Probabilistic Methods for Solving the AX=XB Method in the Presence of Noise

M. Kendal Ackerman
Department of Mechanical Engineering
Johns Hopkins University
Baltimore, MD 21218
Email: mackerm4@jhu.edu

Qianli Ma
Department of Mechanical Engineering
Johns Hopkins University
Baltimore, MD 21218
Email: mqianli1@jhu.edu

Gregory S. Chirikjian
Department of Mechanical Engineering
Johns Hopkins University
Baltimore, MD 21218
Email: gregc@jhu.edu

The "AX=XB" sensor calibration problem is ubiquitous in the fields of robotics and computer vision. In this problem A, X, and B are each homogeneous transformations (i.e., rigid-body motions) with A and B given from sensor measurements, and X is the unknown that is sought. For decades this problem is known to be solvable for X when a set of exactly measured compatible A's and B's with known correspondence is given. However, in practical problems, it is often the case that the data streams containing the A's and B's will present at different sample rates, they will be asynchronous, and each stream may contain gaps in information. We therefore present a method for calculating the calibration transformation, X, that works for data without any a priori knowledge of the correspondence between the As and Bs.

Nomenclature

A_i A rigid body transformation ($A_i \in SE(3)$), usually assocated with sensor measurements

B_i A rigid body transformation ($B_i \in SE(3)$), usually assocated with sensor measurements

X The rigid body transformation ($X_i \in SE(3)$) that relates A_i to B_i

R_G The rotation matrix of any general transformation matrix $G \in SE(3)$

$\mathbf{t}_G$ The translationvector of any general transformation matrix $G \in SE(3)$

$\mathbf{n}_G$ The axis of rotation for R_G

θ_G The angle of rotation for R_G about $\mathbf{n}_G$

1 INTRODUCTION

In the fields of robotics and computer vision, sensor calibration problems are often codified using the "AX=XB" formulation. Example applications include camera calibration, Cartesian robot hand calibration, robot eye-to-hand calibration [?], aerial vehicle sensor calibration [?] and image guided therapy (IGT) sensor calibration [?]. In the "AX=XB" formulation A, X, and B are each homogeneous transformations (i.e., elements of the special Euclidean group, $SE(3)$) with each pair of measurements (A,B) coming from sensors such as cameras, US probes, optical or electromagnetic pose tracking systems, etc., and X is the unknown rigid-body motion that is found as a result of solving $AX = XB$.

It is well known that it is not possible to solve for a unique X from a single pair of exact (A,B), but if there are two instances of independent exact measurements, (A_1,B_1) and (A_2,B_2), then the problem can be solved. However, in practice sensor noise is always present, and an exact solution is not possible, and the goal becomes one of finding an X with least-squared error given corresponding noisy pairs (A_i,B_i) for i=1,2,...,n.

2 THE $AX = XB$ FORMULATION

Any (proper) rigid-body motion in three-dimensional space can be described as a 4×4 homogeneous transformation matrix of the form

$$H(R,\mathbf{t}) = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix} \quad (1)$$

where $R \in SO(3)$ is a 3×3 (proper) rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ is a translation vector. The set of all such matrices can be identified with $SE(3)$, the group of rigid-body motions, where the group law is matrix multiplication.

Given

$$AX = XB \tag{2}$$

for $A, B, X \in SE(3)$ it is well known that, in non-degenerate cases, there are two unspecified degrees of freedom to the problem for a single pair of sensor measurements, (A, B). This situation is rectified by considering two pairs of exact measurements of the form in (5), i.e., $A_1X = XB_1$ and $A_2X = XB_2$, provided that some mild conditions are observed for the selection of the pairs (A_1, B_1) and (A_2, B_2) [**?**, **?**, **?**]. Additionally, if there is sensor error, then it may not be possible to find compatible pairs that reproduce the exact value of X. For this reason, minimization and least squared approaches are often taken over large sets of A's and B's.

Performing the matrix multiplication of homogeneous transformations in (5) and separating out the rotational and translational parts results in two equations of the form

$$R_A R_X = R_X R_B \quad \text{and} \quad R_A \mathbf{t}_X + \mathbf{t}_A = R_X \mathbf{t}_B + \mathbf{t}_X. \tag{3}$$

The strategy to solve (5) would appear to reduce to first solving the part of (3) with only rotations, and then rearranging the second equation so as to find acceptable values of $\mathbf{t}_X$: $(R_A - \mathbb{I}_3)\mathbf{t}_X = R_X\mathbf{t}_B - \mathbf{t}_A$. However, there are some problems with this naive approach. As pointed out in [?, ?], in non-degenerate cases there is a one-parameter set of solutions to the first equation in (3), and the matrix $R_A - \mathbb{I}_{3\times3}$ in general has rank 2. Hence, there are two unspecified degrees of freedom to the problem, and it cannot be solved uniquely unless additional measurements are taken.

However, if there is sensor error, then it may not be possible to find compatible pairs that reproduce the exact value of X. For this reason, minimization approaches are often taken where for $n > 2$ a cost function

$$C(X) = \sum_{i=1}^{n} w_i d^2(A_iX, XB_i) \tag{4}$$

is computed for some distance metric $d(\cdot,\cdot)$ on $SE(3)$ and $\{w_i\}$ is a set of weights which can be taken to be a partition of unity.

3 EXISTING SOLUTION METHODS

The problem of solving

$$AX = XB \tag{5}$$

for X when multiple corresponding pairs of A's and B's are presented has a history that goes back more than a quarter of a century [**?**, **?**, ?, ?, **?**], with the earliest proposed by Tsai [1] and Shiu [2], and applications involving this problem remain active today [**?**]- [**?**].

For the following methods, given

$$AX = XB, \tag{6}$$

decompose equation (6) into the rotation component and the translation component:

$$R_A R_X = R_X R_B \tag{7}$$

$$(R_A - I)\mathbf{t}_x = R_X \mathbf{t}_B - \mathbf{t}_A \tag{8}$$

3.1 Closed-Form Solutions

need some form of introductory statement about closed form solutions here.

3.1.1 Shiu and Ahmad [2]

Let,

$$R_X = Rot(\mathbf{n}_A, \beta) R_{X_P} \tag{9}$$

where

$$R_{X_p} = Rot(\mathbf{n}_X, \theta_X)$$
$$\mathbf{n}_X = \mathbf{n}_B \times \mathbf{n}_A$$
$$\theta_X = atan2(|\mathbf{n}_B \times \mathbf{n}_A|, \mathbf{n}_B \cdot \mathbf{n}_A)$$

and β is an angle with arbitrary value.

Equation (9) shows that R_x has one degree of freedom. Combined with equation (8), it can be concluded that translation part $\mathbf{t}_x$ also has one degree of freedom. Thus, two (A, B) pairs are needed to calculate the unique solution of matrix X.

A Least squares method is employed to first solve for the rotation part.

$$CY = D \tag{10}$$

C and D are very large matrices but we still need to define the method a little better I think. where $Y = \big(\cos(\beta_1), \sin(\beta_1), \cos(\beta_2), \sin(\beta_2)\big)^T$.
Then, given the known rotation matrix, a least squares method is used again to solve for the translation part.

$$\begin{pmatrix} R_{A_1} - I \\ R_{A_2} - I \end{pmatrix} \mathbf{t}_X = \begin{pmatrix} R_X \mathbf{t}_{B_1} - \mathbf{t}_{A_1} \\ R_X \mathbf{t}_{B_2} - \mathbf{t}_{A_2} \end{pmatrix} \tag{11}$$

The Shiu and Ahmad method requires that, for a unique solution, the axes of rotation of A_1 and A_2 are not parallel or anti-parallel, and the rotation angles are not 0 or π. Additionally a true closed-form solution of X (neither rotation part R_X or translation part $\mathbf{t}_X$) described by the two pairs of $\{A_i, B_i\}$, was not given fully. This will be discussed in the next section.

3.2 Lie Group Method [3]

3.2.1 Closed Form Solution with Two Pairs

In Lie group method, the closed-form solution for R_X is achieved as described in (12).

$$R_X = \mathcal{A}\mathcal{B}^{-1} \tag{12}$$

where

$$\begin{aligned} \mathcal{A} &= (\mathbf{n}_{A_1}, \mathbf{n}_{A_2}, \mathbf{n}_{A_1} \times \mathbf{n}_{A_2}) \\ \mathcal{B} &= (\mathbf{n}_{B_1}, \mathbf{n}_{B_2}, \mathbf{n}_{B_1} \times \mathbf{n}_{B_2}) \\ \mathbf{n}_{A_i} &= log(A_i) \\ \mathbf{n}_{B_i} &= log(B_i) \end{aligned}$$

The solution for $\mathbf{t}_X$ is uniquely defined by (13), and the closed-form solution for X is complete.

$$\begin{pmatrix} R_{A_1} - I \\ R_{A_2} - I \end{pmatrix} \mathbf{t}_X = \begin{pmatrix} R_X \mathbf{t}_{B_1} - \mathbf{t}_{A_1} \\ R_X \mathbf{t}_{B_2} - \mathbf{t}_{A_2} \end{pmatrix} \tag{13}$$

3.2.2 Estimation of X Using Multiple Pairs with Noise

It follows the equations below when computing the X matrix using multiple pairs of As and Bs with noise.

$$R_X = (M^T M)^{-\frac{1}{2}} M^T \tag{14}$$

where

$$M = \sum \mathbf{n}_{B_i} \mathbf{n}_{A_i}^T$$

And here $i \geq 3$ is a necessary condition for M to a non-singular matrix.

$$\mathbf{t}_x = (C^T C)^{-1} C^T d \tag{15}$$

where

$$C = \begin{pmatrix} I - R_{X_{A_1}} \\ I - R_{X_{A_2}} \\ . \\ . \\ . \\ I - R_{X_{A_n}} \end{pmatrix} \quad d = \begin{pmatrix} \mathbf{t}_{A_1} - R_X \mathbf{t}_{B_1} \\ \mathbf{t}_{A_2} - R_X \mathbf{t}_{B_2} \\ . \\ . \\ . \\ \mathbf{t}_{A_n} - R_X \mathbf{t}_{B_n} \end{pmatrix}$$

3.3 Quaternion Method

3.3.1 Closed Form Solution with Two Pairs (No Noise) [4]

$$R_A R_X = R_X R_B \;\; \Leftrightarrow \;\; \mathbf{q}_A \otimes \mathbf{q}_X = \mathbf{q}_X \otimes \mathbf{q}_B \tag{16}$$

where $\mathbf{q}_A$, $\mathbf{q}_B$ and $\mathbf{q}_X$ are the unit quaternions that represent the rotation parts of matrices A, B and X, and $\otimes$ denotes quaternion multiplication. Note that this shouldn't be confused with the operative symbol of Kronecker product.

$$E\vec{\mathbf{q}}_X = \mathbf{0} \tag{17}$$

where E is a 4 by 4 matrix as a result of the manipulation of $\mathbf{q}_A$ and $\mathbf{q}_B$ into matrix form, and $\vec{\mathbf{q}}_X$ is a 4 by 1 vector representation of unit quaternion $\mathbf{q}_X$.
For the unit quaternion representing the rotation part of X

$$\vec{\mathbf{q}}_X = V_2 \mathbf{y_2} \tag{18}$$

where

$$\begin{aligned} E &= \sin(\theta_{A|B}/2) M \\ M &= U \Sigma V^T \\ V &= (V_1, V_2) \\ \mathbf{y} &= V^T \mathbf{q}_x \\ \mathbf{y} &= (\mathbf{y_1^T}, \mathbf{y_2^T})^\mathbf{T} \end{aligned}$$

For the translation part

$$\mathbf{t}_x = \mathbf{n_A} z_3 - \frac{1}{2}\left(\cot(\frac{\theta_A}{2})\hat{\mathbf{n}}_A + I\right)\mathbf{c} \quad z_3 = arbitrary \tag{19}$$

where

$$\mathbf{c} = (\mathbf{A} - \mathbf{I})\mathbf{t_x} \tag{20}$$

And $\theta_{A|B} := \theta_A = \theta_B$ is the constraint that corresponding A and B should have the same angle of rotation.
A unique solution can be calculated given two pairs of As and Bs using equation (18) and (20)

3.3.2 Estimation of X Using Multiple Pairs (With Noise) [5]

$$\begin{aligned} f(R_X) &= \textstyle\sum_{i=1}^n ||\mathbf{n}_{A_i} - \mathbf{q}_X \otimes \mathbf{n}_{Bi} \otimes \bar{\mathbf{q}}_X||^2 \\ &= \vec{\mathbf{q}}_X{}^T \tilde{A} \vec{\mathbf{q}}_X \end{aligned} \tag{21}$$

where $\tilde{A} = \sum_{i=1}^n \tilde{A}_i$.

$$\min_q f = \min_q (\vec{\mathbf{q}}_X{}^T \tilde{A} \vec{\mathbf{q}}_X + \lambda(1 - \vec{\mathbf{q}}_X{}^T \vec{\mathbf{q}}_X)) \tag{22}$$

$$\tilde{A}\vec{\mathbf{q}}_X = \lambda \vec{\mathbf{q}}_X \tag{23}$$

And the unit quaternion $\mathbf{p}_X$ that minimizes f is the eigenvector of $\tilde{A}$ associated with its smallest eigenvalue (23).

3.4 Dual Quaternion Method [6]

$$AX = XB \ \Leftrightarrow\ \check{\mathbf{a}} = \check{\mathbf{q}} \circ \check{\mathbf{b}} \circ \bar{\check{\mathbf{q}}} \tag{24}$$

where $\check{\mathbf{a}}$, $\check{\mathbf{b}}$ and $\check{\mathbf{q}}$ are the dual quaternions that represent matrices A, B and X, and $\bar{\check{\mathbf{q}}}$ is the conjugate of $\check{\mathbf{q}}$. The operation symbol $\circ$ denotes dual quaternion multiplication.

$$\check{q} = \begin{pmatrix} \cos(\frac{\theta+\varepsilon d}{2}) \\ \sin(\frac{\theta+\varepsilon d}{2})(\vec{\mathbf{l}} + \varepsilon\vec{\mathbf{m}}) \end{pmatrix} \tag{25}$$

where θ, d, $\vec{\mathbf{l}}$ and $\vec{\mathbf{m}}$ are screw parameters.

$$S_i \begin{pmatrix} \mathbf{q} \\ \mathbf{q}' \end{pmatrix} = \mathbf{0} \tag{26}$$

where

$$S_i = \begin{pmatrix} \vec{\mathbf{a}} - \vec{\mathbf{b}} & (\vec{\mathbf{a}} + \vec{\mathbf{b}})^\wedge & \mathbf{0_{3\times 1}} & \mathbf{0_{3\times 3}} \\ \vec{\mathbf{a}}' - \vec{\mathbf{b}}' & (\vec{\mathbf{a}}' + \vec{\mathbf{b}}')^\wedge & \vec{\mathbf{a}} - \vec{\mathbf{b}} & (\vec{\mathbf{a}} + \vec{\mathbf{b}})^\wedge \end{pmatrix} \tag{27}$$

Under the constraint of

$$\mathbf{q}^T\mathbf{q} = 1 \ \text{ and } \ \mathbf{q}^T\mathbf{q}' = 0 \tag{28}$$

$$T = \begin{pmatrix} S_1^T\ S_2^T\ \ldots\ S_n^T \end{pmatrix}^T \tag{29}$$

By using SVD(singular value decomposition) on $T = U\Sigma V^T$, the dual quaternion for matrix X can be generated by the linear combination of the last two right-singular vectors, which are the last two columns of matrix V.

Different from quaternion method (16), dual quaternion method solves the rotational part and translational part in a united way, and it contains all the information to reconstruct the transformation matrix X.

3.5 Kronecker Product Method [7]

$$AX = XB$$

$$\Updownarrow$$

$$\begin{pmatrix} I_9 - R_B \otimes R_A & 0_{9\times 3} \\ \mathbf{t}_B^T \otimes I_3 & I_3 - R_A \end{pmatrix} \begin{pmatrix} vec(R_X) \\ \mathbf{t}_X \end{pmatrix} = \begin{pmatrix} 0_9 \\ \mathbf{t}_A \end{pmatrix} \tag{30}$$

where the symbol $\otimes$ denotes Kronecker product.
For multiple pairs of As and Bs with noise, the Kroneker product is reformulated as (31) and (32).

$$\begin{pmatrix} I_9 - R_{B_1} \otimes R_{A_1} \\ I_9 - R_{B_2} \otimes R_{A_2} \\ \vdots \\ I_9 - R_{B_n} \otimes R_{A_n} \end{pmatrix} vec(R_X) = \mathbf{0}_{9n\times 1} \tag{31}$$

$$\begin{pmatrix} I_3 - R_{A_1} \\ I_3 - R_{A_2} \\ \vdots \\ I_3 - R_{A_n} \end{pmatrix} \mathbf{t}_X = \begin{pmatrix} \mathbf{t}_{A_1} - R_X\mathbf{t}_{B_1} \\ \mathbf{t}_{A_2} - R_X\mathbf{t}_{B_2} \\ \vdots \\ \mathbf{t}_{A_n} - R_X\mathbf{t}_{B_n} \end{pmatrix} \tag{32}$$

Orthogonalization is implemented on R_X to make it a rotation matrix.

$$R_{X_e} = R_X(R_X^T R_X)^{-1/2} \tag{33}$$

[8] where R_{X_e} denotes the orthogonalized R_X.
The orthogonalized matrix R_X is further normalized as follows, then the rotation matrix $R_X \in SO(3)$ is obtained.

$$R_X = \frac{sign(\det(R_X))}{|\det(R_X)|^{\frac{1}{3}}} R_X \tag{34}$$

After obtaining the estimation of R_X, least square method is implemented on (32) to recover the value of $\mathbf{t}_X$.

A commonality of all the methods in the previous section is that exact knowledge of the A_i and B_i correspondence is assumed. This is not always the case. There are many instances in the literature when the sensor data used in calibration becomes "unsynchronized". Different attempts have been implemented to solve this problem, such as time stamping the data, developing dedicated software modules for syncing the data [**?**], and analyzing components of the sensor data stream to determine a correlation [?], to varying effects. The solution methodology presented in this paper bypasses these issues altogether without tracking, or recomputing, correspondence. By modeling the set of A's and B's as probability distributions on $SE(3)$, the data can be taken as an unordered, uncorrelated "batch" and a solution for X can be generated. Additionally this new formulation can be expanded to more explicitly model the error in X associated with the noise in the A's and B's. The error can then be minimized to further refine X.

4 THE BATCH METHOD (ASSUMING A UNIQUE X)

This section presents methods to solve for an X wherein there does not need to be any a priori knowledge of the correspondence between A's and B's. In other words, the sets of A's and B's each can be given as unordered "batches".

4.1 The Batch Method Formulation

Given a large set of pairs $(A_i, B_i) \in SE(3) \times SE(3)$ for $i = 1, ..., n$ that exactly satisfy the equation

$$A_i X = X B_i \tag{35}$$

we attempt to develop algorithms to find $X \in SE(3)$. We address a generalization of the standard problem in which the sets $\{A_i\}$ and $\{B_j\}$ are provided with elements written in any order and it is known that a correspondence exists between the elements of these sets such that (35) holds, but we do not know a priori this correspondence between each A_i and B_j.

We begin by defining a Gaussian probability distribution on $SE(3)$ (assuming the norm $\|\Sigma\|$ is small) as

$$\rho(H; M, \Sigma) = \frac{1}{(2\pi)^3 |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}F(M^{-1}H)}$$

where $|\Sigma|$ denotes the determinant of Σ and

$$F(H) = [\log^\vee(H)]^T \Sigma^{-1} [\log^\vee(H)].$$

When H is parameterized with exponential coordinates, $H = \exp Z$, this means that $F(\exp Z) = \mathbf{z}^T \Sigma^{-1} \mathbf{z}$ where $\mathbf{z} = Z^\vee$ and $\rho(\exp Z; \mathbb{I}_4, \Sigma)$ becomes exactly a zero-mean Gaussian distribution on the Lie algebra $se(3)$, with covariance Σ, that is 'lifted up' to the Lie group $SE(3)$.

Using formulations of probability theory on SE(3), we can think of (35) as the equation

$$(\delta_{A_i} * \delta_X)(H) = (\delta_X * \delta_{B_i})(H). \tag{36}$$

Whereas the addition (as opposed to multiplication) of homogeneous transformation matrices is nonsensical, the addition of real-valued functions $f_1(H) + f_2(H)$ is a perfectly reasonable operation and since convolution is a linear operation on functions, we can write all n instances of (36) into a single equation of the form

$$(f_A * \delta_X)(H) = (\delta_X * f_B)(H) \tag{37}$$

where

$$f_A(H) = \frac{1}{n}\sum_{i=1}^{n} \delta(A_i^{-1}H) \text{ and } f_B(H) = \frac{1}{n}\sum_{i=1}^{n} \delta(B_i^{-1}H).$$

The above functions are normalized to be probability densities:

$$\int_{SE(3)} f_A(H)dH = \int_{SE(3)} f_B(H)dH = 1.$$

Let the mean and covariance of a probability density $f(H)$ be defined by the conditions

$$\begin{aligned} &\int_{SE(3)} \log(M^{-1}H) f(H) dH = \mathbb{O} \text{ and} \\ &\Sigma = \int_{SE(3)} \log^\vee(M^{-1}H)[\log^\vee(M^{-1}H)]^T f(H) dH. \end{aligned} \tag{38}$$

If $f(H)$ is of the form of $f_A(H)$ given above, then

$$\begin{aligned} &\sum_{i=1}^{n} \log(M_A^{-1}A_i) = \mathbb{O} \text{ and} \\ &\Sigma_A = \frac{1}{n}\sum_{i=1}^{n} \log^\vee(M_A^{-1}A_i)[\log^\vee(M_A^{-1}A_i)]^T. \end{aligned} \tag{39}$$

It can be shown that if these quantities are computed for two highly focused functions, f_1 and f_2, that the same quantities for the convolution of these functions can be computed as

$$M_{1*2} = M_1 M_2 \text{ and } \Sigma_{1*2} = Ad(M_2^{-1})\Sigma_1 Ad^T(M_2^{-1}) + \Sigma_2 \tag{40}$$

where

$$Ad(H) = \begin{pmatrix} R & \mathbb{O} \\ \hat{\mathbf{x}}R & R \end{pmatrix}$$

and $\hat{\mathbf{a}}$ is the skew-symmetric matrix such that $\hat{\mathbf{a}}\mathbf{b} = \mathbf{a} \times \mathbf{b}$.

The mean of $\delta_X(H)$ is $M_X = X$, and its covariance is the zero matrix. Therefore, (37) together with (40) gives two "Batch Method" equations:

$$\boxed{M_A X = X M_B} \tag{41}$$

and

$$\boxed{Ad(X^{-1})\Sigma_A Ad^T(X^{-1}) = \Sigma_B} \tag{42}$$

For the problem discussed in this paper, there is no loss of generality in assuming that $\|\Sigma_A\|$ and $\|\Sigma_B\|$ are small because the constraint equation (42) is linear in Σ_A and Σ_B, and so if they are not small, they can both be normalized resulting in $\Sigma_A' = \Sigma_A/(\|\Sigma_A\|)$ and likewise $\Sigma_B' = \Sigma_B/(\|\Sigma_A\|)$. Note that here we have normalized by the same quantity on both sides. We cannot use $\|\Sigma_A\|$ on one side of the equation and $\|\Sigma_B\|$ on the other because the Frobenius norm is not Ad-invariant for $SE(3)$.

Moreover, standard tests from multivariate statistical analysis such as q-q plots can be used to assess whether or not the data are Gaussian. If they are not, they can be made Gaussian without loss of information or by introducing changes to the original mean and covariance in a simple way. Since $A_i = XB_iX^{-1}$, it follows that $A_i^p = XB_i^pX^{-1}$ for any power $p \in \mathbb{R}$. This means that each measured data point can be replaced with a continuum of equivalent data points parameterized by p. Practically speaking, we can sample p at fractional powers in the range $p \in [-1,1]$ and introduce multiple instances of samples with a Gaussian weighting that depends on p. This would cause the resulting augmented data set to behave as a Gaussian. But since the mean and covariance would be unchanged, there is no need to implement this thought experiment – Gaussians can be used in place of data even if the data are not Gaussian.

4.2 A Batch Method Solution

These two equations can be solved in a similar way to how the two equations $A_1X = XB_1$ and $A_2X = XB_2$ are solved in the closed form case.

First, we seek the rotational component, R_X, of X. From (41) we have that,

$$\mathbf{n}_{M_A} = R_X \mathbf{n}_{M_B} \tag{43}$$

where $\mathbf{n}_H$ is the direction of the screw axis of the homogeneous transform H.

If we decompose Σ_{M_A} and Σ_{M_B} into blocks as

$$\Sigma_i = \begin{pmatrix} \Sigma_i^1 & \Sigma_i^2 \\ \Sigma_i^3 & \Sigma_i^4 \end{pmatrix}$$

where $\Sigma_i^3 = (\Sigma_i^2)^T$, then we can take the first two blocks of (42) and write

$$\begin{aligned} \Sigma_{M_B}^1 &= R_X^T \Sigma_{M_A}^1 R_X \text{ and} \\ \Sigma_{M_B}^2 &= R_X^T \Sigma_{M_A}^1 R_X (\widehat{R_X^T t_x}) + R_X^T \Sigma_{M_A}^2 R_X \end{aligned} \tag{44}$$

We can then find the eigendecomposition, $\Sigma_i = Q_i \Lambda Q_i^T$, where Q_i is the square matrix whose ith column is the eigenvector of Σ_i and Λ is the diagonal matrix with corresponding eigenvalues as diagonal entries and write the first block equation of (44) as,

$$\Lambda = Q_{M_B}^T R_X^T Q_{M_A} \Lambda Q_{M_A}^T R_X Q_{M_B} = Q \Lambda Q^T \tag{45}$$

The set of Qs that satisfy this equation is given as,

$$Q = \left\{ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \right\}$$

with the simple condition that Q_i is constrained to be a rotation matrix. This means that the rotation component of X is given by,

$$R_x = Q_{M_A} Q Q_{M_B}^T \tag{46}$$

The correct solution, from the set of 4 possibilities of R_X (given (46)) can be found by applying (56) and choosing the one that minimizes $\|\mathbf{n}_{M_A} - R_X \mathbf{n}_{M_B}\|$. Once R_X is found in this way, $\mathbf{t}_X$ can be found easily from blocks the 2 and 4 of (42).

5 THE EXPANDED BATCH METHOD (X AS A DISTRIBUTION)

When there is measurement noise, in place of the sums of delta functions defining f_A and f_B we have

$$f_A(H) \longrightarrow (f_A * \rho_A)(H) \text{ and } f_B(H) \longrightarrow (f_B * \rho_B)(H)$$

where $\rho_A(H)$ is the noise distribution for A, and likewise $\rho_B(H)$ is the noise distribution for B. These distributions may be unbiased (in which case $\rho_A(H)$ and $\rho_B(H)$ have means at the identity, or they may be biased.

For this case, a single X will not solve every instance of (35) and hence the delta function in (37) will not solve the problem. Instead, we seek a distribution $\rho_X(H)$ with (a priori unknown) mean X and covariance Σ_X such that

$$(f_A * \rho_X)(H) = (\rho_X * f_B)(H). \tag{47}$$

where here $f_A(H)$ and $f_B(H)$ come from the noisy measurements. There is no a priori guarantee that a distribution $\rho_X(H)$ exists that solves the problem, but we can nevertheless attempt to find it.

As convolution is a ”smearing” process, the distribution, ρ_X will blur both sides of (47) in a similar fashion. This may not always be the ideal, as in the case when $f_A(H)$ and $f_B(H)$ do not have similarly spread covariances, and so blurring each of them with an identical ρ_X may never satisfy (47). For example, if the noise characteristics of the sensors are different, e.g. if the $A's$ are noisier than the $B's$, then the noise distributions (ρ_A and ρ_B from above) and therefore $f_A(H)$ and $f_B(H)$ will have different spreads.

We can re-write (35) as

$$A_i = XB_iX^{-1} \tag{48}$$

and derive the probabilistic form (with noise) as

$$f_A(H) = (\rho_X * f_B * \rho_{X^{-1}})(H). \tag{49}$$

Using (40) applied to both sides of (49) we get

$$\boxed{M_A = M_X M_B M_{X^{-1}}} \tag{50}$$

and

$$\begin{aligned}\Sigma_A &= \Sigma_X * \Sigma_B * \Sigma_{X^{-1}} \\ &= [Ad(M_{B^{-1}})\Sigma_X Ad^T(M_{B^{-1}}) + \Sigma_B] * \Sigma_{X^{-1}} \\ &= Ad(M_X)[Ad(M_{B^{-1}})\Sigma_X Ad^T(M_{B^{-1}}) \\ &\qquad + \Sigma_B]Ad^T(M_X) + \Sigma_{M_{X^{-1}}}.\end{aligned}$$

It can be shown that for $g \in SE(3)$, $Ad(X^{-1}) = Ad(X)^{-1}$, $M_{g^{-1}} = M_g^{-1}$, and $\Sigma_{g^{-1}} = Ad(g)\Sigma_g Ad^T(g)$. (should we prove these?) We can therefore write*

$$\begin{aligned}\Sigma_A =& Ad(X)Ad(B^{-1})\Sigma_X Ad^T(B^{-1})Ad^T(X) \\ &+ Ad(X)\Sigma_B Ad^T(X) + Ad(X)\Sigma_X Ad^T(X)\end{aligned}$$

(*For ease of readability, through a slight abuse of notion, denote $A = M_A$, $X = M_X$, and $B = B$), which gives

$$Ad(X^{-1})\Sigma_A Ad^T(X^{-1}) = Ad(B^{-1})\Sigma_X Ad^T(B^{-1}) + \Sigma_B + \Sigma_X$$

$$\boxed{\begin{aligned}\rightarrow \Sigma_X + Ad(B^{-1})\Sigma_X Ad^T(B^{-1}) = \\ Ad(X^{-1})\Sigma_A Ad^T(X^{-1}) - \Sigma_B.\end{aligned}} \tag{51}$$

As in the noise-free case, (A, Σ_A) and (B, Σ_B) are known from the data and are computed from (39). The only difference is that now we seek (X, Σ_X) instead of just X, and the associated additional degrees of freedom allows for a better fit to capture the effects of noise in the measurements.

5.1 Numerical Validation of Σ_X

To verify that this formulation produced a meaningful Σ_X, we used a numerical approach that would allow us to, a prioir, know the true distribution of X. We simulate an $AX = XB$ calibration by generating simulated A and B data streams. The B's are chosen as poses along a trajectory in $SE(3)$. Since we must use relative motions, we generate $B^{ij} = B_i^{-1}B_{i+1}$ where B_i's are drawn from the two sample "s-shaped" trajectories on a sphere. After forming relative motions, B_{ij}'s, we calculate $A^{ij} = {}^1X_{ij}\, B_i\, {}^2X_{ij}^{-1}$, where ${}^1X_{ij}$ and ${}^2X_{ij}$ are drawn from a distribution with known mean (X) and covariance. Each calculation of ${}^kX_{ij}$ is tweaked as such:

$${}^kX_{ij} \doteq X\, {}^k\Delta_{ij} \tag{52}$$

where ${}^k\Delta_{ij}$ is a matrix of small noises of the the form

$${}^k\Delta_i \doteq \exp\begin{pmatrix}\Omega(\delta t) & \mathbf{v}(\delta t) \\ \mathbf{0}^T & 0\end{pmatrix} \approx \mathbb{I}_4 + (\delta t)\begin{pmatrix}\Omega & \mathbf{v} \\ \mathbf{0}^T & 0\end{pmatrix}.$$

where $\Omega = -\Omega^T$ and $\mathbf{v}$ are random angular and translational velocities with components drawn independently from a Gaussian distribution with variance σ^2, and δt is a small finite time.

After calculating Σ_X from 51, we can compare to the calculated sample mean and covariance of $\{X_{ij}\} = \{{}^1X_{ij}\} \cup \{{}^2X_{ij}\}$ and compare them to the solved for values.

5.2 A least-squares approach using the L^2 Norm

One possible method to search for an appropriate X could be by minimizing the cost function

$$C_2(X) = \left(\int_{SE(3)} \|f_1(H) - f_2(H)\|^2\, dH\right)^{\frac{1}{2}}. \tag{53}$$

where

$$f_1(H) = \rho(H; A, \Sigma_A)$$

and

$$f_2(H) = \rho(H; XBX^{-1},\, \Sigma_2). \tag{54}$$

where

$$\Sigma_2 = Ad(X)[Ad(B^{-1})\Sigma_X Ad^T(B^{-1}) + \Sigma_B + \Sigma_X]Ad^T(X)$$

In general, the integral in this cost function cannot be solved in closed form because the log function is nonlinear, and in terms of exponential coordinates $dH = |J(\mathbf{z})|d\mathbf{z}$ where $|J(\mathbf{0})| = 1$, but this Jacobian is a nonlinear function of $\mathbf{z}$.

However, if we a priori limit the search for X to the cylinder defined in (**??**), automatically, $XBX^{-1} = A$. Then, we can define a new variable $K = M_A^{-1}H$ and using the property of invariance of integration under shifts, can write

$$C_2(X(\phi, s)) = \left(\int_{SE(3)} \left|f'_A(K) - f'_B(K)\right|^2 dK\right)^{\frac{1}{2}}$$

where

$$f'_A(K) = \rho(K; \mathbb{I}_4, \Sigma_A)$$

and

$$f'_B(K) = \rho(K; \mathbb{I}_4\,, Ad(X(\phi,s))[Ad(B^{-1})\Sigma_{X(\phi,s)}Ad^T(B^{-1}) + \Sigma_B + \Sigma_X(\phi,s)]Ad^T$$

And if covariances are scaled to be small, the benefit of this is that it reduces to the integral of the norm of the difference of two Gaussians. This has a closed-form solution

$$C_2(X(\phi, s)) =$$

—————— Aside: Additional Questions

I have some things worked out for this, but I want to talk about how we are defining a Gaussian on a Lie Group.

---

which can be minimized by a 2D search over the parameters (ϕ, s) in a small neighborhood around the solution obtained using the no noise approach.

5.2.1 Iterative Optimization by Σ_X Minimization with 2-D Search

A second method would be a simple iterative procedure to solve (50) and (51). First, set $\Sigma_X = \mathbb{O}$ and solve as if the data had no noise. Second, take the resulting value of X, and solve (51)

$$\Sigma_X + Ad(B^{-1})\Sigma_X Ad^T(B^{-1}) = Ad(X^{-1})\Sigma_A Ad^T(X^{-1}) - \Sigma_B$$

for Σ_X. This is done by taking the Kronecker product of both sides and writing

$$[Ad(B^{-1}) \otimes Ad(B^{-1}) + \mathbb{I}_{36}]\mathrm{vec}(\Sigma_X) = \mathrm{vec}(Ad(X^{-1})\Sigma_A Ad^T(X^{-1}) - \Sigma_B), \tag{55}$$

from which we obtain $\mathrm{vec}(\Sigma_X)$, and hence Σ_X. Since $[Ad(B^{-1}) \otimes Ad(B^{-1}) + \mathbb{I}_6 \otimes \mathbb{I}_6]$ in general will be singular, the SVD and pseudoinverse can be used to find the 'best' Σ_X. Starting from the initial "guess" of X from the no noise case, we gain perform a gradient descent on the two degrees of freedom of (**??**), minimizing the quadratic cost $C(\Sigma_X) = \|\Sigma_X\|_F^2 = \mathrm{tr}(\Sigma_X \Sigma_X^T)$.

5.2.2 Block decomposition

As in the no noise case, we can obtain the rotational component, R_X, of X. From (50) we have that,

$$\mathbf{n}_A = R_X \mathbf{n}_B \tag{56}$$

.

If we decompose Σ_A and Σ_B into blocks as $\Sigma_i = \begin{pmatrix} \Sigma_i^1 & \Sigma_i^2 \\ \Sigma_i^3 & \Sigma_i^4 \end{pmatrix}$
where $\Sigma_i^3 = (\Sigma_i^2)^T$, then we can take (51)

$$\Sigma_X + Ad(B^{-1})\Sigma_X Ad^T(B^{-1}) = Ad(X^{-1})\Sigma_A Ad^T(X^{-1}) - \Sigma_B$$

and write

$$\begin{bmatrix} \Sigma_X^1 & \Sigma_X^2 \\ \Sigma_X^3 & \Sigma_X^4 \end{bmatrix} + \begin{bmatrix} R_B^T & 0 \\ (\widehat{-R_B^T t_B})R_B^T & R_B^T \end{bmatrix} \begin{bmatrix} \Sigma_X^1 & \Sigma_X^2 \\ \Sigma_X^3 & \Sigma_X^4 \end{bmatrix} \begin{bmatrix} R_B & R_B(\widehat{R_B^T t_B}) \\ 0 & R_B \end{bmatrix} = \begin{bmatrix} R_X^T & 0 \\ (\widehat{-R_X^T t_X})R_X^T & R_X^T \end{bmatrix} \begin{bmatrix} \Sigma_A^1 & \Sigma_A^2 \\ \Sigma_A^3 & \Sigma_A^4 \end{bmatrix} \begin{bmatrix} R_X & R_X(\widehat{R_X^T t_X}) \\ 0 & R_X \end{bmatrix} - \begin{bmatrix} \Sigma_B^1 & \Sigma_B^2 \\ \Sigma_B^3 & \Sigma_B^4 \end{bmatrix}. \tag{57}$$

Given that

$$\begin{bmatrix} R^T & 0 \\ (\widehat{-R^T t})R^T & R^T \end{bmatrix} \begin{bmatrix} \Sigma^1 & \Sigma^2 \\ \Sigma^3 & \Sigma^4 \end{bmatrix} \begin{bmatrix} R & R(\widehat{R^T t}) \\ 0 & R \end{bmatrix} = \begin{bmatrix} R^T\Sigma^1 R & R^T\Sigma^1 R(\widehat{R^T t}) + R^T\Sigma^2 R \\ (\widehat{-R^T t})R^T\Sigma^1 R + R^T\Sigma^3 R & \begin{matrix} (\widehat{-R^T t})R^T\Sigma^1 R(\widehat{R^T t}) + R^T\Sigma^3 R(\widehat{R^T t}) \\ +(\widehat{-R^T t})R^T\Sigma^2 R + R^T\Sigma^4 R \end{matrix} \end{bmatrix} \tag{58}$$

we can write the block equations as:

$$\Sigma_X^1 + R_B^T\Sigma_X^1 R_B = R_X^T\Sigma_A^1 R_X - \Sigma_B^1 \tag{59}$$

$$\Sigma_X^2 + R_B^T\Sigma_X^1 R_B(\widehat{R_B^T t_B}) + R_B^T\Sigma_X^2 R_B = R_X^T\Sigma_A^1 R_X(\widehat{R_X^T t_X}) + R_X^T\Sigma_A^2 R_X - \Sigma_B^2 \tag{60}$$

$$\Sigma_X^3 + (\widehat{-R_B^T t_B})R_B^T\Sigma_X^1 R_B + R_B^T\Sigma_X^3 R_B = (\widehat{-R_X^T t_X})R_X^T\Sigma_A^1 R_X + R_X^T\Sigma_A^3 R_X - \Sigma_B^3 \tag{61}$$

$$\Sigma_X^4 + (\widehat{-R_B^T t_B})R_B^T\Sigma_X^1 R_B(\widehat{R_B^T t_B}) + R_B^T\Sigma_X^3 R_B(\widehat{R_B^T t_B}) + (\widehat{-R_B^T t_B})R_B^T\Sigma_X^2 R_B + R_B^T\Sigma_X^4$$
$$(\widehat{-R_X^T t_X})R_X^T\Sigma_A^1 R_X(\widehat{R_X^T t_X}) + R_X^T\Sigma_A^3 R_X(\widehat{R_X^T t_X}) + (\widehat{-R_X^T t_X})R_X^T\Sigma_A^2 R_X + R_X^T\Sigma_A^4 R_X - \tag{62}$$

—————— Aside: Working to get further...

I have not gone much past calculating the block equations. I doubt there is a closed form solution without additional constraints (as now we have Σ_X to solve for also), but there may be some other way to couple these with a minimization or search method.

Acknowledgements

ASME Technical Publications provided the format specifications for the Journal of Mechanical Design, though they are not easy to reproduce. It is their commitment to ensuring quality figures in every issue of JMD that motivates this effort to have authors review the presentation of their figures.

Thanks go to D. E. Knuth and L. Lamport for developing the wonderful word processing software packages TEX and LATEX. We would like to thank Ken Sprott, Kirk van Katwyk, and Matt Campbell for fixing bugs in the ASME style file `asme2ej.cls`, and Geoff Shiflett for creating ASME bibliography stype file `asmems4.bst`.

References

[1] Tsai, R. Y., and Lenz, R. K., 1989. "A new technique for fully autonomous and efficient 3d robotics hand/eye calibration". *Robotics and Automation, IEEE Transactions on,* **5**(3), pp. 345–358.

[2] Shiu, Y. C., and Ahmad, S., 1989. "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form ax = xb". *Robotics and Automation, IEEE Transactions on,* **5**(1), pp. 16–29.

[3] Park, F. C., and Martin, B. J., 1994. "Robot sensor calibration: solving ax= xb on the euclidean group". *Robotics and Automation, IEEE Transactions on,* **10**(5), pp. 717–721.

[4] Chou, J. C., and Kamel, M., 1991. "Finding the position and orientation of a sensor on a robot manipulator using quaternions". *The international journal of robotics research,* **10**(3), pp. 240–254.

[5] Horaud, R., and Dornaika, F., 1995. "Hand-eye calibration". *The international journal of robotics research,* **14**(3), pp. 195–210.

[6] Daniilidis, K., 1999. "Hand-eye calibration using dual quaternions". *The International Journal of Robotics Research,* **18**(3), pp. 286–298.

[7] Andreff, N., Horaud, R., and Espiau, B., 1999. "On-line hand-eye calibration". In 3-D Digital Imaging and Modeling, 1999. Proceedings. Second International Conference on, IEEE, pp. 430–436.

[8] Horn, B. K. P., 1986. *Robot Vision: MIT ELECTRICAL ENGINEERING AND COMPUTER SCIENCE SE*. the MIT Press.

Appendix A: Head of First Appendix

Avoid Appendices if possible.

Appendix B: Head of Second Appendix

Subsection head in appendix

The equation counter is not reset in an appendix and the numbers will follow one continual sequence from the beginning of the article to the very end as shown in the following example.

$$a = b + c. \tag{63}$$