A fast and accurate new algorithm for hand–eye calibration on SO(3)×R^3

Shuwei Qiu *, Miaomiao Wang, Mehrdad R. Kermani
Electrical and Computer Engineering, Western University, London, Ontario, Canada

A R T I C L E   I N F O

Keywords:
Hand–eye calibration
AX = XB
Point set matching
Nonlinear estimator
Lie group SO(3)

A B S T R A C T

In this paper, we propose a new solution for an old problem in robotics related to the calibration of the hand, i.e., the robot’s end-effector, and the eye, i.e., a stereo vision camera. The problem is formulated as a point set matching problem and a nonlinear estimator on manifold SO(3)×R^3 used for obtaining the solution. The main advantage of the proposed approach is that it allows to decouple the error associated with the rotational estimation on Lie group SO(3) from the error of the translational estimation on R^3, which subsequently allows to tune the learning rates for the rotation and translation estimation, separately. This will result in significant increase in the convergence speed of the proposed approach. To show the advantages of our approach, we compare the results with those obtained from other conventional hand–eye calibration solutions as well as those based on point set matching. The experimental results will demonstrate that our proposed hand–eye calibration approach outperforms other approaches in terms of accuracy and computational speed.

1. Introduction

The new collaborative robot manipulators intended for interactive manipulation tasks use exteroceptive sensors such as stereo cameras as a common sensor. The need of using these cameras necessitates the camera calibration — a subject under in-depth research for decades (Harris & Teeder, 1993). The camera used for this purpose can be either on-board the robot (eye-in-hand) or mounted near the robot as an independent device (eye-to-hand). The information acquired from the camera is always represented in the coordinate system of the camera. To use the data from the camera along with the robot’s proprioceptive measurements, it is necessary to represent the acquired data within the coordinate system of the robot. This requires estimating the transformation matrix between the camera’s and the robot’s coordinate system, which is the goal of the hand–eye calibration problem (Bishop & Spong, 1999). However, it is difficult to precisely estimate such a transformation matrix (Xiao et al., 2004). Variations of this problem exist in other application fields such as navigation systems for autonomous vehicles (Figueroa & Mahajan, 1994; Wang & Tayebi, 2020a, 2020b), vision-based autonomous landing for aerial vehicles (Bhargavapuri, Shastri, Sinha, Sahoo, & Kothari, 2019), visual servoing for autonomous underwater vehicles (Allibert, Hua, Krupinski, & Hamel, 2019), to name a few.

The hand–eye calibration problem is conventionally formulated as AX = XB or AX = YB problems (Shah, Eastman, & Hong, 2012). The AX = XB formulation provides a solution for the unknown transformation matrix (X) between the robot end-effector and the camera. The AX = YB formulation, on the other hand, provides solutions for both the unknown transformation matrix between the robot end-effector and the camera (X) as well as the unknown transformation matrix between the world and the robot base (Y). The objective of the current study is to obtain X in AX = XB formulation. A typical configuration for solving this problem is shown in Fig. 1 in that A_i (i = 1, 2) denotes the homogeneous transformation matrix between the robot base and the robot gripper for the two different poses, B_i (i = 1, 2) denotes the homogeneous transformation matrix between the camera’s frame and the world’s frame affixed to the checkerboard pattern, and X is the homogeneous transformation matrix to be obtained. In this context, it is not difficult to show that,

\[ A_i X B_1 = A_i X B_2 = A_i^{-1} A_1 X = X B_2 B_1^{-1} = AX = XB \] (1)

where A = A_i^{-1} A_1, and B = B_2 B_1^{-1}. To make these notations consistent with those that are common in robotics literature, X, A_i, and B are relabeled as \( T \) (between the gripper’s frame and the camera’s frame), \( T \) (between the robot base’s frame and the gripper’s frame), and \( T \) (between the camera’s frame and the world’s frame), respectively. Different methods have been proposed for obtaining \( T \) (or X) within this context (Horaud & Dornaika, 1995; Park & Martin, 1994; Shah, 2013; Wu, Sun, Wang, & Liu, 2019). However, these methods treat the data from stereo cameras in exactly the same way as that from single cameras, and they overlook the benefit of depth recovering feature of stereo cameras. In the proposed solution, we leverage the depth

---

\( \ast \) This research was supported in part through funds received from the Natural Sciences and Engineering Research Council of Canada, Canada Foundation for Innovation, and Ontario Centres of Excellence, Canada.

\( \ast \) Corresponding author.

E-mail addresses: sqiu47@uwo.ca (S. Qiu), mwang448@uwo.ca (M. Wang), mkerman2@uwo.ca (M.R. Kermani).

https://doi.org/10.1016/j.conengprac.2021.104726
Received 29 June 2020; Received in revised form 21 October 2020; Accepted 31 December 2020
Available online xxxx
0967-0661/© 2021 Elsevier Ltd. All rights reserved.
re recuperating feature of stereo cameras to solve the hand–eye calibration problem as a point set matching problem. By doing so, a better calibration accuracy is achieved as shown experimentally.

In what follows, we briefly review the point set matching formulation of the hand–eye calibration problem. If \( B_T \), the homogeneous transformation matrix between the robot base frame and the camera frame, is available, then the relation among \( B_T \), \( B_T \) (the homogeneous transformation matrix between the robot base frame and the gripper frame) and \( C_T \) (the homogeneous transformation matrix between the gripper frame and the camera frame) can be expressed as,

\[
C_T = B_T^{-1} B_T \tag{2}
\]

By rearranging (2), one can deduce \( C_T \) as,

\[
X = C_T = B_T^{-1} B_T \tag{3}
\]

in that \( B_T \) is available through robot kinematics and \( B_T \) is the transformation matrix between the robot base frame (\([B]\)) and the camera frame (\([C]\)). Since matching the points represented in two frames (i.e., point set matching) is a common approach for estimating the homogeneous transformation matrix between two frames, one can formulate the hand–eye calibration problem as a point set matching problem. The point set matching approaches will be reviewed in Section 2. In our previous work, we studied this idea of formulating the hand–eye calibration problem as a point set matching problem and proposed a solution based on the gradient descent algorithm on the special Euclidean group SE(3) (Qiu, Wang, & Kermani, 2020). We named this solution as GD-SE(3). In this solution, the gradients of the rotation matrix \( R \) and the translation vector \( p \) were grouped in a homogeneous transformation matrix in SE(3). Because of this integration, the convergence speed of GD-SE(3) is slow especially when the initial value of the unknown transformation is far from the optimal solution.

In this paper, we propose a new solution on manifold \( \text{SO}(3) \times \mathbb{R}^3 \) for the hand–eye calibration problem motivated by the nonlinear estimators proposed in Wang and Tayebi (2019, 2020a). The proposed algorithm offers both accurate results and fast convergence speed that rivals state-of-the-art solutions for the hand-eye calibration problem. We call our proposed algorithm HI-SO(3)R3 for its high convergence speed. The contributions of this work are summarized as follows:

- We propose a new algorithm on manifold \( \text{SO}(3) \times \mathbb{R}^3 \) for matching two rigidly related point sets to estimate the homogeneous transformation matrix between them.
- We extend the nonlinear continuous observer proposed in Wang and Tayebi (2019, 2020a) to a nonlinear discrete observer (estimator) for point set matching.
- We show that the error of the rotational estimation on \( \text{SO}(3) \) are decoupled from that of the translational estimation on \( \mathbb{R}^3 \). With this decoupling property, the learning rates for rotation and translation estimation can be tuned separately, which significantly increases the convergence speed of the proposed algorithm without affecting its accuracy.
- We propose a strategy for selecting suitable learning rates for the estimation of the unknown rotation matrix and translation vector such that the convergence is guaranteed.
- We provide exhaustive comparison results with state-of-the-art solutions to demonstrate the effectiveness of our proposed algorithm.

The rest of the paper is organized as follows: Section 2 briefly outlines related works in the field of hand–eye calibration and point set matching. Section 3 introduces the proposed algorithm, provides corresponding mathematical derivations, and offers guidelines for selecting the learning rates. Section 4 experimentally evaluates the effectiveness of the proposed approach. At the end, Section 5 concludes the paper.

2. Related works

2.1. AX = XB calibration

The most conventional formulation of the hand–eye calibration problem is given as \( AX = XB \). This well-known formulation was first developed in Shi and Ahmad (1989) and Tsai and Lenz (1989). Using homogeneous transformation matrix one can expand \( AX = XB \) formulation as follows,

\[
\begin{bmatrix}
X_A & p_A \\
X_B & p_B
\end{bmatrix}
\begin{bmatrix}
R_X & p_X \\
0 & 1
\end{bmatrix}
= \begin{bmatrix}
X_B & p_B \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
R_X & p_X \\
0 & 1
\end{bmatrix}
\Rightarrow
\begin{cases}
R_A p_A = R_X p_B \\
R_A p_A + p_A = R_X p_B + p_X
\end{cases}
\tag{4}
\]

where \( R \) represents the rotation matrix and \( p \) represents the translation vector for \( i = A, B, \) and \( X \), respectively. Different methods have been proposed to solve this problem that can be categorized as separable, simultaneous, and iterative methods (Shah et al., 2012).

In separable methods, the rotation \( (R_X) \) and the translation \( (p_X) \) portion of the \( X \) matrix are solved in sequence. Park and Martin (1994) found a least-square solution for \( AX = XB \) using Lie theory. They solved \( R_X \) and \( p_X \) by minimizing the following error functions,

\[
e_1 = \sum_{i=1}^{m} \| R_X \beta_i - a_i \|^2 
\tag{5}
\]

\[
e_2 = \sum_{i=1}^{m} \| (R_X I - I) p_X - R_X p_B + p_A \|^2 
\tag{6}
\]

where \( m \) is the number of measurements, \( a_i = \log R_i \), and \( \beta_i = \log R_i \). \( R_X \) and \( p_X \) were solved as,

\[
R_X = (M^T M)^{-1} M^T p_X = (C^T C)^{-1} C^T d
\tag{7}
\]

in that \( M \) and \( C \) were defined as,

\[
M = \sum_{i=1}^{m} \beta_i a_i^T \\
C = \begin{bmatrix}
I - R_{A} & \vdots & 0 \\
\vdots & \ddots & \vdots \\
I - R_{A_{m}} & \vdots & 0
\end{bmatrix}
\]

\[
d = \begin{bmatrix}
(p_{A})_{1} - p_X p_B \\
\vdots \\
(p_{A_{m}})_{1} - p_X p_B_{m}
\end{bmatrix}
\]

Wu et al. (2019) solved \( AX = XB \) problem using 4D Procrustes analysis and they proposed, for the first time, to solve the Procrustes problem through eigendecomposition. Shah (2013) solved the Robot-World/hand–eye calibration problem in closed-form by exploiting the Kronecker product. The separable methods suffer from a common problem — the estimation error of \( R_X \) exacerbates the estimation of \( p_X \) (Shah et al., 2012; Zhang, 2016).
The simultaneous methods overcome this problem by solving for the rotational and translational components simultaneously. Horauld and Dornaika (1995) solved $R_X$ and $p_X$ simultaneously through a nonlinear technique after representing $R_X$ as a unit quaternion. They were the first to use this nonlinear technique for simultaneous minimization of the rotation quaternion and the translation part (Danilidis, 1999). This approach was relatively slow.

The iterative methods solve $R_X$ and $p_X$ iteratively by employing optimization techniques (Shah et al., 2012; Zhang, 2016). Because of using the optimization technique, the error propagation from $R_X$ to $p_X$ is avoided. With real and simulated data, Tabb and Youssef (2017) compared various iterative methods with varying formulations, different parameterizations, and diverse cost functions. They proposed, for the first time, to use camera re-projection error for hand–eye calibration which prevented the artifacts and errors related to camera calibration from being propagated to the process of hand–eye calibration. However, this method is sensitive to the choice of the initial solution. To find an approximate solution to minimize the camera re-projection error, an initial solution obtained using an alternative method was used as an initial solution for their experiments. The readers are referred to Shah et al. (2012) for a more exhaustive review of the hand–eye calibration algorithm.

2.2. Point set matching

The problem of point set matching (or point set registration) is to estimate the homogeneous transformation matrix between two point sets, often called target points and moving points. This transformation matrix is rigid when the distance between the corresponding points is constant, otherwise, it is non-rigid (Maiseli, Gu, & Gao, 2017).

The most notable algorithm for point set matching is the Iterative Closest Point (ICP) algorithm. The seminal work of this algorithm was reported in Besl and McKay (1992) and Zhang (1994). ICP estimates the corresponding points between the two point sets based on the distance between different points. ICP relies on the singular value decomposition (SVD) of the covariance matrix between two point sets to iteratively minimize the distance between all corresponding points. Several other variants of ICP have been reported in the literature that better estimate the correspondence and improve the computational efficiency of the algorithm (He, Liang, Yang, Li, & He, 2017; Tikhonikh, Makoverskii, & Kuznetsov, 2016; Tykkälä, Audras, & Comport, 2011).

The problem of point set matching can also be considered as an estimation problem of the probability density. The Coherent Point Drift (CPD) algorithm (Myronenko & Song, 2010) was proposed within this context. CPD represents two sets of points as the centroids of SO(3) × SE(3) (Qiu et al., 2020) has much faster convergence rate. The details of CPD solvers can be found in (Tykkälä, Audras, & Comport, 2011).

Moreover, using (3) and the result of (9) (i.e., $c^T T_c$), $\hat{c}^T T$ can be obtained as

$$\hat{c}^T T = \hat{c} R c = \hat{c} R^{-1} R c = \hat{c} c$$

in that $\hat{c}^T T$ is derived from the robot kinematics. Thus, the hand–eye calibration problem for estimating $\hat{c}^T T$ boils down to the problem of estimating the homogeneous transformation matrix $\hat{c}^T T$ between the two sets of points ($c^T X_c$ and $c^T X_m$) within the context of hand–eye calibration.

3. Nonlinear estimator design

In this section, we provide the design of a nonlinear estimator to estimate $\hat{c}^T T$ and minimize the error function defined in (9). To this effect, we define $\hat{c} R_k$ and $\hat{c} \beta_k$ as the estimated value of these variables at the $k$th iteration of the algorithm. We also define the re-projection error of the $i$th point, $e_i$, at the $k$th iteration as follows:

$$e_i = e_i R_k = \hat{c} R_k c X_i + c \beta_k$$

Given the relation between the coordinates of the points in two frames, i.e., $c X_i = \hat{c} R c X_i + \hat{c} \beta$ and the one given in (8), one can verify that $e_i = (\hat{c} R_k c X_i + \hat{c} \beta_k) - (\hat{c} R_k c X_i + c \beta_k) = - \hat{c} \beta_k - \hat{c} \beta_k = - \hat{c} \beta_k - \hat{c} \beta_k = - \hat{c} \beta_k - \hat{c} \beta_k$. Let $c X_i = \frac{1}{n} \sum_{i=1}^{n} c X_i$ be the center of the point set $c X_i \{1 \}$ with $n$ points. We use $\hat{e}_i$ to introduce the following two innovation terms:

$$\Delta R = \sum_{i=1}^{n} e_i R_k c X_i - c X_i$$

$$\Delta \beta_k = \sum_{i=1}^{n} e_i c \beta_k$$

We then propose the following discrete nonlinear estimator for estimating the values of $c R_k$ and $c \beta_k$ and its update criteria as,

$$\hat{c} R_{k+1} = R(\Delta R) c R_k$$

$$\hat{c} \beta_k + c \beta_k = R(\Delta R) \left( c \beta_k + a c \beta_k - c X_i \right) + c X_i$$
where \( R(\Delta R) = \exp \left( -\frac{\alpha}{2}(\Delta R - \Delta R^P) \right) \), and the scalars \( \alpha_R \) and \( \alpha_p \) are the learning rates for the estimation of \( \hat{R} \) and \( \hat{p} \), respectively. Our proposed solution is motivated by the continuous nonlinear observers presented in Wang and Tayebi (2019, 2020a). However unlike the continuous observers, our estimator updates discretely and the data from an IMU (inertial measurement unit) is not required. Hence, the convergence results and the selection of the gains in Wang and Tayebi (2019, 2020a) are not applicable to our method.

Although a large value of the learning rate can increase the convergence speed, the value of \( \alpha_R \) and \( \alpha_p \) cannot be arbitrarily selected otherwise the estimation process would not converge. To guarantee the convergence, the selection of \( \alpha_R \) and \( \alpha_p \) is discussed next.

3.3. Learning rate selection

3.3.1. Selecting \( \alpha_R \)

Let us define the rotational error between the truth and the estimate of \( \hat{R} \) at the \( k \)th iteration as,

\[
E_k^R = \frac{\hat{R}}{c} R c \hat{R}^T
\]

where \( \frac{\hat{R}}{c} R c \) is the truth and \( \frac{\hat{R}}{c} \) is its estimate at the \( k \)th iteration. According to Wang and Tayebi (2020a), the innovation term \( \Delta_R \) defined in (12) can be rewritten as \( \Delta_R = (I - E_k^R)^T \mathcal{M} \) with \( \mathcal{M} = \sum_{i=1}^{n} \hat{R}_i - X_i X_i^T \), and it follows from (14) and (16) that

\[
E_k^{R^T} = E_k^R R(\Delta_R)^T
\]

(17)

It has been shown that Berkane and Tayebi (2019, Theorem 1), if there exist at least three non-colinear sample points in the point set \( \{X_i\} \) with noise-free measurements \( \{c X_i\} \), then the rotation estimation error \( E_k^R \) will converge to \( I \) for any initial condition that satisfies \( \text{tr}(E_k^{R^0}) \neq -1 \) (i.e., the angle of initial rotational error \( E_k^{R0} \) is strictly less than 180°) when the value of \( \alpha_R \) in the estimator in (14) is selected as,

\[
0 < \alpha_R (\text{tr}(\mathcal{M}) - \lambda_i^M) < 1, \quad i = 1, 2, 3
\]

(18)

in that \( \text{tr}(\mathcal{M}) \) denotes the trace of \( \mathcal{M} \), and \( \lambda_i^M \) is the \( i \)th eigenvalue of \( \mathcal{M} \). The stability analysis for this approach has been previously reported and is not repeated here for the sake of brevity (Berkane & Tayebi, 2019, Theorem 1).

Remark 1. The condition in (18) for the scalar gain \( \alpha_R \) was developed for the worst cases. If \( \alpha_R \) is selected slightly larger than the bound in (18), the rotational estimation error \( E_k^R \) may still converge to \( I \), especially when \( E_k^R \) is close to \( I \). The experimental results will show that the proposed algorithm works when \( \alpha_R \) is slightly larger than the bound given in (18).

Remark 2. Theoretically, in the absence of measurement noise, \( \Delta_R - \Delta_R^P = \Omega_{c X_i} \) implies either \( E_k^R = I \) or \( \text{tr}(E_k^{R^0}) = -1 \), in that the latter at \( k = 0 \) may cause the convergence issue of our estimator (Wang & Tayebi, 2020a). However, due to the unavoidable measurement noise in practice, the estimation error \( E_k^R \) with \( \text{tr}(E_k^{R^0}) = -1 \) will leave the undesired equilibrium point and converge to \( I \).

3.3.2. Selecting \( \alpha_p \)

Let us define the translational error between the truth and the estimate of \( \frac{\hat{p}}{c} \) at the \( k \)th iteration as,

\[
E_k^P = \frac{\hat{p}}{c} - X_i - E_k^R \frac{\hat{p}}{c} R_k - X_i
\]

(19)

where \( X_i \) is the center of the point set \( \{X_i\} \), \( \frac{\hat{p}}{c} \) is the truth translation vector, and \( E_k^R \frac{\hat{p}}{c} R_k \) is its estimated value at the \( k \)th iteration. Using (13), (16) and (19), one can rewrite \( \Delta_p \) in terms of estimation errors \( E_k^R \) and \( E_k^P \) as,

\[
\Delta_p = n(E_k^R)^T E_k^P
\]

(20)

where \( n \) is the number of points in the point set. From (14)–(16), (19) and (20), one can show that,

\[
E_{k+1}^P = E_k^P - \eta \frac{\hat{p}}{c} R_k - E_k^R \frac{\hat{p}}{c} R_k - X_i
\]

\[
\Rightarrow E_{k+1}^P = E_k^P - \eta \alpha_p E_k^R \frac{\hat{p}}{c} R_k
\]

\[
\Rightarrow E_{k+1}^P = E_k^P - \eta \alpha_p E_k^R \frac{\hat{p}}{c} R_k^T E_k^P
\]

\[
\Rightarrow E_{k+1}^P = E_k^P - \eta \alpha_p E_k^R \frac{\hat{p}}{c} R_k^T E_k^P
\]

\[
(21)
\]

where \( E_k^P \) denotes the initial translational error (at \( k = 0 \)). It is obvious from (21) that \( E_k^P \) will exponentially converge to \( 0 \) with noise-free measurements if \( \alpha_p \) is selected as

\[
0 < \eta \alpha_p < 1.
\]

(22)

It is worth to note that \( \alpha_p \) and \( \alpha_R \) including the matrix \( \mathcal{M} \) and \( \lambda_i^M (i = 1, 2, 3) \) (the eigenvalues of \( \mathcal{M} \)) are calculated before conducting the iterative estimation of the proposed algorithm (i.e., they are calculated off-line).

Remark 3. If \( \alpha_p \) does not satisfy the bound in (22), \( E_k^P \) may still converge but the convergence is not guaranteed (Wang & Tayebi, 2020a). The experimental results will show that the proposed algorithm works if \( \alpha_p \) is selected slightly larger than the bound given in (22).

Remark 4. As seen in (17) and (21) the convergence of the estimation errors \( E_k^R \) and \( E_k^P \) are independent. In other words, the convergence time and accuracy of \( E_k^R \) and \( E_k^P \) will not be affected by each other. This decoupling property results from the special design of the innovation terms \( \Delta_R \) and \( \Delta_p \). This decoupling property results in a dramatic reduction in the algorithm convergence time. Note also that \( E_k^R = \frac{\hat{R}}{c} R_k c \hat{R}^T = I \) and \( E_k^P = \frac{\hat{p}}{c} - \frac{\hat{p}}{c} R_k - E_k^R \frac{\hat{p}}{c} R_k - X_i \) imply that \( \frac{\hat{R}}{c} R_k = \frac{\hat{R}}{c} R^P \) and \( \frac{\hat{p}}{c} R_k = \frac{\hat{p}}{c} R^P \), respectively. Consequently, we obtain \( \frac{\hat{R}}{c} R^P = \frac{\hat{R}}{c} R^P \).

3.4. Proposed algorithm

Having introduced the nonlinear estimator and the requirements for selecting the learning rates, the flow chart of our proposed algorithm, HI-SO(3)R3, is given in Fig. 2 followed by the details of the algorithm.

The initial value of \( \frac{\hat{R}}{c} R \) can be randomly chosen in SO(3) and the initial value of \( \frac{\hat{p}}{c} R \) can be chosen to be \( 0 \). Alternatively, \( \frac{\hat{R}}{c} R \) and \( \frac{\hat{p}}{c} R \) can be initialized by singular value decomposition (SVD). A simple method is applied to terminate iterations appropriately. Considering the recent
three iterations, we calculate the average change of the rotation matrix, “dR” and the translation vector “dp” to determine the algorithm termination. The algorithm will stop either when both “dR” and “dp” are below a given tolerance “σ”, or when the maximum number of iteration N is surpassed. The following pseudo-codes provides more details regarding the algorithm.

Algorithm The HI-SO(3)R3 algorithm

**Input:** two point sets (\{\vec{\overline{X}}_i\}_1, \{\vec{\overline{C}}_i\}_1, i = 1, 2, ..., n) with established correspondence, maximum iteration (N), learning rates (\alpha_R and \alpha_p), the tolerance (\sigma), initial guess (\vec{R}_0 and \vec{p}_0).

**Output:** \( bT \) (the estimated transformation matrix between the two input point sets).

Initialization: \( k = 0 \), \( b\hat{R}_k = R_{0b} \), \( b\hat{p}_k = p_0 \).

1. \( bX_c = \frac{1}{n} \sum_{i=1}^{n} bX_i \)

repeat

4: Convergence detection: \( \sigma_c = \text{rot2quat}(b\hat{R}_k) \) \( \Rightarrow \) Convert \( b\hat{R}_k \) into quaternion

6: if \( 0 < k < 3 \) then

\[ dR = \frac{1}{n} \sum_{i=1}^{n} \frac{b(\hat{R}_k \vec{X}_i - b\hat{p}_k - \vec{C}_i)}{||b\hat{p}_i||} \] ;

\[ dp = \frac{1}{n} \sum_{i=1}^{n} \frac{\vec{C}_i - b\hat{R}_k \vec{X}_i}{||b\hat{p}_i||} \]

end if

8: if \( k \geq 3 \) then

\[ dR = \frac{1}{n} \sum_{j=0}^{k-1} \frac{b(\hat{R}_j \vec{X}_i - b\hat{p}_j - \vec{C}_i)}{||b\hat{p}_j||} \] ;

\[ dp = \frac{1}{n} \sum_{j=0}^{k-1} \frac{\vec{C}_i - b\hat{R}_j \vec{X}_i}{||b\hat{p}_j||} \]

end if

10: if \( dR < \sigma_c \) and \( dp < \sigma_p \) then

Return \( b\hat{R}_k = \begin{bmatrix} b\hat{R}_k & b\hat{p}_k \\ 0 & 1 \end{bmatrix} \)

end if

14: \( \Delta_R = 0 \), \( \Delta_p = 0 \)

16: while \( 1 \leq l \leq n \) do

\( \vec{e}_p = b\hat{X}_l - (b\hat{R}_k \vec{C}_l + b\hat{p}_k) \)

\( \Delta_R = \Delta_R + \vec{e}_p \)

\( \Delta_p = \Delta_p + \vec{e}_p \)

end while

22: \( c\hat{R}_{k+1} = R^{-1}(dR) c\hat{R}_k \)

\( c\hat{p}_{k+1} = R(dR)(c\hat{p}_k + \alpha_p \Delta_p - b\hat{X}_l) + b\hat{X}_l \)

24: \( k = k + 1 \)

until \( k = N \)

26: return \( b\hat{R}_k = \begin{bmatrix} b\hat{R}_k & b\hat{p}_k \\ 0 & 1 \end{bmatrix} \)

4. Experimental results

This section outlines the experiments designed to demonstrate the performance of HI-SO(3)R3 in comparison with some other hand–eye calibration algorithms and point set matching algorithms.

4.1. Hardware setup

We used a KUKA Light-Weight Robot (LWR) IV to conduct our experiments. A special fixture that included a screw with a sharp tip was fabricated and attached to the KUKA robot. The fixture was designed such that the tip of the screw coincided with the gripper frame. The calibration device used in these experiments was a typical checkerboard pattern containing 54 corners. These components are shown in Fig. 3. We used two different stereo cameras, Intel RealSense D435 and KYT-U100-960R301, also shown in Fig. 3 for conducting our experiments. The Intel camera has better resolution than the KVT camera. All the same data were collected using the KUKA robot with 16 GB RAM.

4.2. Experiment data collection

4.2.1. Conventional formulation

At the outset of our experimental studies, we aimed at evaluating the performance of some of the conventional methods of hand–eye calibration in comparison to our proposed algorithm. To this end, we captured 100 different images of the checkerboard pattern. During image acquisition, we also recorded the gripper poses (i.e., position and orientation) using the internal encoders of the KUKA robot. The A and B matrix in (1) were calculated as \( A = A^{-1}_{1}A_{1} = \frac{1}{3} A_{1} \) and \( B = B_{1}A_{1} = A_{2}B_{2}T_{2}^{1}, \) respectively, where \( bT \) as defined previously is the transformation matrix between the base of the KUKA robot and the gripper and \( u_{w}T \) is the transformation matrix between the camera’s and the world’s frame. Different implementations of the traditional methods reported in Horaud and Dornaika (1995), Park and Martin (1994), Shah (2013) and Wu et al. (2019) were used for this evaluation.

4.2.2. Point set matching formulation

We also examined and compared the performance of our algorithm with some widely used point set matching algorithms. The coordinates of the checkerboard corners expressed in the robot base frame were collected when the tip of the screw made physical contact with each corner as shown in Fig. 3(a). Referring to (8), these coordinates constituted \( bP \). By taking advantage of the stereo camera, we estimated the checkerboard corners’ 3D coordinates expressed in the camera frame from the images acquired previously. These coordinates constituted \( cP \). The collected data (i.e. \( bP \) and \( cP \)) was used to compare the performance of HI-SO(3)R3, GD-SE(3), ICP, and CPD algorithms. To this effect, 100 sets of \( cP \) were obtained among which 70% were used for training and the remaining 30% were used for validation.

4.3. Performance evaluation

The performance of the implemented algorithms was assessed and compared using the previously acquired data.

4.3.1. Training phase

During the training phase, we used the obtained values of \( bT \) and \( u_{w}T \) matrices (i.e., A and B matrices) and several conventional algorithms (Horaud & Dornaika, 1995; Park & Martin, 1994; Shah, 2013; Wu et al., 2019) to estimate \( cT \) (i.e., X matrix). We also used \( dP \) and \( dP \) to estimate \( cT \) as per (8) by implementing ICP (Besl & McKay,
matrix $\mathbb{R}$ solution (e.g., ICP, GD-SE(3), and HI-SO(3)R3), the initial rotation algorithm. More specifically, for the algorithms that required an initial parameters that achieved the best performance of each implemented from the training data sets.

For noises, the estimation of $\hat{G}$ was randomly generated in SO(3) and the initial translation $0$ was set to $0$. In (10), to reduce the influence of the measurement noises, the estimation of $\hat{G}$ was obtained through averaging the results from the training data sets.

As for parameter selection in these algorithms, we selected the aforementioned parameters were the same with different cameras. To detect the convergence of the algorithm and the maximum number of iteration ($N$) in HI-SO(3)R3 and ICP were chosen to be $1 \times 10^4$ and $1 \times 10^5$, respectively. To obtain the same accuracy, $\sigma$ and $N$ used in GD-SE(3) were selected to be $1 \times 10^{-7}$ and $1 \times 10^5$, respectively. The aforementioned parameters were the same with different cameras. To achieve its best performance, the $\sigma$ parameter in Wu’s method (Wu et al., 2019) was chosen as $1 \times 10^{-14}$ and $1 \times 10^6$ for the tests with Intel camera and the KYT camera, respectively.

The correspondence between $\hat{C}P$ and $\hat{B}P$ is a necessary component of HI-SO(3)R3, GD-SE(3), and ICP algorithms whether it is known or is calculated as part of the algorithm. To shed light on the effect of known vs. calculated correspondence, we used the CPD algorithm to estimate the correspondence between the points as part of the algorithm. Comparing the results from these algorithms will demonstrate the importance of the correspondence between two point sets in the hand–eye calibration scenario.

4.3.2. Validation phase

The implemented algorithms were assessed and compared using the validation data sets. To this end, the reconstruction accuracy error (RAE) and the root mean error of the combined rotation and translation error (RMCE) were used to compare the results. RAE is calculated as the RMSE (root mean square error) of the Euclidean distance between the estimated points (i.e., $\hat{B}P$) and the truth points (i.e., $B^TP$) as,

$$RAE = \left( \frac{1}{n} \sum_{i=1}^{n} \| \hat{B}P_i - B^TP_i \| \right)^{\frac{1}{2}}$$

where $n$ is the number of points in the validation set. Also, RMCE is defined as,

$$RMCE = \frac{1}{m} \left( \sum_{i=1}^{m} \| A_iX - XB_i \| \right)^{\frac{1}{2}}$$

where $A_i$ and $B_i$ are the robot and camera motion matrices described in (1), and $m$ is the number of matrices in the validation set.

4.4. Results and discussion

In this section, the results from all experiments are provided and compared.

4.4.1. Results of the training phase

We conducted the training phase following the procedure described in Section 4.3.1. The training results for HI-SO(3)R3, GD-SE(3), ICP, and CPD using two different cameras (Intel RealSense and KYT) are shown in Fig. 4. The training errors and the training time of these algorithms are listed in Tables 1 and 2. As shown in Fig. 4, $\hat{B}P$ (the estimated points) obtained using these algorithms seems to match well with $B^TP$ (the truth points). However, for the CPD algorithm, its RAE value is much larger than other algorithms. The reason is that the CPD algorithm cannot estimate the correspondence between $C^P$ and $B^P$ effectively. This problem was exacerbated because of the rectangular arrangement of $C^P$ and $B^P$, which made the CPD algorithm more-likely to ill-estimate the correspondence between the two sets of points.

$$\begin{equation}
\left\{ \begin{array}{l}
\alpha = \frac{1}{\| A_i(S_1 - S_2) \|} \quad \text{and} \\
\beta = \frac{1}{n} \quad \text{in that} \\
\lambda = \frac{1}{M} \quad \text{is the minimum eigenvalue of the} \\
\text{the} \quad M \quad \text{and} \\
s \quad \text{the number of points. The tolerance} \\
\text{for} \\
\text{detecting the convergence of the algorithm and the maximum number of} \\
\text{iteration} (N) \quad \text{in HI-SO(3)R3 and ICP were chosen to be} \\
1 \times 10^{-4} \quad \text{and} \\
1 \times 10^5, \quad \text{respectively. To obtain the same accuracy,} \\
\sigma \quad \text{and} \\
N \quad \text{used in} \\
GD-SE(3) \quad \text{were selected to be} \\
1 \times 10^{-7} \quad \text{and} \\
1 \times 10^5, \quad \text{respectively. The} 
\end{array} \right. 
\end{equation}$$

$$\begin{equation}
\begin{array}{c}
\frac{1}{m} \left( \sum_{i=1}^{m} \| A_iX - XB_i \| \right)^{\frac{1}{2}}
\end{array}
\end{equation}$$

where $A_i$ and $B_i$ are the robot and camera motion matrices described in (1), and $m$ is the number of matrices in the validation set.

4.4. Results and discussion

In this section, the results from all experiments are provided and compared.

4.4.1. Results of the training phase

We conducted the training phase following the procedure described in Section 4.3.1. The training results for HI-SO(3)R3, GD-SE(3), ICP, and CPD using two different cameras (Intel RealSense and KYT) are shown in Fig. 4. The training errors and the training time of these algorithms are listed in Tables 1 and 2. As shown in Fig. 4, $\hat{B}P$ (the estimated points) obtained using these algorithms seems to match well with $B^TP$ (the truth points). However, for the CPD algorithm, its RAE value is much larger than other algorithms. The reason is that the CPD algorithm cannot estimate the correspondence between $C^P$ and $B^P$ effectively. This problem was exacerbated because of the rectangular arrangement of $C^P$ and $B^P$, which made the CPD algorithm more-likely to ill-estimate the correspondence between the two sets of points.

$$\begin{equation}
\left\{ \begin{array}{l}
\alpha = \frac{1}{\| A_i(S_1 - S_2) \|} \quad \text{and} \\
\beta = \frac{1}{n} \quad \text{in that} \\
\lambda = \frac{1}{M} \quad \text{is the minimum eigenvalue of the} \\
\text{the} \quad M \quad \text{and} \\
s \quad \text{the number of points. The tolerance} \\
\text{for} \\
\text{detecting the convergence of the algorithm and the maximum number of} \\
\text{iteration} (N) \quad \text{in HI-SO(3)R3 and ICP were chosen to be} \\
1 \times 10^{-4} \quad \text{and} \\
1 \times 10^5, \quad \text{respectively. To obtain the same accuracy,} \\
\sigma \quad \text{and} \\
N \quad \text{used in} \\
GD-SE(3) \quad \text{were selected to be} \\
1 \times 10^{-7} \quad \text{and} \\
1 \times 10^5, \quad \text{respectively. The} 
\end{array} \right. 
\end{equation}$$

$$\begin{equation}
\begin{array}{c}
\frac{1}{m} \left( \sum_{i=1}^{m} \| A_iX - XB_i \| \right)^{\frac{1}{2}}
\end{array}
\end{equation}$$

where $A_i$ and $B_i$ are the robot and camera motion matrices described in (1), and $m$ is the number of matrices in the validation set.

Fig. 4. Training examples: estimated points (\(\hat{B}P\), denoted by red crosses) vs. the truth points (\(B^TP\), denoted by blue circles).

Fig. 5. Validation examples with different cameras: \(\hat{B}P\) (estimated points, denoted by red crosses) vs. \(B^TP\) (truth points, denoted by blue circles).
Attaining a drill’s partial point clouds from various camera positions

Fig. 6. Point cloud reconstruction using hand-eye calibration results.

(a) The drill and various camera positions          (b) Partial point clouds of the drill

(c) HI-SO(3)R3          (d) GD-SE(3)          (e) ICP          (f) CPD

(g) Park’s              (h) Horaud’s            (i) Wu’s             (j) Shah’s

Table 1
Experimental results with the Intel RealSense camera.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Average RAE in training (mm)</th>
<th>Training time (s)</th>
<th>Average RAE in validation (mm)</th>
<th>RMCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HI-SO(3)R3</td>
<td>0.9436</td>
<td>0.3368</td>
<td>4.2517</td>
<td>$1.1 \times 10^{-3}$</td>
</tr>
<tr>
<td>GD-SE(3) (Qiu et al., 2020)</td>
<td>0.9436</td>
<td>121.29</td>
<td>4.2516</td>
<td>$1.1 \times 10^{-3}$</td>
</tr>
<tr>
<td>Traditional algorithms</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horaud’s method (Horaud &amp; Dornaika, 1995)</td>
<td>–</td>
<td>0.0083</td>
<td>30.273</td>
<td>$1.1 \times 10^{-3}$</td>
</tr>
<tr>
<td>Park’s method (Park &amp; Martin, 1994)</td>
<td>–</td>
<td>0.0042</td>
<td>30.273</td>
<td>$1.1 \times 10^{-3}$</td>
</tr>
<tr>
<td>Wu’s method (Wu et al., 2019)</td>
<td>–</td>
<td>0.0024</td>
<td>34.724</td>
<td>$1.1 \times 10^{-3}$</td>
</tr>
<tr>
<td>Shah’s method (Shah, 2013)</td>
<td>–</td>
<td>0.0026</td>
<td>134.06</td>
<td>$1.3 \times 10^{-3}$</td>
</tr>
<tr>
<td>Point set matching algorithms</td>
<td>ICP (Besl &amp; McKay, 1992)</td>
<td>0.9436</td>
<td>0.5711</td>
<td>4.2517</td>
</tr>
<tr>
<td></td>
<td>CPD (Myronenko &amp; Song, 2010)</td>
<td>102.97</td>
<td>2.0650</td>
<td>204.92</td>
</tr>
</tbody>
</table>

It is clear that the HI-SO(3)R3 algorithm outperformed all other point set matching algorithms with respect to the training time. In comparison to our previous algorithm GD-SE(3), the speedup achieved in HI-SO(3)R3 is due to the fact that the rotational estimation error is decoupled from the translational estimation error as discussed in Section 3.3. Even though the ICP algorithm required less number of
on the other hand, define a cost function such as availability of the point correspondence. The conventional algorithms, due to the fact that the point set matching algorithms benefit from the priority of point set matching algorithms to traditional methods. This is results properly. On the other hand, the RAE clearly shows the super-

inconsistent outcome seen in the values of RMCE vs. RAE. The RMCE

respondence between the points. Another important observation is the adverse effect of not having the correct cor-

iterations, it took longer to train due to its reliance on the singular value decomposition (SVD) for estimating the rotation matrix in each iteration. It is also important to note that the implemented conventional algorithms (Horaud & Dornaika, 1995; Park & Martin, 1994; Shah, 2013; Wu et al., 2019) required much less training time. This is due to the fact that these algorithms use closed-form solutions for the results. The trade-off here was in much larger validation error (RAE) values as these algorithms use closed-form solutions for the results.

2013; Wu et al., 2019) required much less training time. This is due to

algorithms (Horaud & Dornaika, 1995; Park & Martin, 1994; Shah,

value decomposition (SVD) for estimating the rotation matrix in each

Fig. 7. Performance of HI-SO(3)R3 for different $a_k$ values and $a_p = \frac{0.3}{n}$.

iterations, it took longer to train due to its reliance on the singular value decomposition (SVD) for estimating the rotation matrix in each iteration. It is also important to note that the implemented conventional algorithms (Horaud & Dornaika, 1995; Park & Martin, 1994; Shah, 2013; Wu et al., 2019) required much less training time. This is due to the fact that these algorithms use closed-form solutions for the results. The trade-off here was in much larger validation error (RAE) values as these algorithms use closed-form solutions for the results.

2013; Wu et al., 2019) required much less training time. This is due to

algorithms (Horaud & Dornaika, 1995; Park & Martin, 1994; Shah,

value decomposition (SVD) for estimating the rotation matrix in each

Fig. 7. Performance of HI-SO(3)R3 for different $a_k$ values and $a_p = \frac{0.3}{n}$.

4.4.2. Result of the validation phase

After the training phase, the validation phase was conducted following the procedure described in Section 4.3.2. Examples of the validation results obtained from different implemented algorithms are shown in Fig. 5 and the results are listed in the last two columns of Tables 1 and 2. As seen, the results are consistent with those obtained during the training phase. Once again, the degraded performance of the CPD algorithm highlights the adverse effect of not having the correct correspondence between the points. Another important observation is the inconsistent outcome seen in the values of RMCE vs. RAE. The RMCE cannot represent the effectiveness of various methods and separate the results properly. On the other hand, the RAE clearly shows the superiority of point set matching algorithms to traditional methods. This is due to the fact that the point set matching algorithms benefit from the availability of the point correspondence. The conventional algorithms, on the other hand, define a cost function such as $\sum_{i=1}^{n} \| A_i X - X B_i \|^2$ to solve for X through optimization. In reality, such optimization, as also seen in our result in Tables 1 and 2 does not necessarily minimize the RAE values.

It should be pointed out that Shah’s method (Shah, 2013) is designed for robot-world and hand–eye calibration using $AX = YB$ formulation to provide a solution for both X and Y. The results in Tables 1 and 2 only present the accuracy of the X matrix for this algorithm. Since in Shah’s algorithm, both X and Y matrices are estimated together, the errors from one matrix propagate into another matrix resulting in the least accurate results among conventional methods.

4.4.2. Result of the validation phase

After the training phase, the validation phase was conducted following the procedure described in Section 4.3.2. Examples of the validation results obtained from different implemented algorithms are shown in Fig. 5 and the results are listed in the last two columns of Tables 1 and 2. As seen, the results are consistent with those obtained during the training phase. Once again, the degraded performance of the CPD algorithm highlights the adverse effect of not having the correct correspondence between the points. Another important observation is the inconsistent outcome seen in the values of RMCE vs. RAE. The RMCE cannot represent the effectiveness of various methods and separate the results properly. On the other hand, the RAE clearly shows the superiority of point set matching algorithms to traditional methods. This is due to the fact that the point set matching algorithms benefit from the availability of the point correspondence. The conventional algorithms, on the other hand, define a cost function such as $\sum_{i=1}^{n} \| A_i X - X B_i \|^2$ to solve for X through optimization. In reality, such optimization, as also seen in our result in Tables 1 and 2 does not necessarily minimize the RAE values.

It should be pointed out that Shah’s method (Shah, 2013) is designed for robot-world and hand–eye calibration using $AX = YB$ formulation to provide a solution for both X and Y. The results in Tables 1 and 2 only present the accuracy of the X matrix for this algorithm. Since in Shah’s algorithm, both X and Y matrices are estimated together, the errors from one matrix propagate into another matrix resulting in the least accurate results among conventional methods.

It should be pointed out that Shah’s method (Shah, 2013) is designed for robot-world and hand–eye calibration using $AX = YB$ formulation to provide a solution for both X and Y. The results in Tables 1 and 2 only present the accuracy of the X matrix for this algorithm. Since in Shah’s algorithm, both X and Y matrices are estimated together, the errors from one matrix propagate into another matrix resulting in the least accurate results among conventional methods.

![Table 2](image)

**Table 2**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Average RAE (mm)</th>
<th>Training time (s)</th>
<th>Average RAE (mm)</th>
<th>RMCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HI-SO(3)R3 (Shah, 2013)</td>
<td>1.7001</td>
<td>0.106</td>
<td>20.8487</td>
<td>$2.4 \times 10^{-3}$</td>
</tr>
<tr>
<td>GD-SE(3) (Qiu et al., 2020)</td>
<td>1.7001</td>
<td>0.0110</td>
<td>20.4808</td>
<td>$2.4 \times 10^{-3}$</td>
</tr>
<tr>
<td>Traditional algorithms</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Horace’s method</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Park’s method</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wu’s method</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shah’s method</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Point set matching</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICP (Besl &amp; McKay, 1992)</td>
<td>1.7001</td>
<td>0.6183</td>
<td>20.4837</td>
<td>$2.4 \times 10^{-3}$</td>
</tr>
<tr>
<td>CPD (Myronenko &amp; Song, 2010)</td>
<td>60.357</td>
<td>2.3733</td>
<td>267.5175</td>
<td>$7.8 \times 10^{-1}$</td>
</tr>
</tbody>
</table>

![Fig. 8](image)

**Fig. 8.** Performance of HI-SO(3)R3 for different $a_k$ values and $a_p = \frac{1.5}{\text{tr} (\hat{M}) - \min M}$. (The darker the color is, the less the training time).

![Fig. 9](image)

**Fig. 9.** Average training time (s) of HI-SO(3)R3 with different values of $a_k$ and $a_p$.

It is clear that our proposed algorithm, in comparison to other state-of-the-art algorithms, offers a much better computation speed without compromising the accuracy of the results.

To further assess the performance of implemented methods, we applied the hand–eye calibration results (i.e., the estimated transformation matrix from the gripper frame to the camera frame, $\hat{G}^C_T$) to reconstruct a complete point cloud of a power drill from its partial point clouds (see Fig. 6). Point Cloud Reconstruction (also known as Point Cloud Stitching) is the process of constructing a complete point cloud of a scene/object through combining/stitching several partial point clouds belonging to the same scene/object. As shown in Fig. 6(a)(b), partial point clouds of a power drill were acquired with different camera poses using the Intel RealSense stereo camera. From $\hat{G}^C_T$ and $\hat{G}^G_T$ (computed from robot kinematics), the estimated transformation matrix from the robot base frame to the camera frame (i.e., $\hat{R}^C_T$) can be calculated as $\hat{R}^C_T = \hat{G}^C_T \hat{G}^G_T$. Using $\hat{R}^C_T$, we can convert the partial point clouds represented in different camera frames into the same robot base frame.

If $\hat{U}^G_T$ is estimated accurately, we can attain a more accurate point cloud with high quality expressed in the robot base frame. As shown in Fig. 6, the methods with lower RAE values perform better than those with higher RAE values.
To show the effect of the selection of tolerance and initial guess to HI-SO(3)R3 and ICP, additional experiments were conducted using HI-SO(3)R3 with the data acquired from the Intel RealSense stereo camera. The results are shown in Figs. 7, 8, and 9. Figs. 7 and 8 show that larger values of $\alpha_R$ and $\alpha_p$ result in faster convergence as long as these values satisfy the conditions described in (18) and (22). Fig. 9 shows the average training time of HI-SO(3)R3 with different choices of learning rates while achieving the same accuracy as that listed in Table 2. The infinite training time in Fig. 9 means HI-SO(3)R3 did not converge using the selected learning rate. As mentioned in Section 3.3, with slightly larger learning rates $\alpha_R, \alpha_p$, than the bounds given in (18) and (22), the estimated error may still converge to its minimum (see Figs. 7, 8, and 9), but the convergence is not guaranteed for other systems. Hence, $\alpha_R$ and $\alpha_p$ should be selected according to (18) and (22) when no prior information is available.

To show the effect of the selection of tolerance and initial guess, additional experiments were conducted using HI-SO(3)R3 and ICP with the data acquired using the Intel RealSense stereo camera. In these experiments, the maximum number of iterations for HI-SO(3)R3 and ICP were chosen to be $1 \times 10^4$. The learning rates in HI-SO(3)R3 were selected as $\alpha_R = \frac{1.5}{\text{Min}_R}$ and $\alpha_p = \frac{0.9}{n}$. During these experiments, the initial rotation matrix was obtained from various rotation angles (ranging from $0^\circ$ to $350^\circ$) around the rotation axis computed from the best calibration results of the previous tests. Similarly, the initial translation vector was obtained from various translation distances (ranging from $0\text{mm}$ to $500\text{mm}$).
from 0 to 1000 mm) along the translation direction computed from the best calibration results of the previous tests. The experiments regarding different tolerances and initial guesses were divided into two parts: (1) the experiments about various tolerances and initial rotation angles with the initial translation vector being $\Omega^0$, and (2) the experiments about various tolerances and initial translation distances with the initial rotation matrix being $I$. The results from these experiments are shown in Fig. 10. Although the accuracy of ICP was less sensitive to the selection of tolerance and initial guess, its training time was much longer than that of HI-SO(3)R3. On the other hand, the accuracy of HI-SO(3)R3 is sensitive to the selection of tolerance and the best result was obtained when the tolerance was chosen to be less or equal to $1 \times 10^{-5}$ for both the rotation matrix and translation vector. Hence the best value of tolerance for HI-SO(3)R3 is suggested to be $1 \times 10^{-5}$ to achieve the best accuracy and computational efficiency. As for the selection of the initial guess, however, no clear relation between the initial guess and the accuracy and efficiency of HI-SO(3)R3 can be deduced from Fig. 10. In practice, using singular value decomposition (SVD) is a common approach to estimate the rigid transformation matrix between two sets of points. When the initial guess was computed using SVD, the performances of HI-SO(3)R3 and ICP with the data acquired from the Intel RealSense stereo camera are listed in Table 3. During the tests with SVD initialization, the tolerance was set to $1 \times 10^{-5}$ for both the rotation matrix and the translation vector, and the learning rates of HI-SO(3)R3 were $\alpha_R = \frac{1.5}{(n(\lambda_1-\lambda_\text{min})^2}$ and $\alpha_p = \frac{0.9}{n}$. With SVD initialization, the initial guess is expected to be close to the optimal solution. In this case, the computational efficiency of HI-SO(3)R3 is even more superb compared to the ICP algorithm.

**Remark 5.** To achieve the best accuracy and computational efficiency, the suggested tolerance for HI-SO(3)R3 algorithm is $1 \times 10^{-5}$ and HI-SO(3)R3 algorithm is best to be initialized using the SVD method.

### 5. Conclusions

In this paper, we formulated and solved the hand–eye calibration problem as a problem of point set matching. In this light, we proposed a new algorithm on manifold $\text{SO}(3) \times \mathbb{R}^3$ with a nice decoupling property between the rotational estimation error and translational estimation error. As a result, the convergence speed of the algorithm was significantly increased. We called our algorithm HI-SO(3)R3 for its high convergence speed. The performance of HI-SO(3)R3 was evaluated and compared with some conventional algorithms for hand–eye calibration and some widely used point set matching algorithms. The results manifested that better accuracy in practice could be achieved by formulating the problem of hand–eye calibration as a problem of point set matching. HI-SO(3)R3 offers a superior and convenient alternative to conduct hand–eye calibration for robot manipulators. The experimental results also highlighted a drawback of the algorithm that the robot is required to physically move and make contact with the calibration object to attain the coordinates in the robot frame. Our future work will focus on eliminating this requirement and also solve the robot-world/hand–eye calibration problem (i.e., $AX = YB$).

---

**Table 3**

<table>
<thead>
<tr>
<th></th>
<th>HI-SO(3)R3</th>
<th>ICP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training time (s)</td>
<td>0.001044</td>
<td>0.4724</td>
</tr>
<tr>
<td>Ave. RAE (mm)</td>
<td>4.2517</td>
<td>4.2517</td>
</tr>
</tbody>
</table>

---

**Declaration of competing interest**

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

**References**


