Proceedings of the 2001 IEEE
International Conference on Robotics & Automation
Seoul, Korea · May 21-26, 2001

An Iterative Approach to the Hand-Eye and Base-World Calibration Problem

Robert Lee Hirsh Guilherme Nelson DeSouza Avinash C. Kak
Purdue University
{rhirsh, gdesouza, kak}@ecn.purdue.edu

Abstract

In visual servoing applications using a position-based approach and an end-effector-mounted camera, the position and orientation of the camera with respect to the end-effector must be known. This information is frequently represented in the form of a Homogeneous Transformation Matrix (HTM). For special "noise-free" cases, a closed-form solution for this calibration problem can be determined. However, in the real world, such a solution is not adequate and a least-squares approach or an adaptive algorithm must be used. In this paper, we describe a new algorithm that can simultaneously calculate the Base-World, and the Hand-Eye (camera to end-effector) HTMs. This method is robust to noise, and converges to a valid solution very quickly.

1 Introduction

In recent years much effort has been made towards solving the Hand-Eye Calibration problem [8, 9, 7]. This problem is also known as the AX=XB problem and it was first formulated in this manner by Shiu and Ahmad [11]. In this form, the calibration problem becomes finding X, the unknown Homogeneous Transformation Matrix (HTM) that relates the End-effector coordinate frame (Hand) to the camera standard coordinate frame (Eye), while A and B are other known HTMs.

In order to solve this problem, some researchers like Tsai and Lenz [13] rewrote the equation above using only the rotation part of the HTMs and solved it using a least mean square approach. Others, like Wang [14] compared the solutions from three different calibration procedures; Chou and Kamel [3] used the quaternion representation of the HTMs and singular value decomposition; Rahardja [10] used a BFGS optimization algorithm; Horaud and Dornaika [7] also rewrote the original equation as $MX = XM^T$, where M includes the intrinsic parameters of the camera and X is solved using a nonlinear optimization algorithm. Later, in [5], they proposed both a closed form and a minimization method to simultaneously solve the hand-eye and base-world problems; Ma [9] used a self-calibration technique; and finally, Li [8] also used a nonlinear optimization technique to solve for the rotation part of the HTM. One of the problems with some of the above solutions is that they require the world to robot-base HTM to be perfectly known ahead of time. In addition, some of the previous work offer closed form solutions for the equation AX=XB (or similar), which is not adequate in the presence of noise.

In this paper, we propose an iterative approach to solve the more general form of the Hand-Eye problem where both Hand-Eye and Base-World calibrations are solved simultaneously. In our formulation $A = XBY$, where X is the hand-eye HTM and Y is the world to robot-base HTM, and both need to be calibrated. In addition, A and B are known HTMs derived from the camera calibration and the robot kinematics. Our method proved to be robust to noise, very simple to implement, and converges very quickly to within a reasonably specified tolerance (it converged in less than one second to within 0.0001 of the correct answer when run on a 350MHz PII).

2 Background

2.1 Notation

In this paper, we followed the Denavit-Hartenberg (D-H) notation [4]. In this notation, a HTM H is a 4x4 matrix that relates a homogeneous vector represented in one coordinate frame with its representation in terms of another coordinate frame. The super-script before the letter H stands for the new frame the vector is to be converted into, and the subscript after the H indicates the frame in which the vector was originally represented. For example, ${}^bH_w$ is the HTM that transforms a 4x1 homogeneous vector from the *World* coordinate frame into the *Base* coordinate frame. The

0-7803-6475-9/01/$10.00© 2001 IEEE

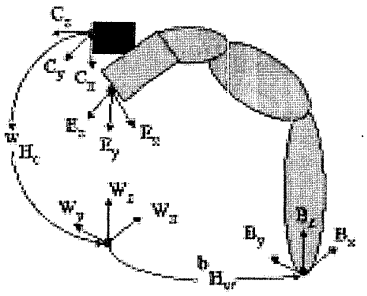

Figure 1: Coordinate Frames used: World, Base, End-Effector, Camera

inverse of this matrix is denoted by ${}^{w}H_{b}$, and it transforms a vector in the *Base* coordinate frame into the *World* coordinate frame. The same notation is applied for the rotational part of the HTMs. For example, ${}^{b}R_{w}$ is the rotation of a vector from the *World* frame into the *B*ase coordinate frame. The inverse of this matrix is denoted by ${}^{w}R_{b}$.

2.2 Quaternion Representation

Throughout this paper, quaternions are 4-dimensional unit vectors used to represent rotations of rigid objects. A quaternion can be thought of as a pair (s, v), where s is a real number representing an *amount* of rotation, and v represents the axis around which this rotation is performed. Quaternions define a closed, non-commutative field under multiplication. That is, any two quaternions can be multiplied to form a new valid quaternion, but the order in which they are multiplied can affect the result. Additionally, every quaternion has an inverse which is also a quaternion. Due to these properties, it can be shown that any rotation matrix can be specified by a quaternion, and vice versa [12, 2]. Finally, since quaternions are unit vectors, it is reasonable to consider the true measure of the size of a quaternion to be its s-component. If this component is one, then, regardless the value of the v-component, the equivalent rotation matrix is the identity matrix.

In order to summarize these and other concepts regarding quaternions, we list the following properties:

1. There are two possible and equivalent quaternion representations for a rotation of angle θ with respect to the e axis (the unit vector along the axis of rotation). That is, $q = \{s, v\} = \{-s, -v\}$, where:

$$\begin{aligned} s &= cos(\frac{\theta}{2}) \\ v &= sin(\frac{\theta}{2}) * e \end{aligned}$$

Since any quaternion with a negative s-component can be converted to an equivalent one with a positive s-component, we assume, without loss of generality, that all quaternions have positive s-components.

2. The product of two quaternions is given by:

$$\begin{aligned} q_3 &= q_1 * q_2 = (s,v) \\ &= (s_1 s_2 - v_1 \cdot v_2,\ s_1 v_2 + s_2 v_1 + v_1 \times v_2) \end{aligned}$$

3. The inverse (or complement) of q is:

$$\begin{aligned} \overline{q} &= (s, -v) \\ &= (-s,\ v) \end{aligned}$$

4. The bisector, q_3, of two quaternions, q_1 and q_2, is defined as the normalized vector average along all four dimensions (components). If the normalization factor, d, is equal to zero, then q_1 and q_2 are the inverse (complement) of each other. In that case, the bisection is the identity, and v_3 can be made the zero-vector and s_3 set to one.

$$\begin{aligned} d &= \left| \left(\frac{q_1 + q_2}{2} \right) \right| \\ q_3 &= \frac{1}{d} * \left(\frac{q_1 + q_2}{2} \right) \end{aligned}$$

3 Proposed Method

The proposed method uses an iterative algorithm to simultaneously refine the values of two matrices: ${}^{b}R_{w}$ and ${}^{c}R_{e}$. The process begins by writing three equations in the form $A = XBY$. Each equation represents the relationship of the HTMs (Fig. 1) for one configuration of the end-effector and camera. We use subscripts for each of the three end-effector and camera frames, e_i and c_i respectively, to reflect the fact that those coordinate frames are different in the three configurations. In contrast, the world, w, and the base, b, coordinate frames are the same in all frames so no subscripts are necessary.

$$\begin{aligned} {}^{c_1}R_{w} &= {}^{c_1}R_{e_1} * {}^{e_1}R_{b} * {}^{b}R_{w} \quad (1) \\ {}^{c_2}R_{w} &= {}^{c_2}R_{e_2} * {}^{e_2}R_{b} * {}^{b}R_{w} \quad (2) \\ {}^{c_3}R_{w} &= {}^{c_3}R_{e_3} * {}^{e_3}R_{b} * {}^{b}R_{w} \quad (3) \end{aligned}$$

Next, given an initial guess for ${}^{b}R_{w}$, we can compute the values for ${}^{c_1}R_{e_1}$, ${}^{c_2}R_{e_2}$, and ${}^{c_3}R_{e_3}$ that satisfy equations (1)-(3) above. But since the camera is stationary with respect to the end-effector and ${}^{c_1}R_{e_1}$, ${}^{c_2}R_{e_2}$, and ${}^{c_3}R_{e_3}$ are in fact estimates of the same rotation matrix, we "average" these three rotations to find an updated matrix that will represent ${}^{c}R_{e}$. This averaging is done via the quaternion representation of the rotations above, and the average quaternion is later converted back into its rotation matrix representation: the updated value of ${}^{c}R_{e}$. Next, this updated ${}^{c}R_{e}$ is used in equations (1)-(3) to obtain three values for the rotation matrix ${}^{b}R_{w}$. Once again, since these three values are estimates of the same rotation matrix, we average them to obtain an updated estimate for ${}^{b}R_{w}$. This process continues until all estimates have converged to the correct values – at least to within the specified tolerance. The proof of this convergence is given in the next section.

In order to make the process converge faster, a rough approximation for ${}^{b}R_{w}$ can be used, but this is not necessary. For our experiments, we used the identity matrix as the initial guess for ${}^{b}R_{w}$.

Once two rotation matrices have been determined, the only remaining portion of the HTMs left to be found are the translation vectors. This can be done in many different ways [8, 4, 13]. Our method uses an initial guess for the translation vector of ${}^{b}H_{w}$, and uses equations (4)-(6) to find three estimates for the translations of ${}^{c_i}H_{c_i}$. These three estimates are then averaged to obtain an updated value for the translation vector of ${}^{c}H_{e}$, which is in turn used to calculate three estimates for the translation vector of ${}^{b}H_{w}$. An averaging, similar to the previous one, yields an updated value for ${}^{b}H_{w}$. The process continues until all three translation vectors converge to a constant value (again, within the specified tolerance), and the algorithm halts.

$$ {}^{c_1}H_{w} = {}^{c_1}H_{e_1} * {}^{e_1}H_{b} * {}^{b}H_{w} \quad (4) $$

$$ {}^{c_2}H_{w} = {}^{c_2}H_{e_2} * {}^{e_2}H_{b} * {}^{b}H_{w} \quad (5) $$

$$ {}^{c_3}H_{w} = {}^{c_3}H_{e_3} * {}^{e_3}H_{b} * {}^{b}H_{w} \quad (6) $$

4 Proof of Convergence (Rotation part)

Since each camera configuration provides one equation of the form (1), and a general rotation matrix has three degrees of freedom, if we calibrated the camera for less than three configurations, we would obtain an equal number of equations and the system of equations would be under-constrained. In that case, there would be an infinite number of consistent solutions. Therefore, in order to uniquely determine the rotation matrix relating the end-effector and camera coordinate frames, ${}^{c}R_{e}$, simultaneously with the matrix relating world and base frames, ${}^{b}R_{w}$, the camera must be calibrated in at least three different configurations. Once the camera has been calibrated in these three configurations, the rotation matrices ${}^{c_i}R_{w}$ can be extracted from the calibration matrices (extrinsic parameters).

Now, let ${}^{b}\widehat{R}_{w}$ be an initial guess for ${}^{b}R_{w}$. We can think of ${}^{b}\widehat{R}_{w}$ as composed of two parts, one of which is the true ${}^{b}R_{w}$ and the other is a matrix A, which represents any possible errors in the guess.

$$ {}^{b}\widehat{R}_{w} = {}^{b}R_{w} * A \quad (7) $$

No matter what is the value of A, there must exist[1] an $\widehat{A_i}$ such that

$$ \begin{aligned} {}^{c_i}R_{w} &= \widehat{A_i} *^{c_i} R_{w} * A \\ &= \widehat{A_i} *^{c_i} R_{e_i} *^{e_i} R_{b} *^{b} R_{w} * A \\ &= {}^{c_i}\widehat{R}_{e_i} *^{e_i} R_{b} *^{b} \widehat{R}_{w} \end{aligned} $$

where

$$ {}^{c_i}\widehat{R}_{e_i} = \widehat{A} *^{c_i} R_{e_i} \quad (8) $$

$$ = {}^{c_i}R_{w} *^{w} \widehat{R}_{b} *^{b} R_{e_i} \quad (9) $$

and $\widehat{A_i}$ is mathematically determined by

$$ \widehat{A_i} = {}^{c_i}R_{w} * A^{-1} *^{w} R_{c_i} \quad (10) $$

And using the estimate for ${}^{b}\widehat{R}_{w}$, we can directly calculate ${}^{c_i}\widehat{R}_{e_i}$ from equation (9) above.

From equation (10), one can think of $\widehat{A_i}$ as a "compensation" for the error caused by using an A matrix that is not the identity matrix. If A is the identity matrix, that implies that the guess for ${}^{b}\widehat{R}_{w}$ has no error. In that case, our solutions for ${}^{c_i}\widehat{R}_{e_i}$ has no error as well, and $\widehat{A_i}$ is also the identity matrix[2].

At this point, a natural question that arises is how much rotation A represents in comparison to $\widehat{A_i}$. The result – proved via quaternions in Appendix A – is that the magnitude of the s-component from $\widehat{A_i}$ has exactly the same magnitude as the original s-component from A (when both matrices are considered as quaternions). On the other hand, the direction of the axis of rotation for $\widehat{A_i}$ is determined both by ${}^{w}R_{c_i}$ (from the camera calibration process) and the original A, as it is shown

[1] Euler [6] proved that any specified rotation can be obtained by applying a single rotation to any reference orientation (rotation). So letting ${}^{b}\widehat{R}_{w}$ be the specified rotation, and ${}^{b}R_{w}$ be the reference rotation, then there must exist A that satisfies (7).

[2] An important point to consider is that neither A nor $\widehat{A_i}$ can be solved directly. A is the unknown part of the initial guess for ${}^{b}\widehat{R}_{w}$ as represented in (7).

in (10). But for a given A, this axis will vary depending only on ${}^{w}R_{c_i}$.

Now consider the three quaternions q_1, q_2, and q_3, corresponding to $\widehat{A}_1$, $\widehat{A}_2$, and $\widehat{A}_3$, respectively. Each q_i represents the differences between the estimate ${}^{c_i}\widehat{R}_{e_i}$ and the true value ${}^{c}R_e$. As we mentioned above, the s-component in q_i is the same as the s-component obtained from the original A matrix. Therefore, q_1, q_2, and q_3 must have the same magnitude of rotation. When the average quaternion, $\overline{q} = \{\overline{s}, \overline{v}\}$, of these three quaternions is taken, its s-component will be larger than the original three s-components, unless the correct result has already been obtained. This assumption of increasing s-values will be explained shortly.

Since $\overline{s}$ represents a rotation of $\overline{\theta}$, and all the q_i have the same rotation of θ, then for some positive constant ϵ strictly less than one,

$$|\overline{\theta}| < \epsilon * |\theta|$$

This means that the updated ${}^{c}R_e$ has at most ϵ times the rotational error of the original guess for ${}^{b}R_w$. Next, we use the rotation matrix corresponding to $\overline{q}$ as ${}^{c}\widehat{R}_e$ (the guess for ${}^{c}R_e$), and break ${}^{c}\widehat{R}_e$ into the real ${}^{c}R_e$ times a rotational error, as was done for ${}^{b}\widehat{R}_w$. The same process described above yields a new estimate for ${}^{b}\widehat{R}_w$ that has only ϵ times the rotational error of ${}^{c}\widehat{R}_e$ (or $\overline{q}$). At this point, an estimate for ${}^{b}\widehat{R}_w$, which has only ϵ^2 times the error of the original guess for ${}^{b}\widehat{R}_w$, is obtained.

After one cycle is completed, the ${}^{b}\widehat{R}_w$ resulting from the first cycle is used as the new estimate for the next cycle, and the process continues. When this is done for n cycles, an estimate with only ϵ^{2n} times the original error will be obtained. And since $\epsilon < 1$, ϵ^{2n} will approach 0 as n tends to infinity.

Since the maximum magnitude of any rotation represented by a quaternion is π radians, the original rotational error in A has a magnitude of at most π radians. Therefore, the final rotational error will be upper bounded by $\pi * \epsilon^{2n}$. This error can be made smaller than the specified tolerance by choosing an appropriately large n. In summary, within any desired accuracy, the rotation matrices ${}^{b}R_w$ and ${}^{c}R_e$ can be obtained in a finite number of steps.

Assumption of increasing s-values

As we mentioned earlier, the average, $\overline{q} = \{\overline{s}, \overline{v}\}$, of three quaternions with equal s-components, has an s-component larger than the original ones. There are two cases that need to be investigated to prove this claim: either all q_i's are equal; or at least two q_i's have different v-components.

If all three v-components are equal, that means we have a single pair $({}^{c}\widehat{R}_e, {}^{b}\widehat{R}_w)$ that satisfies equations (1)-(3). As long as these equations are independent, they provide three constraints on the value of ${}^{b}\widehat{R}_w$, which has only three degrees of freedom. Assuming the three camera configurations are not chosen about a common axis, this condition is satisfied and there is only one solution for ${}^{b}\widehat{R}_w$ that can simultaneously solve all three equations. So, if all three q_i's are equal, then the correct value for ${}^{b}\widehat{R}_w$ has been found.

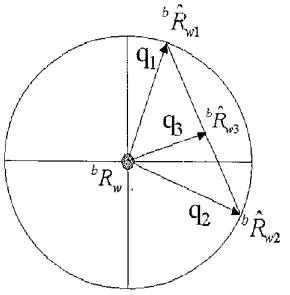

Figure 2: 2D illustration of quaternion averaging

On the other hand, if at least two v-components are different, then the axes of those two quaternions are different and so are the quaternions, even though the s-components are the same. If the bisection of these two quaternions is taken, the resulting s-component must be larger than the original s-components. Although quaternions actually have three dimensional v-components, a 2-D analogy, such as in Fig. 2, provides a good example of this principle. In this example, q_1 and q_2 have v-components, v_1 and v_2, with the same magnitude, but pointing in different directions. Since they point in different directions, the length of their average, $\frac{(v_1+v_2)}{2}$, will be strictly less than the length of either v_1 or v_2. However, because the s-components are identical, the average s will still be the same as the originals. Finally, since the average v has smaller length than either v_1 or v_2, and the s-component is the same, the nonnormalized average quaternion has size less than one. So the average quaternion needs to be normalized in order to obtain a valid unit quaternion. This means that all components are going to be multiplied by a constant greater than one and therefore the new s-component will be larger than the original one.

The only exception to this discussion above is if s is originally zero. This happens only when the initial guess ${}^{b}\widehat{R}_w$ is rotated 180 degrees with respect to the true value of ${}^{b}R_w$. In this case, the value of s never grows, and the algorithm does not converge to a valid solution. This potential problem can be eliminated by randomly choosing another guess for ${}^{b}R_w$, until the algorithm converges to a solution.

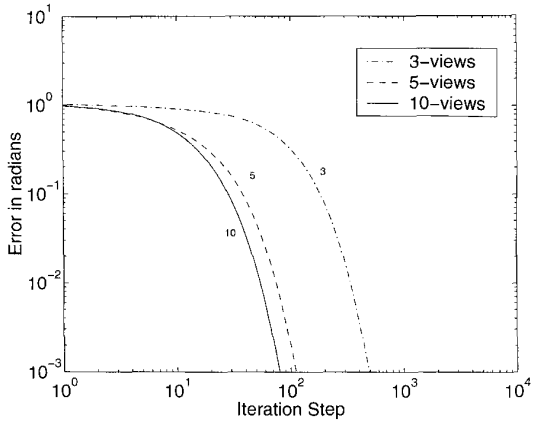

Figure 3: Error vs. Iteration using 3,5, or 10 views (without noise)

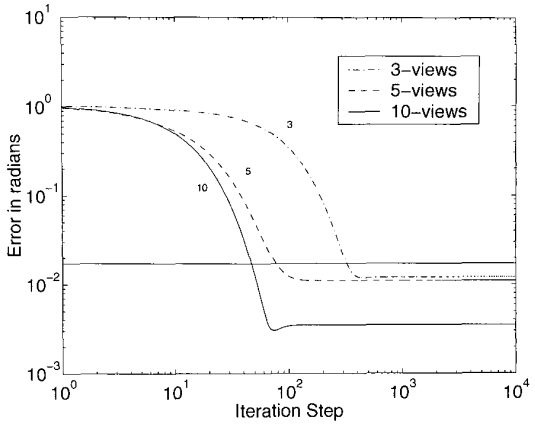

Figure 4: Error vs. Iteration using 3,5, or 10 views (with noise)

5 Results:

In order to check the algorithm, we created a test program that would calculate n theoretical ${}^{c_i}H_w$ matrices, given an arbitrary pair $({}^{b}H_w, {}^{c}H_e)$, and n ${}^{e_i}H_b$'s – one for each desired configuration of the end-effector. The calculated values for the ${}^{c_i}H_w$'s were then input into our algorithm to see if it could find the same given values for ${}^{b}H_w$ and ${}^{c}H_e$. At the same time, the test program converted the given matrix ${}^{b}H_w$ into its quaternion representation $\widetilde{q}$, which became the expected solution. Then, for each iteration of the algorithm, the test program converted the current estimate for ${}^{b}H_w$ into a quaternion, q_i, and measured the error between the two quaternions: expected, $\widetilde{q}$, and current estimate, q_i. This error measurement was performed by computing the dot-product between the two quaternions as 4-D vectors. Since quaternions have unit length, this dot-product reduces to the cosine of the angle between the two vectors, and the error can be interpreted as the angular difference between the two quaternions.

The error measurements, presented in Fig. 3, show that the values obtained for ${}^{b}H_w$ converged very quickly to the expected value. That was true for any value of ${}^{b}H_w$ used as initial guess in the algorithm. Also in Fig. 3, it is depicted the convergence of the error for various numbers of camera configurations – 3, 4, and 5-views. This information shows that in general a larger number of views improves the speed of convergence.

Despite the good results obtained above, this first test is unrealistic because it is performed using *noise-free* data. In order to test our algorithm in a more realistic situation – where the camera calibration process would be imperfect and the matrices obtained would present noise – a second set of tests was performed. For these tests, we corrupted the calculated values of the ${}^{c_i}H_w$'s with *noise*. These noisy calibration matrices were obtained by rotating each of the ${}^{c_i}H_w$'s by one degree about random axes. These random axes of rotation were selected separately for each calibration matrix.

Fig. 4 shows the results for the noisy ${}^{c_i}H_w$'s, using various numbers of camera configurations. It also displays a horizontal line representing the error originated from the one-degree noise. The result from the algorithm, as expected, converged to a value below that line.

Validity of the test method - In practice, the camera calibration process is affected by numerical errors, errors in pixel localization, quantization errors, assumptions about the camera model, etc. Ultimately, all these errors cause differences in the angles of the calibration matrix, more specifically in ${}^{c_i}R_w$. By adding random one-degree rotations to the values of the ${}^{c_i}H_w$'s, we can simulate these differences in the angles of the calibration matrices and approximate our test data to the real data more effectively.

Real data could also be collected and used for our tests. But then it would be difficult to determine the source of any measured error, since it could as well be originated from our algorithm, from our measurement of the position of the world, or from the calibration process.

Appendix A

Let A, B, and C all be quaternions, and B^{-1} be the inverse of B, which can be obtained by inverting the s component of B. So we have $A = \{s_a, v_a\}$, $B = \{s_b, v_b\}$, $C = \{s_c, v_c\}$, and $B^{-1} = \{-s_b, v_b\}$. Where the v's are 3-component vectors, and the s's are scalars.

Assuming $A = B * C * B^{-1}$, then what is the magnitude of s_a? Let $D = B * C$. Then by definition, $D = \{s_d, v_d\}$, where

$$\begin{aligned} v_d &= (v_b \times v_c) + s_b * v_c + s_c * v_b \\ s_d &= s_b * s_c - (v_b \cdot v_c) \end{aligned}$$

Finally, take $A = D * B^{-1}$. By definition,

$$s_a = s_d * (-s_b) - (v_d \cdot v_b)$$

Examine both terms individually.

$$\begin{aligned} s_d * (-s_b) &= [[s_b * s_c - (v_b \cdot v_c)] * (-s_b)] \\ &= -s_b^2 * sc + s_b * (v_b \cdot v_c) \\ (v_d \cdot v_b) &= (((v_b \times v_c) + s_b * v_c + s_c * v_b) \cdot v_b) \\ &= ((v_b \times v_c) \cdot v_b) + ((s_b * v_c) \cdot v_b) + \\ &\quad ((s_c * v_b) \cdot v_b) \\ &= 0 + s_b * (v_c \cdot v_b) + s_c * (v_b \cdot v_b) \end{aligned}$$

Since s_a is the difference of these terms:

$$\begin{aligned} s_a &= s_d * (-s_b) - (s_d \cdot s_b) \\ &= -s_b^2 * sc - s_c \ltimes (v_b \cdot v_b) \\ &= -s_c * (s_b^2 + (v_b \cdot v_b)) \\ &= -s_c * (s_b^2 + v_{b_x}^2 + v_{b_y}^2 + v_{b_z}^2) \\ &= -s_c \end{aligned}$$

Therefore, the magnitude of the rotation in A will be equal to the rotation in C if $A = B * C * B^{-1}$.

Acknowledgment

The authors would like to thank Ford Motor Company for supporting this work.

References

[1] Arvo, J. Graphics Gems II. New York: Academic Press, pp. 351-354 and 377-380, 1994.

[2] Baker, A. L. "Quaternions as the Result of Algebraic Operations", New York: Van Nostrand, 1911.

[3] Chou, J. C. K., and Kamel, M., "Finding the position and orientation of a sensor on a robot manipulator using quaternions", International Journal of Robotics Research, vol.10, no.3, June 1991, pp.240-254.

[4] Denavit, J., and Hartenberg, R., S., "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices", Journal of Applied Mechanics, vol. 77, pp. 215-221, 1955.

[5] Dornaika, F., and Horaud, R., "Simultaneous robot-world and hand-eye calibration," IEEE Transactions on Robotics & Automation, vol.14, no.4, pp.617-622, Aug. 1998.

[6] Euler, Leonhard. "Decouverte d'un nouveau principe de mécanique(1752)", Opera omnia, Ser. secucunda, v. 8, Orell Fusli Turici, Lausannae.

[7] Horaud, R., and Dornaika, F., "Hand-eye calibration," International Journal of Robotics Research, vol.14, no.3, pp.195-210, June 1995.

[8] Li, M., "Kinematic calibration of an active head-eye system," IEEE Transactions on Robotics & Automation, vol.15, no.1, Feb 1998, pp. 153-158.

[9] Ma, S., "A self-calibration technique for active vision systems," IEEE Transactions on Robotics & Automation, vol.12, no.1, Feb 1996, pp. 114-120.

[10] Rahardja, Krisnawan, "Vision-Based Bin-Picking: Recognition and Localization of Multiple Complex Objects Using Simple Visual Cues", Master Thesis, Purdue University, May 1996.

[11] Shiu, Y. C., and Ahmad, S., "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form AX=XB", IEEE Transactions on Robotics & Automation, vol.5, no.1, Feb. 1989, pp.16-29. USA.

[12] Shoemake, Ken. Animating Rotation with Quaternion Curves. Computer Graphics V.19 N.3, 1985

[13] Tsai, R. Y., and Lenz, R. K., "A new technique for fully autonomous and efficient 3-D robotics hand/eye calibration", IEEE Transactions on Robotics & Automation, vol.5, no.3, June 1989, pp.345-358. USA.

[14] Wang C-C "Extrinsic calibration of a vision sensor mounted on a robot", IEEE Transactions on Robotics & Automation, vol.8, no.2, April 1992, pp.161-75. USA.