the image plane $O_I x_I y_I$. The camera locates on the origin of the camera frame $O_C x_C y_C z_C$. The target is to compute the poses of camera relative to these three objects, namely $\boldsymbol{T}_1 = \mathrm{SE}_3(\boldsymbol{R}_1, \boldsymbol{t}_1)$, $\boldsymbol{T}_2 = \mathrm{SE}_3(\boldsymbol{R}_2, \boldsymbol{t}_2)$, and $\boldsymbol{T}_3 = \mathrm{SE}_3(\boldsymbol{R}_3, \boldsymbol{t}_3)$, which give the object frames of $O_i x_i y_i z_i$ for $i = 1, 2, 3$. Four vertices of the tetrahedron are $\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3$, and $\boldsymbol{r}_4$, respectively, in the world frame $O_W x_W y_W z_W$, while the corresponding points on the image plane are $\boldsymbol{p}_i, i = 1, 2, 3, 4$. In common, we can also obtain relative poses between these objects by learning techniques like PoseCNN [1], i.e., $\boldsymbol{T}_{12} = \mathrm{SE}_3(\boldsymbol{R}_{12}, \boldsymbol{t}_{12})$ and $\boldsymbol{T}_{32} =$

$\mathrm{SE}_3(\boldsymbol{R}_{32}, \boldsymbol{t}_{32})$ in Fig. 1, or in some specific applications like aerospace engineering, when the two ellipsoids are planets or asteroids, the relative poses can be easily acquired via ephemeris [2].

Given 2-D interest point coordinates $\boldsymbol{p}_i, i = 1, 2, \ldots$ in the image plane and corresponding points $\boldsymbol{r}_i, i = 1, 2, \ldots$ in the 3-D world frame, the PnP problem is

$$s \underbrace{\left(\boldsymbol{p}_i^\top, 1\right)^\top}_{\tilde{\boldsymbol{p}}_i \in \mathbb{R}^3} = \boldsymbol{K}\left(\boldsymbol{R}\boldsymbol{r}_i + \boldsymbol{t}\right), \ i = 1, 2, \ldots \tag{5}$$

in which $s > 0$ denotes the scale (depth) and $\tilde{\boldsymbol{p}}_i$ is the homogeneous representation of $\boldsymbol{p}_i$ and $\boldsymbol{K} \in \mathbb{R}^{3\times 3}$ represents the camera intrinsic matrix that projects the 3-D points to the 2-D image plane. Vectorizing $\boldsymbol{R}$ and stacking multiple measurement functions in (5) obtains an approximate solution of $\boldsymbol{R}$ and $\boldsymbol{t}$, which is the direct linear transform (DLT) [3]. Although s is constrained by $\boldsymbol{R}, \boldsymbol{t}$, it can be separately estimated to obtain an initial guess, which has been achieved by the EPnP algorithm [4]. Poses will then be refined using Gauss–Newton (GN) or Levenberg–Marquardt (LM) iterations from these initial guesses. However, when the outlier ratio is high, the accuracy of initial guesses may not be satisfactory, which will lead to the convergence to a local minimum. To solve these issues, in the past ten years, researchers focus on finding global optimal solutions to PnP problems, mainly using algebraic elimination methods and semidefinite programming (SDP). In algebraic elimination, the Gröbner-basis method [5], [6] is important, and has been applied to many PnP solvers [7]–[9]. However, as pointed out by [10], these solvers do not always achieve a robust estimate. This reason is that when generating Gröbner basis, some monomials may not occur in the particular problem, which leads to nonexistent elimination steps that causing singularity. The SDP method, although can relax the nonconvex problem and solve it globally optimally, the computational efficiency is not satisfactory that limits its real-time utilization, since the relaxation may be sophisticated and the tightness is difficult to be guaranteed for all problems [11].

Perspective geometries are similar. Because if we downsample the lines and conics, they become sets of perspective points. From Fig. 1, it can be seen that the vertices of tetrahedron form six lines, while four of them are projected onto the image plane. When the line correspondences are considered, the problem becomes perspective-n-lines (PnL) one [12]. Furthermore, when both point and line features occur, the problem of the perspective-n-points and lines (PnPL) problem is involved [13]. Actually, line correspondences occur quite frequently in the Manhattan and Atlanta world model for visual SLAM [14]. Recently, there have been many efficient algorithms for solving PnL and PnPL problems [15]. There has been a consensus that PnP, PnL, and PnPL can actually be united as a single fundamental problem [16]–[18]. Therefore, pose determination from perspective points and lines has been unified.

When the conic correspondences are taken into account, the problem becomes to solving [19]

$$\boldsymbol{\mathcal{G}}^\top \boldsymbol{A} \boldsymbol{\mathcal{G}} = \gamma \boldsymbol{B} \tag{6}$$

in which $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{S}^3$ denote the projected conic and the original conic respectively; $\boldsymbol{\mathcal{G}} = (\mathfrak{r}_1, \mathfrak{r}_2, \boldsymbol{t})$ with $\boldsymbol{R} = (\mathfrak{r}_1, \mathfrak{r}_2, \mathfrak{r}_3)$ and $\gamma > 0$ denotes the scale factor. γ can be estimated according to the similarity of $\boldsymbol{A}$ and $\boldsymbol{B}$. Conics-based pose determination is useful in computer vision for navigation and camera calibration. When the problem (6) is decomposed to solve the rotation, it can be written into the following form:

$$\tilde{\boldsymbol{A}}\boldsymbol{R} = \boldsymbol{R}\tilde{\boldsymbol{B}} \tag{7}$$

where $\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{B}} \in \mathbb{S}^3$ are equivalent symmetric parameter matrices accounting also for the scale.

2) Localization and Calibration: The form (7) is actually very similar to the hand–eye calibration problem [20], [21], that aims to find out the optimal transformation between robotic gripper and attached camera. The hand–eye problem can be modeled as

$$\boldsymbol{A}\boldsymbol{X} = \boldsymbol{X}\boldsymbol{B} \tag{8}$$

in which $\boldsymbol{A}$ and $\boldsymbol{B}$ account for relative transformations from the end effector of industrial robot and the camera, respectively; $\boldsymbol{X} = \mathrm{SE}_3(\boldsymbol{R}, \boldsymbol{t})$ denotes the unknown pose to be figured out. Equations (7) and (8) own a similar form. However, there is no unified solution to them because of a conflict that we name here as the *rigidity and symmetry*. The rigidity and symmetry say that, there is a relation between rigidity and symmetry that the closest rigid transformation of a symmetric matrix is always the identity matrix. This can be proven by matrix orthonormalization method in [22] via singular value decomposition (SVD). Therefore, most analytical solutions to (8) cannot be manipulated to solve (7), and vice versa.

The aforementioned problems are all QPEPs, but QPEPs are not limited to these problems. The point-to-plane registration problem is also a QPEP, which obtains optimal transformation between two point clouds by considering the point-to-plane constraint, such that

$$\underset{\boldsymbol{R}\in\mathrm{SO}(3), \boldsymbol{t}\in\mathbb{R}^3}{\arg\min} \sum \left[\boldsymbol{n}_i^\top \left(\mathfrak{b}_i - \boldsymbol{R}\mathfrak{u}_i - \boldsymbol{t}\right)\right]^2 \tag{9}$$

where $\mathfrak{b}_i$ and $\mathfrak{u}_i$ are ith corresponding points from two point clouds and $\boldsymbol{n}_i$ is the ith normal vector of the target plane. Point-to-plane registration is a typical nonconvex pose estimation problem and is essential to scene reconstruction and simultaneous localization and mapping (SLAM). Previously, (9) is solved by GN and small-angle approximation of $\boldsymbol{R}$ [23], [24]. Therefore, the globally optimal solution to (9) is still open. Moreover, the multi-GNSS attitude determination is important for robotics and aerospace engineering [25], and is also a QPEP because it can be parameterized as the following problem [26], [27]:

$$\underset{\boldsymbol{R}\in\mathrm{SO}(3)}{\arg\min} \ \mathrm{tr}\left(\boldsymbol{R}\boldsymbol{A}\boldsymbol{R}^\top\boldsymbol{B} - \boldsymbol{R}\boldsymbol{C}\right) \tag{10}$$

where $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ are 3×3 matrices. Equation (10) is also not easy-to-solve, in [26], the GN estimator is suggested and some other methods use the Kalman filter [28].

There are also some other problems seemingly not like QPEPs but can be transformed into QPEPs via some techniques. For

instance, the 3-D relative pose estimation from range measurements of mobile robots can be parameterized as [29], [30]

$$(\mathfrak{p}_{1,i} - \boldsymbol{t})^\top \boldsymbol{R}\mathfrak{p}_{2,i} + \mathfrak{p}_{1,i}^\top \boldsymbol{t} = \epsilon_i, i = 1, 2, 3, \dots \tag{11}$$

in which $\mathfrak{p}_{k,i}$ denotes the position of the kth robot at ith time instant, ϵ_i is a known quantity. Since there is a strong coupling between $\boldsymbol{R}$ and $\boldsymbol{t}$, stacking all subequations in (11) and differentiating them will generate a sophisticated system in which $\boldsymbol{t}$ cannot be separated with $\boldsymbol{R}$. In [29], an algebraic elimination method with 27 constraints is proposed. In both [29] and [30], the authors solve the system via homotopy continuation, a general but inefficient method for nonlinear polynomial equations. In fact, following the idea of [29], if we set a temporary variable $\boldsymbol{x} = \boldsymbol{R}^\top \boldsymbol{t}$ as independent variable, adding also the constraint that $\boldsymbol{x}^\top \boldsymbol{x} = \boldsymbol{t}^\top \boldsymbol{t}$ gives a new system. Then, the gradient of this system forms a QPEP (see supplementary material). The forward kinematics of the Stewart platform aims to find the pose of the upper platform from measurements of lengths of six legs, each of them satisfies [31]

$$l_i^2 = \|\mathfrak{y}_i - \boldsymbol{R}\mathfrak{e}_i - \boldsymbol{t}\|^2, \quad i = 1, \dots, 6 \tag{12}$$

where $\mathfrak{y}_i$ and $\mathfrak{e}_i$ are ith coordinates of the bottom and upper platforms connected by one single leg with length of l_i, respectively. Like (11), forward kinematics of the Stewart platform also has strong coupling between $\boldsymbol{R}$ and $\boldsymbol{t}$. Therefore, this problem is also challenging in the computational aspect. However, via similar manipulations, we can also turn (12) into a QPEP (also see the supplementary material).

C. Solvability and Observability

There is currently no general solvability and observability rules for general QPEPs, to our limited knowledge. However, for some specific problems, there are detailed discussions. QPEPs are usually parameterized as polynomial optimization problems. Thus, the solvability and observability shift to the polynomial system theory. For instance, the minimal case of the PnP problem, i.e., the perspective-3-point (P3P) one, has complete root classification [32]. For the PnL problem, sufficient conditions for solvability are presented in [15]. Also, discussion in [33] gives potential information for the observability of perspective line correspondences. There also exist many discussions on the solvability and observability of hand–eye calibration [34], [35]. The relative pose estimation from range measurements has quite sophisticated form, whose observability is investigated via the nonlinear system theory in [29]. There also exist some discussion on relative range-based navigation in [36]. For the Stewart platform, considerable efforts have been paid in finding symbolic rules for deducing the root numbers of the algebraic system [37]–[40]. Till now, it is still hard to combine all these rules into a unified one, which requires further investigation.

D. Uncertainty Description

The uncertainty description of the solution is important for robotic state estimation pipelines. The uncertainty in this article is identical to covariance. In fact, due to the nonlinearity of the rotation group $\mathrm{SO}(3)$, the error propagation using the Lie algebra is not so easy as posed in [41], since Lie exponentials are approximated with high orders. This leads to the fact that obtaining the analytical expression of the covariance of the Lie algebra is also not easy. Barfoot *et al.* proposed uncertainty propagation rules for the pose estimation problem on $\mathrm{SO}(3)$ and $\mathrm{SE}(3)$ [42]. The conclusion was that for most problems, fourth-order results are close to ground truth, while lower order results are unsatisfactory. Mangelson *et al.* refined the theory by introducing cross covariance [43] and pointed out that in some cases, cross-covariance information will be essential to the final accuracy of uncertainty. Learning techniques are also widely studied for pose estimation and uncertainty determination [44], [45]. These methods have quite deep potential in dealing with localization problems in unstructured environments but also suffer from overfitting and long training process and large input samples. For some specific problems, covariance estimation algorithms have been designed. Nguyen *et al.* studied the uncertainty framework of the hand–eye calibration problem $\boldsymbol{AX} = \boldsymbol{XB}$ [46]. A maximum likelihood estimation (MLE) method is invoked for iterative solution and covariance estimation. The MLE also helps build up probabilistic solution to the PnP problem [47], which gives covariance information according to the noise distribution of input points. The Kalman filter is one of the feasible tools for uncertainty propagation, which has been successfully applied in SLAM [48]–[50]. There are some limitations for uncertainty propagation using the Kalman filter, one is that the process and observation models should not be highly nonlinear, otherwise even an extended Kalman filter (EKF) will suffer from instability. The second one is that to use a Kalman filter, one must have reliable dynamic process model, which may not exist in some scenarios, such as purely visual navigation. For other problems, due to the complexity of many solvers, the covariance propagation can hardly be derived. In the work of semidirect visual odometry (SVO) [51], the authors also pointed out that, due to lack of effective uncertainty description, the fusion with IMU has not been effectively developed. Recently, to cope with this problem, deep neural networks are employed to estimate the uncertainty [52], which relies on the training process with mandatory sample data.

III. Proposed Solutions

A. Preliminaries

Following Section II-A, directly solving $\boldsymbol{R}, \boldsymbol{t}$ from (3) is not easy since there are many nonlinear constraints of elements on $\mathrm{SO}(3)$ and what is more, two terms $\boldsymbol{R}$ and $\boldsymbol{t}$ are nonlinearly coupled together. Currently, there are many attitude representations, e.g., quaternion, Rodrigues vector, Euler angles, Cayley parameters, etc. However, the nonlinearity and singularity of each of these representations are quite different. Rodrigues vector and Euler angles are minimal 3-D attitude representations but all have singularity problems, e.g., the gimbal lock effect of Euler angles when the pitch angle approaches $\pm\pi/2$. Therefore, we use quaternion in the following contents to represent $\boldsymbol{R}$. In this way, (3) can be cast into

$$\underset{\boldsymbol{q}^\top \boldsymbol{q}=1,\ \boldsymbol{t}\in\mathbb{R}^3}{\arg\min} \ \mathcal{L}(\boldsymbol{q}, \boldsymbol{t}) \tag{13}$$

which has the Lagrangian of $\tilde{\mathcal{L}}(\boldsymbol{q},\boldsymbol{t},\lambda)=\mathcal{L}(\boldsymbol{q},\boldsymbol{t})-\lambda(\boldsymbol{q}^\top\boldsymbol{q}-1)$. All local minima of the problem (13) can be found by zeroing the system of gradient

$$\nabla_{\boldsymbol{y}}\tilde{\mathcal{L}}(\boldsymbol{q},\boldsymbol{t},\lambda)=\boldsymbol{0} \tag{14}$$

where $\boldsymbol{y}=(\boldsymbol{q}^\top,\boldsymbol{t}^\top,\lambda)^\top$ is the total variable to be solved. All QPEPs can be manipulated to the form of

$$\begin{cases} f\left(\boldsymbol{q}^4,\lambda\right)=\boldsymbol{0} \\ \boldsymbol{q}^\top\boldsymbol{q}=1 \end{cases} \tag{15}$$

via some algebraic operations, while $\boldsymbol{t}$ can be represented in the linear combination of elements in $\boldsymbol{q}^2$, say $\boldsymbol{t}=\boldsymbol{\mathcal{T}}(\boldsymbol{q}^2)$, where $\boldsymbol{\mathcal{T}}$ represents such a linear map of elements in $\boldsymbol{q}^2$ to $\boldsymbol{t}$. This gives the result that (15) can be further transformed into

$$\boldsymbol{M}\left(\boldsymbol{q}^2\right)\boldsymbol{q}=(\boldsymbol{Q}-\lambda\boldsymbol{I})\,\boldsymbol{q} \tag{16}$$

where $\boldsymbol{M}(\boldsymbol{q}^2)$ is a matrix in the linear form of $\boldsymbol{q}^2$ and $\boldsymbol{Q}\in\mathbb{R}^{4\times4}$ is a parameter matrix. From (16), one can see that $\boldsymbol{q}$ is the eigenvector of $\boldsymbol{Q}-\boldsymbol{M}(\boldsymbol{q}^2)$ associated with the eigenvalue λ. Therefore, (16) is actually a recursive eigenvalue problem, that is, the eigenvalue λ is recursively related to $\boldsymbol{q}$ in a nonlinear manner. By denoting $\boldsymbol{u}=\boldsymbol{q}\otimes\boldsymbol{q}\otimes\boldsymbol{q}\in\mathbb{R}^{64}$, where $\boldsymbol{u}$ represents one of many permutations of elements in $\boldsymbol{q}^3$, we can write (16) as

$$\boldsymbol{W}\boldsymbol{u}=(\boldsymbol{Q}-\lambda\boldsymbol{I})\,\boldsymbol{q},\quad \text{s.t. } \boldsymbol{q}^\top\boldsymbol{q}=1 \tag{17}$$

which is the essential form of the unified mathematical framework for QPEPs, where $\boldsymbol{W}\in\mathbb{R}^{4\times64}$ is a pure parameter matrix. Equation (17) is a three-order multivariate polynomial system that cannot be solved algebraically due to the theorem of Ruffini–Abel [53].

Remark 1: For those cases that $\boldsymbol{R}\to\boldsymbol{I}$, we may use $\boldsymbol{R}\approx\boldsymbol{I}+\boldsymbol{\xi}_\times$ for an approximate rotation, where $\boldsymbol{\xi}=[\log(\boldsymbol{R})]^\wedge$. In this condition, solving $\boldsymbol{R}$ and $\boldsymbol{t}$ from (3) becomes a linear unconstrained least-square problem with respect to $\boldsymbol{\xi}$ and $\boldsymbol{t}$, say

$$\boldsymbol{\mathcal{A}}\boldsymbol{x}=\boldsymbol{b} \tag{18}$$

where $\boldsymbol{x}=(\boldsymbol{\xi}^\top,\boldsymbol{t}^\top)^\top$, $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{b}$ come from input data. Equation (18) is a linear system that can be solved easily via QR decomposition, LU decomposition, and SVD, considering different conditions of $\boldsymbol{\mathcal{A}}$. After solving $\boldsymbol{x}$, the obtained rotation may not be exactly on $\mathrm{SO}(3)$, which should be orthonormalized. The practitioners may turn to [54] and [55] for orthonormalization on $\mathrm{SO}(3)$, and moreover, turn to [56] for the refinement on $\mathrm{SE}(3)$.

B. Solution Strategies

To solve $\boldsymbol{q}$ and λ from (17), one must conduct variable elimination to decrease the quantity of unknowns. In this article, we use the Gröbner-basis method to deal with this problem.

Definition 2 (Gröbner-basis Method): The Gröbner-basis method finds out the Gröbner basis of a polynomial system and solves the system from the lowest order terms sequentially. By generating the symbolic factor matrix $\boldsymbol{\mathcal{M}}$ together with all monomials $\mathfrak{g}$ involved in the Gröbner basis, it is able for us to write the polynomial system

$$\boldsymbol{\mathcal{M}}\mathfrak{g}=\boldsymbol{0}. \tag{19}$$

Note that the first several terms in $\mathfrak{g}$ are 1 and the unknown $\boldsymbol{q}$ to be solved, which form the vector $\mathfrak{g}_1$ and the remainder is $\mathfrak{g}_2$ so that $\mathfrak{g}=(\mathfrak{g}_1^\top,\mathfrak{g}_2^\top)^\top$. The elimination template is the symbolic procedure to eliminate variables so that $\mathfrak{g}$ is ordered and all variables are presented in a maximally independent fashion. That is to say, the elimination template is an algebraically manipulated version of $\boldsymbol{\mathcal{M}}$, but with reduced size. After an elimination template is constructed, the action matrix can be obtain using the Schur complement for solution of only those required variables. For better understanding of the Gröbner-basis method, see [5] and [6]. Thus solving all solutions from (19) is equivalent to finding $\mathfrak{g}_1$ via Schur complement and eigendecomposition of $\boldsymbol{\mathcal{M}}$. Generating $\boldsymbol{\mathcal{M}}$ and $\mathfrak{g}$ is not difficult in symbolic computation. By evaluating (19), one can conclude that (17) has at most 40 complex and real solutions, according to the Bezout theorem [53]. However, directly solving (17) using (19) is usually not efficient and robust. The reasons are threefold.

1) There will be zero terms or linearly dependent columns/rows of $\boldsymbol{W}$ and $\boldsymbol{Q}$ in some cases, leading to the fact that some fundamental monomials vanish in $\mathfrak{g}$ and the solution may be trivial.
2) Since λ acts as a penalty factor, for cases with small noise, λ will be very close to zero. Namely, when eliminating λ from (17), the induced system may be very sensitive to input noise and the computed λ will be meaningless.
3) When eliminating λ, the size of the elimination template significantly grows. If the quantity of all unknowns is large, the size of $\boldsymbol{\mathcal{M}}$ will also be large, i.e., the computation of the Schur complement and eigendecomposition will be inefficient.

The elimination of λ is not unique. For one example, (17) indicates that

$$\lambda^2=(\boldsymbol{Q}\boldsymbol{q}-\boldsymbol{W}\boldsymbol{u})^\top(\boldsymbol{Q}\boldsymbol{q}-\boldsymbol{W}\boldsymbol{u}). \tag{20}$$

If we write (17) into

$$\begin{cases} \boldsymbol{q}^\top(q_0\boldsymbol{\mathcal{P}}_{11}+q_1\boldsymbol{\mathcal{P}}_{12}+q_2\boldsymbol{\mathcal{P}}_{13}+q_3\boldsymbol{\mathcal{P}}_{14})\,\boldsymbol{q}-\boldsymbol{b}_1^\top\boldsymbol{q}+\lambda q_0=0 \\ \boldsymbol{q}^\top(q_0\boldsymbol{\mathcal{P}}_{21}+q_1\boldsymbol{\mathcal{P}}_{22}+q_2\boldsymbol{\mathcal{P}}_{23}+q_3\boldsymbol{\mathcal{P}}_{24})\,\boldsymbol{q}-\boldsymbol{b}_2^\top\boldsymbol{q}+\lambda q_1=0 \\ \boldsymbol{q}^\top(q_0\boldsymbol{\mathcal{P}}_{31}+q_1\boldsymbol{\mathcal{P}}_{32}+q_2\boldsymbol{\mathcal{P}}_{33}+q_3\boldsymbol{\mathcal{P}}_{34})\,\boldsymbol{q}-\boldsymbol{b}_3^\top\boldsymbol{q}+\lambda q_2=0 \\ \boldsymbol{q}^\top(q_0\boldsymbol{\mathcal{P}}_{41}+q_1\boldsymbol{\mathcal{P}}_{42}+q_2\boldsymbol{\mathcal{P}}_{43}+q_3\boldsymbol{\mathcal{P}}_{44})\,\boldsymbol{q}-\boldsymbol{b}_4^\top\boldsymbol{q}+\lambda q_3=0 \end{cases} \tag{21}$$

via algebraic induction, where $\boldsymbol{\mathcal{P}}_{ij}\in\mathbb{S}^4$ for $i,j=1,2,3,4$ denote symmetric factor matrices, while $\boldsymbol{b}_i\in\mathbb{R}^4$ for $i=1,2,3,4$ represent factors for first-order quaternion terms, we can see that there are four different ways of eliminating λ. Comparing (20) and (21), it is evident that two elimination approaches are quite different. However, due to the rotating symmetry of $q_0^2+q_1^2+q_2^2+q_3^2=1$, the four elimination methods of λ in (21) are identical. Therefore, using the transformation of the first subequation of (21), i.e.,

$$\lambda=\left[\boldsymbol{b}_1^\top\boldsymbol{q}-\boldsymbol{q}^\top\sum_{i=0}^{3}q_i\boldsymbol{\mathcal{P}}_{1(i+1)}\boldsymbol{q}\right]/q_0 \tag{22}$$

one may derive a new system, which is free of λ, such that

$$\boldsymbol{D}\boldsymbol{v} + \boldsymbol{G}\boldsymbol{y} + \boldsymbol{c} = \boldsymbol{0}, \quad \text{s.t. } \boldsymbol{q}^\top \boldsymbol{q} = 1 \tag{23}$$

where $\boldsymbol{v}$ denotes a certain subset of complete minimal set of $\boldsymbol{q}^4$ and $\boldsymbol{y}$ represents a certain subset of complete minimal set of $\boldsymbol{q}^2$, while $\boldsymbol{D}, \boldsymbol{G},$ and $\boldsymbol{c}$ are factor matrices and vectors. Solving the Gröbner bases of (23) seems not easy and as a result, the newly transformed system has at most 128 solutions, also according to the Bezout theorem. This indicates that (23) is more sophisticated than (17) but (23) actually provides a simplest form of QPEPs that only relates to the quaternion, which can be conveniently employed for uncertainty analysis.

To address the difficulties, we first need to analyze the structures of $\boldsymbol{W}$ and $\boldsymbol{Q}$ using symbolic engines like MATLAB and Mathematica. We use symbolic tools to deduce the zero and linearly dependent rows/columns of $\boldsymbol{W}$ and $\boldsymbol{Q}$. We also perform a detailed checking of all repeated terms that appear not in the same rows and columns. After that, we will also check the linear independence of $\boldsymbol{W}$ and $\boldsymbol{Q}$ when solving QPEPs using the Gauss–Jordan elimination. As $\lambda \to 0$ for some small-noise cases, we neglect λ for these cases, namely solving a reduced system

$$\begin{cases} \tilde{\boldsymbol{W}}\boldsymbol{u} - \tilde{\boldsymbol{Q}}\boldsymbol{q} = \boldsymbol{0} \\ \boldsymbol{q}^\top \boldsymbol{q} - 1 = 0 \end{cases} \tag{24}$$

where $\tilde{\boldsymbol{W}} \in \mathbb{R}^{3\times 64}$ and $\tilde{\boldsymbol{Q}} \in \mathbb{R}^{3\times 4}$ are reduced matrices of $\boldsymbol{W}$ and $\boldsymbol{Q}$, respectively, by selecting 3 out of 4 rows with four row permutations $\{(1,2,3),(1,2,4),(1,3,4),(2,3,4)\}$. Thus, in this way, four potential sets of solutions are generated from (24). To choose the best solution, one can insert all these solutions back into the loss function (3) and select the one that corresponds to the least loss-function value. Since these four branches have the same elimination procedures, they can be executed in parallel with hardware parallelism or multithreading programming. As we can see, in (24), λ is cancelled, the generated $\boldsymbol{\mathcal{M}}$ matrix has a smaller size than previous ones, which leads to more efficient implementation. The final optimal solution may not be exactly accurate since there will be numerical loss in computation of $\boldsymbol{\mathcal{M}}$. When a best solution is found out, one may refine the solution using GN and LM techniques by (23), which treat (23) as an error vector in terms of $\boldsymbol{q}$.

In fact, from (16) and (20), it can be deduced that (24) provides an optimal initial guess to guarantee the smallest λ^2. The expression of $\boldsymbol{\mathcal{M}}$ may not be unique since there are many conditions that some monomials vanish. Also, for those cases that solutions are ambiguous, the size of $\boldsymbol{\mathcal{M}}$ reduces. In practice, we can choose the $\boldsymbol{\mathcal{M}}$, which is the most reducible. The sizes of elimination template are summarized in Table II. Smaller sizes indicate much faster computation speed for eigendecomposition and solution verification. However, it should also be noted that, a smaller template usually accompanies lower stability, that is, the elimination may be failed and not all of potential solutions can be found out.

The solution sizes also vary with different action matrix sizes, e.g., 27 solutions for 239×239 and 249×249 templates, 16 solutions for 129×129, 151×151, and 189×189 templates.

TABLE II
ELIMINATION TEMPLATE SIZES OF QPEPS

Problems	Elimination Template Size
PnPL	189×16
PnP and PnL	151×16
Conic-based camera Pose	239×27
Rotation-only Conic Pose	129×16
Hand-eye Calibration	249×27
Point-to-Plane Registeation	249×27
Multi-Robot Localization	249×27
Stewart Forward Kinematics	249×27
Multi-GNSS Attitude	189×16

Remark 2: Another technique to generate all solutions to (23) is by further elimination, which is not utilized in the proposed QPEP. Note that we can still eliminate one more quaternion component, e.g., q_0 from (23) using the substitutions as follows:

$$\begin{aligned} q_0^4 &= \left(1 - q_1^2 - q_2^2 - q_3^2\right)^2 \\ q_0^3 &= \left(1 - q_1^2 - q_2^2 - q_3^2\right) q_0 \\ q_0^2 &= \left(1 - q_1^2 - q_2^2 - q_3^2\right). \end{aligned} \tag{25}$$

We suggest the substitution ordering in (25) that the terms are replaced from the most complex ones. In this way, one will obtain a system where q_0 only exists in the first order, with which one can eliminate q_0 and construct three equations of q_1, q_2, q_3. The new equation has 70 subterms of the degree 7 and is no longer depending on the quaternion norm constraint $\boldsymbol{q}^\top \boldsymbol{q} = 1$. Generating the Gröbner basis for this new system is harder than previous quaternion ones, since there are more complicated monomials involved, and inevitably, more extraneous roots are introduced. We can still solve this system by finding the Sylvester matrix $\boldsymbol{\mathcal{S}}$ in q_3 so that a linear system

$$\boldsymbol{\mathcal{S}}\boldsymbol{\phi} = \boldsymbol{0} \;\Rightarrow\; \det(\boldsymbol{\mathcal{S}}) = 0 \tag{26}$$

can be generated, where $\boldsymbol{\phi}$ contains all monomials related to q_1 and q_2. The equality $\det(\boldsymbol{\mathcal{S}}) = 0$ forms a univariate polynomial of q_3 in the degree of 40. Then, solving all roots of q_3 generate the complete set of potential local minima. Note that here in (26), the determinant of the Sylvester matrix also equals to the resultant of the system to be solved. Thus, using symbolic resultant solvers, similar forms of $\boldsymbol{\mathcal{S}}$ can also be derived. However, what should be noted here is that since the derivation of $\boldsymbol{\mathcal{S}}$ is sophisticated, the generated factor matrices will be extremely lengthy, which means that using this approach for a complete elimination is not efficient. Besides, to solve all potential roots, we need to solve all roots of the 40-degree univariate polynomial, which significantly suffers from the so-called Runge phenomenon that occurs in high-order polynomials [57]. The Runge phenomenon actually describes the fact that a high-order polynomial is very sensitive to small perturbation of input variable. That is to say, it is hard to distinguish very accurate solutions when the solutions are small. As a result, arbitrary-digit arithmetic libraries will be highly required, which significantly reduces the computational efficiency.

C. Solvability and Observability Analysis

1) Solvability: It can be verified through symbolic engines that $\boldsymbol{u} = \frac{1}{3}\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{u}}\boldsymbol{q}$. Therefore, combining with (17), we can see that

$$\boldsymbol{S}(\boldsymbol{q}^2)\boldsymbol{q} = \lambda\boldsymbol{q} \tag{27}$$

in which $\boldsymbol{S}(\boldsymbol{q}^2) = \boldsymbol{Q} - \frac{1}{3}\boldsymbol{W}\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{u}}$. The eigenvalue system (27) indicates that λ is the root of the characteristic polynomial

$$\det\left[\boldsymbol{S}(\boldsymbol{q}^2) - \lambda\boldsymbol{I}\right] = 0. \tag{28}$$

Expanding (28), one may arrange the equation in terms of λ so that

$$\lambda^4 + \tau_1\lambda^3 + \tau_2\lambda^2 + \tau_3\lambda + \tau_4 = 0 \tag{29}$$

where $\tau_1, \ldots, \tau_4$ are functions of $\boldsymbol{q}$. Previous algebraic results of quartic polynomials [57] show that (29) can be manipulated to a simpler form without cubic term

$$\lambda^4 + \tilde{\tau}_1\lambda^2 + \tilde{\tau}_2\lambda + \tilde{\tau}_3 = 0 \tag{30}$$

in which $\tilde{\tau}_1, \ldots, \tilde{\tau}_3$ are manipulated coefficients. In this way, the solvability of $\boldsymbol{q}$ becomes quantifying the feasible regions of λ. One may need to answer the problem that whether (30) has real roots, and moreover, whether these roots are distinct or repeated. Since $\boldsymbol{q} \in \mathbb{R}^4$, λ should also be real numbers. However, the structure of $\boldsymbol{S}(\boldsymbol{q}^2)$ does not guarantee that it is always real symmetric. Using the discriminants [53] of (30), i.e., $A = \tilde{\tau}_1^2 + 12\tilde{\tau}_3$, $B = 9\tilde{\tau}_2^2 + 2\tilde{\tau}_1(\tilde{\tau}_1^2 - 4\tilde{\tau}_3)$, and $C = 6\tilde{\tau}_1\tilde{\tau}_2^2 + (\tilde{\tau}_1^2 - 4\tilde{\tau}_3)^2$. The following rules deduce the quantities of the roots of (29).

Rules 1 (Solvability of a QPEP).

1) Two distinct real roots and two distinct complex roots: $B^2 - 4AC > 0$ (RS1).
2) When $B^2 - 4AC < 0$ (RS2).
1) Four distinct complex roots: $B \geq 0$ or $A \geq 0$ (RS21).
2) Four real roots: $B < 0$ and $A < 0$ (RS22).
3) When $B^2 - 4AC = 0$ (RS3), there must have been repeated roots.
1) Two real roots and two complex roots: $B \neq 0$ and $\tilde{\tau}_1 B^2/2 + (\tilde{\tau}_2 A)^2 > 0$ (RS31).
2) Four complex roots: $B = 0, A \neq 0$ and $\tilde{\tau}_1 > 0$ (RS32).
3) Four real roots: Otherwise (RS33).

Remark 3: Following these rules, if the roots are real and repeated while the repeated ones exactly have the smallest absolute values, the system has multiple solutions, i.e., the system is partially solvable. One classical case is the PnP problem with only four noncollinear point pairs, namely, the P4P problem. The P4P problem normally has unique solution. However, when one point is collinear with another, the problem degenerates. In such a problem setup, the $\boldsymbol{S}(\boldsymbol{q}^2)$ matrix will have exactly two equal reals λ that correspond to different quaternions. For proper solutions, we must have at least one feasible λ in the real domain. This indicates that (RS2)$\cap$(RS21) and (RS3)$\cap$(RS32) must not happen. These inequalities define the infeasible regions of the pose, where point correspondences degenerate. These infeasible regions actually characterize those workspaces where pose is not completely estimated.

2) Observability: In the condition that one problem is solvable, the observability specifies the quality for one system to observe the estimated pose components. Given the system (23), it is easy for us to write out its Jacobian of first subequations

$$\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{e}} = \boldsymbol{D}\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}} + \boldsymbol{G}\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{y}} \qquad \in \mathbb{R}^{3\times 4} \tag{31}$$

where the residual is denoted as

$$\boldsymbol{e} = \boldsymbol{D}\boldsymbol{v} + \boldsymbol{G}\boldsymbol{y} + \boldsymbol{c}. \tag{32}$$

The complete Jacobian of (23) reads

$$\boldsymbol{\mathcal{J}}_{\mathrm{eq}} = \left[\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{e}}\right)^{\top}, 2\boldsymbol{q}\right]^{\top} \in \mathbb{R}^{4\times 4}. \tag{33}$$

The Jacobian in (33) is of full-rank if $\boldsymbol{q}$ is completely observable to the system, and is rank-deficient if $\boldsymbol{q}$ is partially observable or unobservable. To analyze the rank of $\boldsymbol{\mathcal{J}}_{\mathrm{eq}}$, we can perform the Gauss–Jordan elimination so that one can obtain a reduced row echelon form of $\boldsymbol{\mathcal{J}}_{\mathrm{eq}}$, say

$$\tilde{\boldsymbol{\mathcal{J}}}_{\mathrm{eq}} = \begin{pmatrix} 1 & 0 & 0 & \vartheta_1 \\ 0 & \vartheta_6 & 0 & \vartheta_2 \\ 0 & 0 & \vartheta_5 & \vartheta_3 \\ 0 & 0 & 0 & \vartheta_4 \end{pmatrix}. \tag{34}$$

We do not perform SVD because this operation cannot generate explicit symbolic expressions. The following rules specify different observability conditions.

Rules 2 (Observability of a QPEP)

1) If $\vartheta_4 \neq 0, \vartheta_5 \neq 0, \vartheta_6 \neq 0$, we have $\mathrm{rank}(\tilde{\boldsymbol{\mathcal{J}}}_{\mathrm{eq}}) = 4$, the system is completely observable. (RO1)
2) If $\vartheta_4 = 0, \vartheta_5 \neq 0, \vartheta_6 \neq 0$, we have $\mathrm{rank}(\tilde{\boldsymbol{\mathcal{J}}}_{\mathrm{eq}}) = 3$, the system is partially observable and owns the observable attitude freedom of 2 (one quaternion has 3 degrees of freedom). (RO2)
3) If $\vartheta_4 = 0, \vartheta_5 = 0, \vartheta_6 \neq 0$, we have $\mathrm{rank}(\tilde{\boldsymbol{\mathcal{J}}}_{\mathrm{eq}}) = 2$, the system is partially observable and owns the observable attitude freedom of 1 (one quaternion has 3 degrees of freedom). (RO3)
4) If $\vartheta_4 = 0, \vartheta_5 = 0, \vartheta_6 = 0$, we have $\mathrm{rank}(\tilde{\boldsymbol{\mathcal{J}}}_{\mathrm{eq}}) = 1$, the system is completely unobservable. (RO4)

The intersection (RO2)$\cap$(RO3)$\cap$(RO4) will form infeasible regions. By avoiding these infeasible regions, the system (23) possesses full observability of $\boldsymbol{q}$.

Remark 4: Evaluating $\vartheta_1 \sim \vartheta_6$ may be challenging even for a modern symbolic computation system. In engineering, one usually cares about whether the system is fully observable, rather than its specific observable degrees of freedom. Therefore, using determinant, we can *taste* the system. The word "taste" means that we may first check if the system is solvable or not, and then, deduce that whether we need to solve the problem. Following this criterion, the feasible region of $\boldsymbol{q}$ enables $\det(\boldsymbol{\mathcal{J}}_{\mathrm{eq}}) \neq 0$, which roughly gives a measure of the observability.

D. Uncertainty Analysis

Prior to uncertainty description, we perform a detailed error analysis of some terms. The difficulty of error analysis concentrates in the rotation terms according to the many constraints.

Therefore, we use the pure-quaternion equation (23) to conduct error and covariance analysis about $\boldsymbol{q}$. The residual in (32) leads to the following perturbation model:

$$\Delta \boldsymbol{e} = \Delta \boldsymbol{D}\boldsymbol{v} + \boldsymbol{D}\Delta\boldsymbol{v} + \Delta \boldsymbol{G}\boldsymbol{y} + \boldsymbol{G}\Delta\boldsymbol{y} + \Delta\boldsymbol{c}. \tag{35}$$

Using the Kronecker product $\mathrm{vec}(\boldsymbol{AXB}) = (\boldsymbol{B}^\top \otimes \boldsymbol{A})\mathrm{vec}(\boldsymbol{X})$, it follows that

$$\Delta \boldsymbol{D}\boldsymbol{v} = \left(\boldsymbol{v}^\top \otimes \boldsymbol{I}\right)\Delta\mathrm{vec}(\boldsymbol{D}) = \boldsymbol{\mathcal{V}}\Delta\boldsymbol{d} \tag{36}$$

$$\Delta \boldsymbol{G}\boldsymbol{y} = \left(\boldsymbol{y}^\top \otimes \boldsymbol{I}\right)\Delta\mathrm{vec}(\boldsymbol{G}) = \boldsymbol{\mathcal{Y}}\Delta\boldsymbol{g} \tag{37}$$

where $\boldsymbol{\mathcal{V}} = \boldsymbol{v}^\top \otimes \boldsymbol{I}$, $\boldsymbol{\mathcal{Y}} = \boldsymbol{y}^\top \otimes \boldsymbol{I}$, and $\boldsymbol{d} = \mathrm{vec}(\boldsymbol{D})$, $\boldsymbol{g} = \mathrm{vec}(\boldsymbol{G})$. The complete error analysis of $\boldsymbol{v}$ and $\boldsymbol{y}$ is conducted as follows. First, we have an algebra fact that $\boldsymbol{v} = \frac{1}{4}\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\boldsymbol{q}$, $\boldsymbol{y} = \frac{1}{2}\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{y}}\boldsymbol{q}$, which can be easily reproduced via symbolic engines. Discretizing them leads to $\Delta\boldsymbol{v} = \frac{1}{4}(\Delta\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\boldsymbol{q} + \boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\Delta\boldsymbol{q})$, $\Delta\boldsymbol{y} = \frac{1}{2}(\Delta\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{y}}\boldsymbol{q} + \boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{y}}\Delta\boldsymbol{q})$. For $\boldsymbol{v}$, the Jacobian difference part can be computed via

$$\Delta\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\boldsymbol{q} = \sum_{j=1}^{4}\Delta\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]} q_{j-1} \tag{38}$$

where another algebraic fact is that

$$\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]} = \frac{1}{3}\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\boldsymbol{q}. \tag{39}$$

This can be further derived to

$$\Delta\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]} = \frac{1}{3}\left(\Delta\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\boldsymbol{q} + \frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\Delta\boldsymbol{q}\right) \tag{40}$$

where we have

$$\Delta\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\boldsymbol{q} = \sum_{k=1}^{4}\Delta\left[\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\right]_{[k]} q_{k-1} \tag{41}$$

$$\left[\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\right]_{[k]} = \frac{1}{2}\frac{\partial}{\partial\boldsymbol{q}}\left\{\left[\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\right]_{[k]}\right\}\boldsymbol{q}. \tag{42}$$

By repeating such mechanisms over and over again, the high-order Jacobians become constant matrices and cannot be performed any more. The final result is shown as follows: (43) shown at the bottom of this page

Simplifying these results recursively gives an interesting fact that $\Delta\boldsymbol{v} = \boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\Delta\boldsymbol{q}$ and likewise for $\boldsymbol{y}$, $\Delta\boldsymbol{y} = \boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{y}}\Delta\boldsymbol{q}$. This shows that Jacobian matrices of $\boldsymbol{v}$ and $\boldsymbol{y}$ *completely* describe the nonlinearity and are *unbiased* for covariance propagation using

$$\begin{aligned}\boldsymbol{\Sigma}_{\boldsymbol{v}} &= \boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\boldsymbol{\Sigma}_{\boldsymbol{q}}\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)^\top \\ \boldsymbol{\Sigma}_{\boldsymbol{y}} &= \boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{y}}\boldsymbol{\Sigma}_{\boldsymbol{q}}\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{y}}\right)^\top.\end{aligned} \tag{44}$$

This is very different from approaches from [42] and [46], where error propagation of the Lie algebra on $\mathrm{SO}(3)$ and $\mathrm{SE}(3)$ has been invoked. Therefore, due to the fact in (44), the Jacobian covariance propagation is potentially more accurate than that in the Lie theory, namely, the proposed method will not require any high-order approximation of Lie exponentials. Lie-algebra uncertainty propagation also suffers from the fact that solving the globally optimal Lie algebra representation of the pose from a nonconvex pose estimation problem is challenging, also due to the Lie exponentials and nonlinearity of Lie logarithms. Therefore, practitioners usually need to convert the results in other pose representations into the Lie algebra for further computation, if they would like to guarantee the global optimality of the solution.

Following these results, we can write $\Delta\boldsymbol{e} = \boldsymbol{\mathcal{V}}\Delta\boldsymbol{d} + \boldsymbol{F}\Delta\boldsymbol{q} + \boldsymbol{\mathcal{Y}}\Delta\boldsymbol{g} + \Delta\boldsymbol{c}$, where $\boldsymbol{F} = \boldsymbol{D}\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}} + \boldsymbol{Y}\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{y}}$. All these error terms contribute to the autocovariance $\boldsymbol{\Sigma}_{\boldsymbol{e},\mathrm{auto}} = \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{F}\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{F}^\top + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c}}$ and cross-covariance part of

$$\begin{aligned}\boldsymbol{\Sigma}_{\boldsymbol{e},\mathrm{cross}} = {} & \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d},\boldsymbol{q}}\boldsymbol{F}^\top + \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d},\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top + \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d},\boldsymbol{c}} + \boldsymbol{F}\boldsymbol{\Sigma}_{\boldsymbol{q},\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \\ & \boldsymbol{F}\boldsymbol{\Sigma}_{\boldsymbol{q},\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top + \boldsymbol{F}\boldsymbol{\Sigma}_{\boldsymbol{q},\boldsymbol{c}} + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g},\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g},\boldsymbol{q}}\boldsymbol{F}^\top + \\ & \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g},\boldsymbol{c}} + \boldsymbol{\Sigma}_{\boldsymbol{c},\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c},\boldsymbol{q}}\boldsymbol{F}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c},\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top\end{aligned}$$

such that $\boldsymbol{\Sigma}_{\boldsymbol{e}} = \boldsymbol{\Sigma}_{\boldsymbol{e},\mathrm{auto}} + \boldsymbol{\Sigma}_{\boldsymbol{e},\mathrm{cross}}$. Since for globally optimal $\boldsymbol{q}$, one always has $\boldsymbol{e} = \boldsymbol{0}$, it also follows that $\boldsymbol{\Sigma}_{\boldsymbol{e}} = \boldsymbol{0}$. Moreover, from $\Delta\boldsymbol{e} = \boldsymbol{0}$, one obtains $-\boldsymbol{F}\Delta\boldsymbol{q} = \boldsymbol{\mathcal{V}}\Delta\boldsymbol{d} + \boldsymbol{\mathcal{Y}}\Delta\boldsymbol{g} + \Delta\boldsymbol{c}$. Postmultiplying last equation by $\Delta\boldsymbol{q}^\top\boldsymbol{F}^\top$ and computing the expectation will give $-\boldsymbol{F}\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{F}^\top = \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d},\boldsymbol{q}}\boldsymbol{F}^\top + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g},\boldsymbol{q}}\boldsymbol{F}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c},\boldsymbol{q}}\boldsymbol{F}^\top$, which turns $\boldsymbol{\Sigma}_{\boldsymbol{e}}$ into

$$\begin{aligned}\boldsymbol{\Sigma}_{\boldsymbol{e}} = {} & \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c}} + \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d},\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top + \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d},\boldsymbol{c}} + \\ & \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g},\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g},\boldsymbol{c}} + \boldsymbol{\Sigma}_{\boldsymbol{c},\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c},\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top - \boldsymbol{F}\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{F}^\top.\end{aligned}$$

Therefore, we have

$$\boldsymbol{F}\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{F}^\top = \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c}} + \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d},\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top +$$

$$\begin{aligned}\Delta\left[\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\right]_{[k]} &= \frac{1}{2}\left(\Delta\frac{\partial}{\partial\boldsymbol{q}}\left\{\left[\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\right]_{[k]}\right\}\boldsymbol{q} + \frac{\partial}{\partial\boldsymbol{q}}\left\{\left[\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\right]_{[k]}\right\}\Delta\boldsymbol{q}\right) \\ &= \frac{1}{2}\left(\left\{\sum_{l=1}^{4} q_{l-1}\frac{\partial}{\partial\boldsymbol{q}}\left[\left(\frac{\partial}{\partial\boldsymbol{q}}\left\{\left[\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\right]_{[k]}\right\}\right)_{[l]}\right]\right\} + \frac{\partial}{\partial\boldsymbol{q}}\left\{\left[\frac{\partial\left(\boldsymbol{\mathcal{J}}_{\boldsymbol{q}}^{\boldsymbol{v}}\right)_{[j]}}{\partial\boldsymbol{q}}\right]_{[k]}\right\}\right)\Delta\boldsymbol{q}.\end{aligned} \tag{43}$$

$$\boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d},\boldsymbol{c}} + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g},\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g},\boldsymbol{c}} + \boldsymbol{\Sigma}_{\boldsymbol{c},\boldsymbol{d}}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c},\boldsymbol{g}}\boldsymbol{\mathcal{Y}}^\top = \boldsymbol{\Sigma}_{\text{meas}}(\boldsymbol{q}) \tag{45}$$

where we denote the right side as $\boldsymbol{\Sigma}_{\text{meas}}(\boldsymbol{q})$, meaning that this part comes from measurements and estimated quaternion. All the covariances on the right side are related to input data, which can be evaluated very efficiently either using first-order approximation or Monte-Carlo simulation. Equation (45) presents a covariance criterion for the entire QPEP system. To estimate $\boldsymbol{\Sigma}_{\boldsymbol{q}}$, one can use a single pair of measurements to construct *ad hoc* equalities

$$\begin{aligned}\boldsymbol{F}_i\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{F}_i^\top \approx\ & \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d}_i}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g}_i}\boldsymbol{\mathcal{Y}}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c}_i} + \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d}_i,\boldsymbol{g}_i}\boldsymbol{\mathcal{Y}}^\top + \\ & \boldsymbol{\mathcal{V}}\boldsymbol{\Sigma}_{\boldsymbol{d}_i,\boldsymbol{c}_i} + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g}_i,\boldsymbol{d}_i}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\mathcal{Y}}\boldsymbol{\Sigma}_{\boldsymbol{g}_i,\boldsymbol{c}_i} + \\ & \boldsymbol{\Sigma}_{\boldsymbol{c}_i,\boldsymbol{d}_i}\boldsymbol{\mathcal{V}}^\top + \boldsymbol{\Sigma}_{\boldsymbol{c}_i,\boldsymbol{g}_i}\boldsymbol{\mathcal{Y}}^\top \\ =\ & \boldsymbol{\Sigma}_{\text{meas},i}(\boldsymbol{q})\end{aligned} \tag{46}$$

in which the subscript i denotes the index of employed measurements. Suppose that we have n sets of measurements to compute the globally optimal QPEP, the following optimization is constructed:

$$\underset{\boldsymbol{\Sigma}_{\boldsymbol{q}}\in\mathbb{S}^4,\boldsymbol{\Sigma}_{\boldsymbol{q}}\succeq 0}{\arg\min} \sum_{i=1}^{n} \left\| \boldsymbol{F}_i\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{F}_i^\top - \boldsymbol{\Sigma}_{\text{meas},i}(\boldsymbol{q}) \right\|^2. \tag{47}$$

However, (47) is not enough to solve $\boldsymbol{\Sigma}_{\boldsymbol{q}}$, because $\boldsymbol{F} \in \mathbb{R}^{3\times 4}$ so the optimization has infinity solutions. To fix the most appropriate one, the following two sufficient constraints must be added.

1) To guarantee the total covariance of QPEP, the equality (45) always holds.
2) The quaternion is constrained by $\|\boldsymbol{q}\| = 1$. However, in normalization of quaternion, there are always some very small numerical errors. Therefore, one can set an upper bound $\alpha > 0$ of such normalization error so that $\boldsymbol{\Sigma}_{\boldsymbol{q}^\top\boldsymbol{q}} \le \boldsymbol{\Sigma}_{\boldsymbol{q}^\top\boldsymbol{q},\max}$, which can be reduced to $4\boldsymbol{q}^\top\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{q} \le \alpha$.

The final optimization then comes to

$$\begin{aligned}&\underset{\boldsymbol{\Sigma}_{\boldsymbol{q}}\in\mathbb{S}^4,\boldsymbol{\Sigma}_{\boldsymbol{q}}\succeq 0}{\arg\min} \sum_{i=1}^{n} \left\| \boldsymbol{F}_i\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{F}_i^\top - \boldsymbol{\Sigma}_{\text{meas},i}(\boldsymbol{q}) \right\|^2 \\ &\text{s.t. } \boldsymbol{F}\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{F}^\top = \boldsymbol{\Sigma}_{\text{meas}}(\boldsymbol{q}) \\ &\boldsymbol{q}^\top\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{q} \le \tilde{\alpha}\end{aligned} \tag{48}$$

in which $\tilde{\alpha}$ is an equivalent threshold for the quaternion normalization error. Each of constraints in (48) will be important. The first equality constraint specifies the relative scale of $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ with respect to $\boldsymbol{\Sigma}_{\text{meas}}(\boldsymbol{q})$. One quaternion although has four components, it only has a three degrees of freedom. From the first constraint, it can be seen that the optimization without quaternion normalization constraint is rank-deficient, namely the covariance of q_0 is not fully observable for some cases. The second constraint fixes the uncertainty scale of q_0. Therefore, the estimation of $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ is complete. This concept can also be used for recovering quaternion covariance from a rotation covariance $\boldsymbol{\Sigma}_{\boldsymbol{R}}$ whose details are shown in the supplementary material. The positive semidefinite condition will be equivalent to another condition that the determinants of all principal subminors of $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ are positive. These determinants are in the polynomial form of $\boldsymbol{z} = \text{vec}(\boldsymbol{\Sigma}_{\boldsymbol{q}})$, which can be stacked as $m(\boldsymbol{z}) \le \boldsymbol{0}$. Then, using $(\boldsymbol{F}_i \otimes \boldsymbol{F}_i)\boldsymbol{z} \approx \text{vec}[\boldsymbol{\Sigma}_{\text{meas},i}(\boldsymbol{q})]$, (48) can be simplified as $\arg\min_{\boldsymbol{z}\in\mathbb{R}^{16}} \boldsymbol{z}^\top\boldsymbol{A}\boldsymbol{z} - \boldsymbol{h}^\top\boldsymbol{z}$ subject to the conditions: $(\boldsymbol{F}\otimes\boldsymbol{F})\boldsymbol{z} = \boldsymbol{\mathfrak{d}}, \boldsymbol{q}^\top\boldsymbol{\Sigma}_{\boldsymbol{q}}\boldsymbol{q} \le \tilde{\alpha}, m(\boldsymbol{z}) \le \boldsymbol{0}$, where $\boldsymbol{A} = \sum_{i=1}^{n}(\boldsymbol{F}_i\otimes\boldsymbol{F}_i)^\top(\boldsymbol{F}_i\otimes\boldsymbol{F}_i)$, $\boldsymbol{h} = \sum_{i=1}^{n} 2(\boldsymbol{F}_i\otimes\boldsymbol{F}_i)^\top\boldsymbol{\mathfrak{d}}$ with $\boldsymbol{\mathfrak{d}} = \text{vec}[\boldsymbol{\Sigma}_{\text{meas},i}(\boldsymbol{q})]$. By reducing the unknowns using $\boldsymbol{w} = \text{symvec}(\boldsymbol{\Sigma}_{\boldsymbol{q}})$, the optimization is

$$\underset{\boldsymbol{w}\in\mathbb{R}^{10}}{\arg\min}\, \boldsymbol{w}^\top\tilde{\boldsymbol{A}}\boldsymbol{w} - \tilde{\boldsymbol{h}}^\top\boldsymbol{w} \ \text{ s.t. } \boldsymbol{\mathcal{F}}\boldsymbol{w} = \boldsymbol{\psi},\ \tilde{m}(\boldsymbol{w}) \le \boldsymbol{s} \tag{49}$$

where $\tilde{\boldsymbol{A}} = (\boldsymbol{\mathcal{J}}_{\boldsymbol{w}}^{\boldsymbol{z}})^\top\boldsymbol{A}\boldsymbol{\mathcal{J}}_{\boldsymbol{w}}^{\boldsymbol{z}}, \tilde{\boldsymbol{h}}^\top = \boldsymbol{h}^\top\boldsymbol{\mathcal{J}}_{\boldsymbol{w}}^{\boldsymbol{z}}$, $\boldsymbol{\mathcal{F}} = (\boldsymbol{F}\otimes\boldsymbol{F})\boldsymbol{\mathcal{J}}_{\boldsymbol{w}}^{\boldsymbol{z}}$, $\boldsymbol{\psi} = \text{vec}[\boldsymbol{\Sigma}_{\text{meas}}(\boldsymbol{q})]$, $\boldsymbol{s} = (\tilde{\alpha}, \boldsymbol{0}^\top)^\top$, and $\tilde{m}(\boldsymbol{w})$ is the equivalent stacked inequality constraint. Equation (49) is a polynomial quadratic programming problem with nonlinear inequality constraints and linear equality constraints. By introducing Lagrange multipliers $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$, the Karush–Kuhn–Tucker (KKT) condition meets $2\tilde{\boldsymbol{A}}\boldsymbol{w} - \boldsymbol{h} + \boldsymbol{\mathcal{F}}^\top\boldsymbol{\lambda}_1 + \frac{\partial\tilde{m}(\boldsymbol{w})}{\partial\boldsymbol{w}}\boldsymbol{\lambda}_2 = \boldsymbol{0}$, $\boldsymbol{\mathcal{F}}\boldsymbol{w} - \boldsymbol{\psi} = \boldsymbol{0}$, and $\tilde{m}(\boldsymbol{w}) - \boldsymbol{s} = \boldsymbol{0}$, which also denote a polynomial system of $\boldsymbol{w}, \boldsymbol{\lambda}_1$, and $\boldsymbol{\lambda}_2$. Therefore, the optimization (49) can be solved globally optimally via global polynomial optimization, which can be relaxed via semidefinite programming.

Remark 5: It is also noted that since (48) follows linear matrix inequalities (LMIs), the globally optimal optimization may also be conducted via many relaxation techniques of LMI. The reasons of global optimality are as follows:

1) the optimization is typical and can be solved via the convex optimization theory;
2) the polynomial form of the KKT condition guarantees that all conditioning states can be solved globally optimally via polynomial equation solvers.

For instance, the `GpoSolver`[1] [58] and `Gloptipoly`[2] [59] achieve such semidefinite relaxation for globally optimal solution of polynomial optimization, which can generate C++ codes for highly efficient robotic estimation problems. Therefore, the solution for uncertainty description of $\boldsymbol{q}$ can be guaranteed accurate and robust. When $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ is computed, the covariance of $\boldsymbol{t}$ can be determined using

$$\boldsymbol{\Sigma}_{\boldsymbol{t}} = \frac{\partial\boldsymbol{\mathcal{T}}(\boldsymbol{q}^2)}{\partial\boldsymbol{q}}\boldsymbol{\Sigma}_{\boldsymbol{q}}\left[\frac{\partial\boldsymbol{\mathcal{T}}(\boldsymbol{q}^2)}{\partial\boldsymbol{q}}\right]^\top + \frac{\partial\boldsymbol{\mathcal{T}}(\boldsymbol{q}^2)}{\partial\boldsymbol{\varrho}}\boldsymbol{\Sigma}_{\boldsymbol{\varrho}}\left[\frac{\partial\boldsymbol{\mathcal{T}}(\boldsymbol{q}^2)}{\partial\boldsymbol{\varrho}}\right]^\top \tag{50}$$

where $\boldsymbol{\varrho}$ denotes all coefficient of $\boldsymbol{q}^2$ in $\boldsymbol{\mathcal{T}}(\boldsymbol{q}^2)$. Since $\partial\boldsymbol{\mathcal{T}}(\boldsymbol{q}^2)/\partial\boldsymbol{q}$ completely describes the nonlinearity of $\boldsymbol{\mathcal{T}}(\boldsymbol{q}^2)$ according to results in (43), the induced covariance of $\boldsymbol{t}$ is also unbiased and accurate.

There are still some problems while implementing the covariance estimation problem. The following are some practical considerations.

1) Scaling the Optimization: Another problem in implementation is that sometimes covariance matrices from measurements are very small. This will cause numerical degeneration problems

[1][Online]. Available: http://cmp.felk.cvut.cz/gposolver
[2][Online]. Available: http://homepages.laas.fr/henrion/software/gloptipoly

when optimizing (48). A practical method is to scale the problem in a proper manner. Note that if we premultiply a scaling factor $\varrho > 1$ to $\boldsymbol{w}$, forming a new variable $\tilde{\boldsymbol{w}} = \varrho \boldsymbol{w}$, the KKT condition and the optimality does not change. This will lead to a new scaled optimization

$$\underset{\tilde{\boldsymbol{w}} \in \mathbb{R}^{10}}{\arg\min}\, \tilde{\boldsymbol{w}}^\top \tilde{\boldsymbol{A}} \tilde{\boldsymbol{w}} - \varrho \tilde{\boldsymbol{h}}^\top \tilde{\boldsymbol{w}},\ \text{s.t.}\ \boldsymbol{\mathcal{F}} \tilde{\boldsymbol{w}} = \varrho \boldsymbol{a},\ \tilde{m}(\tilde{\boldsymbol{w}}) \leq \tilde{\boldsymbol{s}} \quad (51)$$

where $\tilde{\boldsymbol{s}}$ denotes an equivalent inequality upper bound. Via (51), the optimization will be conducted much more friendly to many optimizers.

2) Covariance Estimation: Simplified Optimization: For a system with huge loads of data, evaluating $\boldsymbol{F}_i$ for all measurements is computationally inefficient. It is notable that when $\boldsymbol{\Sigma}_{\boldsymbol{q}} \succeq \boldsymbol{0}$, the term $\boldsymbol{q}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}} \boldsymbol{q}$ will always be positive. Therefore the optimization (48) can be simplified to minimizing the value of $\boldsymbol{q}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}} \boldsymbol{q}$ such that

$$\underset{\boldsymbol{\Sigma}_{\boldsymbol{q}} \in \mathbb{S}^4, \boldsymbol{\Sigma}_{\boldsymbol{q}} \succeq \boldsymbol{0}}{\arg\min}\ \boldsymbol{q}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}} \boldsymbol{q},\ \text{s.t.}\ \boldsymbol{F} \boldsymbol{\Sigma}_{\boldsymbol{q}} \boldsymbol{F}^\top = \boldsymbol{\Sigma}_{\text{meas}}(\boldsymbol{q}) \quad (52)$$

whose form is much more elegant but will loose a little bit information. Equation (52) is also applicable to those cases that $\boldsymbol{D}$ is almost singular, e.g., those ones with ill-conditioned measurements. Writing (52) in a different manner, we observe that the resulting optimization

$$\underset{\boldsymbol{z} \in \mathbb{R}^{16}}{\arg\min} \left(\boldsymbol{q}^\top \otimes \boldsymbol{q}^\top\right) \boldsymbol{z},\quad \text{s.t.}\ \ (\boldsymbol{F} \otimes \boldsymbol{F})\, \boldsymbol{z} = \boldsymbol{\psi}, \boldsymbol{\Sigma}_{\boldsymbol{q}} \succeq \boldsymbol{0} \quad (53)$$

which is identical to

$$\underset{\boldsymbol{w} \in \mathbb{R}^{10}}{\arg\min}\, \boldsymbol{\zeta}^\top \boldsymbol{w},\quad \text{s.t.}\ \ \boldsymbol{\Xi} \boldsymbol{w} = \text{symvec}\left[\boldsymbol{\Sigma}_{\text{meas}}(\boldsymbol{q})\right]$$
$$\text{symmat}(\boldsymbol{w}) \succeq \boldsymbol{0} \quad (54)$$

where $\boldsymbol{\zeta} = \frac{\partial (\boldsymbol{q}^\top \otimes \boldsymbol{q}^\top) \boldsymbol{z}}{\partial \boldsymbol{w}}$, and $\boldsymbol{\Xi} = \frac{\partial (\boldsymbol{F} \otimes \boldsymbol{F}) \boldsymbol{z}}{\partial \boldsymbol{w}}$. Equation (54) is a conic programming problem, such that $\boldsymbol{\Sigma}_{\boldsymbol{q}} \in \mathbb{S}^+$, where $\mathbb{S}^+$ denotes the symmetric positive definite cone $\mathbb{S}^+ := \{\boldsymbol{X} \in \mathbb{S}^n, \text{mineig}(\boldsymbol{X}) \succeq 0\}$. Such a conic programming problem can be solved very efficiently on a modern computer at the level of several milliseconds.

IV. Experimental Results

A. Overview

The algorithms proposed in this article are implemented using C++ and MATLAB. The algorithmic procedures are summarized in Algorithm 1. For C++, we use the C++11 standard for programming. The practitioners can turn to https://github.com/zarathustr/LibQPEP for details of compilation and usage. Most of the comparisons are produced on a personal MacBook Pro 2019 computer with i7-8core 2.4-GHz processor. The MATLAB version is R2020a. We implement many Gröbner-basis solvers proposed in this article, which are used for comparisons with different types of QPEPs. In the efficiency test, we use the one with largest elimination template, namely the 249×249 one with action matrix size of 27×27. To deal with the computation of the Lie algebra on $\text{SO}(3)$ and $\text{SE}(3)$, the Sophus[3] library is employed. For all comparisons, the rotation, translation, and pose errors are defined as

$$\mathcal{E}_{\text{rot}} = \arccos \frac{\text{tr}(\boldsymbol{R}^\top \boldsymbol{R}_{\text{true}}) - 1}{2}$$
$$\mathcal{E}_{\text{trans}} = \|\boldsymbol{t} - \boldsymbol{t}_{\text{true}}\|^2$$
$$\mathcal{E}_{\text{pose}} = \|\boldsymbol{t}_{\text{pose}}\|$$

in which $\boldsymbol{R}_{\text{true}}$ and $\boldsymbol{t}_{\text{true}}$ stand for the reference rotation and translation, respectively; $\boldsymbol{t}_{\text{pose}}$ is the translation of the error pose $\boldsymbol{T}^{-1} \boldsymbol{T}_{\text{true}}$, with $\boldsymbol{T}$ the estimated pose and $\boldsymbol{T}_{\text{true}}$ the true value of the pose. For covariance estimation, we use CSDP[4] as the optimization solver. The CSDP is written in C programming language, which is portable to C++ applications. The pose graph optimization in this article is solved by the g2o[5] library, which is also written in C++. In the evaluation of covariance, we do not distinguish $\boldsymbol{\Sigma}_{\boldsymbol{R}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ because we develop in this article a method for lossless conversion between them (see supplementary material). Besides, we only compare the rotation results since the translation is just a function of rotation (see Section III-D) and (50). The covariance ellipse is employed to visualize covariance matrices in details regarding both the autocovariance and cross-covariance.

B. Accuracy and Efficiency: PnP Cases

A perspective model from a real camera is employed for simulation, with the focal lengths of $f_x = 810.27$, $f_y = 825.58$, principle points of $c_x = 568.66$, $c_y = 321.98$, and image size of 1280×720. World points are generated randomly. We use the perspective model (5) to compute image points and neglect

Algorithm 1: The QPEP Algorithmic Procedure.

Step 1: Construct the optimization objective function in (13), check if it is a QPEP via its gradient form (14). If it is a QPEP, transform it into (15), then generate mandatory matrices $\boldsymbol{W}, \boldsymbol{Q}$ via (17).

Step 2: Do variable elimination via (22), and derive a fundamental system in (23). Establish $\tilde{\boldsymbol{W}}$ and $\tilde{\boldsymbol{Q}}$ via (24). Save these matrices in symbolic functions. Then, obtain all local minima of the problem via the Gröbner-basis method in (19). Finally, inserting these local minima into the objective function, we obtain the least function value that corresponds to the globally optimal solution.

Step 3: If the users need secure certificate, they can deduce the solvability and observability via Section III-C.

Step 4: To obtain the covariance, first, we need to get the Jacobian matrices in (45) at the solved globally optimal solution. via `Step 2`. Finally, we construct the optimization (48) or (54) to get the covariance via convex programming.

[3][Online]. Available: https://github.com/strasdat/Sophus

[4][Online]. Available: https://github.com/coin-or/csdp

[5][Online]. Available: https://github.com/RainerKuemmerle/g2o

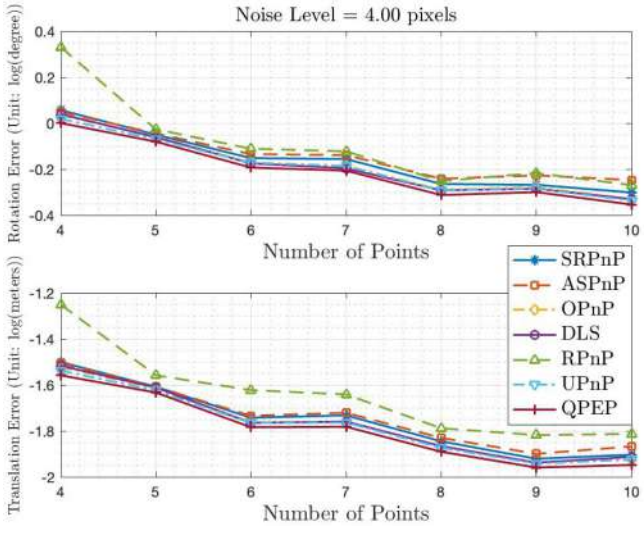


Fig. 2. Mean accuracy sensitivity to point numbers of multiple PnP solvers.

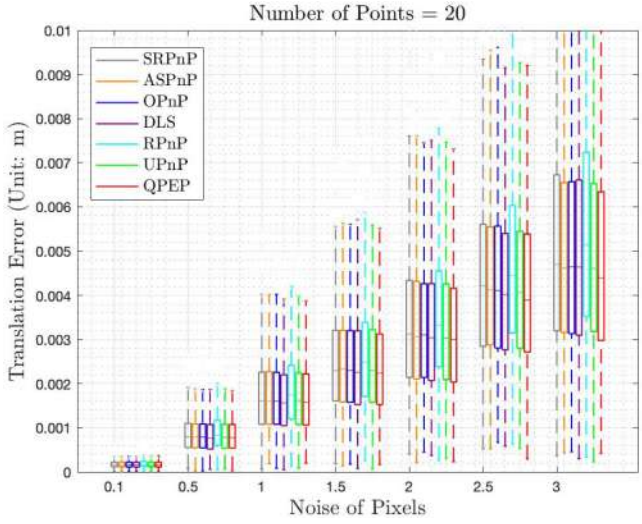


Fig. 4. Translation-error boxplots of different PnP solvers with different point numbers.

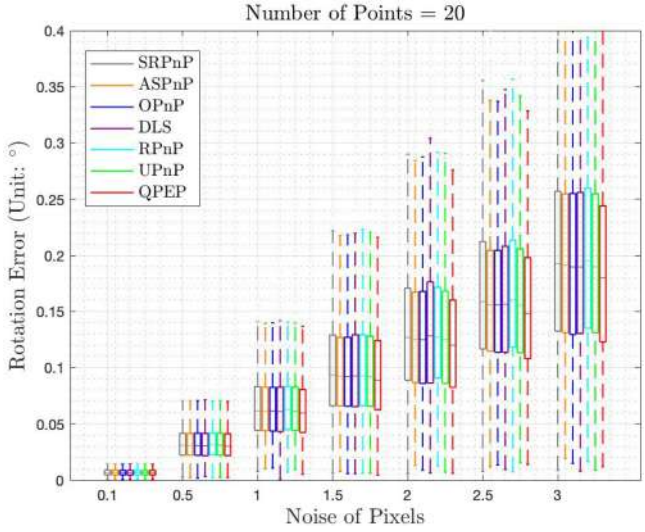


Fig. 3. Rotation-error boxplots of different PnP solvers with different point numbers.

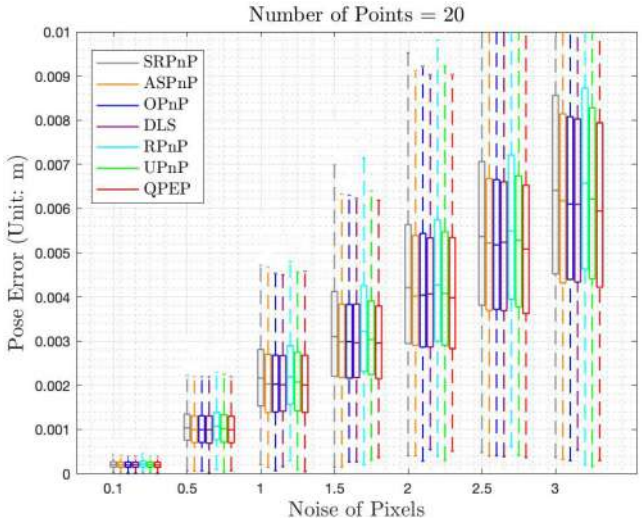


Fig. 5. Pose-error boxplots of different PnP solvers with different point numbers.

those ones with negative depth or out of the image boundary. To show the accuracy performance, the accuracy sensitivity to point numbers has been studied. We generate synthetic datasets with point number of $4 \sim 10$. The SRPnP [60], ASPnP [61], OPnP [9], DLS [62], RPnP [8], and UPnP [10] are employed for comparisons. We do not compare the accuracy with analytical but not globally optimal ones, e.g., the EPnP [4]. All these globally optimal candidates are implemented with no GN or LM optimizers, for fair comparison of global optimality. In Fig. 2, the studied case is contaminated by image noise with the level of 4 pixels. Then, using a different noise level, we depict the mean, maximum, and minimum error performances in Figs. 3 –5. These two pictures are presented in the way of `boxplot`. From the results, we can see that with increasing point numbers, all the algorithms reach better accuracy. For different candidates, the errors when point number equals 4 are worst. In all conditions, the proposed QPEP solver and OPnP have almost identical accuracy. The UPnP algorithm, which is efficient for having a smaller size of elimination template, does not work well for some cases. The reason is simple that small-size elimination template although leads to much faster computation of action matrix and solutions, will also inevitably bring about unstable behavior in variable elimination.

What we show next is the computational efficiency of various candidates. All these candidates are implemented in C++ programming language. For UPnP, we inherit the implementation from original authors.[6] We also implement the proposed QPEP with multicore (8-core CPU) support via `OpenMP` and `OpenCL`.

[6][Online]. Available: https://github.com/laurentkneip/opengv

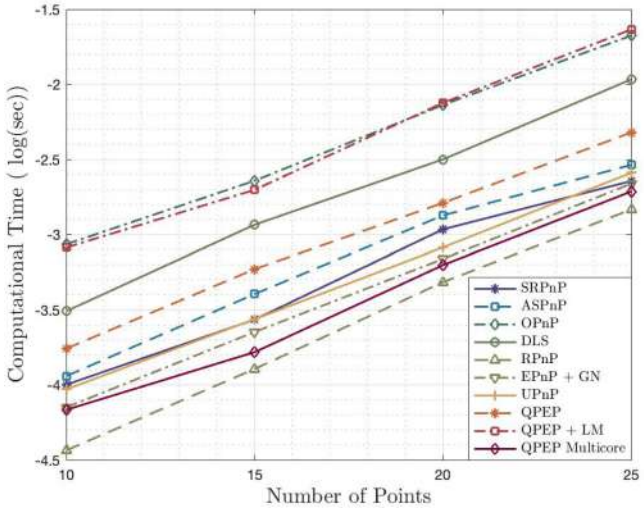


Fig. 6. Mean runtime performances of various PnP solvers (10 000 trials).

TABLE III
COMPUTATIONAL STATISTICS OF VARIOUS PNP METHODS (25 POINTS)

Algorithms	Max Time (s)	Min Time (s)	Complexity
SRPnP	5.1×10^{-3}	4.8×10^{-3}	$O(1.21n \log n)$
ASPnP	5.3×10^{-3}	4.9×10^{-3}	$O(1.32n \log n)$
OPnP	2.5×10^{-2}	2.2×10^{-2}	$O(6.1n \log n)$
RPnP	4.2×10^{-3}	2.7×10^{-3}	$O(n \log n)$
EPnP + GN	5.0×10^{-3}	4.9×10^{-3}	$O(1.15n \log n)$
UPnP	5.3×10^{-3}	5.0×10^{-3}	$O(1.30n \log n)$
QPEP	6.6×10^{-3}	6.0×10^{-3}	$O(1.57n \log n)$
QPEP + LM	2.6×10^{-2}	2.4×10^{-2}	$O(6.21n \log n)$
QPEP Multicore	4.8×10^{-3}	1.1×10^{-3}	$O(1.12n \log n)$

The execution time of different algorithms are collected using the platform shown in Section IV-A. A test with 10 000 times run has been conducted for average performance. The internal timer of MATLAB gives the runtime stats of the compared methods. The results are shown in Fig. 6 with computational statistics shown in Table III, from which we can see that when the number of points is small, the RPnP takes the least computational time. However, when the point number increases, the computational burden will be higher and the UPnP becomes the fastest one. The proposed method and its LM refined version is slower than those faster ones, but still holds an level of $10^{-3} \sim 10^{-2}$. We can also see from last experiment that, all these fast methods do not perform better than the proposed one. That is to say, the proposed method is slower, just because of it satisfactory robustness, which is inevitable for introducing a higher computational burden. In summary of computational complexity, we set the fastest algorithm as $O(n \log n)$ in the time unit of $\log \sec$. It should also be noticed that, with modern computational technology, parallel computing makes the matrix computation of the QPEP much faster. The QPEP with multicore is only a little slower than RPnP, and slightly faster EPnP with GN. The EPnP does not guarantee global optimality and may fail in corner cases. The RPnP does not guarantee a better accuracy than QPEP as shown in previous tests.

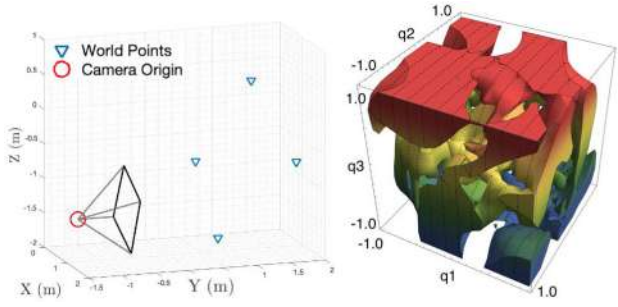


Fig. 7. (Left) Example of P4P problem. (Right) Feasible region satisfying solvability rule of (RS1) in Section III-C. The right subfigure does not have a colormap. The colors only represent feasible region areas.

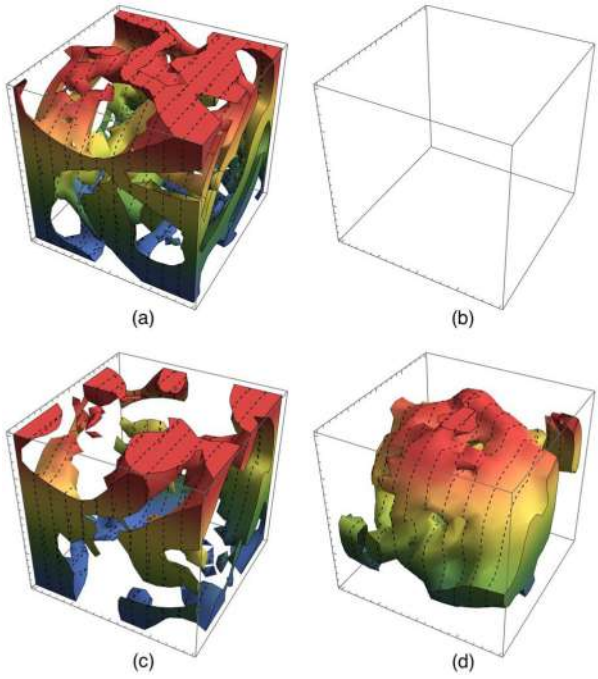


Fig. 8. Feasible region satisfying solvability rules of: (a) (RS31); (b) (RS32); (c) (RS21); and (d) (RS22) in Section III-C. The colored regions correspond to those $\boldsymbol{q}$ values that meet certain conditions. The figure axes are the same with that in right subfigure of Fig. 7. Likewise, the right subfigure does not have a colormap. The colors only represent feasible region areas.

C. Solvability/Observability Analysis: Examples of PnP

Consider a P4P problem with world points of $\boldsymbol{r}_1 = (1.489, 0.251, -0.5547)^\top$, $\boldsymbol{r}_2 = (1.719, 0.9978, 0.6086)^\top$, $\boldsymbol{r}_3 = (0.8988, 0.7825, -1.838)^\top$, and $\boldsymbol{r}_4$=$(1.192, 1.846, -0.7485)^\top$. When the perspective (pose) is $\boldsymbol{q} = (0.68471, 0.58041, 0.32478, 0.29802)^\top$, $\boldsymbol{t} = (-0.3266, -1.616, 1.341)^\top$ (m), camera pose and world points are illustrated in the left figure of Fig. 7. Following solvability rules in Section III-C, the feasible region of (RS1) is shown in the right plot of Fig. 7. Likewise, other feasible regions are plotted in Fig. 8, including (RS31), (RS32), (RS21), and (RS22). Note that, to generate these feasible regions, we use the equality $q_0 = \sqrt{1 - q_1^2 - q_2^2 - q_3^2} > 0$. For the part $q_0 < 0$,

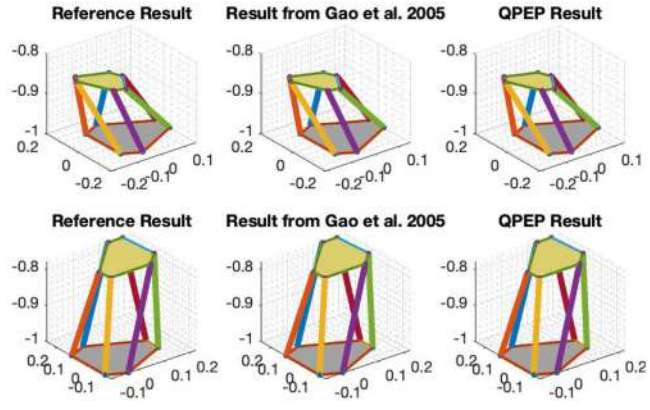


Fig. 9. Forward kinematics solutions of a hexapod Stewart platform. There are two poses in two different rows. The numerical differences between the method of Gao *et al.* and ours are at the level of 10^{-8}.

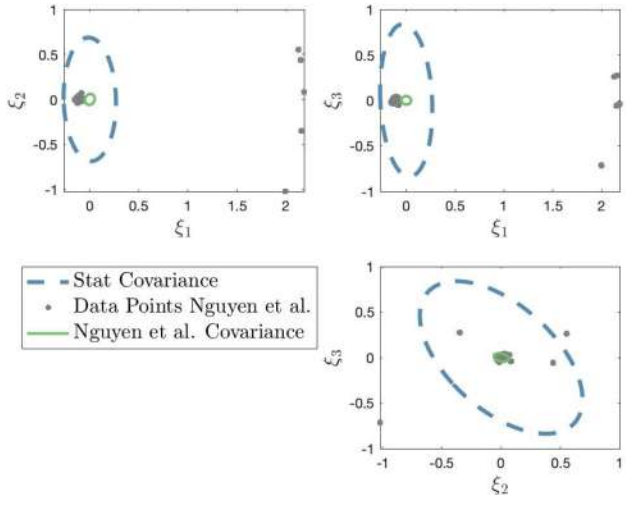


Fig. 10. Hand–eye calibration and uncertainty estimation using method of Nguyen *et al.* [46].

the geometry is dual. Therefore, studying $q_0 > 0$ is sufficient for analyzing the solvability. One may see that (RS32) does not hold. It also can be verified that the rule (RS3) never holds for the studied case. Therefore, for this case, there will be no repeated λ, i.e., no any triplet of these world points are collinear. Fig. 8(d) contains a set of all quaternions such that λ are fully complex. Therefore, Fig. 8(d) shows the complete infeasible region of the studied P4P case. This indicates the following.

1) Only for feasible poses, the imaged points have positive scale (depth) and is within the imaging plane. Those infeasible poses will break these conditions.
2) In some cases, the P4P problem degenerates, e.g., part of points coincide, which lead to infeasible solutions.

D. Uncertainty Description: Hand–Eye Calibration Experiments

In industrial engineering, many types of robots help intelligent manufacturing and automation. Typical ones include serial robots and parallel robots. To achieve precision grasping tasks, hand–eye calibration must be performed. The hand–eye calibration requires robot kinematics and camera poses as input. The forward kinematics of serial robots has been extensively studied. We now study the forward kinematics of the Stewart platform, which is a parallel one that constrained by mechanical structures. In Fig. 9, we show the computed poses of the moving platform of the Stewart platform. The six coordinates of the base and moving platform are given in the supplementary material. The compared algorithm is the one by Gao *et al.* [39] that follows symbolic elimination. From the results, we can see that the proposed method well computes the forward kinematics. The readers may reproduce the results via `test_stewart.m` in the demo kit.

Conventionally, the camera pose of the hand–eye calibration is obtained by PnP or the differential pose can be acquired from two-view algorithms. Thus, one may see that the uncertainty description of the hand–eye calibration involves the following three parts:

1) the uncertainty of robot kinematics;
2) the uncertainty of camera poses;
3) the uncertainty of hand–eye calibration propagated from covariances of robot kinematics and camera poses.

We first deals with the third step. In previous work of Nguyen *et al.* [46], the uncertainty of camera poses is evaluated offline with statistical observations. However, the PnP covariance is actually affected by its perspective, which is not considered previously. We refine this problem by evaluating the covariance of camera pose of PnP using results from Section III-D. The simplified optimization in Section III-D2 is utilized for covariance computation. The codes of Nguyen *et al.* [46] are programmed via Python of version 2.7.[7] We use the Python functionality of the MATLAB to generate the comparison results (see `nguyen_covariance.m` in the demo kit). We replicate the original dataset of covariance as shown in the codes of Nguyen *et al.* [46]. In our synthetic experiment, we first generate samples in which only three sets of hand–eye measurements exist. From previous results, one may know that the hand–eye calibration will be solvable if two pairs of measurements are available. However, the method of Nguyen *et al.* [46] cannot continue with only two pairs of measurements, and thus, we reach the current choice. We conduct 100 repeated simulations for different samples. The statistical covariance matrix is computed via statistical mean from solved pose data points of different methods. The covariance is visualized in the way of covariance ellipse. The results from method of Nguyen *et al.* [46] are shown in Fig. 10 and those from the proposed method are presented in Fig. 11 . The method of Nguyen *et al.* [46] uses a strategy to obtain estimation and uncertainty in a simultaneous MLE manner. However, seen from Fig. 10, it is clear that this method does not always achieve convergent estimates. Rather, since there are only three pairs of measurements in each sample, some of them are almost singular, making the algorithm hard to converge to a global minimum. The proposed QPEP avoids this problem, that it first gives global optimal solutions, and then,

[7][Online]. Available: https://github.com/dinhhuy2109/python-cope

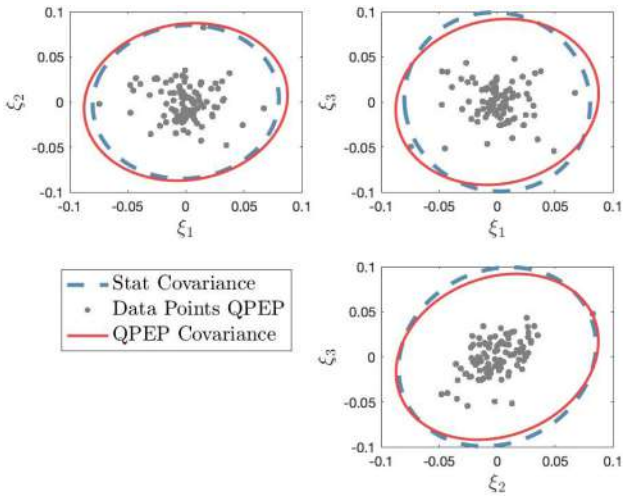


Fig. 11. Hand–eye calibration and uncertainty estimation using the proposed QPEP method.

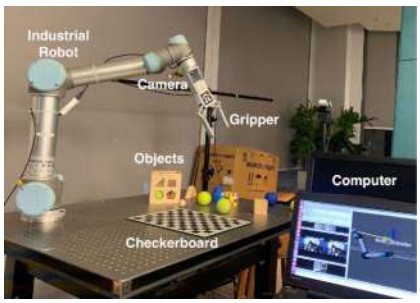


Fig. 12. Experimental setup and related devices.

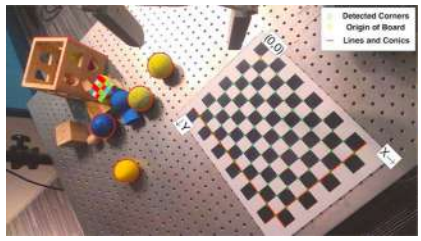


Fig. 13. Segmented multiple types of features during hand–eye calibration tasks.

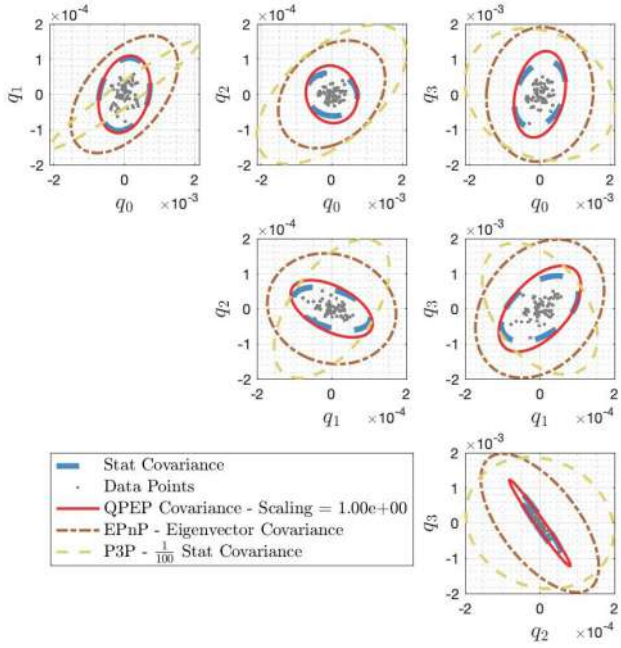


Fig. 14. PnP covariance matrices obtained from various methods. The scaling in this experiment for covariance estimation with (51) is $\varrho = 1.0$, namely, the optimization is not scaled.

estimates the covariance also in a global optimal framework. The synthetic results also indicate that the QPEP method is capable of estimating highly accurate pose and covariance even in the presence of very small numbers of measurements, which significantly enhances the robustness of the solution.

We then perform a standard industrial hand–eye calibration experiment, whose setup is shown in Fig. 12. An industrial robot UR5 and an Intel Realsense D435 camera are attached together. A gripper is working as an end-effector for grasping objects on the table. There is a 12×9 checkerboard as calibration pattern. The UR5 robot has the positioning accuracy of 3 mm and the covariance of the forward kinematics can be gathered statistically. Fig. 13 shows a view during calibration that illustrates multiple kinds of features that contribute to camera pose estimation. Nevertheless, due to the robustness issues, we only use point features for obtaining camera poses.

Now, the PnP uncertainty analysis is performed. For many PnP solvers, the covariance information is missing simply because of sophisticated algorithmic mechanisms. Here, we pick the EPnP and P3P as candidates for comparison. As is known, the EPnP can be cast into an eigendecomposition solver, and thus, the covariance of EPnP is shifted to the uncertainty analysis of eigenvalue problem. Thanks to Liounis *et al.*, a general uncertainty framework has been proposed in [63]. For P3P, the random sample consensus (RANSAC) is employed for solving the initial pose from three of the many points and verifying the pose with another fourth point. Further techniques can also achieve the maximum sample consensus, e.g., [64]. The covariance of the P3P is generated using the statistical mean. We choose inner corners of the 12×9 checkerboard, so that 88 point correspondences are presented. We use the platform in Fig. 12 to collect 63 samples. The covariance matrices calculated using different methods are shown in Fig. 14. In Fig. 14, we only show $1/100$ magnitude of the P3P covariance because the actual covariance of P3P is far larger than others. From the results, we can see that the eigenvector covariance of EPnP is not so accurate. The reason is that EPnP is not a global optimal solution. Yet, the uncertainty description does not consider the constraint of the system (17), and thus, the covariance estimation is also nonoptimal.

In fact, the covariance estimation scheme (51) is affected by different scalings. The results shown in Fig. 14 use the scaling of $\varrho = 1.0$, which is unscaled. Using different scalings, we are able to refine the covariance in a better fashion. Fig. 15 depicts

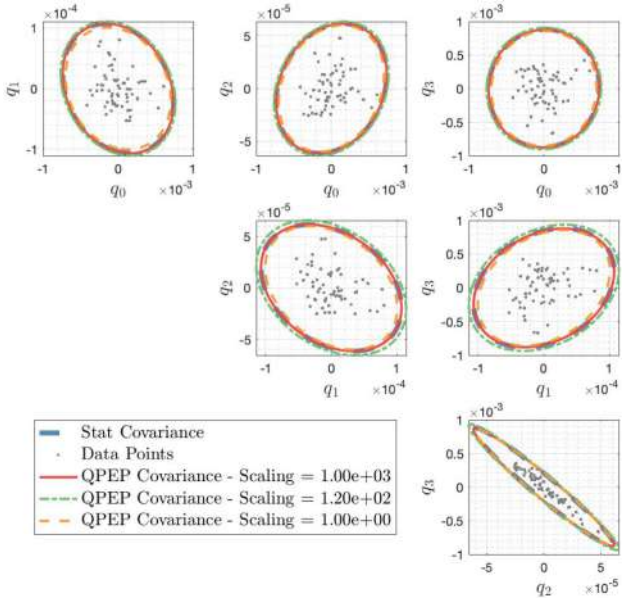


Fig. 15. Estimated PnP covariance matrices using QPEP with different scalings.

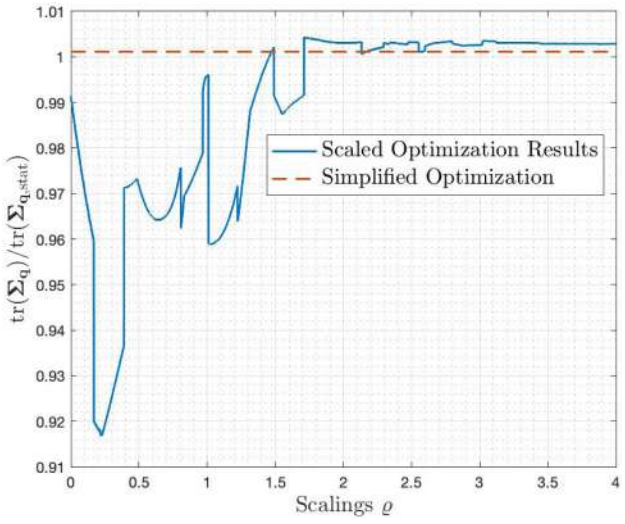


Fig. 16. Covariance metric values subject to different algorithms and scalings.

the performances with various scalings. One can observe that when $\varrho = 1 \times 10^3$, the estimated covariance is the best. When $\varrho = 1.2 \times 10^2$, the covariance bound is a little bit larger than the statistical covariance. When the practitioners are not sure which scaling should be selected, the simplified optimization (54) can be employed. Multiple tests with 4000 different ϱ are conducted, the metric $\mathrm{tr}(\boldsymbol{\Sigma}_{\boldsymbol{q}})/\mathrm{tr}(\boldsymbol{\Sigma}_{\boldsymbol{q},\mathrm{stat}})$ is proposed for evaluating the consistency of the estimated covariance with statistical one. The covariance is regarded as more accurate as ϱ approaches 1. The results are shown in Fig. 16.

One may see that tuning the scaling may be very difficult since the curve of the metric values seems very nonlinear with respect to ϱ. The simplified optimization almost achieves the metric value of 1 and is thus more adaptive and flexible than a manually tuned one.

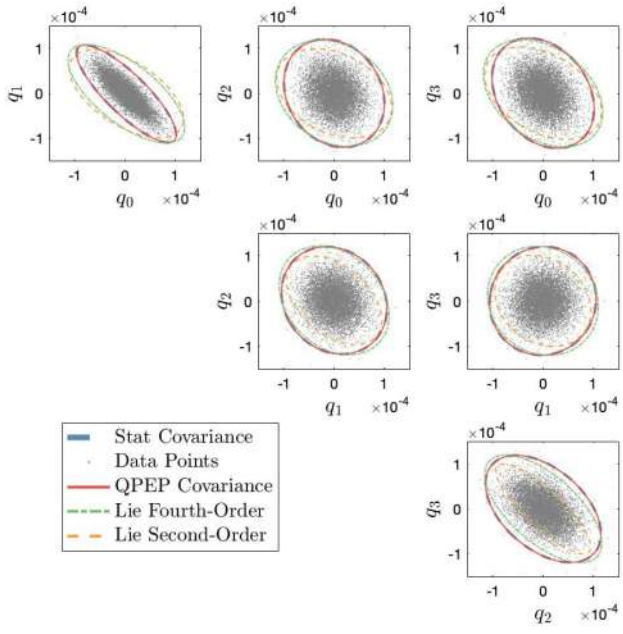


Fig. 17. Averaged hand–eye calibration covariance after 10 000 tests.

Finally, using PnP and hand–eye covariance experiences introduced previously, we can perform a complete uncertainty propagation for the hand–eye calibration system. Note that in hand–eye equation $\boldsymbol{AX} = \boldsymbol{XB}$, $\boldsymbol{A}$ and $\boldsymbol{B}$ are relative poses from robot kinematics and camera poses, respectively. Therefore, when the covariance matrices of two camera poses are obtained, the covariance of $\boldsymbol{B}$ is computed by compounding two poses and their uncertainty information. This part has been studied in the supplementary material in the framework of quaternion. As the performance of Nguyen *et al.* may not be satisfactory [46], we compare the results with the method of Barfoot *et al.* that propagates uncertainty in a Lie-algebra framework [42]. From the 63 samples gathered using platform in Fig. 12, we randomly pick up 10 000 subsets, while in each subset 30 samples are selected. The second-order and fourth-order Lie covariance expressions are evaluated for covariance computation. All covariances are averaged to obtain mean performances. From Fig. 17, it can be seen that, compared with the statistical one, the QPEP still obtains the best uncertainty information. The fourth-order Lie method is very close to the statistical one, but is still not perfect. The second-order Lie method, according to limited expansion of Lie exponentials, does not achieve a good uncertainty estimation. The proposed one is theoretically unbiased and globally optimal according to its optimization scheme. Another principle for classifying these candidates is the computational efficiency. The method of Nguyen *et al.* [46] has to iterate many times until a convergent solution. The fourth-order Lie expansion needs much longer formulae for covariance propagation. We compare all these variants together via C++ codes on the personal computer described in Section IV-A. Gathered runtime stats are shown in Table IV.

TABLE IV
RUNTIME PERFORMANCES FOR COVARIANCE ESTIMATION (10 000 CASES)

Algorithms	Computational Time
Proposed QPEP Method	4.3268 sec
Lie Second-order	4.0122 sec
Lie Fourth-order	10.4445 sec
Nguyen et al. [46]	532.26 sec

The results indicate that the QPEP method has a fast computation speed for covariance estimation. It is only a little bit slower than the second-order Lie method, which is verified to be not so accurate. The method of Nguyen *et al.* [46], due to its iterative nature, is not computationally efficient, and for real-time applications, the utilization may be limited.

E. Robotic Mapping: Point-to-Plane Registration and RGB-D Case

RGB-D scene reconstruction is powerful for robotic indoor navigation and mapping. An RGB-D sensor comprises two cameras: an RGB one and a depth camera. There are several emerging issues in RGB-D SLAM to be discussed in this experiment, which are as follows:

1) extrinsic calibration between RGB and depth cameras;
2) motion estimation between successive RGB and depth frames.

First, the extrinsic calibration between cameras can be regarded as an application of the hand–eye calibration. For the second issue, we present a method that combines the RGB-Depth motion and depth–depth motion in a hybrid QPEP. Since from each frame, we can pick up some feature points via feature extraction techniques, the motion between frames can be obtained by aligning RGB points and depth point cloud, namely, PnP. For continuous depth point clouds, we are also able to register them via 3-D registration. Here, the point-to-plane metric is chosen since in the indoor environment, there are many point-to-plane correspondences. A general pipeline is shown in Fig. 18, from which we can see that each step related to motion estimation or calibration can be solved by the proposed QPEP method. We select the ETH3D[8] indoor dataset for illustration. To compare the reconstruction and odometry performances, we pick up the method of bundle adjusted direct RGB-D SLAM (BADSLAM)[9] [65] and RGBD SLAM v2[10] [66]. The BADSLAM employs geometric (point-to-plane depth registration) and photometric (RGB luminous intensity) errors and bundle adjustment (BA) for refinement of the RGB-D pose. The BADSLAM does not rely on feature extraction for each frame, rather, using the direct photometric method, it can efficiently track nonopportunistic features. The RGBD SLAM v2 computes the initial pose from epipolar matching of feature extraction. Then, it verifies the pose via consensus of successive depth point clouds. In the end, RGBD SLAM v2 refines the odometry via a pose graph. Generally, for most cases,

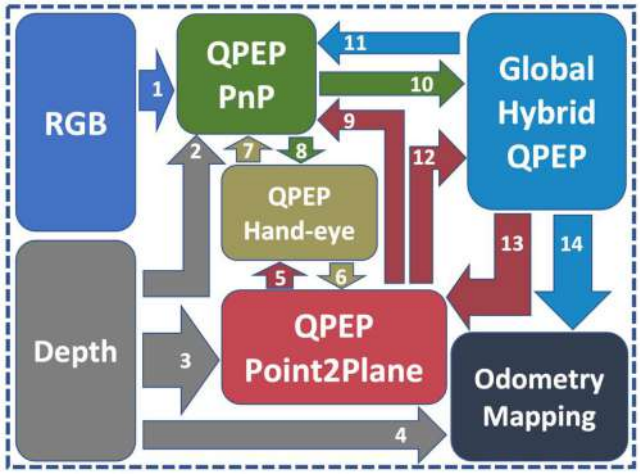


Fig. 18. Diagram of the developed RGB-D odometry/mapping scheme. The key steps are as follows. 1. Image transfer to the globally optimal QPEP-based PnP solver. 2. Depth point cloud transfer to PnP nodes as world points. 3. Depth point cloud transfer to the globally optimal QPEP-based point-to-plane registration solver. 4. Depth point cloud for scene stitching. 5. Depth transfer to hand–eye calibration. 6 and 7. Hand–eye calibration for alignment of depth point cloud and image. 8. Image transfer to hand–eye calibration. 9. The point-to-plane solver offers initial guess to PnP solver. 10 and 12. The PnP and point-to-plane solvers contribute to a hybrid QPEP problem. 11 and 13. Solving a hybrid QPEP problem with globally optimal solution gives a refined solution for PnP and point-to-plane registration. 14. The final estimated pose stitches all scenes. The final step outputs the odometry and mapping results together with uncertainty description.

BADSLAM is much more reliable and accurate. The reason is that BADSLAM extracts more substantial information than RGBD SLAM v2. Also, due to the utilization of the photometric error, the pose estimation and mapping can be conducted in real time. However, the BADSLAM also has some problems. First, the photometric-based optical flow assumes that the gray scales of the images are invariant, which may not hold in many applications. Second, it does not consider the unknown extrinsic calibration between RGB and depth cameras. Third, in the BA refinement, no covariance information is considered.

Prior to RGB-D scene reconstruction, because both PnP and hand–eye calibration has been studied in previous subsections, we take a brief survey of the QPEP performance for solving the point-to-plane registration. The Stanford bunny and dragon[11] laser scans are used for point cloud matching. Multiple datasets are chosen for inspection. For each pair of point clouds, ten random initial poses are generated for matching based on the iterative closest principle [67]. The best alignment is selected by choosing the one with the least point-to-plane matching error. Using linear approximation of rotation in (18), which is extensively used in popular softwares like PCL[12] and MATLAB, together with the developed QPEP method, the registration results under the point-to-plane metric are shown in Figs. 19 and 20.

It is clear that the QPEP method achieves a better registration accuracy than the linear approximation. This indicates that the

[8][Online]. Available: https://www.eth3d.net
[9][Online]. Available: https://github.com/ETH3D/badslam
[10][Online]. Available: https://github.com/felixendres/rgbdslam_v2
[11][Online]. Available: https://graphics.stanford.edu/data/3Dscanrep
[12][Online]. Available: https://github.com/PointCloudLibrary/pcl

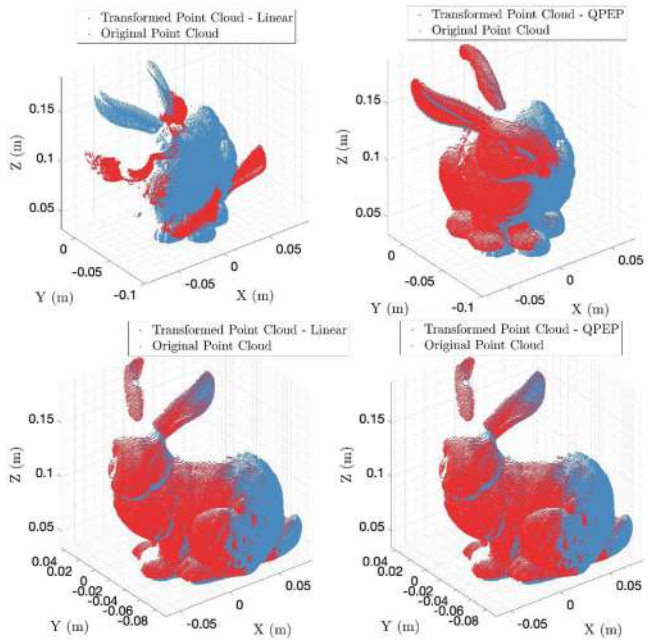


Fig. 19. Point-to-plane registration results using linear approximation and QPEP. (Upper) Results from `bun270.ply/bun315.ply`. (Bottom) Results from `bun000.ply/bun045.ply`.

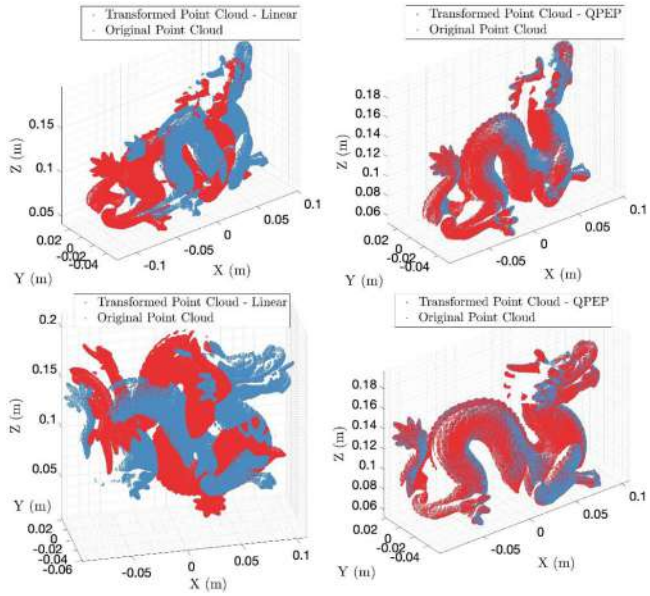


Fig. 20. Point-to-plane registration results using linear approximation and QPEP. (Upper) Results from `dragonStandRight_144.ply/dragonStandRight_168.ply`. (Bottom) Results from `dragonStandRight_168.ply/dragonStandRight_192.ply`.

globally optimal solution tends to make the system easier to converge to a global optimum, which also reflects that a linear approximation of the rotation matrix may not be sufficiently accurate for matching. Nevertheless, the linear approximation is more efficient than the QPEP method, which also accelerates the matching process.

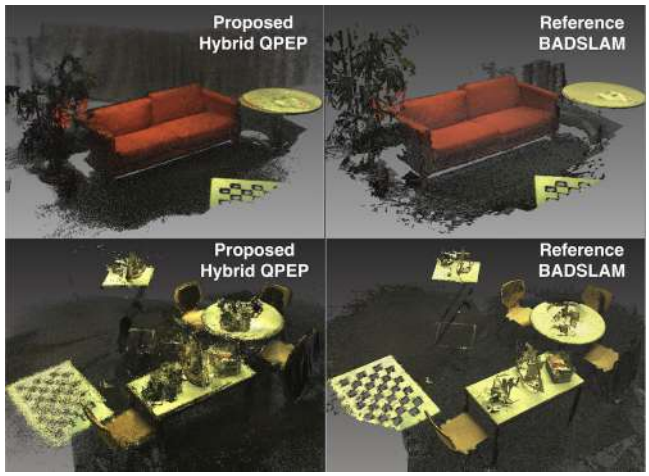


Fig. 21. Reconstructed scenes using proposed QPEP-based pipelines (*without* pose graph) and BADSLAM. The upper one is from the `plane_scene_1` dataset and the bottom one is from `desk_changing_1`.

Now that all the key steps in Fig. 18 have been verified to be effective when solved by the QPEP method. This pipeline first obtains a bare initial relative pose between depth frames using the QPEP-based point-to-plane registration. Then, it uses an approximation of the extrinsic calibration, say the identity matrix, for mapping the feature points in the RGB image to the depth frame. This gives rise to further refinement of the depth point cloud correspondences and extrinsic calibration, and finally, the PnP correspondences are also refined. For our method, the maximum iteration for such an interlaced refinement is set to 10. It is noticeable that the final pose is solved by a hybrid QPEP that combines the PnP and point-to-plane together. The globally optimal solution can be easily obtained by using the solution strategies in Section III-B. Likewise, we also employ the pose graph for the final refinement of poses, but more differently, the information matrix between two poses, as can be interpreted as an inverse of the covariance matrix, is not set to identity as appeared in many refinement pipelines. We use the covariance derived from the QPEP to better refine the pose graph. By picking up ETH3D datasets `plane_scene_1` and `desk_changing_1`, the reconstruction results using the proposed QPEP method (without pose graph) and the BADSLAM are shown in Fig. 21. The QPEP (without pose graph) results in Fig. 21 are a little bit noisy as compared with that of BADSLAM. This is caused by inevitable noise in the measurement but we still can see that the QPEP method already achieves good initial poses that to be refined by the pose graph (see Fig. 22). In Fig. 23, we show such refined poses in comparison with representatives. When building up the pose graph, loop detection using features has been employed [68], namely the FBoW[13] library. In the figure, the camera poses denote the selected keyframes according to the motion modal.

After a pose graph optimization (PGO) with covariance information from the QPEP, the refined trajectory almost coincides with the ground truth, which is slightly better than nonweighted

[13][Online]. Available: https://github.com/rmsalinas/fbow

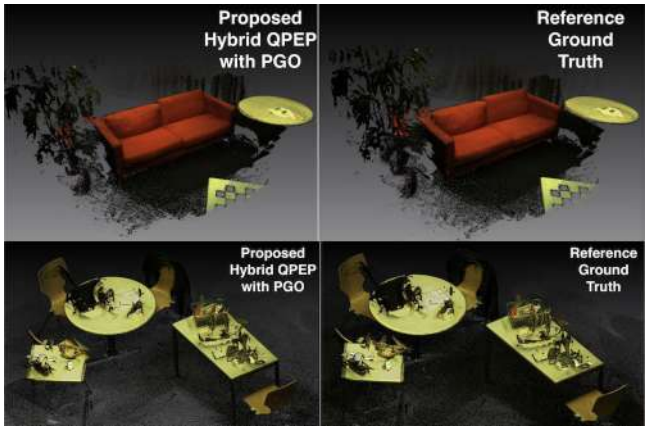


Fig. 22. Reconstructed scenes using proposed QPEP-based pipelines (*with* pose graph) and BADSLAM. The datasets are the same in Fig. 21.

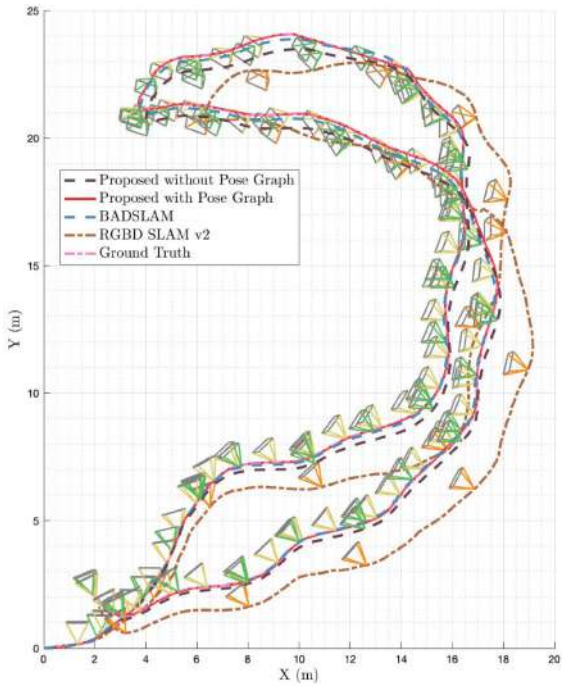


Fig. 23. RGB-D poses of different methods for dataset `plane_scene_1`.

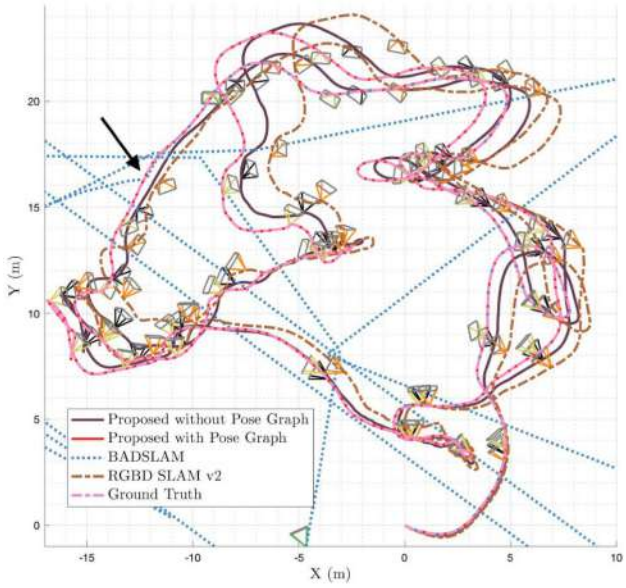


Fig. 24. RGB-D poses of different methods for dataset `desk_2`.

TABLE V
RMSEs OF TRAJECTORIES FOR DATASET `plane_scene_1` AND `desk_2`

Algorithms	X (m)	Y (m)	Z (m)
Proposed QPEP with PGO	**0.311**	**0.435**	**0.382**
Proposed QPEP without PGO	2.124	1.987	2.012
BADSLAM (with BA)	**0.282**	**0.294**	**0.317**
RGBD SLAM v2	5.821	5.053	6.178
Algorithms	X (m)	Y (m)	Z (m)
Proposed QPEP with PGO	**0.627**	**0.592**	**0.761**
Proposed QPEP without PGO	**2.783**	**3.472**	**3.709**
BADSLAM (with BA)	NA	NA	NA
RGBD SLAM v2	4.541	4.257	5.951

TABLE VI
COMPUTATIONAL EFFICIENCY OF RGBD METHODS

Algorithms	Single-Frame Computation Time (s)
Proposed QPEP with PGO	0.03465
Proposed QPEP without PGO	0.02879
BADSLAM (with BA)	0.03011
RGBD SLAM v2	0.18927

BA from BADSLAM. The PGO can quickly converge to an optimal value within 0.01 s with given initial estimate from QPEP pipeline. BADSLAM may fail when the gray-scale assumption is broken. A typical example is the ETH3D dataset `desk_2`, whose results are presented in Fig. 24. In the initial stage, BADSLAM works well with the input data. However, it diverges since the position indicated by the arrow and never converges again. The root mean-squared errors (RMSEs) of the trajectory for various methods are shown in Table V. In the table, the black bold figures denote the best method, while the blue bold figures denotes the second best algorithm. BADSLAM combines the geometric and photometric errors in one unified error metric and solves it by nonlinear optimization. When one of these errors degenerates, the entire optimization may also degrade, leading to unexpected crash. In our pipeline shown in Fig. 18, each key step can be verified by interlaced checking, thus makes the final results much more reliable. The flexibility of solving *ad hoc* QPEPs as an entirely hybrid one also gives convenience to the implementation and verification, which does not require those sophisticated operations like online numerical Jacobian computation. The computational efficiency of various method has been summarized in Table VI, where the single-frame statistics are obtained in average using a dataset with 4 512 images. One may observe that the proposed QPEP with PGO is slightly slower than BADSLAM with BA. The reason is that BADSLAM utilizes the photometric error, which is fundamentally faster than the feature-matching method employed in the QPEP-based RGBD SLAM.

V. Conclusion

In this article, many important pose estimation problems were revisited. They were unified as QPEPs. We showed that although the estimation was nonconvex, globally optimal solutions can be obtained with efficient approaches. We also gave solvability/observability analysis and optimal uncertainty description of the derived solutions. The proposed method was verified to be accurate and efficient for solving many problems including PnP, hand–eye calibration, point-to-plane registration, etc. The capability of solving hybrid QPEPs also gave rise to a new effective scheme of the RGB-D scene reconstruction. The generality of the proposed method gave a universal tool for solving many robotic pose estimation problems and can be a candidate for real-time applications. However, we also observed that the proposed method was inevitably slower than some compared candidates. The computational burden of the proposed method is expected to be optimized in the near future.

From (17), one can see that the kernel problem of solving QPEPs is the solution to polynomial equations. It was also shown that (17) is an eigenvalue problem. In the future, we will pay our efforts to finding more efficient solvers of these problems. Moreover, from the derived quaternion-only equation (24), if one solves it again using the gradient-descent method, a new equation of higher order will be generated. This equation has the same subset of solutions with (17). That is to say, there is a passage of roots between lower order and higher order polynomial systems, which may be quite essential for elegant solution of such problems by order reduction.

Acknowledgment

The authors would like to thank Prof. F. C. Park from Seoul National University, Republic of Korea, and Prof. F. Thomas from Institut de Rob'otica i Inform'atica Industrial, Spain, for fruitful discussions on this work. The authors would also like to thank Tencent Robotics X and Unity Drive Inc. for providing essential experimental devices and services for this work.

References

[1] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," 2017, *arXiv:1711.00199*.

[2] J. Wu, "Unified attitude determination problem from vector observations and hand-eye measurements," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 56, no. 5, pp. 3941–3957, Oct. 2020.

[3] L. Quan and Z. Lan, "Linear N-point camera pose determination," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 8, pp. 774–780, Aug. 1999.

[4] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate O(n) solution to the PnP problem," *Int. J. Comput. Vis.*, vol. 81, no. 2, pp. 155–166, 2009.

[5] Z. Kukelova, M. Bujnak, and T. Pajdla, "Polynomial eigenvalue solutions to minimal problems in computer vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1381–1393, Jul. 2012.

[6] V. Larsson, K. Åström, and M. Oskarsson, "Efficient solvers for minimal problems by syzygy-based reduction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2383–2392.

[7] G. Schweighofer and A. Pinz, "Globally optimal $O(n)$ solution to the PnP problem for general camera models," in *Proc. Brit. Mach. Vis. Conf.*, 2008, pp. 1–10.

[8] S. Li, C. Xu, and M. Xie, "A robust $O(n)$ solution to the perspective-n-point problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1444–1450, Jul. 2012.

[9] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi, "Revisiting the PnP problem: A fast, general and optimal solution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 2344–2351.

[10] L. Kneip, H. Li, and Y. Seo, "UPnP: An optimal $O(n)$ solution to the absolute pose problem with universal applicability," in *Proc. Brit. Mach. Vis. Conf.*, 2014, pp. 127–142.

[11] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Rev.*, vol. 38, no. 1, pp. 49–95, 1996.

[12] H. H. Chen and T. S. Huang, "Matching 3-D line segments with applications to multiple-object motion estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 10, pp. 1002–1008, Oct. 1990.

[13] A. Vakhitov, J. Funke, and F. Moreno-Noguer, "Accurate and linear time pose estimation from points and lines," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 583–599.

[14] D. Zou, Y. Wu, L. Pei, H. Ling, and W. Yu, "StructVIO: Visual-inertial odometry with structural regularity of man-made environments," *IEEE Trans. Robot.*, vol. 35, no. 4, pp. 999–1013, Aug. 2019.

[15] C. Xu, L. Zhang, L. Cheng, and R. Koch, "Pose estimation from line correspondences: A complete analysis and a series of solutions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1209–1222, Jun. 2017.

[16] L. Zhang, H. Lu, X. Hu, and R. Koch, "Vanishing point estimation and line classification in a Manhattan world with a unifying camera model," *Int. J. Comput. Vis.*, vol. 117, no. 2, pp. 111–130, 2016.

[17] S. Agostinho, J. Gomes, and A. Del Bue, "CvxPnPL: A unified convex solution to the absolute pose estimation problem from point and line correspondences," 2019, *arXiv:1907.10545*.

[18] L. Zhou, D. Koppel, and M. Kaess, "A complete, accurate and efficient solution for the perspective-n-line problem," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 699–706, Apr. 2021.

[19] S. De Ma, "Conics-based stereo, motion estimation, and pose determination," *Int. J. Comput. Vis.*, vol. 10, no. 1, pp. 7–25, 1993.

[20] R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3D robotics hand/eye calibration," *IEEE Trans. Robot. Autom.*, vol. 5, no. 3, pp. 345–358, Jun. 1989.

[21] Y. C. Shiu and S. Ahmad, "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form $AX = XB$," *IEEE Trans. Robot. Autom.*, vol. 5, no. 1, pp. 16–29, Feb. 1989.

[22] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-9, no. 5, pp. 698–700, Sep. 1987.

[23] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1991, pp. 2724–2729.

[24] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," *Robot, Sci. Syst.*, vol. 2, no. 4, 2009, Art. no. 435.

[25] S. T. Goh and K.-S. Low, "Survey of global-positioning-system-based attitude determination algorithms," *J. Guid. Control. Dyn.*, vol. 40, no. 6, pp. 1321–1335, 2017.

[26] F. C. Park, J. Kirn, and C. Kee, "Geometric descent algorithms for attitude determination using the global positioning system," *J. Guid. Control. Dyn.*, vol. 23, no. 1, pp. 26–33, 2000.

[27] C. Chun and F. C. Park, "Dynamics-based attitude determination using the global positioning system," *J. Guid. Control. Dyn.*, vol. 24, no. 3, pp. 466–473, 2001.

[28] D. Willi and M. Rothacher, "GNSS attitude determination with non-synchronized receivers and short baselines onboard a spacecraft," *GPS Solutions*, vol. 21, no. 4, pp. 1605–1617, 2017.

[29] N. Trawny, X. S. Zhou, K. Zhou, and S. I. Roumeliotis, "Interrobot transformations in 3-D," *IEEE Trans. Robot.*, vol. 26, no. 2, pp. 226–243, Apr. 2010.

[30] X. S. Zhou and S. I. Roumeliotis, "Determining 3-D relative transformations for any combination of range and bearing measurements," *IEEE Trans. Robot.*, vol. 29, no. 2, pp. 458–474, Apr. 2013.

[31] W. Khalil and S. Guegan, "Inverse and direct dynamic modeling of Gough-Stewart robots," *IEEE Trans. Robot.*, vol. 20, no. 4, pp. 754–761, Aug. 2004.

[32] X.-S. Gao, X.-R. Hou, J. Tang, and H.-f. Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 8, pp. 930–943, Aug. 2003.

[33] H. H. Chen, "Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 6, pp. 530–541, Jun. 1991.

[34] K. Daniilidis, "Hand-eye calibration using dual quaternions," *Int. J. Robot. Res.*, vol. 18, no. 3, pp. 286–298, 1999.

[35] L. Wu, J. Wang, L. Qi, K. Wu, H. Ren, and M. Q. Meng, "Simultaneous hand-eye, tool-flange, and robot-robot calibration for comanipulation by solving the $\boldsymbol{AXB} = \boldsymbol{YCZ}$ problem," *IEEE Trans. Robot.*, vol. 32, no. 2, pp. 413–428, Apr. 2016.
[36] B. Jiang, B. D. Anderson, and H. Hmam, "3-D relative localization of mobile systems using distance-only measurements via semidefinite optimization," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 56, no. 3, pp. 1903–1916, Jun. 2020.
[37] M. Raghavan, "The stewart platform of general geometry has 40 configurations," 1993.
[38] P. Dietmaier, "The Stewart-Gough platform of general geometry can have 40 real postures," in *Advances in Robot Kinematics: Analysis and Control*. Berlin, Germany: Springer, 1998, pp. 7–16.
[39] X.-S. Gao, D. Lei, Q. Liao, and G.-F. Zhang, "Generalized Stewart–Gough platforms," *IEEE Trans. Robot.*, vol. 21, no. 2, pp. 141–151, Apr. 2005.
[40] J. Borras and F. Thomas, "On the primal and dual forms of the Stewart platform pure condition," *IEEE Trans. Robot.*, vol. 28, no. 6, pp. 1205–1215, Dec. 2012.
[41] Y. Wang and G. S. Chirikjian, "Error propagation on the Euclidean group with applications to manipulator kinematics," *IEEE Trans. Robot.*, vol. 22, no. 4, pp. 591–602, Aug. 2006.
[42] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Trans. Robot.*, vol. 30, no. 3, pp. 679–693, Jun. 2014.
[43] J. G. Mangelson, M. Ghaffari, R. Vasudevan, and R. M. Eustice, "Characterizing the uncertainty of jointly distributed poses in the lie algebra," *IEEE Trans. Robot.*, vol. 36, no. 5, pp. 1371–1388, Oct. 2020.
[44] N. Yang, L. V. Stumberg, R. Wang, and D. Cremers, "D3VO: Deep depth, deep pose and deep uncertainty for monocular visual odometry," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1281–1292.
[45] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1851–1858.
[46] H. Nguyen and Q.-C. Pham, "On the covariance of x. in $\boldsymbol{AX} = \boldsymbol{XB}$," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1651–1658, Dec. 2018.
[47] M. Raghavan, "The Stewart platform of general geometry has 40 configurations," *in Proc. Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf. Amer. Soci. Mech. Eng.*, 1991. pp. 397–402.
[48] F. M. Mirzaei and S. I. Roumeliotis, "A Kalman-filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 1143–1156, Oct. 2008.
[49] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *Int. J. Robot. Res.*, vol. 32, no. 6, pp. 690–711, 2013.
[50] M. Li and A. I. Mourikis, "Online temporal calibration for camera-IMU systems: Theory and algorithms," *Int. J. Robot. Res.*, vol. 33, no. 7, pp. 947–964, 2014.
[51] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. Robot.*, vol. 33, no. 2, pp. 249–265, Apr. 2017.
[52] G. Costante and M. Mancini, "Uncertainty estimation for data-driven visual odometry," *IEEE Trans. Robot.*, vol. 36, no. 6, pp. 1738–1757, Dec. 2020.
[53] D. Wang, *Elimination Practice: Software Tools and Applications*. Singapore: World Scientific, 2004.
[54] S. Sarabandi, A. Shabani, J. M. Porta, and F. Thomas, "On closed-form formulas for the 3-D nearest rotation matrix problem," *IEEE Trans. Robot.*, vol. 36, no. 4, pp. 1333–1339, Aug. 2020.
[55] J. Wu, M. Liu, Z. Zhou, and R. Li, "Fast symbolic 3-D registration solution," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 2, pp. 761–770, Apr. 2020.
[56] C. Belta and V. Kumar, "An SVD-based projection method for interpolation on SE(3)," *IEEE Trans. Robot. Autom.*, vol. 18, no. 3, pp. 334–345, Jun. 2002.
[57] R. B. King, *Beyond the Quartic Equation*. Berlin, Germany: Springer, 2009.
[58] J. Heller and T. Pajdla, "GpoSolver: A MATLAB/C toolbox for global polynomial optimization," *Optim. Methods Softw.*, vol. 31, no. 2, pp. 405–434, 2016.
[59] D. Henrion, J. B. Lasserre, and J. Lofberg, "GloptiPoly 3: Moments, optimization and semidefinite programming," *Optim. Methods Softw.*, vol. 24, no. 4/5, pp. 761–779, 2009.
[60] P. Wang, G. Xu, Y. Cheng, and Q. Yu, "A simple, robust and fast method for the perspective-n-point problem," *Pattern Recognit. Lett.*, vol. 108, pp. 31–37, 2018.
[61] Y. Zheng, S. Sugimoto, and M. Okutomi, "ASPnP: An accurate and scalable solution to the perspective-n-point problem," *IEICE Trans. Inf. Syst.*, vol. 96, no. 7, pp. 1525–1535, 2013.
[62] J. A. Hesch and S. I. Roumeliotis, "A direct least-squares (DLS) method for PnP," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 383–390.
[63] A. J. Liounis and J. A. Christian, "Techniques for generating analytic covariance expressions for eigenvalues and eigenvectors," *IEEE Trans. Signal Process.*, vol. 64, no. 7, pp. 1808–1821, Apr. 2016.
[64] F. Wen, R. Ying, Z. Gong, and P. Liu, "Efficient algorithms for maximum consensus robust fitting," *IEEE Trans. Robot.*, vol. 36, no. 1, pp. 92–106, Feb. 2020.
[65] T. Schops, T. Sattler, and M. Pollefeys, "BADSLAM: Bundle adjusted direct RGB-D SLAM," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 134–144.
[66] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-D mapping with an RGB-D camera," *IEEE Trans. Robot.*, vol. 30, no. 1, pp. 177–187, Feb. 2014.
[67] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.
[68] D. Galvez-Lopez and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1188–1197, Oct. 2012.

Jin Wu (Member, IEEE) was born in May 1994, in Zhenjiang, China. He received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China. He is currently working toward the Ph.D. degree with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong.

Since 2018, he has been a Research Assistant with the Hong Kong University of Science and Technology. From 2012 to 2018, he was in the UAV industry. In 2013, he was a visiting student with Groep T, KU Leuven. From 2019 to 2020, he was with Tencent Robotics X, Shenzhen, China.

Yu Zheng (Senior Member, IEEE) received the B.E. and Ph.D. degrees from Shanghai Jiao Tong University, Shanghai, China, in 2001 and 2007, respectively, and the Ph.D. degree from the University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, in 2014.

From 2014 to 2018, he was an Assistant Professor with the University of Michigan–Dearborn. In 2018, he joined Tencent Robotics X, Shenzhen, China, where he is currently a Principal Research Scientist. His research interests include multicontact/multibody robotic systems, robotic grasping and manipulation, legged robots, and geometric algorithms for robotics.

Dr. Zheng serves as an Associate Editor for the IEEE ROBOTICS AND AUTOMATION LETTERS.

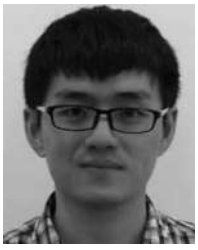
Zhi Gao (Senior Member, IEEE) received the B.Eng. and the Ph.D. degrees from Wuhan University, Wuhan, China in 2002 and 2007, respectively.

He is currently a Full Professor with Wuhan University. Since 2019, he has been supported by the Distinguished Professor program of Hubei Province and the National Young Talent Program, China. He has authored and co-authored more than 70 research papers in top journals and conferences, such as *International Journal of Computer Vision*, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, Conference on Computer Vision and Pattern Recognition, European Conference on Computer Vision, etc. His research interests include computer vision, machine learning, remote sensing and their applications.

Yi Jiang (Member, IEEE) was born in Ezhou, Hubei, China. He received the B.E., M.S., and Ph.D. degrees from Northeastern University, Shenyang, China.

He is currently a Postdoc with the City University of Hong Kong, Hong Kong. His research interests include networked control systems, industrial process operational control, reinforcement learning, and event-triggered control.

Dr. Jiang is an Associate Editor for *Advanced Control for Applications: Engineering and Industrial Systems* (Wiley). He was the recipient of Excellent Doctoral Dissertation Award from Chinese Association of Automation in 2021.



Xiangcheng Hu was born in 1995, in Macheng, Hubei, China. He received the B.Sc. degree from the North University of China, Taiyuan, China, in 2017, and the M.S. degree from Beihang University, Beijing, China, in 2020. He is currently working toward the Ph.D. degree with the Robotics and Multi-Perception Laboratory, Hong Kong University of Science and Technology, Hong Kong, supervised by Prof. M. Liu.

His current research interests include LiDAR mapping and sensor fusion for robotics.

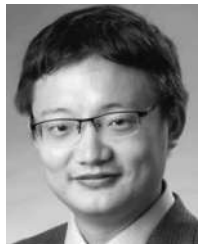
Yilong Zhu was born in 1994. He received the B.Sc. degree in 2017 from the Harbin Institute of Technology, Harbin, China, and the M.S. degree in electrical engineering in 2018 from the Hong Kong University of Science and Technology, Hong Kong, where he is currently working toward the Ph.D. degree with the RAM-LAB, supervised by Prof. M. Liu.

His current research interests include LiDAR and sensor fusion for robotics.

Jianhao Jiao was born in 1994. He received the B.Eng. degree in instrument science from Zhejiang University, Hangzhou, China, in 2017, and the Ph.D. degree from the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong, in 2021, supervised by Prof. M. Liu.

His research interests include state estimation, SLAM, sensor fusion, and computer vision. He has authored and co-authored his research works in top robotics venues including IEEE TRANSACTIONS ON ROBOTICS, International Conference on Robotics and Automation, and IEEE/RSJ International Conference on Intelligent Robots and Systems.

Dr. Jiao is a Reviewer for the IEEE TRANSACTIONS ON ROBOTICS.



Ming Liu (Senior Member, IEEE) received the B.A. degree in automation from Tongji University, Shanghai, China, in 2005, and the Ph.D. degree from the Department of Mechanical and Process Engineering, ETH Zurich, Zurich, Switzerland, in 2013, supervised by Prof. R. Siegwart.

He is currently with the Department of Electronic and Computer Engineering and Computer Science and Engineering Department, Robotics Institute, The Hong Kong University of Science and Technology, Hong Kong, as an Associate Professor. He is currently also the Chairman with Shenzhen Unity Drive Inc., China. His research interests include dynamic environment modelling, deep-learning for robotics, 3-D mapping, machine learning, and visual control. He has authored and co-authored many popular papers in top robotics journals including IEEE TRANSACTIONS ON ROBOTICS, and IEEE ROBOTICS AND AUTOMATION MAGAZINE.

Dr. Liu was the recipient of the European Micro Aerial Vehicles competition (EMAV) 2009 (second place) and two awards from International Aerial Robotics Competition (IARC) 2014 as a Team Member, the Best Student Paper Award as first author from EEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI) 2012, the Best Paper Award in information from the IEEE International Conference on Information and Automation (IEEE ICIA) 2013 as first author, the Best Paper Award Finalists as co-author and the Best RoboCup Paper Award from the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2013, the Best Conference Paper Award from IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems 2015, the Best Student Paper Finalist for IEEE International Conference on Real-time Computing and Robotics (RCAR) 2015, the Best Student Paper Finalist for IEEE International Conference on Robotics and Biomimetics (IEEE ROBIO) 2015, the Best Student Paper from IEEE ROBIO 2019, the Best Student Paper Award from IEEE International Conference on Advanced Robotics 2017, the Best Paper in Automation Award from IEEE ICIA 2017, twice the innovation contest Chunhui Cup Winning Award in 2012 and 2013, and the Wu Wenjun AI Award in 2016. He was the Program Chair of the IEEE RCAR 2016 and International Robotics Conference in Foshan 2017. He was the Conference General Chair of International Conference on Computer Vision Systems 2017 (Springer). He is currently an Associate Editor for IEEE ROBOTICS AND AUTOMATION LETTERS, *International Journal of Robotics and Automation*, *IET Cyber-Systems and Robotics*, IEEE IROS Conference 2018, 2019, and 2020. He served as a Guest Editor of special issues in IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING. He was a Program Member of *Robotics: Science and Systems* (RSS) 2021.