2011 IEEE International Conference on Robotics and Automation
Shanghai International Conference Center
May 9-13, 2011, Shanghai, China

Hand-Eye Calibration Using Convex Optimization

Zijian Zhao

Abstract— **Hand-eye calibration is a very important task in robotics. Many algorithms have been proposed for it, but they almost apply the L_2 optimization, which is usually in the form of nonlinear optimization. In this paper, we propose new hand-eye calibration algorithms using convex optimization, and it can be solved in the form of a global linear optimization without starting values. Experiments with both simulated and real data are performed to test our algorithms. The experimental results show the robustness and validity of our algorithms. Considering both the computing errors and the time consuming, our algorithm based on quaternions is a good option for real applications.**

I. INTRODUCTION

When computer vision techniques are applied in robotics, such as robot assisted surgery, the cameras are usually mounted on the end-effector of a robot to form a hand-eye robot system. Using the cameras mounted on the robot, we can estimate the position of a target to grasp or to reach in camera coordinate. However, the control commands can be expressed only in the robot base coordinate system. It is necessary for us to know which is the end-effector's motion in the camera frame. Hand-eye calibration has classically been used for computing the rigid transformation from the end-effector of a robot to a camera mounted on it.

The usual way to describe the hand-eye calibration problem is by means of homogeneous transformation matrices. As shown in Fig.1, $\mathbf{A}_i$ is the transformation matrix from the camera to the world frame, and $\mathbf{B}_i$ is the transformation matrix from the robot base to the end-effector at the ith pose of the robot's "hand". Shiu and Ahmad [1] have firstly proposed the well-known hand-eye calibration equation:

$$\mathbf{AX} = \mathbf{XB}, \tag{1}$$

where $\mathbf{X}$ is the unknown transformation from camera to end-effector; $\mathbf{A} = \mathbf{A}_2\mathbf{A}_1^{-1}$ and $\mathbf{B} = \mathbf{B}_2^{-1}\mathbf{B}_1$. Every homogeneous matrix is in the form of

$$\begin{bmatrix} \mathbf{R} & \vec{\mathbf{t}} \\ \mathbf{0}^T & 1 \end{bmatrix},$$

so equation (1) can be written as the following equations,

$$\mathbf{R}_A\mathbf{R}_X = \mathbf{R}_X\mathbf{R}_B, \tag{2}$$

$$\mathbf{R}_A\vec{\mathbf{t}}_X + \vec{\mathbf{t}}_A = \mathbf{R}_X\vec{\mathbf{t}}_B + \vec{\mathbf{t}}_X. \tag{3}$$

Many approaches are proposed to solve the hand-eye calibration equation with the linear solutions. Tasi and Lenz [2] proposed an efficient linear algorithm for solving the equation (2,3), and also pointed out that at least two rotations containing motions with nonparallel rotation axes are required. Chou and Kamel [3] presented an algorithm using quaternions as the rotation representation, and gave a SVD-based linear solution. Lu and Chou [4] introduced an eight-space formulation for hand-eye calibration based on quaternions with least-squares solutions. Chen [5] utilized the screw motion theory to analyze hand-eye calibration, and indicated the constraints of hand-eye geometry. Based on Chen's work, Daniilidis [6] introduced a hand-eye calibration algorithm using dual quaternions, which is capable of determining rotation and translation simultaneously with a SVD approach. A method based on screw motions, very similar to Daniilidis' algorithm, was proposed by Zhao and Liu [7].

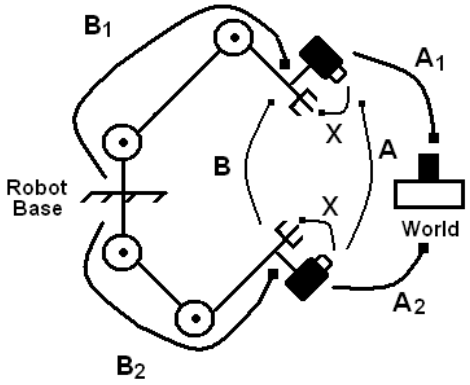


Fig. 1. Hand-eye calibration by solving $\mathbf{AX} = \mathbf{XB}$.

Considering the influence of random noise, many people applied the optimization in hand-eye calibration. Zhuang and Shiu [8] applied nonlinear optimization for both rotational and translational parts, using a similar formulation to Frobenius norms as the cost function. A similar optimization strategy was presented by Fassi and Legnani [9]. Park and Martin [10] added the canonical coordinates to hand-eye calibration, and simplified the characterization of parameters in optimization. Horaud and Dornaika [11] performed nonlinear optimization with the rotational parameters formulated in quaternions. They also introduced how to solve robot-world and hand-eye transformation simultaneously in nonlinear optimization [12]. Strobl and Hirzinger [13] proposed a new metric on the rigid transformations $SE(3)$ for nonlinear optimization in hand-eye calibration, and the metric has the ability of automatic optimal weighting. Although all works

This work has been supported by French National Research Agency (ANR) through TecSan program (project DEPORRA nANR-09-TECS-006).

Z. Zhao is with the GMCAO group of the UJF-Grenoble 1 / CNRS / TIMC-IMAG UMR 5525, Grenoble, F-38041, France, Zijian.Zhao@imag.fr.

achieved good results, they have a common disadvantage: without a good initial value, the optimization process may get stuck in local minima .

We try to apply the convex optimization [14] to hand-eye calibration in this paper. Convex optimization is actually a kind of global optimization. It is not necessary to find an initial value for the optimization process. Our method was implemented and compared with other methods using local optimization, showing its good performance. In the applications with real data, our method also proved its validity.

This paper is organized as follows. In Section II, we analyze the past optimization based methods and find the differences between local optimization and convex optimization. Section III describes how our method is implemented to convex optimization in two different algorithms. Experiments with both simulated and real data are carried out in Section IV. Section V concludes this work.

II. Problem Description and Analysis

The past hand-eye calibration algorithms ([8], [9], [10], [11], [12], [13]) based on optimization have all applied the local optimization techniques such as Newton-based methods. Their objective functions usually have the following form:

$$f = \sum_{i=1}^{N} (\|r_i\|^2 + \lambda \|t_i\|^2), \tag{4}$$

where r_i is the rotation error term, t_i is the translation error term; N is the number of error terms in the problem, λ is the weight factor; $\|\cdot\|$ is a distance metric on the Euclidean group of rigid motions $SE(3)$. For example, Horaud and Dornaika [11] optimize nonlinearly the following function

$$\begin{aligned} f(\mathbf{q}, \vec{\mathbf{t}}) = & \sum_{i=1}^{N} \|\mathbf{v}_i' - \mathbf{q} * \mathbf{v}_i * \bar{\mathbf{q}}\|_2^2 + \\ & \lambda \sum_{i=1}^{N} \|\mathbf{q} * \mathbf{p}_i * \bar{\mathbf{q}} - (\mathbf{K}_i - \mathbf{I})\vec{\mathbf{t}} - \mathbf{p}_i'\|_2^2, \end{aligned} \tag{5}$$

with the constraint $\|\mathbf{q}\|_2 = 1$. Minimizing a sum-of-squares objective function as (4) is known to be a troublesome non-convex optimization problem in computer vision ([15], [17]). It usually has the phenomena of multiple minima, which cause the difficulties (stuck in local minima) for local optimization techniques. That is why a good initial value is necessary for nonlinear optimization of (5).

The sum-of-squares objective function is actually a kind of L_2-norm error formulation, and it has been widely used in multi-view geometry, for example the triangulation problems [17]. Although the L_2-norm error has the statistical meaning in some multi-view problems under the disturbance of independent Gaussian noise, we can not make the same conclusion for hand-eye calibration problems. For L_2-norm based minimization, most hand-eye calibration algorithms ([8], [9], [10], [11], [12], [13]) utilized nonlinear optimization, which is very time-consuming. Kahl et al. ([15], [16]) applied the $L_\infty-$norm in the global optimization of multi-view geometry, and showed its good performances over the L_2-norm. In the light of Kahl's work, we use the $L_\infty-$norm based optimization in hand-eye calibration.

The L_∞ optimization of hand-eye calibration is actually the minimax problem

$$\min_{\mathbf{x}} \max_{i} f_i(\mathbf{x}), \tag{6}$$

where $f_i(\mathbf{x})$ is the error term corresponding to the ith robot's motion ($i = 1, 2, \ldots, N$), variable $\mathbf{x}$ contains all the hand-eye transformation parameters. If each $f_i(\mathbf{x})$ is a quasi-convex function [14] on a convex domain D, and the minimization takes place on D, then the optimization of (6) can be converted to a convex optimization problem. We will introduce how to construct the function $f_i(\mathbf{x})$ and apply the convex optimization in the next section.

III. Hand-Eye Calibration Problems Solved in L_∞ Optimization

There are two ways to formulate the rotation part of the hand-eye transformation parameters, one is by orthonormal matrix, the other is by quaternion. Then we have two possible approaches to construct the function $f_i(\mathbf{x})$ and apply L_∞ Optimization in the hand-eye calibration problem.

A. Formulation By Orthonormal Matrix

For constructing the convex function $f_i(\mathbf{x})$, we have to introduce the Kronecker product $\otimes$ and the column vector operator vec [18] firstly. If $\mathbf{M}$ is a $m \times n$ matrix, $\mathbf{N}$ is a $o \times p$ matrix, then the Kronecher product $\mathbf{M} \otimes \mathbf{N}$ is defined as

$$\mathbf{M} \otimes \mathbf{N} = \begin{bmatrix} M_{11}\mathbf{N} & \cdots & M_{1n}\mathbf{N} \\ \vdots & \ddots & \vdots \\ M_{m1}\mathbf{N} & \cdots & M_{mn}\mathbf{N} \end{bmatrix}, \tag{7}$$

the column vector operator is defined as

$$vec(\mathbf{M}) = [M_{11}, \ldots, M_{1n}, \ldots, M_{21}, \ldots, M_{mn}]^T. \tag{8}$$

Using the properties of equation (7,8), we can rewrite the equation (2,3) for all robot motions ($i = 1, 2, \ldots, N$) as the linear system (also seen in [19]):

$$\begin{bmatrix} \mathbf{I}_9 - \mathbf{R}_{Ai} \otimes \mathbf{R}_{Bi} & \mathbf{0}_{9\times 3} \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{Bi}^T & \mathbf{I}_3 - \mathbf{R}_{Ai} \end{bmatrix} \begin{bmatrix} vec(\mathbf{R}_X) \\ \vec{\mathbf{t}}_X \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{9\times 1} \\ \vec{\mathbf{t}}_{Ai} \end{bmatrix} \tag{9}$$

Suppose

$$\mathbf{x} = \begin{bmatrix} vec(\mathbf{R}_X) \\ \vec{\mathbf{t}}_X \end{bmatrix},$$

$$\mathbf{C}_i = \begin{bmatrix} \mathbf{I}_9 - \mathbf{R}_{Ai} \otimes \mathbf{R}_{Bi} & \mathbf{0}_{9\times 3} \\ \mathbf{I}_3 \otimes \vec{\mathbf{t}}_{Bi}^T & \mathbf{I}_3 - \mathbf{R}_{Ai} \end{bmatrix},$$

$$\mathbf{d}_i = \begin{bmatrix} \mathbf{0}_{9\times 1} \\ \vec{\mathbf{t}}_{Ai} \end{bmatrix},$$

then we can construct the error function

$$f_i(\mathbf{x}) = \|\mathbf{C}_i\mathbf{x} - \mathbf{d}_i\|_2, \tag{10}$$

which is obviously a convex function.

The problems in hand-eye calibration that we will be considering in this paper may be written in the following form:

$$\min_{\mathbf{x}} \max_{i} \|\mathbf{C}_i\mathbf{x} - \mathbf{d}_i\|_2. \tag{11}$$

It is easily seen that this problem may be transformed, by introducing an additional variable δ, into an equivalent problem of the form

$$\begin{aligned} &\min_{\delta,\mathbf{x}} \delta, \\ &\text{subject to } \|\mathbf{C}_i\mathbf{x} - \mathbf{d}_i\|_2 \leq \delta, \\ &\text{for } i = 1, \ldots, N. \end{aligned} \tag{12}$$

The particular convex optimization problem for solving equation (12) is called *second-order cone program* (SOCP). The SOCP problem is easily solvable using commonly available tool-boxes, for example SeDuMi [20].

B. Formulation By Quaternion

If we represent rotation by a unit quaternion $\mathbf{q}$, we should rewrite equation (2,3) in another form. There are two possible methods to reformulate equation (2,3). One method is using dual quaternions (seen in [6]). The other method is using screw motion constraint (seen in [7]). We take the first method as an example. By using dual quaternions method, equation (2,3) can be transformed to

$$\mathbf{a}_i * \mathbf{q} = \mathbf{q} * \mathbf{b}_i, \tag{13}$$

$$\mathbf{a}'_i * \mathbf{q} + \mathbf{a}_i * \mathbf{q}' = \mathbf{q} * \mathbf{b}'_i + \mathbf{q}' * \mathbf{b}_i, \tag{14}$$

where all symbols are denoted by quaternions, and $\mathbf{q}' = \frac{1}{2}\mathbf{t} * \mathbf{q}$, $i = 1, 2, \ldots, N$. In the equation (13,14), the scalar part is redundant. Therefore, we can have the following linear system:

$$\begin{bmatrix} \vec{\mathbf{a}}_i - \vec{\mathbf{b}}_i & Skew(\vec{\mathbf{a}}_i + \vec{\mathbf{b}}_i) & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times3} \\ \vec{\mathbf{a}'}_i - \vec{\mathbf{b}'}_i & Skew(\vec{\mathbf{a}'}_i + \vec{\mathbf{b}'}_i) & \vec{\mathbf{a}}_i - \vec{\mathbf{b}}_i & Skew(\vec{\mathbf{a}}_i + \vec{\mathbf{b}}_i) \end{bmatrix} \begin{bmatrix} \mathbf{q} \\ \mathbf{q}' \end{bmatrix} = \mathbf{0}, \tag{15}$$

where the symbol $\vec{\mathbf{l}}$ is the vector part of quaternion $\mathbf{l}$, $Skew(\vec{\mathbf{l}})$ is the antisymmetric matrix corresponding to the cross-product with $\vec{\mathbf{l}}$. Using the second method [7], we can also get a similar linear system to (15).

Suppose

$$\mathbf{C}_i = \begin{bmatrix} \vec{\mathbf{a}}_i - \vec{\mathbf{b}}_i & Skew(\vec{\mathbf{a}}_i + \vec{\mathbf{b}}_i) & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times3} \\ \vec{\mathbf{a}'}_i - \vec{\mathbf{b}'}_i & Skew(\vec{\mathbf{a}'}_i + \vec{\mathbf{b}'}_i) & \vec{\mathbf{a}}_i - \vec{\mathbf{b}}_i & Skew(\vec{\mathbf{a}}_i + \vec{\mathbf{b}}_i) \end{bmatrix},$$

$$\mathbf{x} = \begin{bmatrix} \mathbf{q} \\ \mathbf{q}' \end{bmatrix},$$

then we can have the error function

$$f_i(\mathbf{x}) = \|\mathbf{C}_i\mathbf{x}\|_2. \tag{16}$$

The L_∞ optimization problem is becoming to the following form

$$\min_{\mathbf{x}} \max_{i} \|\mathbf{C}_i\mathbf{x}\|_2. \tag{17}$$

Obviously, $\mathbf{x} = \mathbf{0}$ is one solution of the equation (17), but it is meaningless in practice. Therefore, we should define a constraint of $\mathbf{x}$ to avoid the zero point. With the additional constraint, the optimization of (17) can be considered as a real convex optimization problem. According to the unit quaternion constraint $\|\mathbf{q}\|_2 = 1$ and some apriority knowledge, we can define a linear constraint in the following form

$$\mathbf{D}\mathbf{x} \geq \mathbf{f}. \tag{18}$$

By introducing an additional variable δ, the L_∞ optimization problem has an equivalent form with the linear constraint,

$$\begin{aligned} &\min_{\delta,\mathbf{x}} \delta, \\ &\text{subject to } \|\mathbf{C}_i\mathbf{x}\|_2 \leq \delta, \\ &\text{for } i = 1, \ldots, N \text{ with } \mathbf{D}\mathbf{x} \geq \mathbf{f}. \end{aligned} \tag{19}$$

The optimization problem of (19) is also a SOCP problem, and can be solved by SeDuMi [20].

C. Motion Data Selection

From the equation (12,19), we can see that the motion data selection [21] can be done naturally during the L_∞ optimization, if a pre-defined threshold e is set for the optimization error. Taking the equation (12) as the example, we can give the algorithm of motion selection:

Algorithm 1: **Given**: the motion data set $\mathbf{C} = \{\mathbf{C}_i, i = 1, 2, ...\}$ and $\mathbf{d} = \{\mathbf{d}_i, i = 1, 2, ...\}$, and the threshold e;

Repeat

Solve the SOCP problem (12);

For all the ith motion,

If $\|\mathbf{C}_i\mathbf{x} - \mathbf{d}_i\|_2 > e$,

Then remove the ith motion from $\mathbf{C}$ and $\mathbf{d}$;

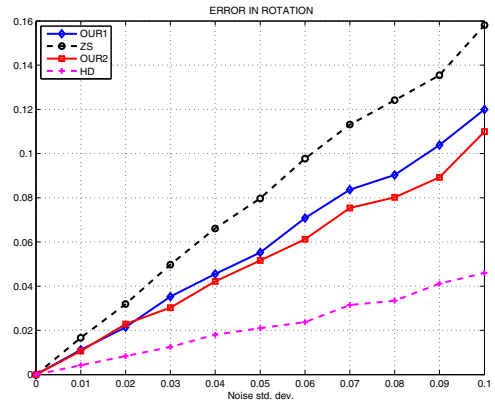
Until $\delta \leq e$.

The above algorithm of motion selection has the direct meaning on geometry and algebra, then it can be understood easily.

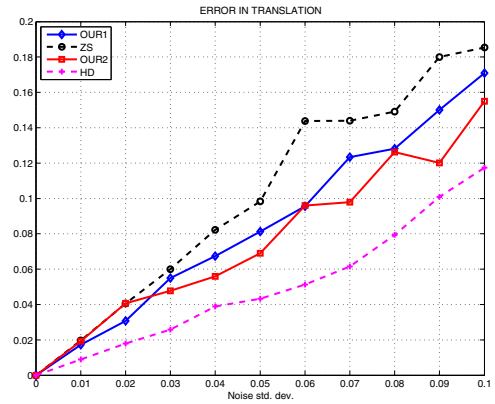
IV. Experiments

In this section, we performed simulations and real experiments to test our method. To compare its performance, we implemented two additional methods from the literatures. The first one is the method proposed by Horaud and Dornaika [11], and is based on the simultaneous computation of rotation and translation without concerning camera parameters. Like every iterative nonlinear minimization algorithm, it must be provided with an initial value. To do so, we particularly applied Daniilidis' method [6] to give the initial value for nonlinear optimization. The second method we applied is the one-stage iterative algorithm described by Zhuang and Shiu [8].

In the following experiments and figures, we denote our method described in Section III-A by "OUR1", our method in the form of quaternion (Section III-B) by "OUR2", the first additional method by "HD", and the second additional method by "ZS".



(a)



(b)

Fig. 2. Errors in rotation and translation with variation in noise level.

A. Simulated Experiments

In the simulations, we synthesized 10 hand motions $(\mathbf{R}_B, \vec{\mathbf{t}}_B)$, and added Gaussian noise of standard deviation σ corresponding to the angle reading. We assumed a hand-eye setup and computed the camera motions $(\mathbf{R}_A, \vec{\mathbf{t}}_A)$ according to it. Gaussian noise with standard deviation σ is also added to the camera motions. We varied the noise standard deviation σ from 0 to 0.1; each method ran $N(N = 100)$ times at every noise level σ and outputed the estimated rotation matrix $\tilde{\mathbf{R}}_X$ and the estimated translation $\tilde{\vec{\mathbf{t}}}_X$ of hand-eye transformation parameters. To qualify the results, we computed the errors associated with rotation and translation as follows:

$$e_R = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \|\mathbf{R}_X - \tilde{\mathbf{R}}_X\|^2}, \tag{20}$$

$$e_t = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N} \|\vec{\mathbf{t}}_X - \tilde{\vec{\mathbf{t}}}_X\|_2^2}}{\|\vec{\mathbf{t}}_X\|_2}, \tag{21}$$

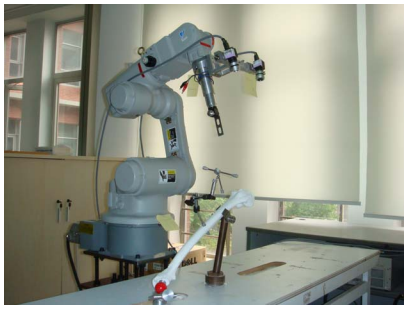
where $(\mathbf{R}_X, \vec{\mathbf{t}}_X)$ is the ground truth.

In Fig.2, we can compare our algorithms' behavior (OUR1 and OUR2) with HD algorithm's and ZS algorithm's under different noise levels with 10 hand-eye motions. As shown in the figure, the algorithms by using unit quaternion exhibit better performance than the algorithms by using orthonormal matrix. That is also why the formulation by quaternion is so widely used in computer vision and robotics. Although the HD algorithm shows smaller errors than our algorithms, it mainly depends on the wonderful starting values provided by Daniilidis' method. Nonlinear minimization without the suitable initial values can not get the good results. That is why the ZS algorithm has the worst performance.

TABLE I

THE RUNNING TIME OF ALL ALGORITHMS.

ALgorithm	Running time 10×100's trials	Average running time
OUR1	193.502632s	0.1935s
ZS	2664.277239s	2.6643s
OUR2	202.474709s	0.2025s
HD	5886.325242s	5.8863s



(a)

(b)

Fig. 3. Robot systems in the real experiments. (a) MOTOMAN HP3 robot with two PointGrey flea2 CCD cameras. (b) EndoControl ViKY robot with an Olympus endoscope

We used MATLAB V7.6 to program all algorithms. Our computer system configuration is as follows: Pentium 4 CPU 3.00 GHz, 1G RAM. We recorded the running time of all algorithms' 10×100 trials respectively, as shown in Table.I. From the table, we can see that Our two algorithms have nearly the same running time, which is less than one tenth of HZ's running time and one twentieth of HD's running time. Our algorithms are much less time consuming than the other two algorithms. Fully Considering the two aspects of error performance and time complexity, we think our algorithms should be good options for hand-eye calibration.

B. Real Experiments

The real experiments utilized two different robot systems. The first one is a MOTOMAN HP3 robot with two PointGrey flea2 CCD cameras mounted on its last joint (seen in Fig.3(a)). The second one is an EndoControl ViKY robot with an Olympus endoscope mounted on its end-effector (seen in Fig.3(b)). Two independent assessments are conducted by using the robot systems in Fig.3.

In the first experiment, we used the first robot system, as shown in Fig.3(a). The cameras were moved to 15 different locations with respect to a calibration object. The 14 camera's motions $\mathbf{A}_i (i = 1, \ldots, 14)$ were computed. We solved the hand-eye transformation $\mathbf{X}$ by using the four algorithms (OUR1, OUR2, HD and ZS) based on the camera and robot motion 1 through 8. Then we predicted the camera motion $\hat{\mathbf{A}}_i (i = 9, \ldots, 14)$ from the robot motion $\mathbf{B}_i (i = 9, \ldots, 14)$ in the following form:

$$\hat{\mathbf{A}}_i = \mathbf{X}\mathbf{B}_i\mathbf{X}^{-1}. \tag{22}$$

We compared $\hat{\mathbf{A}}_i$ and $\mathbf{A}_i$ $(i = 9, \ldots, 14)$, and computed the errors of all motions (from 9 to 14) in rotation and translation. As shown in Fig.4, both quaternion-based algorithms (OUR2 and HD) performed better in rotation and translation. There is also some difference from simulated experiments: OUR2 algorithm was slightly superior to HD algorithm, that is because the real noise maybe does not satisfy the Gaussian distribution that we used in simulated experiments. The main source of errors might be located in the camera's pose extraction.

In the second real experiment, the robot system seen in Fig.3(b) was utilized. We controlled the robot to move 16 times through its controller. At each time, one image of the calibration object (a chess board) and one pose of the robot were captured. By using these 16 images, we could calibrate the endoscope in Zhang's method [22], and get the endoscope's poses with respect to the calibration object. Based on these camera's and robot's poses, the 15 motions of the endoscope and the robot could be computed. Because the EndoControl robot is a kind of medical device, the movement range of its end-effector is smaller than the MOTOMAN robot. Therefore, the motion data we have is not well-suited as described in Schmidt and Niemann's paper [21].

We solved the hand-eye transformation $\mathbf{X}$ by using the four algorithms (OUR1, OUR2, HD and ZS) based on the camera and robot motion 1 through 8. Then we estimated the camera motions (from 9 to 15) according to the equation (22). Using these estimated camera transformations, we could reconstruct the 3D positions of the chess board's corner points in the structure from motion algorithm [17], seen in Fig.5. We took the reconstruction result using the real camera transformations as the ground truth. As shown in the figure, we can see that there are nearly no differences between quaternion based algorithms' results (OUR2,HD) and the ground truth. We also computed the absolute reconstruction errors of each grid's edges in length. Table.II gives the running times and the average errors for all testing algorithms. The experiment

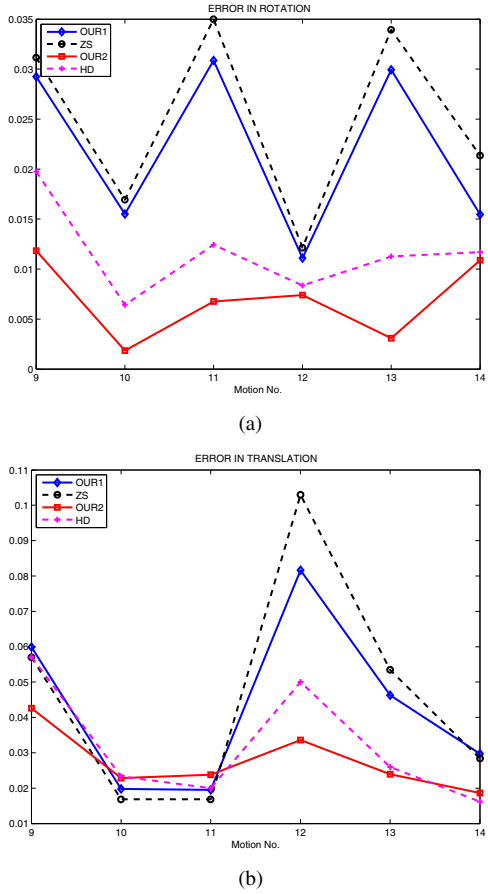


Fig. 4. Error results in the first real experiment.

TABLE II
THE AVERAGE RECONSTRUCTION ERRORS IN THE GRID'S EDGES AND THE RUNNING TIMES OF ALL ALGORITHMS .

ALgorithm	Average reconstruction error (mm)	Running time (s)
OUR1	3.5579	0.1347
ZS	3.4385	0.4766
OUR2	2.4157	0.1805
HD	2.6559	2.0115

is based on the ill-condition data, so the experiment results can reflect the robustness and validity of all testing algorithms. In Table.II, we can see that both quaternion-based algorithms (OUR2 and HD) performed better as before, and OUR2 algorithm was a little superior to HD algorithm. With considering the time complexity, OUR2 algorithm is the best option for real applications.

C. Discussion

In this section, we will discuss the computation complexity of optimization. Suppose there are N robot motion constraints, and the number of variable parameters is n. For solving the nonlinear optimization, we take the LevenbergMarquardt algorithm (LMA) as an example. As described in the reference [23], the iterative update of LMA can be solved in $O(n^3)$ without considering any special

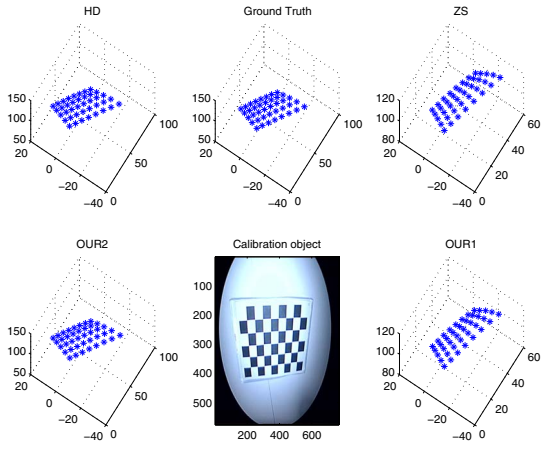


Fig. 5. Reconstruction results in the second real experiment.

acceleration algorithms. For solving the SOCP based optimization, we take the primal-dual interior-point method [24] as an example. In every iteration step of this method, a set of linear equations is solved, and then the plane search process is carried on. The cost of solving linear systems is $O(n^3)$, the plane search is actually a nonlinear minimization with only two variable parameters; so the iteration's cost is roughly equal to $O(n^3)$. It seems that the two kinds of optimizations have similar computation complexity in iterations; however, the LMA always needs a starting value before iterations, and the cost of computing it has something to do with N and n. Therefore, we can see the reasons why our SOCP based algorithms have the shorter running times than other algorithms by using nonlinear optimization.

V. Conclusion

We have proposed a new hand-eye calibration method using convex optimization. Unlike the past algorithms, our algorithms utilize the $L_\infty-$norm optimization instead of theL_2-norm optimization. Then we can apply a kind of linear global optimization–convex optimization to solve the hand-eye calibration problems. It is less time consuming and needs no initial values that the nonlinear optimization should have. The performance of our algorithms was evaluated in the simulated and real experiments. Experimental results showed the robustness and validity of our algorithms. Considering both computing errors and time complexity, our algorithm based on quaternions is a very good option for real applications.

VI. Acknowledgments

The author would like to thank Sandrine Voros, Philippe Cinquin, Shen Cai and Gael Le-Bellego for their technical supports.

References

[1] Y. C. Shiu, S. Ahmad, "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form AX=XB," *IEEE Trans. Robot. Autom.*, vol.5, no.1, pp.16-29, 1989.

[2] R. Y. Tsai, R. K. Lenz, "A new technique for fully autonomous and efficient 3D robotics hand/eye calibration," *IEEE Trans. Robot. Autom.*, vol.5, no.3, pp.345-358, 1989.

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

[4] Y. C. Lu, J. C. Chou, "Eight-space quaternion approach for robotic hand-eye calibration," in *Proc. IEEE Intl. Conf. Systems, Man and Cybernetics*, 1995, pp.3316-3321.

[5] H. H. Chen, "A screw motion approach to uniqueness analysis of head-eye geometry," in *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, 1991, pp.145-151.

[6] K. Daniilidis, "Hand-eye calibration using dual quaternions," *International Journal of Robotics Research*, vol.18, no.3, pp.286-298, 1999.

[7] Z. Zhao, Y. Liu, "A hand-eye calibration algorithm based on screw motions," *Robotica*, vol.27, no.2, pp.217-223, 2009.

[8] H. Zhuang, Y. C. Shiu, "A noise-tolerant algorithm for robotic hand-eye calibration with or without sensor orientation measurement," *IEEE Trans. Systems, Man and Cybernetics*, vol.23, no.4, pp.1168-1175, 1993.

[9] I. Fassi, G. Legnani, "Hand to sensor calibration: a geometrical interpretation of the matrix equation AX=XB," *Journal of Robotic Systems*, vol.22, no.9, pp.497-506, 2005.

[10] F. C. Park, B. J. Martin, "Robot sensor calibration: solving AX=XB on the Euclidean group," *IEEE Trans. Robot. Autom.*, vol.10, no.5, pp.717-721, 1994.

[11] R. Horaud, F. Dornaika, "hand-eye calibration," *International Journal of Robotics Research*, vol.14, no.3, pp.195-210, 1995.

[12] F. Dornaika, R. Horaud, "Simultaneous robot-world and hand-eye calibration," *IEEE Trans. Robot. Autom.*, vol.14, no.4, pp.617-622, 1998.

[13] K. H. Strobl, G. Hirzinger, "Optimal hand-eye calibration," in *Proc. IEEE/RSJ Conf. Intelligent Robots and Systems*, 2006, pp.4647-4653.

[14] S. Boyd, L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[15] F. Kahl, R. Hartley, "Multiple view geometry under the L-infinity norm," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.30, no.9, pp.1603-1617, 2008.

[16] F. Kahl, S. Agarwal, M. Chandraker, D. Kriegman, S. Belongie, "Practical global optimization for multiview geometry," *International Journal of Computer Vision*, vol.79, no.3, pp.271-284, 2008.

[17] R. Hartley, A. Zisserman, *Multiplue View Geometry in Computer Vision*, Cambridge University Press, 2004.

[18] H. Neudecker, "A note on kronecker matrix product and matrix equation systems," *SIAM Journal of Applied Mathematics*, vol.17, no.3, pp.603-606, 1969.

[19] N. Andreff, R. Horaud, B. Espiau, "On-line hand-eye calibration," in *Proc. 2nd Intl. Conf. 3-D Digital Imaging and Modeling*, 1999, pp.430-436.

[20] J. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol.11-12, pp.625-653, 1999.

[21] J. Schmidt, H. Niemann, "Data selection for hand-eye calibration: a vector quantization approach," *International Journal of Robotics Research*, vol.27, no.9, pp.1027-1053, 2008.

[22] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.22, no.11, 2000.

[23] I. Christian, T. Marc, W.S Wan, "Rprop using the natural gradient," *International Series of Numerical Mathematics*, Vol. 151, pp.259-272, 2005.

[24] Y. Nesterov, A. Nemirovsky, "Interior-point polynomial methods in convex programing," *Studies in Applied Mathematics* SIAM, vol.13, 1994.