Optimal least-squares solution to the hand-eye calibration problem

Amit Dekel Linus Härenstam-Nielsen Sergio Caccamo
Univrses
{amit.dekel, linus.arenstam, sergio.caccamo}@univrses.com

Abstract

We propose a least-squares formulation to the noisy hand-eye calibration problem using dual-quaternions, and introduce efficient algorithms to find the exact optimal solution, based on analytic properties of the problem, avoiding non-linear optimization. We further present simple analytic approximate solutions which provide remarkably good estimations compared to the exact solution. In addition, we show how to generalize our solution to account for a given extrinsic prior in the cost function. To the best of our knowledge our algorithm is the most efficient approach to optimally solve the hand-eye calibration problem.

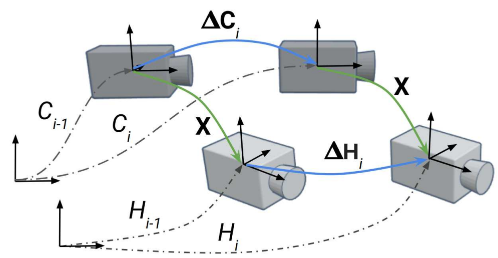


Figure 1: The hand-eye calibration problem for two rigidly attached cameras.

1. Introduction

Hand-eye calibration is a common problem in computer vision where one wishes to find the transformation between two rigidly attached frames. One popular formulation of the problem is in terms of the following equation

$$\Delta C \circ X = X \circ \Delta H, \tag{1}$$

where $\Delta C, \Delta H, X$ are $SE(3)$ elements and $\circ$ is the group multiplication operation [1]. ΔC and ΔH represent the sources relative poses (*e.g.* camera and robot hand) and X is the rigid transformation between the two frames.

In the literature one can find plenty of strategies to solve the problem for a given set of poses, as well as different representations and metrics being used, see [2, 3] for a recent review. Since the $SE(3)$ group is a semi-direct product of the special orthogonal group and the translational group ($SO(3) \ltimes T(3)$), starting with (1), one usually ends up with two equations which we will refer to as the rotational equation (the quotient by the normal subgroup) and the translational equation (the normal subgroup). In practice, due to noisy data these equations cannot be satisfied exactly, and a numerical minimization approach or an approximating procedures are often used.

Some authors approach the problem by solving the two equations separately by first minimizing a term related to the rotational equation and then substituting the solution in the translational equation and minimizing this term in turn, where a closed form solution can be found [4, 5, 6]. Others try to minimize both terms simultaneously.

In the later case, one approach is to perform a (constrained) nonlinear optimization procedure to minimize a cost function which takes into account the two terms (the rotational and translational parts) in some representation and specified metrics. This procedure is more costly in terms of computations, and requires a proper initial estimation in order to avoid local minima [6, 7, 8]. A linear simultaneous solution was presented in terms of dual-quaternions (or equivalently geometric algebra) [9], [10], which aims to solve (1) rather than a minimization problem, being valid for the noise free case, or requires filtering of the data.

As mentioned above, one can use different metrics for the minimization problem, usually related to the chosen representations or noise model assumptions. For the rotational part, the common (non-equivalent [11]) choices are the chordal, quaternionic or the geodesic metrics. Similarly, one can use different minimization terms for the translational part. Furthermore, rotation and translation terms have different units, and a proper relative weighting should be used. A natural choice would be the covariance of the measurements, which is not always available.

In this paper we address the problem of simultaneous minimization using the dual-quaternions (DQs) representation. The difficulties in solving the minimization problems analytically are due to nonlinear constraints. Using DQs, one can construct a quadratic cost function in terms of the DQ d.o.f. which are subject to nonlinear constraints, which inevitably make the problem highly nonlinear.

Our approach is to add the constraints as two Lagrange multiplier terms and to study their analytic properties. The two Lagrange multipliers are related by a polynomial $p(\lambda, \mu)$ of 8th-order in μ and 4th-order in λ, with some special properties. Over the reals, the polynomial defines four curves $\lambda_i(\mu)$, and the optimal solution corresponds to the saddle point $d\lambda/d\mu = 0$ of the smallest λ curve. Thus, our strategy is to efficiently find this special point in the Lagrange multipliers space. We propose several nonequivalent ways to find the optimal solutions as well as a hierarchy of analytic approximations which perform remarkably well.

We show explicitly how to extend our algorithms when adding a prior to the problem, allowing for a maximum a posteriori (MAP) analysis, or regularization in degenerate cases (*e.g.* planar motion).

We perform several experiments using synthetic and real data, and compare our algorithms to other available methods and to nonlinear optimization, showing that our algorithm indeed finds the optimal solution.

The paper is organized as follows: we initially establish the connection between the $SE(3)$ group and unit DQ. Then we formulate the hand-eye calibration problem using DQs, and introduce the minimization problem. Afterwards we discuss the properties of the cost function and present algorithms for solving the problem. Next, we extend the formulation to include a prior term. In the final sections, we present our experiments and comparisons to other existing methods in the literature, along with discussion and conclusions. Some technical details and more detailed experiment results are relegated to appendices.

2. SE(3) dual-quaternions representation

In this paper we use the dual-quaternion representation of the $SE(3)$ group. The group elements are constructed using dual-quaternions which are a sum of an ordinary quaternion and a dual part which is an ordinary quaternion multiplied by the dual-number ϵ, where $\epsilon^2 = 0$, see for example [12, 9].

$SE(3)$ can be represented by unit DQ, $Q = q + \epsilon q'$ with $Q \otimes Q^* = 1$, where $q, q' \in \mathbb{H}$, $\otimes$ is the quaternionic product, and * is the usual quaternion conjugation. As with ordinary quaternion double cover representation of representing $SO(3)$, we identify $Q \sim -Q$. The unit DQ constraint can be equivalently written as $|q| = 1$ and $q \cdot q' = 0$ where q and q' are treated as real four-dimensional vectors, and $\cdot$ is the dot product.

The explicit, a rotation element is given by $R_{(\hat{k},\theta)} = (\hat{k} \sin \frac{\theta}{2}, \cos \frac{\theta}{2})$ with $\hat{k}$ the rotation axis and θ the rotation angle, while a translation element by a vector $\vec{a}$ is given by $T_{\vec{a}} = (\vec{0}, 1) + \epsilon \frac{1}{2}(\vec{a}, 0)$. Our notation $(\vec{q}, q_0)$ refers to the imaginary and real parts of a quaternion respectively. Thus, a general $[R|T]$ transformation can be represented by

$$Q = T \otimes R = R + \epsilon \frac{1}{2}(\vec{a}, 0) \otimes R \equiv q + \epsilon q'. \qquad (2)$$

An equation concerning DQ can always be split in two equations, one for the "primal" part which is a pure quaternion, and one for the "dual" part which is the quaternion coefficient of ϵ.

3. Hand-eye calibration problem in terms of dual-quaternions

The hand-eye calibration problem can be formulated using the following set of equations

$$\Delta C_i \circ X = X \circ \Delta H_i, \quad i = 1, ..., N, \qquad (3)$$

where ΔC_i and ΔH_i are corresponding relative poses of two rigidly attached sources (see fig. 1), which from now on we shorten as C_i and H_i. Using the DQ representation the equation can be written as [9]

$$Q_{C_i} \otimes Q_X = Q_X \otimes Q_{H_i}, \quad i = 1, ..., N, \qquad (4)$$

and more explicitly as

$$\begin{aligned} &q_{C_i} \otimes q_X = q_X \otimes q_{H_i}, \quad i = 1, ..., N, \\ &q_{C_i} \otimes q'_X + q'_{C_i} \otimes q_X = q_X \otimes q'_{H_i} + q'_X \otimes q_{H_i}, \end{aligned} \qquad (5)$$

where we separated the primal and dual parts to two equations. Next we translate the quaternionic multiplications to matrix multiplications using $q \otimes p \equiv \mathbf{L}(q)p \equiv \mathbf{R}(p)q$, where q and p are treated as real four-dimensional vectors on the RHS and $\mathbf{L}, \mathbf{R}$ are 4×4 matrices, and we get

$$\begin{aligned} &\left(\mathbf{L}(q_{C_i}) - \mathbf{R}(q_{H_i})\right) q_X = 0, \quad i = 1, ..., N, \qquad (6) \\ &\left(\mathbf{L}(q'_{C_i}) - \mathbf{R}(q'_{H_i})\right) q_X + \left(\mathbf{L}(q_{C_i}) - \mathbf{R}(q_{H_i})\right) q'_X = 0. \end{aligned}$$

For simplicity of notation we define $A_i \equiv \mathbf{L}(q_{C_i}) - \mathbf{R}(q_{H_i})$, $B_i \equiv \mathbf{L}(q'_{C_i}) - \mathbf{R}(q'_{H_i})$ and $q \equiv q_X$, $q' \equiv q'_X$, so we have

$$A_i q = 0, \quad B_i q + A_i q' = 0, \quad i = 1, ..., N. \qquad (7)$$

As described in [9], in the noise free case where the equations can be solved exactly, the concatenated 8-dimensional vector (q, q') is in the null space of $\left(\begin{smallmatrix} A & 0 \\ B & A \end{smallmatrix}\right)$, where A and B are a stacks of A_i and B_i respectively. Moreover, the relative transformation can only change the screw axis from one frame to another so the angle and screw pitch d.o.f. are omitted in the treatment of [9], which also allows to reduce A_i and B_i to 3×4 matrices instead of 4×4 matrices.

In our analysis we prefer not to cancel the invariant screw parameters since we generally expect noisy data, and these parameters act as sort of relevant weights for our data, thus, we work with the full 4×4 matrices.

3.1. The minimization problem

Our next step is to define the minimization problem. Generally, in the presence of noise, the RHS of (7) will differ from zero. We define our minimization problem as the simultaneous minimization of the squared L_2 norm of residuals of the two equations in (7), namely

$$\arg\min_{q,q'} \sum_{i=1}^{N} \left(|A_i q|^2 + \alpha^2 |B_i q + A_i q'|^2\right),$$
$$\text{subject to: } |q| = 1, \quad q \cdot q' = 0, \tag{8}$$

where we introduced a fixed dimensionful parameter $\alpha \in \mathbb{R}$ with $1/\text{length}$ units, which may account for the different units of the two terms. For simplicity we will treat α as a fixed tunable parameter. More generally one could use a covariance matrix inside the norm based on specific measurements without changing our analysis.

Our choice for the cost function is natural given the dual-quaternion formulation, however one can find plenty of other non-equivalent choices in the literature using different metrics on $SE(3)$, see [2] for detailed discussion.

3.1.1 Minimization equations

In this section we present the minimization problem in more details, and set up the notation for the rest of the paper. First, we express the cost function (8) using two Lagrange multipliers corresponding to the two constraints,

$$\begin{aligned} L = & \sum_i \left(|A_i q|^2 + \alpha^2 |B_i q + A_i q'|^2\right) + \lambda(1 - q^2) - 2\mu q \cdot q' \\ = & \, q^T S q + q'^T M q' + 2 q^T W q' + \lambda(1 - q^2) - 2\mu q \cdot q', \end{aligned} \tag{9}$$

where λ and μ are Lagrange multipliers, and $S \equiv A^T A + \alpha^2 B^T B$, $M = \alpha^2 A^T A$ and $W \equiv \alpha^2 B^T A$. The minimization equations yield

$$\begin{aligned} \frac{1}{2}\frac{\partial L}{\partial q} &= S q + W q' - \lambda q - \mu q' = 0, \\ \frac{1}{2}\frac{\partial L}{\partial q'} &= M q' + W^T q - \mu q = 0, \end{aligned} \tag{10}$$

along with the two constraint equations. These equations imply that $L = \lambda$ on the solution.

Throughout the paper we assume that the matrix M is full rank, which is the case of interest where the input data is noisy, otherwise one can solve analytically along the lines of [9]. Thus, we have

$$q' = M^{-1}(\mu - W^T) q. \tag{11}$$

Plugging q' in the first equation yields

$$Z(\mu) q = \left(Z_0 + \mu Z_1 - \mu^2 Z_2\right) q = \lambda q, \tag{12}$$

where we introduce the notation

$$Z_0 \equiv S - W M^{-1} W^T, \quad Z_1 \equiv W M^{-1} + M^{-1} W^T,$$
$$Z_2 \equiv M^{-1}. \tag{13}$$

The first constraint equations $|q|^2 = 1$ can be trivially satisfied by normalizing the solution, while the second constraint $q \cdot q' = 0$ implies that

$$\mu = \frac{1}{2}\frac{q^T Z_1 q}{q^T Z_2 q}. \tag{14}$$

While we can easily solve (12) as an eigenvalue problem for a given μ, (14) will generally not be satisfied. Unfortunately plugging (14) in (12) results in a highly non-linear problem.

3.1.2 Properties of the minimization equations

In order to solve the problem it is useful to notice some properties of (12) and (14). By construction, the matrix $Z(\mu)$ is real and symmetric for any real μ, so its eigenvalues are real. We further notice that Z_2 is positive semi-definite (since it is the inverse of $A^T A$) and so is Z_0[1].

Considering these properties, the eigenvalue curves $\lambda_i(\mu)$ (each of the four roots ordered by their value) in (12) generally should not intersect as we change μ, as a consequence of the von Neumann-Wigner theorem[2] [13], [14].

A useful way to think of the problem is in terms of the Lagrange multipliers parameter space, given by a real algebraic curve defined by (12),

$$p(\lambda, \mu) = \det(Z(\mu) - \lambda) = \sum_{m=0}^{4} \lambda^m c_{8-2m}(\mu) = 0, \tag{15}$$

which is a 4-th degree polynomial in λ and 8-th degree in μ, and $c_n(\mu)$ is an n-th degree real polynomial in μ (see figure 2). The functions $\lambda_i(\mu)$ can be found analytically, but their explicit expressions are not very illuminating.

The constraint $q(\mu) \cdot q'(\mu) = 0$ dictates which points on the curves $\lambda_i(\mu)$ correspond to the solution. A key observation is that these are the extrema points of $\lambda_i(\mu)$. To show this we use the fact that $\lambda = q^T Z(\mu) q$, so

$$\begin{aligned} \left.\frac{\partial \lambda}{\partial \mu}\right|_{\text{sol}} &= 2\dot{q}^T Z(\mu) q + q^T \dot{Z}(\mu) q \\ &= 2\lambda \dot{q}^T q - 2 q^T M^{-1}(\mu - W^T) q = -2 q^T q', \end{aligned} \tag{16}$$

where we used (12) so we are on the solution, and $q^2 = 1$ which implies $\dot{q}^T q = 0$, and a dot denotes a μ derivative.

[1] Notice that $Z_0 = A^T A + \alpha^2 B^T C B$, where $C \equiv (1_{4N \times 4N} - A(A^T A)^{-1} A^T)$ is positive semi-definite having four zero eigenvalues and the rest are ones. The four zero eigenvalues correspond to the columns of A (notice that $(A^T A)^{-1} A^T$ is the pseudo-inverse of A), and then since the rank of $A(A^T A)^{-1} A^T$ is 4, the rest of the eigenvalues will be one (corresponding to the vectors in the null space of C).

[2] The probability to get an eigenvalue intersection for generic matrices (due to noise) in our problem when changing μ is practically zero. We also observed this phenomena in our experiments, see fig. 2c.

Thus, one can solve the problem by finding the extrema points of $\lambda(\mu)$, which is equivalent to finding the real intersection points of the polynomials $p(\lambda, \mu)$ and $\partial_\mu p(\lambda, \mu)$, which in turn is equivalent to finding the multiplicities of the 8-th degree polynomial in μ, as a function of its λ dependent coefficients. In either case, this is equivalent to the vanishing of the determinant of the Sylvester matrix (the resultant) of the two polynomials $p(\lambda, \mu)$ and $\partial_\mu p(\lambda, \mu)$ w.r.t. λ or μ. In both cases the result is a 28-th degree polynomial in one variable. Thus, the hand-eye calibration problem can be optimally solved by finding the smallest real positive λ root of this polynomial.

However, considering further properties of the problem allows to find the solution in more efficient ways. We notice that for large $|\mu|$ (12) and (11) become

$$-\mu^2 Z_2 q \simeq \lambda q, \quad q' \simeq \mu Z_2 q, \tag{17}$$

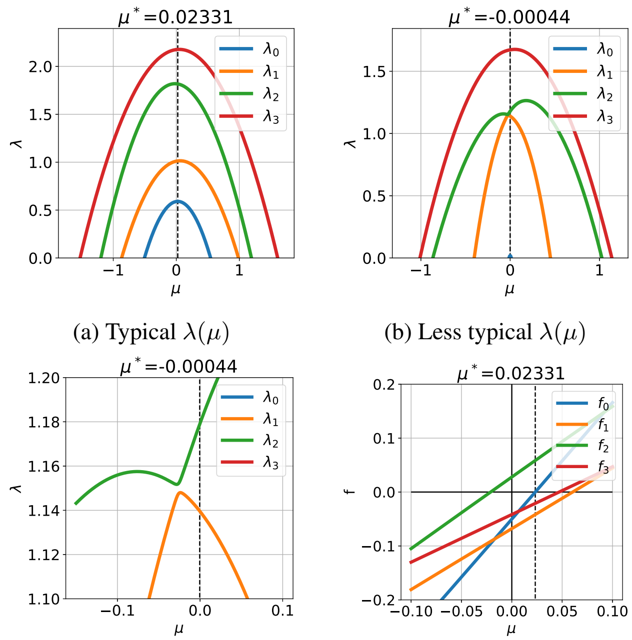
so $\lambda_i(|\mu|) \simeq -\mu^2 \xi_i$ where $\xi_i \geq 0$ are eigenvalues of Z_2. Similarly, $Z(\mu = 0) = Z_0$ which is PSD, so $\lambda_i(\mu = 0) \geq 0$.

Combining these two observations together with the von Neumann-Wigner non-crossing theorem, each of the four eigenvalues $\lambda_i(\mu)$ must cross the μ axis at least twice. However, the number of solutions to $Z(\mu)q = 0$ in terms of μ is at most eight, so we conclude that every eigenvalue $\lambda_i(\mu)$ crosses the μ axis exactly twice, say at $\mu_1 < 0 < \mu_2$, and then goes asymptotically quadratically to $-\infty$ for $\mu \gg \mu_2$ and $\mu \ll \mu_1$ (there could be cases where the two points unite and we get one solution at $\mu = 0$, which corresponds to the noise free case).

Next, we change our point of view and consider $\mu(\lambda)$. We already established that the number of real solutions to $\mu(\lambda = 0)$ must be eight, which is the maximum number of solutions. As we increase λ, the number of real solutions can either stay constant or drop by an even integer number[3]. The first jump from eight to six solutions corresponds to the first maximum of $\lambda(\mu)$, which is the minimal cost solution to our problem. Furthermore, this must correspond to the maximum of $\lambda_{i=0}(\mu)$, since otherwise the number of real solutions to the μ polynomial will exceed the maximum number (eight). Multiple extrema are allowed only for higher $\lambda_{i>0}$ functions, see 2b. Similarly we can look for root multiplicities in terms of the discriminant, since the extrema of $\lambda(\mu)$ correspond to such points.

To summarize, we showed that the function $\lambda_0(\mu)$, the smallest eigenvalue of (12), has one positive maximum, and asymptotically decays quadratically in μ. Finding the optimal solution is equivalent to finding the maximum of this function which is a convex problem.

[3]For a given λ we can rewrite the eigenvalue problem in terms of μ as an 8×8 eigenvalue problem, with a non-symmetric matrix, so in general the properties of μ are less constrained and μ can become complex.



(a) Typical $\lambda(\mu)$ (b) Less typical $\lambda(\mu)$

(c) Less typical $\lambda(\mu)$ (zoomed) (d) Typical $f(\mu)$

Figure 2: Lagrange multipliers space. Subscripts in $\lambda_i(\mu)$ and $f_i(\mu)$ indicate the i-th EV for a given μ in ascending order. The optimal μ is denoted by μ^* and the dashed black line. In (a) and (b) the optimal solution corresponds to the maximum of the blue line. In (b) the number of real roots starts at 8, reduce to 6, 4 and then grows to 6 and reduces to 4, 2 and finally to 0. We also see the level repulsion effect where the eigenvalues almost intersect but eventually repel each other, see (c). In (d) solution μ^* corresponds to the intersection of f_0 with the μ-axis (same data as in (a)).

3.2. Solving the minimization problem

Based on the discussion above, we introduce various strategies to solve the minimization equations, presenting both exact and approximate solution.

3.2.1 Optimal 1D line search (DQOpt)

This is the simplest algorithm we present which yields the optimal solution. We first define the function

$$f_0(\mu) \equiv q_0(\mu) \cdot q_0'(\mu) = -\frac{1}{2}\frac{d\lambda_0(\mu)}{d\mu}, \tag{18}$$

where q_0 is the smallest eigenvalue of $Z(\mu)$, and q_0' is computed using (11). As explained above, this is a monotonic function with a unique root, see figure 2d. To find the optimal solution we first find the root of this function, $f_0(\mu_*) = 0$, and solve for q as the eigenvector corresponding to the smallest eigenvalue of $Z(\mu_*)$ and for q' using (11). Alternatively, one can look for the maximum of $\lambda_0(\mu)$ directly which correspond to the same solutions, see 2a.

We further derived alternative algorithms to arrive at the optimal solution, again, using only a 1D line search based on resultants and roots counting. These algorithms yield the same optimal solution, but are slower to evaluate. Since they are conceptually very different and provide a different view on the problem we believe it is interesting to include them in this work. However, since we shall only use the above optimal solution throughout the paper we relegate their description to appendix B.

3.2.2 Two steps (DQ2steps)

The two steps approximation, solving first the rotation equation, takes a very simple form in our formulation. One first solves for q for finding the eigenvector corresponding to the smallest eigenvalue in

$$Mq = \lambda_0 q, \tag{19}$$

and the corresponding q' is given by

$$q' = M^{-1}\left(\frac{1}{2}\frac{q^T Z_1 q}{q^T Z_2 q} - W^T\right) q. \tag{20}$$

3.2.3 Convex relaxation (DQCovRlx)

Here we approximate the solution while relaxing the $q \cdot q' = 0$ constraint, and then project the solution to the non-convex constraint space using (14), thus we solve

$$Z_0 q = \lambda_0 q, \tag{21}$$

and the corresponding q' is given by (20) again. We can also get a nice expression bounding the gap,

$$\Delta\lambda = \lambda^c - \lambda_0 = \frac{1}{4}\frac{(q^T Z_1 q)^2}{q^T Z_2 q}, \tag{22}$$

where λ^c is the constrained solution. Namely, the true cost for the non-convex problem, λ^*, satisfies $\lambda_0 \leq \lambda^* \leq \lambda^c$.

3.2.4 Second order approximation (DQ2ndOrd)

Starting with the relaxed solution (21), corresponding to $\mu = 0$ (which is true in the noise free case), we can have an analytic expansion in small μ. A detailed derivation of a recursive formula for the dual-quaternion solution to any order in μ is given in appendix A. The resulting second order approximation for yields

$$\begin{aligned}\mu^*_{(2)} =& \frac{1}{2}\frac{Z_1^{00}}{Z_2^{00} - \sum_a \frac{(Z_1^{a0})^2}{\lambda_{0a}}},\\ q_{(2)} =& q_0 + \mu^*_{(2)} \sum_a \frac{Z_1^{a0}}{\lambda_{0a}} q_a + \mu^{*2}_{(2)}\Bigg(-\frac{1}{2} q_0 \sum_a \left(\frac{Z_1^{a0}}{\lambda_{0a}}\right)^2 \\ &+ \sum_a \frac{\sum_b \frac{Z_1^{b0} Z_1^{ab}}{\lambda_{0b}} - Z_2^{a0} - \frac{Z_1^{00} Z_1^{a0}}{\lambda_{0a}}}{\lambda_{0a}} q_a \Bigg),\end{aligned} \tag{23}$$

where the q's and λ's are defined by $Z_0 q_a = \lambda_a q_a$. We also introduced the short-hand notation $Z_i^{ab} \equiv q_a^T Z_i q_b$ and $\lambda_{0a} \equiv \lambda_0 - \lambda_a$ and all the sums are running over $a = 1, 2, 3$. After normalizing $q_{(2)}$ we solve for q' using (20).

Since μ is not a dimensionless parameter, one might prefer to expand in λ which is dimensionless. We can start with $\lambda_0 = 0$ which is the solution in the noise free case. However, this leads to an 8-th degree polynomial in μ when coming to solve (12) which complicates a bit the procedure. Instead, we can expand in λ around the relaxed ($\mu = 0$) solution, so $\lambda = \lambda_0 + \Delta\lambda$ and the expansion is in $\Delta\lambda$, and λ_0 corresponds to the smallest eigenvalue of Z_0. The second order expansions allowing to find q is slightly more elaborated compared to the previous one and can be found in appendix A. Then as in the previous approximation we solve for q' using (20).

The results of this expansion are comparable to the previous one in our experiments, though they are not equivalent (both giving very accurate results).

3.2.5 Iterative solution (DQItr)

Defining the function

$$\tilde{\mu}(\mu) = \frac{1}{2}\frac{q(\mu)^T Z_1 q(\mu)}{q(\mu)^T Z_2 q(\mu)}, \tag{24}$$

where $q(\mu)$ is the eigenvector corresponding to the smallest eigenvalue of (12), the solution to the problem is the fixed point, namely $\tilde{\mu}(\mu) = \mu$. Since $\tilde{\mu}(\mu)$ is bounded as explained in the previous section, iteratively estimating $\tilde{\mu}$ and plugging the value back convergence to the solution. In practice this procedure converges very quickly, though it is unstable in the noise free case.

3.3. Adding a prior

The hand-eye calibration problem is not always well posed, for example in the case of planar motion some d.o.f. are not fixed, and might require a regularization term for a unique solution. Furthermore, constructing the problem from a statistical model perspective, as a MAP probability estimation requires a prior. Thus, it is useful to incorporate a prior term in our formulation.

Let us assume we have some prior knowledge regarding the hand-eye calibration, namely that the unit dual-quaternion representing the calibration is close to some given $\hat{Q}$. We define $Q = \hat{Q} \otimes \delta Q$, such that $\delta Q = \delta q + \epsilon \delta q'$ is the deviation from the prior, which is expected to be small, namely $\delta Q \simeq 1$, or equivalently $\delta q \simeq 1$ and $\delta q' \simeq 0$. Thus, we introduce the prior by penalizing large δQ

$$L_{\text{prior}}(q, q') = L(q, q') + a(1 - \delta q)^2 + b\delta q'^2, \tag{25}$$

where $a, b > 0$ are constant weight parameters (notice that q and δq are not independent). The rotation term has a linear dependence on δq. This term complicates our analysis, where the eigenvalue problem needs to be promoted to an 8×8 instead of the 4×4 problem we had before, and our matrices lose some of their nice properties. Nonetheless, this approach is valid and yields good results.

However, we can avoid these issues by noting that $|\delta q| = 1$, $\delta q \simeq 1$ is equivalent to the imaginary part being small. So we instead modify (25) to

$$L_{\text{prior}} = L + a\delta q^T G \delta q + b\delta q'^2, \quad G = \text{diag}(1,1,1,0). \quad (26)$$

We can minimize this cost in the same way we minimize (9) by noting that

$$\delta Q = \hat{Q}^* \otimes Q = \mathbf{L}(\hat{q}^*)q + \epsilon(\mathbf{L}(\hat{q}'^*)q + \mathbf{L}(\hat{q}^*)q'). \quad (27)$$

Thus, the two new terms are given by

$$\begin{aligned} \delta q^T G \delta q &= q^T \mathbf{L}^{*T} G \mathbf{L}^* q, \\ \delta q'^2 &= q^T \mathbf{L}'^{*T} \mathbf{L}'^* q + 2q^T \mathbf{L}'^{*T} \mathbf{L}^* q' + q'^T q' \\ &= |\hat{q}'|^2 q^T q + 2q^T \tilde{W} q' + q'^T q', \end{aligned} \quad (28)$$

where $\tilde{W} = -\tilde{W}^T = \mathbf{L}'^{*T} \mathbf{L}^* = \mathbf{L}(\hat{t})$. Because of our constraints, the term $|\hat{q}'|^2 q^T q$ is a constant which does not change the cost function, so we can ignore it (notice that by doing that the cost function is not guaranteed to be positive anymore, but is bounded from below by $-b|\hat{q}'|^2$). Thus, we can solve (26) in the same way as (9) by identifying

$$\begin{aligned} S &\to S + a\mathbf{L}^{*T} G \mathbf{L}^*, \\ W &\to W + b\tilde{W}, \\ M &\to M + b\mathbf{1}_{4\times 4}. \end{aligned} \quad (29)$$

Notice that these modifications to (9) do not change the matrix properties discussed above regarding semi-positive definiteness and symmetries.

4. Experiments

In this work we perform two kinds of experiments. First we compare our optimal algorithm results with a nonlinear optimization implementation to show that our algorithm results in the optimal solution. Afterwards we compare the performance of our algorithms with other algorithms, on real and synthetic datasets.

Throughout this section we use the algorithms abbreviations introduced in sec. 3, with the addition of **Dan** [9], as well as **ChVecOpt** and **QuatVecOpt** which minimize

$$\begin{aligned} &\textstyle\sum_i \left(|R_C^i R - R R_H^i|^2 + \alpha^2 |(R_C^i - 1)T - R T_H^i + T_C^i|^2\right), \\ &\textstyle\sum_i \left(|q_C^i q - q q_H^i|^2 + \alpha^2 |(R_C^i - 1)T - R T_H^i + T_C^i|^2\right) \end{aligned} \quad (30)$$

w.r.t. R, T and q, T using nonlinear optimization respectively. Finally, **DQNLOpt** and **DQNLOptRnd** represent nonlinear optimization of our cost function (9) initialized with **DQOpt** and randomly respectively.

All the algorithms are implemented using python with numpy and scipy optimize packages for 1D root finding.

4.1. Synthetic experiments

Our synthetic dataset consists of three scenarios:
- **Random** - uniformly sampled from $SO(3) \times E(3)$.
- **Line** - straight motion + small 3d perturbations.
- **Circle** - 2d circular motion + small 3d perturbations.

The relative poses are related by the extrinsic transformation with addition of noise. For the translations we use multivariate Gaussian noise, while for the rotations we use a uniform distribution on S^2 for the axis and a Gaussian distribution for the angle (more details can be found in appendix D).

4.2. Real data experiments

We recorded poses from two rigidly attached ZED cameras running the built-in visual odometry. The cameras were moved by hand, creating a general 3d motion scenario and three approximately planar scenarios: general, straight, and circular. The collected data is significantly noisy due to the inaccuracy of the tracking algorithm and time synchronization issues. More details on the data acquisition process and pre-filtering can be found in appendix D.

4.3. Weighting factor

The proper choice of the weighting factor α, which was introduced in sec. 3.1, depends on the noise model. In order to have a fair comparison we add the α parameter to all the algorithm implementations, see (30), and by scaling the translation part by α for **Dan**. We introduce α_{best} for rotation and translation, which are the values that give the best results in terms of the rotation and translation mean error respectively w.r.t. the ground-truth. Note that these values are not necessarily the same for all algorithms and even for the same dataset. In this way we treat all the algorithms in a fair way in contrast to setting one value which might benefit certain algorithms more than others.

Relative cost difference $\frac{alg-DQOpt}{alg+DQOpt}$				
alg	mean	std	min	max
DQNLOpt	6.3×10^{-18}	5.8×10^{-16}	-3.0×10^{-15}	2.8×10^{-15}
DQNLOptRnd	6.8×10^{-2}	1×2.1^{-1}	4.3×10^{-12}	8.9×10^{-1}
DQ2ndOrd	6.6×10^{-10}	5.4×10^{-9}	-2.3×10^{-15}	1.3×10^{-7}
DQConvRlx	2.7×10^{-5}	7.0×10^{-5}	2.15×10^{-12}	5.3×10^{-4}
DQ2Steps	8.4×10^{-3}	1.6×10^{-2}	2.4×10^{-5}	9.7×10^{-2}

Timing in μsec				
alg	mean	std	min	max
DQOpt	299	92	158	1199
DQ2ndOrd	155	53	98	818
Dan	92	33	61	1200
DQConvRlx	77	36	43	896
DQ2Steps	63	23	42	1242

Table 1: Upper table: Results for the (signed) relative difference (averaged over 2450 real data runs) between **DQOpt** and our other algorithms. Positive values imply that the competing algorithm has a higher cost function. Lower table: Algorithm's timing averaged over 24500 real data runs.

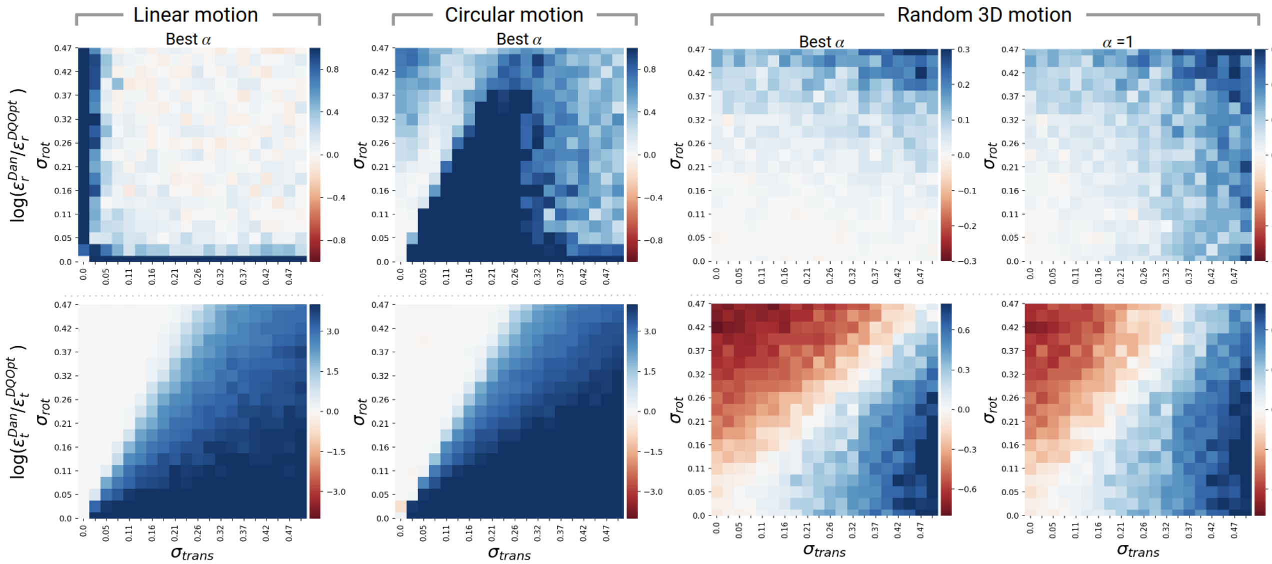


Figure 3: **Synthetic experiments**. We compare between **Dan** and **DQOpt** on the three different motions by showing the error response of the solvers when the relative poses are subject to increasing rotational and translational noise (sec. 4.1). The heat maps show the mean ratio (60 iterations) of the error responses (first row - rotation, second row - translation). A blue (red) color indicates that **DQOpt** has e^{value} times lower (higher) error compared to **Dan**. It is possible to notice how **DQOpt** dramatically outperform **Dan** on degenerate motions (linear and circular). For the 3D case (column 3) both **DQOpt** and **Dan** show high accuracy (see table 2) and the error ratio loses its significance (*i.e.* they both perform well). In the last column, we compare the two solvers for a $\alpha = 1$.

4.4. Verifying optimality

In order to verify that our algorithm finds the optimal solution we compare its cost with the cost of a nonlinear optimization implementation (using scipy's least_squares), designed to minimize the same cost function. When running the nonlinear optimizer we initialized it both with our solution and randomly to allow a search for a different minimum in case our algorithm ended up in a local minimum. In order for our algorithm to be optimal, the nonlinear optimizer should not change the solution when initialized with our solution, and might find higher cost solutions when initialized randomly. The results are summarized in table 1, where our algorithm always returns a lower cost up to numerical precision. We further compare with our approximate methods to show how close they get to the global minimum.

4.5. Timing

We compare the run time of our optimal solution implementation with the approximate solutions and **Dan** [9], which serves as a benchmark, as it is well known to be very efficient. The results are shown in lower table 1, ordered by the mean run time. Our optimal algorithm is found to be $\times 3$ slower than **Dan**. Two of the approximate algorithms are a bit faster, while approximating very well the optimal solution, see upper table 1. The nonlinear optimization algorithms are much slower, but we do not report their performance since we did not optimize their implementation.

5. Conclusions

In this paper we introduced a novel approach to find an optimal solution to the hand-eye calibration problem, based on a dual-quaternions formulation. We showed that our algorithm is guaranteed to find the global minimum by only using a 1D line search of a convex function. This provides a big advantage over nonlinear optimization methods which are less efficient and are sensitive to local minima.

We further introduced a hierarchy of efficient analytic approximate solutions which provide remarkably close approximations to the optimal solution. We showed that our optimal solution is very efficient being only 3 times slower than the analytic solution by Daniilidis [9], and that some of our approximation are faster while providing dramatically better estimations for ill-posed cases (comparable with the optimal solution).

We compared the performance of our algorithms with the well known analytic solution by Daniilidis and nonlinear optimization of other common formulations of the problem with different cost functions, using real and synthetic noisy datasets. We find that all the methods which provide an optimal solution are comparable in terms of accuracy, see table 2. However, for the real data we find that our methods consistently obtain slightly lower median errors in terms of both rotations and translations. The two-steps method degrades in accuracy for almost planar motions due to the ill-posedness of the rotation equation. Daniilidis's algorithm which does

		3d motion			Rotation			Planar			Linear		
α	Algorithm	ε_{r} (deg)	ε_{t} (cm)	α	ε_{r} (deg)	ε_{t} (cm)	α	ε_{r} (deg)	ε_{t} (cm)	α	ε_{r} (deg)	ε_{t} (cm)	α
best rotation α	DQOpt	$2.03\pm^{0.73}_{0.60}$	$5.32\pm^{1.95}_{1.39}$	1.47	$\mathbf{6.91}\pm^{\mathbf{5.45}}_{\mathbf{3.47}}$	$\mathbf{5.05}\pm^{\mathbf{3.08}}_{\mathbf{1.56}}$	0.20	$2.55\pm^{0.44}_{0.41}$	$17.7\pm^{31.4}_{9.61}$	0.37	$\mathbf{2.89}\pm^{\mathbf{0.22}}_{\mathbf{0.17}}$	$26.3\pm^{10.5}_{9.03}$	1.14
	DQ2ndOrd	$2.03\pm^{0.73}_{0.60}$	$5.32\pm^{1.95}_{1.39}$	1.47	$\mathbf{6.91}\pm^{\mathbf{5.45}}_{\mathbf{3.47}}$	$\mathbf{5.05}\pm^{\mathbf{3.08}}_{\mathbf{1.56}}$	0.20	$2.55\pm^{0.44}_{0.41}$	$17.7\pm^{31.4}_{9.61}$	0.37	$\mathbf{2.89}\pm^{\mathbf{0.22}}_{\mathbf{0.17}}$	$26.3\pm^{10.5}_{9.03}$	1.14
	DQConvRlx	$\mathbf{2.00}\pm^{\mathbf{0.76}}_{\mathbf{0.58}}$	$5.32\pm^{1.95}_{1.40}$	1.47	$6.95\pm^{5.39}_{3.50}$	$\mathbf{5.05}\pm^{\mathbf{3.10}}_{\mathbf{1.56}}$	0.20	$\mathbf{2.53}\pm^{\mathbf{0.77}}_{\mathbf{0.51}}$	$\mathbf{17.2}\pm^{\mathbf{27.3}}_{\mathbf{9.64}}$	0.17	$\mathbf{2.89}\pm^{\mathbf{0.23}}_{\mathbf{0.18}}$	$\mathbf{25.9}\pm^{\mathbf{10.8}}_{\mathbf{9.01}}$	1.04
	DQ2Steps	$3.26\pm^{1.24}_{0.99}$	$\mathbf{5.27}\pm^{\mathbf{2.05}}_{\mathbf{1.52}}$	-	$17.0\pm^{13.2}_{9.09}$	$7.59\pm^{4.12}_{2.86}$	-	$7.17\pm^{6.77}_{3.75}$	$19.5\pm^{23.1}_{10.1}$	-	$9.46\pm^{8.92}_{3.20}$	$43.9\pm^{45.8}_{17.4}$	-
	Dan [9]	$\mathbf{2.00}\pm^{\mathbf{0.75}}_{\mathbf{0.60}}$	$5.45\pm^{2.14}_{1.54}$	1.47	$11.4\pm^{9.91}_{4.99}$	$27.4\pm^{73.5}_{20.4}$	17.8	$4.01\pm^{4.20}_{1.20}$	$565.\pm^{1132}_{336.}$	1.90	$18.3\pm^{26.5}_{10.7}$	$5801\pm^{4347}_{2592}$	2.26
	ChVecOpt	$2.08\pm^{0.74}_{0.63}$	$5.96\pm^{1.79}_{1.55}$	2.47	$7.22\pm^{5.00}_{3.72}$	$28.0\pm^{18.0}_{14.8}$	0.31	$2.75\pm^{0.44}_{0.44}$	$45.3\pm^{22.4}_{22.1}$	4.13	$2.95\pm^{0.25}_{0.19}$	$39.6\pm^{23.6}_{15.9}$	1.47
	QuatVecOpt	$2.04\pm^{0.81}_{0.56}$	$5.71\pm^{2.06}_{1.47}$	0.81	$7.37\pm^{5.05}_{4.06}$	$16.9\pm^{11.0}_{8.16}$	0.12	$2.69\pm^{0.41}_{0.37}$	$45.8\pm^{23.0}_{24.4}$	1.00	$2.95\pm^{0.27}_{0.18}$	$38.8\pm^{25.8}_{16.1}$	0.48
best translation α	DQOpt	$3.21\pm^{1.16}_{0.99}$	$5.26\pm^{1.99}_{1.46}$	0.03	$7.75\pm^{5.27}_{3.98}$	$\mathbf{5.31}\pm^{\mathbf{2.77}}_{\mathbf{1.80}}$	0.22	$3.84\pm^{2.92}_{1.89}$	$18.5\pm^{21.1}_{10.3}$	0.05	$3.37\pm^{0.35}_{0.29}$	$23.4\pm^{10.4}_{7.77}$	17.8
	DQ2ndOrd	$3.21\pm^{1.16}_{0.99}$	$5.26\pm^{1.99}_{1.46}$	0.03	$7.75\pm^{5.27}_{3.98}$	$\mathbf{5.31}\pm^{\mathbf{2.77}}_{\mathbf{1.80}}$	0.22	$3.84\pm^{2.92}_{1.89}$	$18.5\pm^{21.1}_{10.3}$	0.05	$3.37\pm^{0.35}_{0.29}$	$23.4\pm^{10.4}_{7.77}$	17.8
	DQConvRlx	$3.21\pm^{1.16}_{0.99}$	$5.26\pm^{1.99}_{1.46}$	0.03	$\mathbf{7.72}\pm^{\mathbf{5.38}}_{\mathbf{3.94}}$	$5.34\pm^{2.74}_{1.82}$	0.22	$\mathbf{3.83}\pm^{\mathbf{2.89}}_{\mathbf{1.89}}$	$18.4\pm^{21.1}_{10.3}$	0.05	$\mathbf{3.35}\pm^{\mathbf{0.36}}_{\mathbf{0.28}}$	$\mathbf{23.3}\pm^{\mathbf{10.3}}_{\mathbf{7.75}}$	17.8
	DQ2Steps	$3.26\pm^{1.24}_{0.99}$	$5.27\pm^{2.05}_{1.52}$	-	$17.0\pm^{13.2}_{9.09}$	$7.59\pm^{4.12}_{2.86}$	-	$7.17\pm^{6.77}_{3.75}$	$19.5\pm^{23.1}_{10.1}$	-	$9.46\pm^{8.92}_{3.20}$	$43.9\pm^{45.8}_{17.4}$	-
	Dan [9]	$\mathbf{2.36}\pm^{\mathbf{0.94}}_{\mathbf{0.72}}$	$\mathbf{5.14}\pm^{\mathbf{2.07}}_{\mathbf{1.43}}$	3.19	$10.7\pm^{11.2}_{4.72}$	$20.2\pm^{55.2}_{13.9}$	32.5	$4.19\pm^{5.62}_{1.18}$	$270.\pm^{445.}_{141.}$	42.1	$19.3\pm^{27.8}_{11.5}$	$5208\pm^{3804}_{2110}$	25.1
	ChVecOpt	$3.20\pm^{1.14}_{0.98}$	$5.88\pm^{1.84}_{1.67}$	0.03	$15.0\pm^{12.9}_{8.79}$	$15.9\pm^{8.88}_{6.63}$	0.03	$7.21\pm^{6.77}_{3.71}$	$18.5\pm^{15.2}_{8.90}$	0.01	$5.74\pm^{1.49}_{1.59}$	$24.6\pm^{9.79}_{7.43}$	0.07
	QuatVecOpt	$3.17\pm^{1.14}_{1.09}$	$5.69\pm^{1.93}_{1.65}$	0.13	$9.82\pm^{8.03}_{5.64}$	$14.8\pm^{9.11}_{6.80}$	0.06	$6.45\pm^{5.12}_{3.29}$	$19.9\pm^{13.5}_{8.62}$	0.01	$4.94\pm^{1.47}_{1.27}$	$24.2\pm^{8.55}_{7.08}$	0.07

		Random			Circular			Linear		
α	Algorithm	ε_{r} (deg)	ε_{t} (cm)	α	ε_{r} (deg)	ε_{t} (cm)	α	ε_{r} (deg)	ε_{t} (cm)	α
best rotation α	DQOpt	$0.0523\pm^{0.0162}_{0.0156}$	$0.1857\pm^{0.0487}_{0.0563}$	0.26	$6.29\pm^{2.17}_{2.02}$	$\mathbf{42.5}\pm^{\mathbf{8.92}}_{\mathbf{9.09}}$	0.57	$8.31\pm^{2.63}_{2.42}$	$\mathbf{45.3}\pm^{\mathbf{8.94}}_{\mathbf{8.33}}$	0.62
	DQ2ndOrd	$0.0523\pm^{0.0162}_{0.0156}$	$0.1857\pm^{0.0487}_{0.0563}$	0.26	$6.29\pm^{2.17}_{2.02}$	$\mathbf{42.5}\pm^{\mathbf{8.92}}_{\mathbf{9.09}}$	0.57	$8.31\pm^{2.63}_{2.42}$	$\mathbf{45.3}\pm^{\mathbf{8.94}}_{\mathbf{8.33}}$	0.62
	DQConvRlx	$0.0523\pm^{0.0162}_{0.0156}$	$0.1858\pm^{0.0487}_{0.0564}$	0.26	$\mathbf{6.25}\pm^{\mathbf{2.22}}_{\mathbf{1.98}}$	$\mathbf{42.5}\pm^{\mathbf{8.91}}_{\mathbf{9.03}}$	0.57	$\mathbf{8.25}\pm^{\mathbf{2.66}}_{\mathbf{2.34}}$	$\mathbf{45.3}\pm^{\mathbf{8.98}}_{\mathbf{8.24}}$	0.62
	DQ2Steps	$0.0532\pm^{0.0171}_{0.0152}$	$0.1846\pm^{0.0591}_{0.0526}$	-	$9.02\pm^{3.24}_{2.86}$	$42.9\pm^{8.91}_{8.90}$	-	$12.9\pm^{4.30}_{3.72}$	$46.2\pm^{8.47}_{8.34}$	-
	Dan [9]	$0.0524\pm^{0.0161}_{0.0157}$	$0.1889\pm^{0.0516}_{0.0548}$	0.26	$17.0\pm^{7.93}_{5.73}$	$347.\pm^{139.}_{98.8}$	42.1	$21.9\pm^{12.1}_{7.22}$	$499.\pm^{242.}_{139.}$	29.9
	ChVecOpt	$\mathbf{0.0491}\pm^{\mathbf{0.0170}}_{\mathbf{0.0140}}$	$0.1783\pm^{0.0521}_{0.0504}$	1.04	$6.43\pm^{2.53}_{1.93}$	$45.6\pm^{9.01}_{9.04}$	0.57	$8.65\pm^{2.54}_{2.38}$	$51.3\pm^{8.93}_{8.32}$	1.00
	QuatVecOpt	$0.0500\pm^{0.0148}_{0.0151}$	$\mathbf{0.1753}\pm^{\mathbf{0.0503}}_{\mathbf{0.0508}}$	0.26	$6.60\pm^{2.16}_{2.07}$	$45.7\pm^{9.52}_{8.19}$	0.26	$8.31\pm^{2.85}_{2.47}$	$50.5\pm^{9.07}_{8.55}$	0.34
best translation α	DQOpt	$0.0527\pm^{0.0171}_{0.0148}$	$0.1786\pm^{0.0555}_{0.0526}$	0.24	$6.56\pm^{2.13}_{2.32}$	$40.9\pm^{10.1}_{8.18}$	0.62	$\mathbf{9.56}\pm^{\mathbf{3.03}}_{\mathbf{2.90}}$	$45.0\pm^{8.34}_{7.71}$	1.14
	DQ2ndOrd	$0.0527\pm^{0.0171}_{0.0148}$	$0.1786\pm^{0.0555}_{0.0526}$	0.24	$6.56\pm^{2.13}_{2.32}$	$40.9\pm^{10.1}_{8.18}$	0.62	$\mathbf{9.56}\pm^{\mathbf{3.03}}_{\mathbf{2.90}}$	$45.0\pm^{8.35}_{7.71}$	1.14
	DQConvRlx	$0.0527\pm^{0.0171}_{0.0148}$	$0.1786\pm^{0.0554}_{0.0525}$	0.24	$\mathbf{6.55}\pm^{\mathbf{2.12}}_{\mathbf{2.40}}$	$40.9\pm^{10.0}_{8.21}$	0.62	$\mathbf{9.56}\pm^{\mathbf{3.03}}_{\mathbf{2.85}}$	$45.0\pm^{8.50}_{7.51}$	1.14
	DQ2Steps	$0.0532\pm^{0.0171}_{0.0152}$	$0.1846\pm^{0.0591}_{0.0526}$	-	$9.02\pm^{3.24}_{2.86}$	$42.9\pm^{8.91}_{8.90}$	-	$12.9\pm^{4.30}_{3.72}$	$46.2\pm^{8.47}_{8.34}$	-
	Dan [9]	$0.0599\pm^{0.0213}_{0.0172}$	$0.1809\pm^{0.0555}_{0.0532}$	0.96	$17.0\pm^{7.93}_{5.73}$	$347.\pm^{139.}_{98.8}$	42.1	$22.9\pm^{12.9}_{7.52}$	$497.\pm^{263.}_{148.}$	21.2
	ChVecOpt	$\mathbf{0.0506}\pm^{\mathbf{0.0154}}_{\mathbf{0.0140}}$	$\mathbf{0.1689}\pm^{\mathbf{0.0558}}_{\mathbf{0.0480}}$	0.96	$8.77\pm^{3.47}_{2.58}$	$30.9\pm^{11.4}_{10.0}$	0.02	$12.6\pm^{4.53}_{3.76}$	$\mathbf{34.0}\pm^{\mathbf{10.5}}_{\mathbf{9.34}}$	0.01
	QuatVecOpt	$0.0514\pm^{0.0155}_{0.0146}$	$0.1719\pm^{0.0516}_{0.0495}$	0.24	$9.15\pm^{2.95}_{2.99}$	$\mathbf{30.4}\pm^{\mathbf{10.3}}_{\mathbf{8.93}}$	0.01	$12.4\pm^{4.09}_{3.48}$	$34.3\pm^{10.2}_{10.1}$	0.02

Table 2: **Real and synthetic data experiments**. We show the median and distance to the 25% and 75% percentiles of the error distribution of the absolute rotation (geodesic) and translation (Euclidean) errors (ε_r and ε_t). We separate the results according to different types of motions. The α values are picked to give either the minimal rotation or translation mean error. We highlight the best median results for each motion type and best rotation/translation α. Notice that it does not necessarily indicate significance, where often the different results are very close. **DQ2Steps** does not depend on α so its estimates are the same for best rotation/translation α.

not solve an optimization problem shows a degradation in accuracy in the same scenarios.

We also introduced a prior to our formulation which does not change the efficiency nor the optimality of our solution. The prior term can help for having maximum a posterior probability (MAP) analysis, or to regularize the estimation for ill-posed scenarios. Other generalizations, such as adding weights or a covariance, can be trivially incorporated to our formulation.

Generally we find that optimal methods are more robust to almost planar motions, however the best performing method will depend on the trajectory and noise in the given problem which dictates the residuals distribution. Our method has the advantage of providing good results comparable with other optimal methods, while being very efficient.

Acknowledgments

This research was done as part of a computer vision research project at Zenuity. The authors gratefully acknowledge funding and support from Zenuity. We would also like to thank Ernesto Nunes for useful discussions.

References

[1] Yiu Cheung Shiu and Shaheen Ahmad. Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form ax= xb. *Robotics and Automation, IEEE Transactions on*, 5(1):16–29, 1989.

[2] Sandro Esquivel. Eye-to-eye calibration: extrinsic calibration of multi-camera systems using hand-eye calibration methods. 2015.

[3] Mili Shah, Roger D. Eastman, and Tsai Hong. An overview of robot-sensor calibration methods for evaluation of perception systems. In *Proceedings of the Workshop on Performance Metrics for Intelligent Systems*, PerMIS '12, pages 15–20, New York, NY, USA, 2012. ACM.

[4] Roger Y. Tsai and Reimer K. Lenz. A new technique for fully autonomous and efficient 3d robotics hand-eye calibration. In *Proceedings of the 4th International Symposium on Robotics Research*, pages 287–297, Cambridge, MA, USA, 1988. MIT Press.

[5] Frank Chongwoo Park and Bryan J. Martin. Robot sensor calibration: solving ax=xb on the euclidean group. *IEEE Trans. Robotics and Automation*, 10:717–721, 1994.

[6] Radu Horaud and Fadi Dornaika. Hand-eye calibration. *The international journal of robotics research*, 14(3):195–210, 1995.

[7] Jack C. K. Chou and M. Kamel. Finding the position and orientation of a sensor on a robot manipulator using quaternions. *The International Journal of Robotics Research*, 10(3):240–254, 1991.

[8] Klaus H. Strobl and Gerd Hirzinger. Optimal hand-eye calibration. *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4647–4653, 2006.

[9] Kostas Daniilidis. Hand-eye calibration using dual quaternions. *I. J. Robotics Res.*, 18:286–298, 1999.

[10] Eduardo Bayro-Corrochano, Kostas Daniilidis, and Gerald Sommer. Motor algebra for 3d kinematics: The case of the hand-eye calibration. *Journal of Mathematical Imaging and Vision*, 13(2):79–100, Oct 2000.

[11] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *International Journal of Computer Vision*, 103(3):267–305, Jul 2013.

[12] Y. B. Jia. Dual quaternion. *Com S 477/577, Fall 2015, Course handouts, 2013. URL http://web. cs.iastate.edu/ cs577/. Date accessed: January 30, 2017.*, 2017.

[13] J. von Neuman and E. Wigner. Uber merkwürdige diskrete Eigenwerte. Uber das Verhalten von Eigenwerten bei adiabatischen Prozessen. *Physikalische Zeitschrift*, 30:467–470, 1929.

[14] Peter D. Lax. *Linear Algebra and Its Applications*. Wiley-Interscience, Hoboken, NJ, second edition, 2007.

[15] James Arvo. Graphics gems iii. chapter Fast Random Rotation Matrices, pages 117–120. Academic Press Professional, Inc., San Diego, CA, USA, 1992.