Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 20, No. 2, pp. 213–228, 2024

A PROXIMAL LINEARIZED ALGORITHM FOR DUAL QUATERNION OPTIMIZATION WITH APPLICATIONS IN HAND-EYE CALIBRATION*

Can Li, Yannan Chen† and Dong-Hui Li

Abstract: Many kinetic problems arising from robotics are modeled in optimization of real function in dual quaternion variables. For example, hand-eye calibration fits data to find dual quaternion variables, which integrate the relative position and orientation between a robot gripper and a camera mounted rigidly on the gripper. But now derivatives of real function in dual quaternion variables are unwieldy. To solve optimization of real function in dual quaternion variables, we utilize eight-dimensional vectors to represent dual quaternions. Then the unit dual quaternion constraint is turned into two quadratic equations. Using optimization theory, we derive an algorithm for computing the projection of dual quaternions onto a unit dual quaternion set, which is closed, nonconvex and unbounded. Based on this projection, we propose a proximal linearized algorithm for optimization of real function in dual quaternion variables. The global convergence of the proximal linearized algorithm is analyzed. Finally, numerical experiments on hand-eye calibration problems and dual quaternion-based regressions illustrate the effectiveness of the proposed proximal linearized algorithm.

Key words: *dual quaternion optimization, unit dual quaternion constraint, nonconvex projection, proximal linearized algorithm, global convergence, hand-eye calibration*

Mathematics Subject Classification: *49M37, 65K05, 90C30, 90C90*

1 Introduction

As a widely known and used hypercomplex numbers, quaternions were invented by W. R. Hamilton in 1843. A quaternion $\boldsymbol{q}$ is defined as the pair of a scalar q_0 and a vector $\vec{\boldsymbol{q}} = (q_1, q_2, q_3)^\top$, i.e.,

$$\boldsymbol{q} = (q_0, \vec{\boldsymbol{q}}) = q_0 + \mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3 \quad \text{with} \quad \mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1,$$

where q_0, q_1, q_2, and q_3 are real numbers and $(1, \mathbf{i}, \mathbf{j}, \mathbf{k})$ is a basis of a four-dimensional vector space over the real number field. Quaternions follow the normal rules of algebra except for the commutative law for multiplication (i.e., $\boldsymbol{ab} \neq \boldsymbol{ba}$). Hence, quaternions only form an associative and division algebra. Owing to the greatly efficiency for analyzing 3D rotations [4] and color images [1], quaternions have been applied in robotics, computer vision and

*The work was supported by the National Natural Science Foundation of China (Nos. 12171168, 12071159, 12271187), Guangdong Basis and Applied Basic Research Foundation (Nos. 2021A1515012032, 2021A1515010368, 2022A1515011123) and Yunnan Natural Science Foundation (No. 202101BA070001-047).
†Corresponding author

DOI: https://doi.org/10.61208/pjo-2023-044

graphics, flight dynamics and navigation systems of aircraft, orbital mechanics, quantum mechanics, etc.

Dual numbers and dual quaternions were introduced by W. Clifford in 1873 [3]. A dual number is expressed as

$$\breve{a} = a + \epsilon a' \quad \text{with} \quad \epsilon^2 = 0 \text{ and } \epsilon \neq 0,$$

where a and a' are real numbers. Particularly, a dual number $\breve{a} = a + \epsilon a'$ can be represented by a matrix $\left(\begin{smallmatrix} a & 0 \\ a' & a \end{smallmatrix}\right)$, which implies that ϵ is understood to be $\left(\begin{smallmatrix} 0 & 0 \\ 1 & 0 \end{smallmatrix}\right)$. Dual numbers form a commutative algebra.

When real numbers a and a' are substituted by quaternions $\boldsymbol{a}$ and $\boldsymbol{a'}$, we obtain a dual quaternion $\breve{\boldsymbol{a}} = \boldsymbol{a} + \epsilon \boldsymbol{a'}$ [3, 11, 12]. The dual quaternion $\breve{\boldsymbol{a}}$ can be regarded as an element of an eight-dimensional vector space over the real number field with basis $(1, \mathbf{i}, \mathbf{j}, \mathbf{k}, \epsilon, \epsilon\mathbf{i}, \epsilon\mathbf{j}, \epsilon\mathbf{k})$. Then, dual quaternions form a non-commutative ring with multiplicative identity. Interestingly, dual quaternions could represent rigid transformations including rotations and translations in 3D space as simple dual quaternion products with clear geometric meaning [4].

As an important problem in many areas such as robot kinematics, hand-eye calibration (more appropriately, sensor-actuator calibration) is the computation of the relative translation and rotation between the robot gripper (hand) and a camera (eye) mounted rigidly on the gripper. In fact, any rigid transformation can always be described by two parameters: a translation vector $\vec{\boldsymbol{t}} \in \mathbb{R}^3$ and a rotation matrix $\boldsymbol{R} \in \mathbb{SO}(3)$. By combining translations and rotations, we obtain homogeneous transformation matrices

$$\begin{pmatrix} \boldsymbol{R} & \vec{\boldsymbol{t}} \\ \vec{\boldsymbol{0}}^\top & 1 \end{pmatrix} \in \mathbb{SE}(3),$$

where $\mathbb{SE}(3)$ stands for the Euclidean group of rigid-body motions. $\mathbb{SE}(3)$ has the structure of both a differentiable manifold and an algebraic group, and is an example of a Lie group. The well-known hand-eye equation [14, 16, 10] is formulated as

$$\boldsymbol{AX} = \boldsymbol{XB}, \tag{1.1}$$

where the homogeneous transformation matrix from gripper to camera $\boldsymbol{X} \in \mathbb{SE}(3)$ is undetermined, $\boldsymbol{A} \in \mathbb{SE}(3)$ is the change in the robot gripper position and $\boldsymbol{B} \in \mathbb{SE}(3)$ is the resulting camera displacement.

Numerical methods for solving hand-eye calibration problems include direct methods and iterative methods. Direct methods can find closed-form solutions. For example, Shiu and Ahmad [14] estimated the orientational component by utilizing the angle-axis formulation of rotations and then the translational component by using standard linear system techniques. Zhuang and Roth [21] and Chou and Kamel [2] used quaternion algebra to represent orientations and re-formulated the determination of the rotation matrix as a homogeneous linear least squares problem. Daniilidis [4] introduced the dual quaternion parameterization and proposed a simultaneous solution for the rotation and the translation using singular value decomposition. Wu et al. [18] utilized symbolic methods to derive a globally optimal solution for the hand-eye equation. Although direct methods can estimate rotational and translational components for the hand-eye calibration problems, many of them involve complicated derivations.

Iterative methods are simple and efficient. Horaud and Dornaika [6] used the Levenberg–Marquardt nonlinear minimization method, which solves for rotation and translation simultaneously for hand-eye calibration. They claimed that nonlinear optimization method seems to be the most robust one with respect to noise and measurement errors. A recent research

[9] also confirmed the assertion. Qiu, Wang, and Kermani [13] used the gradient-descent technique on the special Euclidean group $\mathbb{SE}(3)$. Zhang, Zhang, and Yang [20] presented an alternating minimization for hand-eye calibration using dual quaternions. Heller, Havlena, and Pajdla [5] proposed to integrate the hand-eye calibration problem into a branch-and-bound parameter space search. Wang et al. [17] gave singularity analysis for the solution of the hand-eye calibration and solved a singular-free solution in the form of homogeneous transformation matrices. For a more exhaustive review of the hand-eye calibration, the readers are referred to [7].

In this paper, we consider the hand-eye calibration problem by means of dual quaternions:

$$\breve{\boldsymbol{a}}_i \breve{\boldsymbol{q}} = \breve{\boldsymbol{q}} \breve{\boldsymbol{b}}_i, \tag{1.2}$$

for $i = 1, \ldots, m$, where $\breve{\boldsymbol{a}}_i$ and $\breve{\boldsymbol{b}}_i$ are known and $\breve{\boldsymbol{q}}$ is an undetermined unit dual quaternion. First of all, the hand-eye calibration problem (1.2) is rewritten as a least squares optimization of real function in dual quaternion variables, subject to a unit dual quaternion constraint. Unusually, the set of unit dual quaternions is nonconvex, unbounded and closed. Second, to handle the unit dual quaternion constraint, we exploit the relationship between dual quaternions and eight-dimensional vectors and formulate the unit dual quaternion constraint as two quadratic equations in the field of real numbers. Using optimization theory, we can compute the closed-form solution of the projection from any eight-dimensional vector onto the real feasible region constrained by these two quadratic equations. That is to say, we invent an algorithm for computing the projection of dual quaternions onto the unit dual quaternion set. Third, utilizing the unit dual quaternion projection, we design a proximal linearized algorithm for optimization of real function in dual quaternion variables. The global convergence of the proximal linearized algorithm is analyzed. Finally, numerical experiments on hand-eye calibration and regression problems illustrate the effectiveness of the proposed proximal linearized algorithm.

The remainder of this paper is organized as follows. Section 2 describes foundations on dual quaternions and the relationship between dual quaternions and eight-dimensional vectors. The closed-form solution of the projection from a dual quaternion onto the unit dual quaternion set is derived in Section 3. A proximal linearized algorithm for optimization of real function in dual quaternion variables is customized in Section 4, and the global convergence is also analyzed. Numerical experiments on the proposed proximal linearized algorithm for solving hand-eye calibration and regression problems are illustrated in Section 5. Finally, some concluding remarks are given in Section 6.

2 Preliminary on Dual Quaternions

The purpose of this section is to transform unfamiliar dual quaternion operations to frequently-used linear algebra operations in vector space. In the beginning, linear operations on quaternions and dual quaternions are performed in an element-wise manner.

We concentrate on the multiplication of quaternions

$$\boldsymbol{p}\boldsymbol{q} = (p_0 q_0 - \vec{\boldsymbol{p}}^\top \vec{\boldsymbol{q}}, p_0 \vec{\boldsymbol{q}} + q_0 \vec{\boldsymbol{p}} + \vec{\boldsymbol{p}} \times \vec{\boldsymbol{q}}), \tag{2.1}$$

where $\boldsymbol{p} = (p_0, \vec{\boldsymbol{p}})$ and $\boldsymbol{q} = (q_0, \vec{\boldsymbol{q}})$ are two quaternions. Since the cross product satisfies $\vec{\boldsymbol{p}} \times \vec{\boldsymbol{q}} = -\vec{\boldsymbol{q}} \times \vec{\boldsymbol{p}}$, the quaternion multiplication is not commutative. However, the quaternion multiplication is associative and has a unit element $(1, \vec{\boldsymbol{0}})$. By linear algebra, the cross-product $\vec{\boldsymbol{p}} \times \vec{\boldsymbol{q}}$ could be rewritten as

$$\vec{\boldsymbol{p}} \times \vec{\boldsymbol{q}} = \boldsymbol{W}(\vec{\boldsymbol{p}})\vec{\boldsymbol{q}} = -\boldsymbol{W}(\vec{\boldsymbol{q}})\vec{\boldsymbol{p}},$$

where $\boldsymbol{W}$ is a skew-symmetric matrix

$$\boldsymbol{W}(\vec{\boldsymbol{p}}) := \begin{pmatrix} 0 & -p_3 & p_2 \\ p_3 & 0 & -p_1 \\ -p_2 & p_1 & 0 \end{pmatrix}.$$

Since a quaternion $\boldsymbol{q}$ corresponds to a four-dimensional real vector

$$\mathrm{vec}\boldsymbol{q} := \begin{pmatrix} q_0 \\ \vec{\boldsymbol{q}} \end{pmatrix},$$

there is an interesting lemma.

Lemma 2.1 ([20])**.** *The multiplication between quaternions $\boldsymbol{p} = (p_0, \vec{\boldsymbol{p}})$ and $\boldsymbol{q} = (q_0, \vec{\boldsymbol{q}})$ can be represented as*

$$\mathrm{vec}(\boldsymbol{pq}) = \boldsymbol{L}(\boldsymbol{p})\mathrm{vec}\boldsymbol{q} = \boldsymbol{R}(\boldsymbol{q})\mathrm{vec}\boldsymbol{p},$$

where

$$\boldsymbol{L}(\boldsymbol{p}) := \begin{pmatrix} p_0 & -\vec{\boldsymbol{p}}^\top \\ \vec{\boldsymbol{p}} & p_0 I + \boldsymbol{W}(\vec{\boldsymbol{p}}) \end{pmatrix}, \qquad \boldsymbol{R}(\boldsymbol{q}) := \begin{pmatrix} q_0 & -\vec{\boldsymbol{q}}^\top \\ \vec{\boldsymbol{q}} & q_0 I - \boldsymbol{W}(\vec{\boldsymbol{q}}) \end{pmatrix}.$$

Next, we consider the multiplication between dual quaternions $\breve{\boldsymbol{p}} = \boldsymbol{p} + \epsilon\boldsymbol{p}'$ and $\breve{\boldsymbol{q}} = \boldsymbol{q} + \epsilon\boldsymbol{q}'$:

$$\breve{\boldsymbol{p}}\breve{\boldsymbol{q}} = (\boldsymbol{p} + \epsilon\boldsymbol{p}')(\boldsymbol{q} + \epsilon\boldsymbol{q}') = \boldsymbol{pq} + \epsilon(\boldsymbol{pq}' + \boldsymbol{p}'\boldsymbol{q}), \tag{2.2}$$

where $\boldsymbol{p}, \boldsymbol{p}', \boldsymbol{q}$ and $\boldsymbol{q}'$ are quaternions. Then the following theorem holds.

Theorem 2.2. *By regarding a dual quaternion $\breve{\boldsymbol{q}} = \boldsymbol{q} + \epsilon\boldsymbol{q}'$ as an eight-dimensional real vector*

$$\mathrm{vec}\breve{\boldsymbol{q}} := \begin{pmatrix} \mathrm{vec}\boldsymbol{q} \\ \mathrm{vec}\boldsymbol{q}' \end{pmatrix},$$

the multiplication between dual quaternions could be represented as

$$\mathrm{vec}(\breve{\boldsymbol{p}}\breve{\boldsymbol{q}}) = \boldsymbol{L}(\breve{\boldsymbol{p}})\mathrm{vec}\breve{\boldsymbol{q}} = \boldsymbol{R}(\breve{\boldsymbol{q}})\mathrm{vec}\breve{\boldsymbol{p}},$$

where

$$\boldsymbol{L}(\breve{\boldsymbol{p}}) := \begin{pmatrix} \boldsymbol{L}(\boldsymbol{p}) & \boldsymbol{0}_{4\times 4} \\ \boldsymbol{L}(\boldsymbol{p}') & \boldsymbol{L}(\boldsymbol{p}) \end{pmatrix}, \qquad \boldsymbol{R}(\breve{\boldsymbol{q}}) := \begin{pmatrix} \boldsymbol{R}(\boldsymbol{q}) & \boldsymbol{0}_{4\times 4} \\ \boldsymbol{R}(\boldsymbol{q}') & \boldsymbol{R}(\boldsymbol{q}) \end{pmatrix}.$$

Proof. Using Lemma 2.1 and linear algebra, we have

$$\boldsymbol{L}(\breve{\boldsymbol{p}})\mathrm{vec}\breve{\boldsymbol{q}} = \begin{pmatrix} \boldsymbol{L}(\boldsymbol{p})\mathrm{vec}\boldsymbol{q} \\ \boldsymbol{L}(\boldsymbol{p}')\mathrm{vec}\boldsymbol{q} + \boldsymbol{L}(\boldsymbol{p})\mathrm{vec}\boldsymbol{q}' \end{pmatrix} = \begin{pmatrix} \mathrm{vec}(\boldsymbol{pq}) \\ \mathrm{vec}(\boldsymbol{p}'\boldsymbol{q} + \boldsymbol{pq}') \end{pmatrix} = \mathrm{vec}(\breve{\boldsymbol{p}}\breve{\boldsymbol{q}}).$$

The other equality $\boldsymbol{R}(\breve{\boldsymbol{q}})\mathrm{vec}\breve{\boldsymbol{p}} = \mathrm{vec}(\breve{\boldsymbol{p}}\breve{\boldsymbol{q}})$ can be proved by a similar discussion. □

3 Project onto the Unit Dual Quaternion Set

Let $\breve{\boldsymbol{q}} = \boldsymbol{q} + \epsilon\boldsymbol{q}' = q_0 + \mathbf{i}q_1 + \mathbf{j}q_2 + \mathbf{k}q_3 + \epsilon q_0' + \epsilon\mathbf{i}q_1' + \epsilon\mathbf{j}q_2' + \epsilon\mathbf{k}q_3'$ be a dual quaternion. The conjugate $\bar{\breve{\boldsymbol{q}}}$ of $\breve{\boldsymbol{q}}$ is defined by $\bar{\breve{\boldsymbol{q}}} = \bar{\boldsymbol{q}} + \epsilon\bar{\boldsymbol{q}}' = q_0 - \mathbf{i}q_1 - \mathbf{j}q_2 - \mathbf{k}q_3 + \epsilon q_0' - \epsilon\mathbf{i}q_1' - \epsilon\mathbf{j}q_2' - \epsilon\mathbf{k}q_3'$. For a unit dual quaternion $\breve{\boldsymbol{q}} = \boldsymbol{q} + \epsilon\boldsymbol{q}'$, it holds that

$$\breve{\boldsymbol{q}}\bar{\breve{\boldsymbol{q}}} = \boldsymbol{q}\bar{\boldsymbol{q}} + \epsilon(\boldsymbol{q}\bar{\boldsymbol{q}}' + \boldsymbol{q}'\bar{\boldsymbol{q}}) = q_0^2 + q_1^2 + q_2^2 + q_3^2 + \epsilon 2(q_0q_0' + q_1q_1' + q_2q_2' + q_3q_3') = 1.$$

The real part $\boldsymbol{q}\bar{\boldsymbol{q}} = 1$ and the dual part $\boldsymbol{q}\bar{\boldsymbol{q}'} + \boldsymbol{q}'\bar{\boldsymbol{q}} = 0$ mean $\|\text{vec}\boldsymbol{q}\|_2^2 = 1$ and $(\text{vec}\boldsymbol{q})^\top \text{vec}\boldsymbol{q}' = 0$, respectively. For convenience, let $\mathbf{q} := \text{vec}\boldsymbol{q} \in \mathbb{R}^4$ and $\mathbf{q}' := \text{vec}\boldsymbol{q}' \in \mathbb{R}^4$. Then the unit dual quaternion set corresponds to the following set:

$$\Omega := \left\{ (\mathbf{q}^\top, \mathbf{q}'^\top)^\top \mid \|\mathbf{q}\|_2^2 = 1, \mathbf{q}'^\top \mathbf{q} = 0 \right\} \subseteq \mathbb{R}^8. \tag{3.1}$$

Obviously, the set Ω is a closed, nonconvex and unbounded set. That is to say, the following lemma is valid.

Lemma 3.1. *The unit dual quaternion set is nonempty, closed, nonconvex and unbounded.*

Next, we concentrate on the projection of a generic dual quaternion $\breve{\boldsymbol{a}}$ onto the unit dual quaternion set, i.e., find a solution of the following optimization problem

$$\min_{\breve{\boldsymbol{q}}} \ \frac{1}{2}\|\text{vec}\breve{\boldsymbol{q}} - \text{vec}\breve{\boldsymbol{a}}\|_2^2 \qquad \text{s.t. } \text{vec}\breve{\boldsymbol{q}} \in \Omega, \tag{3.2}$$

which is indeed a quadratically constrained quadratic programming in the field of real numbers

$$\begin{aligned} \min_{\mathbf{q},\mathbf{q}'} \ & \frac{1}{2}\|\mathbf{q} - \mathbf{a}\|_2^2 + \frac{1}{2}\|\mathbf{q}' - \mathbf{a}'\|_2^2 \\ \text{s.t.} \ & \|\mathbf{q}\|_2^2 = 1, \quad \mathbf{q}'^\top \mathbf{q} = 0, \end{aligned} \tag{3.3}$$

where $\breve{\boldsymbol{a}} = \boldsymbol{a} + \epsilon\boldsymbol{a}'$, $\mathbf{a} = \text{vec}\boldsymbol{a}$ and $\mathbf{a}' = \text{vec}\boldsymbol{a}'$.

Using real multipliers λ and μ, the Lagrangian function of (3.3) is

$$\mathcal{L}(\mathbf{q}, \mathbf{q}', \lambda, \mu) = \frac{1}{2}\|\mathbf{q} - \mathbf{a}\|_2^2 + \frac{1}{2}\|\mathbf{q}' - \mathbf{a}'\|_2^2 + \frac{\lambda}{2}(\|\mathbf{q}\|_2^2 - 1) + \mu\mathbf{q}'^\top \mathbf{q}.$$

By optimization theory, we solve the following system of polynomial equations

$$\mathbf{q} - \mathbf{a} + \lambda\mathbf{q} + \mu\mathbf{q}' = \mathbf{0}, \tag{3.4}$$

$$\mathbf{q}' - \mathbf{a}' + \mu\mathbf{q} = \mathbf{0}, \tag{3.5}$$

$$\|\mathbf{q}\|_2^2 - 1 = 0, \tag{3.6}$$

$$\mathbf{q}'^\top \mathbf{q} = 0. \tag{3.7}$$

By taking the inner product between $\mathbf{q}$ and (3.4) and using (3.6) and (3.7), we obtain

$$\lambda = \mathbf{q}^\top \mathbf{a} - 1.$$

Similarly, using (3.5), (3.6), and (3.7), we get

$$\mu = \mathbf{q}^\top \mathbf{a}'.$$

Further, it follows from (3.5) that

$$\mathbf{q}' = \mathbf{a}' - \mu\mathbf{q} = \mathbf{a}' - (\mathbf{q}^\top \mathbf{a}')\mathbf{q}.$$

Substituting these variables into (3.4), we have

$$(\mathbf{q}^\top \mathbf{a})\mathbf{q} - (\mathbf{q}^\top \mathbf{a}')^2\mathbf{q} + (\mathbf{q}^\top \mathbf{a}')\mathbf{a}' - \mathbf{a} = \mathbf{0}. \tag{3.8}$$

By taking an inner product between (3.8) and $\mathbf{a}$ and an inner product between (3.8) and $\mathbf{a}'$, we establish

$$\begin{aligned} (\mathbf{q}^\top\mathbf{a})^2 - (\mathbf{q}^\top\mathbf{a}')^2(\mathbf{q}^\top\mathbf{a}) + (\mathbf{q}^\top\mathbf{a}')(\mathbf{a}^\top\mathbf{a}') - \|\mathbf{a}\|_2^2 &= 0, \\ (\mathbf{q}^\top\mathbf{a})(\mathbf{q}^\top\mathbf{a}') - (\mathbf{q}^\top\mathbf{a}')^3 + (\mathbf{q}^\top\mathbf{a}')\|\mathbf{a}'\|_2^2 - \mathbf{a}^\top\mathbf{a}' &= 0. \end{aligned}$$

Recalling $\mathbf{q}^\top\mathbf{a} = \lambda + 1$ and $\mathbf{q}^\top\mathbf{a}' = \mu$, the above system can be rewritten as

$$(\lambda+1)^2 - (\lambda+1)\mu^2 + \mu\mathbf{a}^\top\mathbf{a}' - \|\mathbf{a}\|_2^2 = 0, \tag{3.9}$$

$$(\lambda+1)\mu - \mu^3 + \mu\|\mathbf{a}'\|_2^2 - \mathbf{a}^\top\mathbf{a}' = 0. \tag{3.10}$$

It yields from (3.10) that

$$(\lambda+1)\mu = \mu^3 - \mu\|\mathbf{a}'\|_2^2 + \mathbf{a}^\top\mathbf{a}'.$$

By substituting the above equation into the product of μ^2 and (3.9), we obtain

$$(\mu^3 - \mu\|\mathbf{a}'\|_2^2 + \mathbf{a}^\top\mathbf{a}')^2 - (\mu^3 - \mu\|\mathbf{a}'\|_2^2 + \mathbf{a}^\top\mathbf{a}')\mu^3 + \mu^3\mathbf{a}^\top\mathbf{a}' - \mu^2\|\mathbf{a}\|_2^2 = 0,$$

which is a polynomial equation in a single variable

$$-\|\mathbf{a}'\|_2^2\mu^4 + 2\mathbf{a}^\top\mathbf{a}'\mu^3 + (\|\mathbf{a}'\|_2^4 - \|\mathbf{a}\|_2^2)\mu^2 - 2\mathbf{a}^\top\mathbf{a}'\|\mathbf{a}'\|_2^2\mu + (\mathbf{a}^\top\mathbf{a}')^2 = 0. \tag{3.11}$$

All real roots of the above quartic equation for μ can be found by approaches in [15, 19]. Clearly, there may exist multiple roots. Once a real root μ is known, we find λ by (3.10) if $\mu \neq 0$, otherwise, we find real λ by (3.9). There may have multiple (μ, λ) pairs.

Whereafter, for each (μ, λ) pair, we obtain a linear system by (3.4) and (3.5)

$$\begin{aligned} (\lambda+1)\mathbf{q} + \mu\mathbf{q}' &= \mathbf{a}, \\ \mu\mathbf{q} + \mathbf{q}' &= \mathbf{a}', \end{aligned}$$

which means

$$\mathbf{q} = \frac{1}{\lambda+1-\mu^2}(\mathbf{a} - \mu\mathbf{a}'), \qquad \mathbf{q}' = \frac{1}{\lambda+1-\mu^2}(-\mu\mathbf{a} + (\lambda+1)\mathbf{a}').$$

If there are multiple $(\mathbf{q}, \mathbf{q}')$ pairs, we find the best one which attains the minimal objective of the projection (3.3). In summary, the detailed algorithm for finding a closed-form solution of the unit dual quaternion projection is described in Algorithm 1.

Theorem 3.2. *The dual quaternion* $\breve{\boldsymbol{a}} = \boldsymbol{a} + \epsilon\boldsymbol{a}'$ *satisfies* $\boldsymbol{a} \neq \kappa\boldsymbol{a}'$ *for all* $\kappa \in \mathbb{R}$. *Then, Algorithm 1 can solve the projection problem* (3.2)*.*

Proof. We only need to prove that Algorithm 1 is well-defined.

If $\mathbf{a}' = \mathbf{0}$, we say $\mathbf{a} \neq \mathbf{0}$. Equation (3.11) has a double root $\mu = 0$. Then, equation (3.9) reduces to $(\lambda+1-\mu^2)(\lambda+1) = \|\mathbf{a}\|_2^2 \neq 0$, which means $\lambda + 1 - \mu^2 \neq 0$. Hence, $\mathbf{q}$ and $\mathbf{q}'$ can be found successfully.

Otherwise, we assume $\mathbf{a}' \neq \mathbf{0}$. Because $\mathbf{a} \neq \kappa\mathbf{a}'$ with $\kappa = 0$, we say $\mathbf{a} \neq \mathbf{0}$. Since $-\|\mathbf{a}'\|_2^2 < 0$ and $(\mathbf{a}^\top\mathbf{a}')^2 \geq 0$, we know the polynomial equation (3.11) has real roots. Next, equations (3.9) and (3.10) could be rewritten as

$$\begin{aligned} (\lambda+1-\mu^2)(\lambda+1) &= (\mathbf{a} - \mu\mathbf{a}')^\top\mathbf{a}, \\ (\lambda+1-\mu^2)\mu &= (\mathbf{a} - \mu\mathbf{a}')^\top\mathbf{a}', \end{aligned}$$

respectively. Let θ be the angle between nonzero vectors $\mathbf{a}$ and $\mathbf{a}'$. It holds that $\theta \in (0, \pi)$ since $\boldsymbol{a} \neq \kappa\boldsymbol{a}'$ for all κ. Hence, at least one of scalars $(\mathbf{a} - \mu\mathbf{a}')^\top\mathbf{a}$ and $(\mathbf{a} - \mu\mathbf{a}')^\top\mathbf{a}'$ is nonzero. Therefore, $\lambda + 1 - \mu^2 \neq 0$ and $\mathbf{q}$ and $\mathbf{q}'$ can be found successfully. □

Algorithm 1 Projection onto the unit dual quaternion set.

Abstract $\mathbf{a} = \mathrm{vec}\boldsymbol{a}$ and $\mathbf{a}' = \mathrm{vec}\boldsymbol{a'}$ from the inputted (nonzero) dual quaternion $\breve{\boldsymbol{a}} = \boldsymbol{a} + \epsilon\boldsymbol{a'}$.

Solve a polynomial equation (3.11)

$$-\|\mathbf{a}'\|_2^2\mu^4 + 2\mathbf{a}^\top\mathbf{a}'\mu^3 + (\|\mathbf{a}'\|_2^4 - \|\mathbf{a}\|_2^2)\mu^2 - 2\mathbf{a}^\top\mathbf{a}'\|\mathbf{a}'\|_2^2\mu + (\mathbf{a}^\top\mathbf{a}')^2 = 0$$

to find all real roots $\mu_1, \mu_2, \dots, \mu_n$. If $\mu_t = 0$, we regard it as a double root.

for $t = 1, 2, \dots, n$ **do**

  Find λ_t according to μ_t by solving (3.10) or (3.9).

  Compute $\mathbf{q}_t$ and $\mathbf{q'}_t$ via

$$\mathbf{q}_t = \frac{1}{\lambda_t + 1 - \mu_t^2}(\mathbf{a} - \mu_t\mathbf{a}'), \qquad \mathbf{q'}_t = \frac{1}{\lambda_t + 1 - \mu_t^2}(-\mu_t\mathbf{a} + (\lambda_t + 1)\mathbf{a}').$$

  Calculate objective

$$f_t = \frac{1}{2}\|\mathbf{q}_t - \mathbf{a}\|_2^2 + \frac{1}{2}\|\mathbf{q'}_t - \mathbf{a}'\|_2^2.$$

end for

Find $t_* = \arg\min_{t=1,\dots,n} f_t$. Set quaternions $\boldsymbol{q}$ and $\boldsymbol{q'}$ such that $\mathrm{vec}\boldsymbol{q} = \mathbf{q}_{t_*}$ and $\mathrm{vec}\boldsymbol{q'} = \mathbf{q'}_{t_*}$.

Return a unit dual quaternion $\breve{\boldsymbol{q}} = \boldsymbol{q} + \epsilon\boldsymbol{q'}$.

Corollary 3.3. *Suppose that the dual quaternion* $\breve{\boldsymbol{a}} = \boldsymbol{a} + \epsilon\boldsymbol{a'}$ *satisfies* $\boldsymbol{a} \neq 0$ *and* $\boldsymbol{a}\bar{\boldsymbol{a'}} + \boldsymbol{a'}\bar{\boldsymbol{a}} = 0$ *(i.e.,* $\mathbf{a}^\top\mathbf{a}' = 0$*). Then the projection of* $\breve{\boldsymbol{a}}$ *onto the unit dual quaternion set is* $\breve{\boldsymbol{q}} = \boldsymbol{a}/\sqrt{\boldsymbol{a}\bar{\boldsymbol{a}}} + \epsilon\boldsymbol{a'}$.

At the end of this section, we give an example.

Example 3.4. Let $\breve{\boldsymbol{a}} = \boldsymbol{a} + \epsilon\boldsymbol{a'} = (1, 0, 0, 0) + \epsilon(10, 10, 10, 10)$. Algorithm 1 produces

$$\breve{\boldsymbol{q}}_{true} = (0.8666, -0.2881, -0.2881, -0.2881) + \epsilon(9.9784, 10.0072, 10.0072, 10.0072).$$

For a naive projection method, which project $\boldsymbol{a}$ onto the unit quaternion set and then project $\boldsymbol{a'}$ onto the hyperplane $\boldsymbol{a}\bar{\boldsymbol{a'}} + \boldsymbol{a'}\bar{\boldsymbol{a}} = 0$, the resulting unit dual quaternion is

$$\breve{\boldsymbol{q}}_{naive} = (1, 0, 0, 0) + \epsilon(0, 10, 10, 10).$$

Obviously, we have

$$\|\breve{\boldsymbol{q}}_{true} - \breve{\boldsymbol{a}}\| = 0.5170 < \|\breve{\boldsymbol{q}}_{naive} - \breve{\boldsymbol{a}}\| = 10.$$

4 A proximal Linearized Algorithm

The hand-eye calibration problem (1.2) is to find a unit dual quaternion $\breve{\boldsymbol{q}}$ such that

$$\breve{\boldsymbol{a}}_i\breve{\boldsymbol{q}} = \breve{\boldsymbol{q}}\breve{\boldsymbol{b}}_i, \qquad \forall\ i = 1, \dots, m. \tag{4.1}$$

We consider the least squares fitting problem for hand-eye calibration

$$\min\ \tilde{f}(\breve{\boldsymbol{q}}) := \frac{1}{2}\sum_{i=1}^{m}\|\mathrm{vec}(\breve{\boldsymbol{a}}_i\breve{\boldsymbol{q}} - \breve{\boldsymbol{q}}\breve{\boldsymbol{b}}_i)\|_2^2 \qquad \text{s.t. } \breve{\boldsymbol{q}} \text{ is a unit dual quaternion.} \tag{4.2}$$

By Theorem 2.2, we have

$$\begin{aligned}\sum_{i=1}^{m}\|\mathrm{vec}(\breve{\boldsymbol{a}}_i\breve{\boldsymbol{q}}-\breve{\boldsymbol{q}}\breve{\boldsymbol{b}}_i)\|_2^2 &= \sum_{i=1}^{m}\|\boldsymbol{L}(\breve{\boldsymbol{a}}_i)\mathrm{vec}\breve{\boldsymbol{q}}-\boldsymbol{R}(\breve{\boldsymbol{b}}_i)\mathrm{vec}\breve{\boldsymbol{q}}\|_2^2 \\ &= (\mathrm{vec}\breve{\boldsymbol{q}})^\top\left(\sum_{i=1}^{m}[\boldsymbol{L}(\breve{\boldsymbol{a}}_i)-\boldsymbol{R}(\breve{\boldsymbol{b}}_i)]^\top[\boldsymbol{L}(\breve{\boldsymbol{a}}_i)-\boldsymbol{R}(\breve{\boldsymbol{b}}_i)]\right)\mathrm{vec}\breve{\boldsymbol{q}}.\end{aligned}$$

Let

$$\boldsymbol{H}:=\sum_{i=1}^{m}[\boldsymbol{L}(\breve{\boldsymbol{a}}_i)-\boldsymbol{R}(\breve{\boldsymbol{b}}_i)]^\top[\boldsymbol{L}(\breve{\boldsymbol{a}}_i)-\boldsymbol{R}(\breve{\boldsymbol{b}}_i)].$$

Then, the following theorem on the positive semidefinite matrix $\boldsymbol{H}$ is interesting.

Theorem 4.1. *The matrix $\boldsymbol{H}$ has a double eigenvalue 0 with associated eigenvectors $\mathrm{vec}\breve{\boldsymbol{q}}$ and $(\boldsymbol{0}_{1\times 4},\mathrm{vec}\boldsymbol{q}^\top)^\top$, if the hand-eye calibration problem (4.1) has a solution $\breve{\boldsymbol{q}}=\boldsymbol{q}+\epsilon\boldsymbol{q}'$.*

Proof. Since $\breve{\boldsymbol{q}}=\boldsymbol{q}+\epsilon\boldsymbol{q}'$ is a solution of (4.1), we have $\breve{\boldsymbol{a}}_i\breve{\boldsymbol{q}}-\breve{\boldsymbol{q}}\breve{\boldsymbol{b}}_i=\breve{\boldsymbol{0}}$ and hence $[\boldsymbol{L}(\breve{\boldsymbol{a}}_i)-\boldsymbol{R}(\breve{\boldsymbol{b}}_i)]\mathrm{vec}\breve{\boldsymbol{q}}=\boldsymbol{0}$ for all $i=1,\ldots,m$. Hence, we say $\boldsymbol{H}\mathrm{vec}\breve{\boldsymbol{q}}=0\mathrm{vec}\breve{\boldsymbol{q}}$.

Next, by Theorem 2.2, it follows from $[\boldsymbol{L}(\breve{\boldsymbol{a}}_i)-\boldsymbol{R}(\breve{\boldsymbol{b}}_i)]\mathrm{vec}\breve{\boldsymbol{q}}=\boldsymbol{0}$ that

$$\begin{pmatrix}\boldsymbol{L}(\boldsymbol{a}_i)-\boldsymbol{R}(\boldsymbol{b}_i) & \boldsymbol{0}_{4\times 4}\\ \boldsymbol{L}(\boldsymbol{a}_i')-\boldsymbol{R}(\boldsymbol{b}_i') & \boldsymbol{L}(\boldsymbol{a}_i)-\boldsymbol{R}(\boldsymbol{b}_i)\end{pmatrix}\begin{pmatrix}\mathrm{vec}\boldsymbol{q}\\ \mathrm{vec}\boldsymbol{q}'\end{pmatrix}=\boldsymbol{0}$$

and hence

$$[\boldsymbol{L}(\boldsymbol{a}_i)-\boldsymbol{R}(\boldsymbol{b}_i)]\mathrm{vec}\boldsymbol{q}=\boldsymbol{0},\qquad \forall\ i=1,\ldots,m.$$

Then, it is straightforward to calculate that

$$[\boldsymbol{L}(\breve{\boldsymbol{a}}_i)-\boldsymbol{R}(\breve{\boldsymbol{b}}_i)]\begin{pmatrix}\boldsymbol{0}_{4\times 1}\\ \mathrm{vec}\boldsymbol{q}\end{pmatrix}=\begin{pmatrix}\boldsymbol{L}(\boldsymbol{a}_i)-\boldsymbol{R}(\boldsymbol{b}_i) & \boldsymbol{0}_{4\times 4}\\ \boldsymbol{L}(\boldsymbol{a}_i')-\boldsymbol{R}(\boldsymbol{b}_i') & \boldsymbol{L}(\boldsymbol{a}_i)-\boldsymbol{R}(\boldsymbol{b}_i)\end{pmatrix}\begin{pmatrix}\boldsymbol{0}_{4\times 1}\\ \mathrm{vec}\boldsymbol{q}\end{pmatrix}=\boldsymbol{0},$$

for all $i=1,\ldots,m$. Therefore, $(\boldsymbol{0}_{1\times 4},\mathrm{vec}\boldsymbol{q}^\top)^\top$ is also an eigenvector of $\boldsymbol{H}$ corresponding to the zero eigenvalue. □

Let $\mathbf{x}:=\mathrm{vec}\breve{\boldsymbol{q}}$. The optimization model (4.1) can be represented as

$$\min\ f(\mathbf{x})=\frac{1}{2}\mathbf{x}^\top\boldsymbol{H}\mathbf{x}+i_\Omega(\mathbf{x}),$$

where Ω is defined in (3.1) and i_Ω is an indicator function such that $i_\Omega(\mathbf{x})=0$ if $\mathbf{x}\in\Omega$ and $i_\Omega(\mathbf{x})=+\infty$ otherwise.

Starting from an initial point $\mathbf{x}_1$ and setting $t\leftarrow 1$, we repeatedly solve the proximal linearized subproblem

$$\mathbf{x}_{t+1}=\arg\min\frac{1}{2}\mathbf{x}_t^\top\boldsymbol{H}\mathbf{x}_t+(\mathbf{x}-\mathbf{x}_t)^\top\boldsymbol{H}\mathbf{x}_t+\frac{1}{2\kappa_t}\|\mathbf{x}-\mathbf{x}_t\|_2^2+i_\Omega(\mathbf{x}),\tag{4.3}$$

which is equivalent to a projection problem

$$\begin{aligned}\mathbf{x}_{t+1} \;=\; &\arg\min\ \|\mathbf{x}-\mathbf{x}_t+\kappa_t\boldsymbol{H}\mathbf{x}_t\|_2^2\\ &\text{s.t.}\quad \mathbf{x}\in\Omega.\end{aligned}\tag{4.4}$$

We note that the subproblem (4.4) can be solved by Algorithm 1. Then, we set $t\leftarrow t+1$. If the difference between two iterative points $\|\mathbf{x}_{t+1}-\mathbf{x}_t\|$ is small enough, we terminate the algorithm. The detailed algorithm is represented in Algorithm 2.

Now, we show that the smallest step size in line 3 of Algorithm 2 is reasonable.

Algorithm 2 Proximal linearized algorithm.

1: Choose $\mathbf{x}_1$. Set parameters $\eta > 0$, $\kappa_0 = 1$, and $t \leftarrow 1$.
2: **while** not convergence **do**
3: Find the largest $\kappa_t \in \{\kappa_{t-1}, \kappa_{t-1}/2, \kappa_{t-1}/4, \ldots, 2^{-\lceil \log_2(\|\boldsymbol{H}\|_2+\eta)\rceil}\}$ such that

$$f(\mathbf{x}_{t+1}) \le f(\mathbf{x}_t) - \frac{\eta}{2}\|\mathbf{x}_{t+1} - \mathbf{x}_t\|_2^2, \tag{4.5}$$

where $\mathbf{x}_{t+1}$ is obtained by solving (4.4).
4: $t \leftarrow t+1$.
5: **end while**

Lemma 4.2. *The step size κ_t used in line 3 of Algorithm 2 is bounded away from zero, i.e.,*

$$\kappa_t \ge 2^{-\lceil \log_2(\|\boldsymbol{H}\|_2+\eta)\rceil}, \qquad \textit{for all } t = 1, 2, \ldots.$$

Proof. Because $\mathbf{x}_t \in \Omega$, we have

$$\begin{aligned} f(\mathbf{x}_{t+1}) &= \frac{1}{2}\mathbf{x}_t^\top \boldsymbol{H}\mathbf{x}_t + (\mathbf{x}_{t+1} - \mathbf{x}_t)^\top \boldsymbol{H}\mathbf{x}_t + \frac{1}{2}(\mathbf{x}_{t+1} - \mathbf{x}_t)^\top \boldsymbol{H}(\mathbf{x}_{t+1} - \mathbf{x}_t) \\ &= \frac{1}{2}\mathbf{x}_t^\top \boldsymbol{H}\mathbf{x}_t + (\mathbf{x}_{t+1} - \mathbf{x}_t)^\top \boldsymbol{H}\mathbf{x}_t + \frac{1}{2\kappa_t}\|\mathbf{x}_{t+1} - \mathbf{x}_t\|_2^2 \\ &\quad - \frac{1}{2}(\mathbf{x}_{t+1} - \mathbf{x}_t)^\top(\kappa_t^{-1} I - \boldsymbol{H})(\mathbf{x}_{t+1} - \mathbf{x}_t) \\ &\le \frac{1}{2}\mathbf{x}_t^\top \boldsymbol{H}\mathbf{x}_t - \frac{1}{2}(\mathbf{x}_{t+1} - \mathbf{x}_t)^\top(\kappa_t^{-1} I - \boldsymbol{H})(\mathbf{x}_{t+1} - \mathbf{x}_t) \\ &\le f(\mathbf{x}_t) - \frac{1}{2}(\kappa_t^{-1} - \|\boldsymbol{H}\|_2)\|\mathbf{x}_{t+1} - \mathbf{x}_t\|_2^2. \end{aligned}$$

When $\kappa_t^{-1} - \|\boldsymbol{H}\|_2 \ge \eta$, the inequality (4.5) holds. Hence, we say that the step size is bounded away from zero. □

On the convergence of Algorithm 2, we establish the following theorem.

Theorem 4.3. *For all bounded subsequences $\{\mathbf{x}_{t'}\}$ of $\{\mathbf{x}_t\}$, we have*

$$\lim_{t'\to\infty} \operatorname{dist}(0, \boldsymbol{H}\mathbf{x}_{t'} + \partial i_\Omega(\mathbf{x}_{t'})) = 0.$$

Proof. Let $T \ge 1$ be an iterative number. According to (4.5) and the nonnegativity, we know

$$f(\mathbf{x}_1) \ge f(\mathbf{x}_1) - f(\mathbf{x}_{T+1}) = \sum_{t=1}^{T} f(\mathbf{x}_t) - f(\mathbf{x}_{t+1}) \ge \frac{\eta}{2}\sum_{t=1}^{T}\|\mathbf{x}_{t+1} - \mathbf{x}_t\|_2^2.$$

Let $T \to \infty$. We get $\sum_{t=1}^{\infty}\|\mathbf{x}_{t+1} - \mathbf{x}_t\|_2^2 < \frac{2}{\eta}f(\mathbf{x}_1)$ and hence

$$\lim_{t\to\infty}\|\mathbf{x}_{t+1} - \mathbf{x}_t\|_2 = 0. \tag{4.6}$$

Because $\mathbf{x}_{t+1}$ is a solution of (4.4) and (4.3), we have

$$\begin{aligned} 0 &\in \boldsymbol{H}\mathbf{x}_t + \frac{1}{\kappa_t}(\mathbf{x}_{t+1} - \mathbf{x}_t) + \partial i_\Omega(\mathbf{x}_{t+1}) \\ &= \boldsymbol{H}\mathbf{x}_{t+1} + \partial i_\Omega(\mathbf{x}_{t+1}) + (\kappa_t^{-1} I - \boldsymbol{H})(\mathbf{x}_{t+1} - \mathbf{x}_t). \end{aligned}$$

For a bounded subsequence $\{\mathbf{x}_{t'}\}$ of $\{\mathbf{x}_t\}$, we get

$$\begin{aligned} \operatorname{dist}(0, \boldsymbol{H}\mathbf{x}_{t'} + \partial i_\Omega(\mathbf{x}_{t'})) &\leq \|(\kappa_{t'-1}^{-1} I - \boldsymbol{H})(\mathbf{x}_{t'} - \mathbf{x}_{t'-1})\|_2 \\ &\leq (2^{\lceil \log_2(\|\boldsymbol{H}\|_2 + \eta)\rceil} + \|\boldsymbol{H}\|_2)\|\mathbf{x}_{t'} - \mathbf{x}_{t'-1}\|_2, \end{aligned}$$

where the last inequality holds owing to Lemma 4.2. It follows from (4.6) straightforwardly that $\operatorname{dist}(0, \boldsymbol{H}\mathbf{x}_{t'} + \partial i_\Omega(\mathbf{x}_{t'})) \to 0$. □

5 Numerical Experiments

We examine the performance of the proposed proximal linearized algorithm for solving the hand-eye calibration and a special regression problem in unit dual quaternion variables.

5.1 Hand-eye calibration

An example of hand-eye calibration is equipped with parameters [14]:

$$\begin{aligned} \boldsymbol{X}_{act} &= \text{Trans}(0.01 \text{ m}, 0.05 \text{ m}, 0.1 \text{ m}) \cdot \text{Rot}([1,0,0]^\top, 0.2 \text{ rad}), \\ \boldsymbol{A}_1 &= \text{Trans}(0 \text{ m}, 0 \text{ m}, 0 \text{ m}) \cdot \text{Rot}([0,0,1]^\top, 3.0 \text{ rad}), \\ \boldsymbol{A}_2 &= \text{Trans}(-0.4 \text{ m}, 0 \text{ m}, 0.4 \text{ m}) \cdot \text{Rot}([0,1,0]^\top, 1.5 \text{ rad}), \end{aligned}$$

and

$$\boldsymbol{B}_1 = \boldsymbol{X}_{act}^{-1}\boldsymbol{A}_1\boldsymbol{X}_{act}, \qquad \boldsymbol{B}_2 = \boldsymbol{X}_{act}^{-1}\boldsymbol{A}_2\boldsymbol{X}_{act}.$$

That is to say, we have

$$\begin{aligned} \breve{\boldsymbol{q}}_{act} &= 0.9950 + 0.0998\mathbf{i} + 0.0000\mathbf{j} + 0.0000\mathbf{k} - 0.0005\epsilon + 0.0050\epsilon\mathbf{i} + 0.0299\epsilon\mathbf{j} + 0.0473\epsilon\mathbf{k}, \\ \breve{\boldsymbol{a}}_1 &= 0.0707 + 0.0000\mathbf{i} + 0.0000\mathbf{j} + 0.9975\mathbf{k} + 0.0000\epsilon + 0.0000\epsilon\mathbf{i} + 0.0000\epsilon\mathbf{j} + 0.0000\epsilon\mathbf{k}, \\ \breve{\boldsymbol{a}}_2 &= 0.7317 + 0.0000\mathbf{i} + 0.6816\mathbf{j} + 0.0000\mathbf{k} + 0.0000\epsilon - 0.2827\epsilon\mathbf{i} + 0.0000\epsilon\mathbf{j} + 0.0100\epsilon\mathbf{k}, \\ \breve{\boldsymbol{b}}_1 &= 0.0707 + 0.0000\mathbf{i} + 0.1982\mathbf{j} + 0.9776\mathbf{k} + 0.0000\epsilon - 0.0499\epsilon\mathbf{i} + 0.0098\epsilon\mathbf{j} - 0.0020\epsilon\mathbf{k}, \\ \breve{\boldsymbol{b}}_2 &= 0.7317 + 0.0000\mathbf{i} + 0.6681\mathbf{j} - 0.1354\mathbf{k} - 0.0000\epsilon - 0.2145\epsilon\mathbf{i} + 0.0006\epsilon\mathbf{j} + 0.0031\epsilon\mathbf{k}. \end{aligned}$$

Starting from a random dual quaternion, the proposed proximal linearized algorithm generates an iterative sequence $\{\breve{\boldsymbol{q}}_t\}$ (satisfying $\mathbf{x}_t = \text{vec}\breve{\boldsymbol{q}}_t$) that always converges fastly to the solution of the hand-eye calibration problem. A typical iterative process is illustrated in Figure 1, where the objective value is $f(\breve{\boldsymbol{q}}_t) = \frac{1}{2}\sum_{i=1}^{2}\|\text{vec}(\breve{\boldsymbol{a}}_i\breve{\boldsymbol{q}}_t - \breve{\boldsymbol{q}}_t\breve{\boldsymbol{b}}_i)\|_2^2$ and the error of dual quaternion is $\|\text{vec}(\breve{\boldsymbol{q}}_t - \breve{\boldsymbol{q}}_{act})\|_2$.

To simulate the rotation error for each homogeneous transformation matrix $\boldsymbol{A}$, a random 1×3 vector, whose entries indicate the rotation Euler angles error, is transformed into rotational error matrix $\delta\boldsymbol{R}$, then acts upon the rotational part of $\boldsymbol{A}$ to generate a disturbed rotation matrix $\boldsymbol{R}_A$ as follows

$$\boldsymbol{R}_A \leftarrow \delta\boldsymbol{R} \cdot \boldsymbol{R}_A.$$

A translation error vector generated from a zero mean normal distribution with small standard deviation is added to translation part of $\boldsymbol{A}$ to simulate the translation error. The rotation error and translation error of $\boldsymbol{B}$ are set by the same way.

We compare the performance of our method with three methods. (I) "Shiu89" is a direct method based on homogeneous transformation matrices [14]. (II) "Daniilidis99" is a direct method based on unit dual quaternions [4]. (III) "Two-step17" is a two-step iteration in dual quaternions [20]. The first two methods have closed-form solutions via solving linear

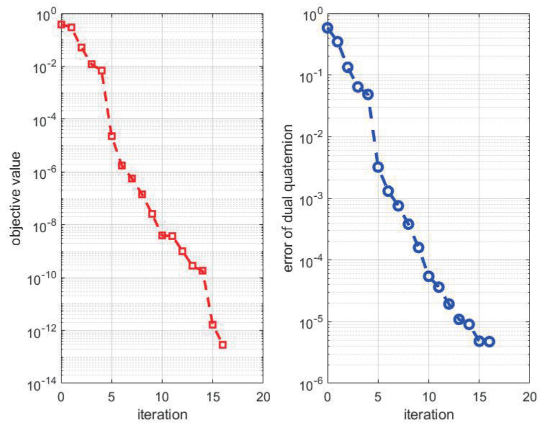

Figure 1: Performance of the proposed proximal linearized algorithm for hand-eye calibration.

Table 1: Average results E_{rotm} and E_{trvec} for the cases of rotation error.

	rotation(rad)	0.0350	0.0700	0.1050	0.1400	0.1750	0.2100	0.2450	0.2800	0.3150	0.3500
E_{rotm}	Shiu89	0.0081	0.0143	0.0248	0.1127	0.3943	0.5894	0.6190	0.7044	0.7303	0.6536
	Daniilidis99	0.0064	0.0115	0.0207	0.1076	0.3897	0.5857	0.6109	0.6995	0.7280	0.6471
	Two-step17	0.0065	0.0117	0.0212	0.1080	0.3901	0.5864	0.6124	0.7000	0.7290	0.6492
	Algorithm 2	0.0064	0.0115	0.0207	0.1076	0.3896	0.5857	0.6109	0.6993	0.7280	0.6470
E_{trvec}	Shiu89	0.0014	0.0028	0.0045	0.0408	0.1602	0.2424	0.2511	0.2827	0.2891	0.2532
	Daniilidis99	0.0014	0.0030	0.0047	0.0418	0.1616	0.2446	0.2537	0.2871	0.2944	0.2590
	Two-step17	0.0032	0.0113	0.0217	0.0746	0.1944	0.2999	0.3462	0.3431	0.3687	0.3680
	Algorithm 2	0.0015	0.0030	0.0049	0.0419	0.1603	0.2418	0.2501	0.2822	0.2882	0.2528

equations, while Two-step17 and our method are iterative methods. For iterative methods, we stop the iteration if the number of iterations exceeds 300 or the Euclidean distance between the t-th and the $(t+1)$-th iterates is less than 10^{-3}. To qualify the results, we take E_{rotq} as the errors in the rotation unit quaternion and E_{trvec} as the errors in the translation vector

$$E_{rotq} = \|\text{vec}(\boldsymbol{q} - \boldsymbol{q}^*)\|_2, \qquad E_{trvec} = \|\vec{\boldsymbol{t}} - \vec{\boldsymbol{t}}^*\|_2,$$

where $\boldsymbol{q}$ is the estimated rotation quaternion, $\vec{\boldsymbol{t}}$ is the estimated translation vector and their ground truth $\boldsymbol{q}^*$ and $\vec{\boldsymbol{t}}^*$.

For any $\boldsymbol{A}_i$ and associated $\boldsymbol{B}_i$, we consider rotation error and translation error separately. The rotation angle error increases from 0.0350 to 0.3500 rad in steps of 0.0350. The translation error increases from 0.002 to 0.02 m in steps of 0.002. For each step, we randomly generate 50 sets of Euler angles with eligible values and run the four methods to obtain the average errors E_{rotq} and E_{trvec} for each method. For each set of Euler angles, we set 100 initial points and run the two iterative methods to obtain the average errors respectively. The results of all four methods for the cases of rotation error are listed in Table 1 and the cases of translation error are listed in Table 2.

With the error increasing, whether rotation or translation, the deviation of the estimation from ground truth becomes more serious for each method. For various rotation errors, Table 1 shows that the performance of Algorithm 2 and Daniilidis99 are comparable and more

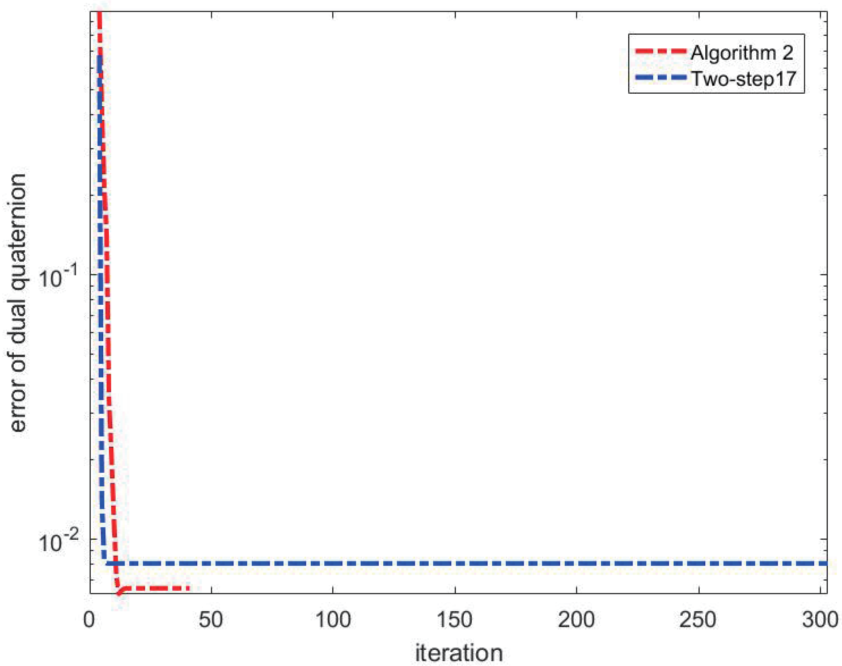

Figure 2: Convergence performance of Two-step17 and Algorithm 2.

accurate than the other two algorithms in rotational part of the estimation. In translational part, Algorithm 2 sometimes performs a little worse than Shiu89 and Daniilidis99 when the rotation error is less than 0.175. But as the rotation error increases again, Algorithm 2 is better than the other three methods. For the cases of translation error, we can see from Table 2 that the difference of the four algorithms is very small, either rotational part or translational part of the estimation. In particular, there is no deviation in rotational part of Shiu89. The reason is that Shiu89 estimates the orientational component and translational component separately, while the other methods estimate simultaneously.

For iterative methods, the sequence generated by Two-step17 may not converge within 300 iterations when the stop tolerance is set to less than 10^{-3}, while Algorithm 2 is always convergent even the accuracy reaches 10^{-12}. Typical convergence performances of Two-step17 and Algorithm 2 are given in Figure 2, in which the rotation Euler angles error are set to empirical value 0.0350 rad and the translation error with standard deviation is 0.002 m. As illustrated in Figure 2, Two-step17 converges quickly in the first three iterations, but little progress is made in later iterations. Although Algorithm 2 is not as fast as Two-step17 in the early stage, the proposed algorithm can always terminate in the less number of iterations and achieve more accurate solution when solving the hand-eye calibration problem.

5.2 A regression problem

Generalized linear regression is an important mathematical model in many areas. In this subsection, we study the log-linear regression problem in dual quaternion variables. In the beginning, for two unit dual quaternions $\breve{\boldsymbol{p}}$ and $\breve{\boldsymbol{q}}$, the screw linear interpolation (ScLERP)

Table 2: Average results E_{rotm} and E_{trvec} for the cases of translation error.

	translation(m)	0.0020	0.0040	0.0060	0.0080	0.0100	0.0120	0.0140	0.0160	0.0180	0.0200
E_{rotm}	Shiu89	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	Daniilidis99	0.0002	0.0005	0.0007	0.0009	0.0014	0.0014	0.0021	0.0017	0.0022	0.0024
	Two-step17	0.0002	0.0005	0.0007	0.0009	0.0014	0.0014	0.0021	0.0016	0.0022	0.0025
	Algorithm 2	0.0003	0.0005	0.0007	0.0009	0.0014	0.0015	0.0021	0.0017	0.0023	0.0025
E_{trvec}	Shiu89	0.0024	0.0043	0.0078	0.0098	0.0107	0.0169	0.0164	0.0192	0.0200	0.0222
	Daniilidis99	0.0024	0.0043	0.0078	0.0098	0.0107	0.0169	0.0164	0.0193	0.0200	0.0222
	Two-step17	0.0024	0.0044	0.0079	0.0098	0.0113	0.0180	0.0176	0.0207	0.0215	0.0245
	Algorithm 2	0.0025	0.0043	0.0077	0.0098	0.0107	0.0169	0.0164	0.0192	0.0200	0.0222

Table 3: Observed noise-corrupted unit dual quaternions.

basis	1	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$	ϵ	$\epsilon\mathbf{i}$	$\epsilon\mathbf{j}$	$\epsilon\mathbf{k}$
$\breve{\boldsymbol{p}}_1$	0.8274	0.3707	0.3893	0.1627	−0.0585	−0.2179	0.1671	0.3940
$\breve{\boldsymbol{p}}_2$	0.3405	0.6244	0.6408	0.2892	−0.2753	−0.3831	0.2236	0.6559
$\breve{\boldsymbol{p}}_3$	−0.3476	0.6223	0.6338	0.3003	−0.4514	−0.6093	0.1103	0.5075
$\breve{\boldsymbol{p}}_4$	−0.8104	0.4170	0.3573	0.2043	−0.4139	−0.6194	−0.2753	0.1044
$\breve{\boldsymbol{p}}_5$	−0.9990	−0.0250	0.0349	−0.0072	−0.0027	−0.5855	−0.5568	−0.2888
$\breve{\boldsymbol{p}}_6$	−0.7894	−0.4254	−0.3868	−0.2150	0.6236	−0.2714	−0.6469	−0.5889
$\breve{\boldsymbol{p}}_7$	−0.2756	−0.6570	−0.6212	−0.3264	1.1127	0.2458	−0.4089	−0.6559
$\breve{\boldsymbol{p}}_8$	0.3250	−0.6006	−0.6567	−0.3200	1.2527	0.7691	0.1438	−0.4661
$\breve{\boldsymbol{p}}_9$	0.7959	−0.3817	−0.4083	−0.2327	0.8765	1.0120	0.7285	0.0598
$\breve{\boldsymbol{p}}_{10}$	0.9990	0.0292	−0.0248	−0.0207	0.0072	1.0803	1.1718	0.4679
$\breve{\boldsymbol{p}}_{11}$	0.8199	0.4053	0.3704	0.1622	−1.0114	0.6900	1.1521	0.7576
$\breve{\boldsymbol{p}}_{12}$	0.3193	0.6051	0.6625	0.3050	−1.8802	−0.0924	0.5772	0.8977
$\breve{\boldsymbol{p}}_{13}$	−0.2892	0.6098	0.6557	0.3383	−2.0415	−0.8750	−0.3203	0.4529
$\breve{\boldsymbol{p}}_{14}$	−0.7828	0.4147	0.4126	0.2120	−1.4183	−1.4608	−1.1253	−0.1892
$\breve{\boldsymbol{p}}_{15}$	−0.9990	0.0051	−0.0346	−0.0260	0.0713	−1.6860	−1.6189	−0.9190
$\breve{\boldsymbol{p}}_{16}$	−0.7929	−0.4191	−0.3814	−0.2241	1.6541	−1.1816	−1.5484	−1.0078

[8] gives

$$\mathrm{ScLERP}(\ell) = \breve{\boldsymbol{p}}(\bar{\breve{\boldsymbol{p}}}\breve{\boldsymbol{q}})^{\ell}, \qquad \ell \in [0,1].$$

In fact, ScLERP is both the shortest path and a constant speed interpolation.

Now, we consider the situation that unit dual quaternions are corrupted by noise, i.e.,

$$\breve{\boldsymbol{p}}_k = \breve{\boldsymbol{p}}_0(\breve{\boldsymbol{q}})^k + \breve{\boldsymbol{\varepsilon}}_k, \qquad k = 0,1,\ldots,m,$$

where $\breve{\boldsymbol{p}}_1,\ldots,\breve{\boldsymbol{p}}_m$ are observed and noise-corrupted unit dual quaternions corresponding to parameters $k = 1,\ldots,m$ respectively, unit dual quaternion $\breve{\boldsymbol{p}}_0$ is a known initial point, unit dual quaternion $\breve{\boldsymbol{q}}$ is unknown, and $\breve{\boldsymbol{\varepsilon}}_k$'s are noises. To find the unknown rigid transformation $\breve{\boldsymbol{q}}$, a regression model is built as

$$\min\ f_R(\breve{\boldsymbol{q}}) := \frac{1}{2}\sum_{k=1}^{m} \|\mathrm{vec}(\breve{\boldsymbol{p}}_k - \breve{\boldsymbol{p}}_0(\breve{\boldsymbol{q}})^k)\|_2^2 \qquad \text{s.t. } \breve{\boldsymbol{q}} \text{ is a unit dual quaternion.} \tag{5.1}$$

Define

$$g_R(\breve{\boldsymbol{q}}) := -\sum_{k=1}^{m} k\left[\boldsymbol{L}(\breve{\boldsymbol{p}}_0)\left(\boldsymbol{L}(\breve{\boldsymbol{q}})\right)^{k-1}\right]^\top \mathrm{vec}(\breve{\boldsymbol{p}}_k - \breve{\boldsymbol{p}}_0(\breve{\boldsymbol{q}})^k) \in \mathbb{R}^8.$$

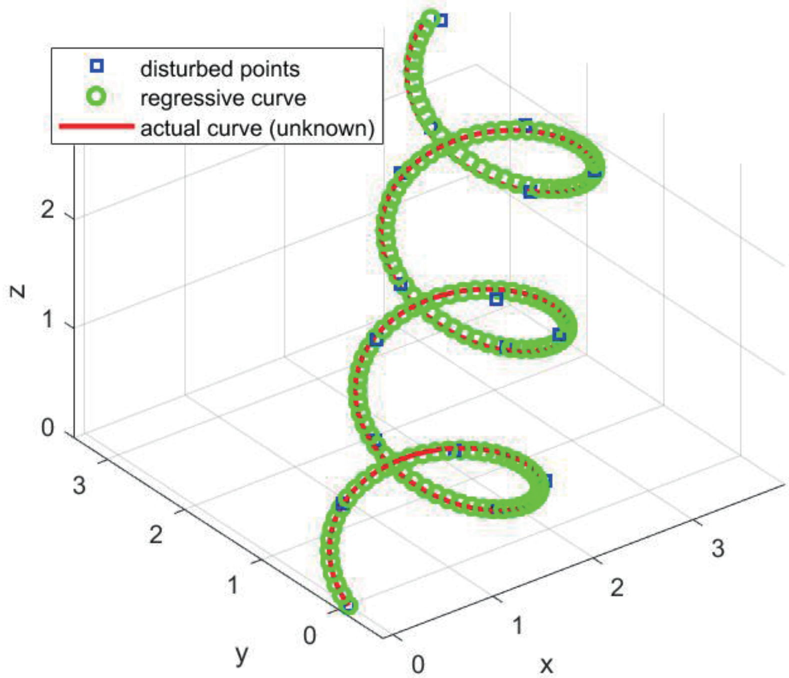

Figure 3: Illustration of a regression problem in unit dual quaternion variables.

Starting from the current iterate $\breve{\boldsymbol{q}}_t$, the proximal linearized algorithm produces a new iterate $\breve{\boldsymbol{q}}_{t+1}$ by solving the following subproblem

$$\begin{aligned} \mathrm{vec}\breve{\boldsymbol{q}}_{t+1} \quad &= \quad \arg\min \ \|\mathrm{vec}\breve{\boldsymbol{q}} - \mathrm{vec}\breve{\boldsymbol{q}}_t + \kappa_t g_R(\breve{\boldsymbol{q}}_t)\|_2^2 \\ & \qquad \text{s.t.} \quad \mathrm{vec}\breve{\boldsymbol{q}} \in \Omega. \end{aligned}$$

Here, Algorithm 1 is applicable. Then, we set $t \leftarrow t+1$ and repeat this process until $g_R(\breve{\boldsymbol{q}}_t)$ is sufficiently small.

As an example, Table 3 lists observed noise-corrupted unit dual quaternions $\breve{\boldsymbol{p}}_1, \ldots, \breve{\boldsymbol{p}}_{16}$. An initial point is $\breve{\boldsymbol{p}}_0 = 1$. Positions of these unit dual quaternions are illustrated in Figure 3 with blue squares. Using the proximal linearized algorithm, we obtain the estimated solution

$$\breve{\boldsymbol{q}}_{est} = 0.8090 + 0.3889\mathbf{i} + 0.3974\mathbf{j} + 0.1908\mathbf{k} - 0.0976\epsilon - 0.1940\epsilon\mathbf{i} + 0.1919\epsilon\mathbf{j} + 0.4094\epsilon\mathbf{k},$$

which corresponds to a regression curve marked by green circles in Figure 3. For comparison, the true curve with

$$\breve{\boldsymbol{q}}_{true} = 0.8090 + 0.3919\mathbf{i} + 0.3919\mathbf{j} + 0.1959\mathbf{k} - 0.0980\epsilon - 0.1959\epsilon\mathbf{i} + 0.1959\epsilon\mathbf{j} + 0.4045\epsilon\mathbf{k}$$

is described by a red curve. Obviously, the proposed proximal linearized algorithm solves this regression problem in unit dual quaternion variables successfully.

6 Conclusion

A proximal linearized algorithm was customized for solving optimization of real function in unit dual quaternion variables. Particularly, we designed a direct method for computing the projection onto the unit dual quaternion set which is nonconvex and unbounded. Using the projection, we proposed and analyzed the proximal linearized algorithm. Preliminary numerical experiments on the hand-eye calibration and a special regression problem illustrated that the proposed algorithm is effective and promising.

Acknowledgments

The authors are grateful to the associate editor and two anonymous referees for their comments which helped us to improve our manuscript essentially.

References

[1] Y. Chen, L. Qi and X. Zhang, Color image completion using a low-rank quaternion matrix approximation, *Pac. J. Optim.* 18 (2022) 55–75.

[2] J.C.K. Chou and M. Kamel, Finding the position and orientation of a sensor on a robot manipulator using quaternions, *Int. J. Rob. Res.* 10 (1991) 240–254.

[3] W.A. Clifford, Preliminary sketch of biquaternions, *Proc. Lond. Math. Soc.* 4 (1873) 381–395.

[4] K. Daniilidis, Hand-eye calibration using dual quaternions, *Int. J. Rob. Res.* 18 (1999) 286–298.

[5] J. Heller, M. Havlena and T. Pajdla, Globally optimal hand-eye calibration using branch-and-bound, *IEEE Trans. Pattern Anal. Mach. Intell.* 38 (2016) 1027–1033.

[6] R. Horaud and F. Dornaika, Hand-Eye Calibration, *Int. J. Rob. Res.* 14 (1995) 195–210.

[7] J. Jiang, X. Luo, Q. Luo, L. Qiao and M. Li, An overview of hand-eye calibration, *Int. J. Adv. Manuf. Technol.* 119 (2022) 77–97.

[8] L. Kavan, S. Collins, C. O'Sullivan and J. Žára, *Dual quaternions for rigid transformation blending*, Technical Report, Trinity College Dublin, 2006.

[9] E. Pedrosa, M. Oliveira, N. Lau and V. Santos, A general approach to hand-eye calibration through the optimization of atomic transformations, *IEEE Trans. Robot.* 37 (2021) 1619–1633.

[10] L. Qi, Standard dual quaternion optimization and its applications in hand-eye calibration and SLAM, *Commun. Appl. Math. Comput.* 5 (2023) 1469–1483.

[11] L. Qi, C. Ling and H. Yan, Dual quaternions and dual quaternion vectors, *Commun. Appl. Math. Comput.* 4 (2022) 1494–1508.

[12] L. Qi and Z. Luo, Eigenvalues and singular values of dual quaternion matrices, *Pac. J. Optim.* 19 (2023) 257–272.

[13] S. Qiu, M. Wang and M.R. Kermani, A new formulation for hand-eye calibrations as point-set matching, *IEEE Trans. Instrum. Meas.* 69(2020) 6490–6498.

[14] Y.C. Shiu and S. Ahmad, Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form $AX = XB$, *IEEE Trans. Rob. Autom.* 5 (1989) 16–29.

[15] S.L. Shmakov, A universal method of solving quartic equations, *Int. J. Pure Appl. Math.* 71 (2011) 251–259.

[16] R.Y. Tsai and R.K. Lenz, A new technique for fully autonomous and efficient 3D robotics hand/eye calibration, *IEEE Trans. Rob. Autom.* 5 (1989) 345–358.

[17] W. Wang, Y. Liu, H. Song and Z. Du, A novel singular-free solution based on principle of transfer theory for the hand-eye calibration problem, *Mech. Mach. Theory* 170 (2022): 104723.

[18] J. Wu, M. Liu, Y. Zhu, Z. Zou, M. Dai, C. Zhang, Y. Jiang and C. Li, Globally optimal symbolic hand-eye calibration, *IEEE ASME Trans. Mechatron.* 26 (2021) 1369–1379.

[19] C. Yuksel, High-performance polynomial root finding for graphics, *Proc. ACM Comput. Graph. Interact. Tech.* 5 (2022): Article 27.

[20] Z. Zhang, L. Zhang, and G.-Z. Yang, A computationally efficient method for hand-eye calibration, *Int. J. Comput. Assist. Radiol. Surg.* 12 (2017) 1775–1787.

[21] H. Zhuang and Z.S. Roth, Comments on "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form $AX = XB$", *IEEE Trans. Rob. Autom.* 7 (1991) 877–878.

Manuscript received 15 March 2023
revised 30 June 2023
accepted for publication 25 July 2023

Can Li
School of Mathematical Sciences
South China Normal University
Guangzhou 510631, China
School of Mathematics and Statistics
Honghe University
Mengzi 661199, China
E-mail address: 021010122@m.scnu.edu.cn

Yannan Chen
School of Mathematical Sciences
South China Normal University
Guangzhou 510631, China
E-mail address: ynchen@scnu.edu.cn

Dong-Hui LI
School of Mathematical Sciences
South China Normal University
Guangzhou 510631, China
E-mail address: lidonghui@m.scnu.edu.cn