A New Approach to Hand-Eye Calibration*

Henrik Malm and Anders Heyden
Centre for Mathematical Sciences, Lund University,
Box 118, S-221 00 Lund, Sweden
email: henrik,heyden@maths.lth.se

Abstract

Traditionally, hand-eye calibration has been done using point correspondences, reducing the problem to a matrix equation. This approach requires reliably detected and tracked points between images taken from fairly widespread locations. In this paper we present a new approach to performing hand-eye calibration. The novelty of the proposed method lies in the fact that instead of point correspondences, normal derivatives of the image flow field are used. Firstly, two different small translational motions are made, enabling the direction of the optical axis to be computed from image derivatives only. Secondly, at least two different rotational motions are made, enabling also the translational part of the hand-eye transformation to be estimated. It is also shown how to compute a depth reconstruction from the information obtained in the hand-eye calibration algorithm. Finally, we discuss how to calculate the derivatives and present some experiments on synthetic data.

1. Introduction

The topic of hand-eye calibration has been studied for several years. The problem consists of determining the relative orientation between the camera coordinate system and a fixed world coordinate system. When the camera is mounted on a robot gripper, it is sufficient to determine the relation between the camera coordinate system and the coordinate system of the robot gripper, since the relation between this system and a fixed world coordinate system is known, via the robot kinematics. The main reason for studying this problem is that it gives the flexibility to mount a camera on a robot hand without measuring the position of the camera, which is sometimes impossible, since the camera coordinate system is not known in relation to the physical camera. When this relation is known the vision system can guide the robot motions and the robot can position the camera at specific locations, which are favorable for obtaining information about the surrounding world.

There are several different formulations of the hand-eye calibration problem. In this paper we will assume that the camera is pre-calibrated and that the kinematics of the robot is known. The problem of both recovering the hand-eye calibration and the robot-hand calibration has been treated in [3, 15, 14]. The classical approach to treat hand-eye calibration is to use (i) a known reference object (calibration object) and (ii) track points on this reference object in order to obtain corresponding points between pairs of images. This approach leads to the study of the equation $AX = XB$, where A, X and B denote 4×4 matrices representing Euclidean transformations. A and B denote the transformations between the first and second position of the robot hand (in the robot coordinate system) and the camera (in the camera system - estimated from point correspondences) respectively and X denotes the transformation between the hand coordinate system and the camera coordinate system, i.e. the hand-eye calibration, see [12, 13, 2, 4].

The hand-eye calibration problem can be simplified considerably by using the possibility to move the robot in a controlled manner and investigating the arising motion field of points in the images. In [9] this fact has been exploited by first only translating the camera and using the focus of expansion in order to obtain the rotational part of the hand-eye transformation. The method for finding the translational part is also presented, but it relies on the ability to detect and track points in the surrounding world.

In this paper we will use a novel approach to deal with the hand-eye calibration problem. Firstly, we will take advantage of the possibility to control the motion of the robot hand and secondly, we will use small motions and calculate the normal flow field instead of tracking feature points. This approach has two great advantages: (i) we need no calibration grid, (ii) we need not to track feature points. Furthermore, the estimation of the hand-eye calibration can be simplified considerably by using controlled motions of the robot hand. Firstly, when only translating the hand in two different directions the rotational part can be estimated.

*Sponsored by the Swedish Research Council for Engineering Sciences (TFR), project 95-64-222

Secondly, using this estimate of the rotational part and making rotational motions, the translational part can be estimated. We like to stress that we are not using an estimation of the motion field obtained from the optical flow. We are only using image derivatives, giving the so-called normal flow field. The idea to use only the normal flow has been used in [1] to make (intrinsic) calibration of a camera. Preliminary investigations on how to do a hand-eye calibration from normal derivatives have been reported in [10]. In this paper we continue this work by investigating how the approximations of the temporal derivatives can be improved by using multiple images in each direction of translation and by exploring the possibility of doing a dense depth reconstruction of the current 3D scene. Improvements in the implementation have also been made to make the results of the experiments more accurate.

2. Preliminaries

Throughout this paper we represent the coordinates of a point in the image plane by small letters (x, y) and the coordinates in the world coordinate frame by capital letters (X, Y, Z). In our work we use the pinhole camera model as our projection model. That is the projection is governed by the following equation

$$x = \frac{X}{Z}, \quad y = \frac{X}{Z} \ . \tag{1}$$

The hand-eye calibration problem boils down to finding the transformation $H = (R, t)$ between the robot hand coordinate system and the camera coordinate system. In the general case this transformation has 6 degrees of freedom, 3 for the position, defined by the 3-vector t and 3 for the orientation, defined by the orthogonal matrix R. We will solve these two parts separately, starting with the orientation.

Our algorithm for hand-eye calibration uses the notion of the normal flow, i.e. the orthogonal projection of the motion field onto the image gradient. Let $E(x, y, t)$ be the intensity at point (x, y) in the image plane at time t. Let $u(x, y)$ and $v(x, y)$ denote components of the motion field in the x and y directions respectively. Using the constraint that the gray-level intensity of the object is (locally) invariant to the viewing angle and distance we obtain the optical flow constraint equation, c.f. [6, 5],

$$E_x u + E_y v + E_t = 0 \ , \tag{2}$$

where

$$u = \frac{\delta x}{\delta t}, \quad v = \frac{\delta y}{\delta t} \ , \tag{3}$$

which denote the motion field.

3. The proposed solution

The hand-eye transformation will be obtained from at least 4 different motions. A reference position of the robot hand, where a reference image is obtained by the camera, will be used. This reference position we use as a common point for the different motions and by this we can effectively eliminate the depth Z from our calculations. We will first obtain the orientation of the camera and after that the position of the camera in relation to the robot hand. This order is important since we use the information about the direction of the optical axis with respect to the robot hand in the second part.

3.1. Orientation

To calculate the orientation of the camera we will, as mentioned above, use purely translational motions. The motion field (u, v) in the image plane that arises from a translation $D = (D_X, D_Y, D_Z)$ in the camera coordinate system is given by

$$\begin{cases} u = \dot{x} = \dfrac{\dot{X}}{Z} - \dfrac{X\dot{Z}}{Z^2} = -\dfrac{1}{Z}(D_X - xD_Z), \\ v = \dot{y} = \dfrac{\dot{Y}}{Z} - \dfrac{Y\dot{Z}}{Z^2} = -\dfrac{1}{Z}(D_Y - yD_Z) \ . \end{cases} \tag{4}$$

This way of expressing the motion field has previously been described similar way in, e.g. [8, 5]. Plugging u and v into equation (2) gives, after multiplication with Z,

$$-E_x D_X - E_y D_Y + (E_x x + E_y y) D_Z + E_t Z = 0 \ . \tag{5}$$

Assume that the image has N pixels. We got one equation of type (5) for each pixel. We also got a different depth Z in each pixel, but the vector D is the same throughout the whole image. That is, we got $N + 3$ unknowns and N equations. To get more equations of the form (5), a new translation direction $\bar{D} = (\bar{D}_x, \bar{D}_y, \bar{D}_z)$ is chosen. The new translation should have the reference position, where the spatial derivatives are calculated, in common with the previous translation. Using the motion field of the new motion together with the optical flow constraint equation (2) gives,

$$-E_x \bar{D}_X - E_y \bar{D}_Y + (E_x x + E_y y)\bar{D}_Z + \bar{E}_t Z = 0 \ . \tag{6}$$

Notice that the spatial derivatives E_x and E_Y is the same in equation (5) and (6), but the time derivative E_t has changed to $\bar{E}_t$. Also notice that the depth parameter Z is the same in (5) and (6). Pairing together these two equations and eliminating Z gives

$$\begin{aligned} &- E_x \bar{E}_t D_X - E_y \bar{E}_t D_Y + (E_x x + E_y y)\bar{E}_t D_Z + \\ &E_x E_t \bar{D}_X + E_y E_t \bar{D}_Y - (E_x x + E_y y) E_t \bar{D}_Z = 0 \ . \end{aligned} \tag{7}$$

Now we have N equations but only 6 unknowns. These unknowns represent the directions in camera coordinate system of the translation directions of the robot hand. From these relations of the directions it is an easy task to compute the rotational part of the hand-eye transformation.

The idea of using what is known about the motion field and plugging it into the optical flow constraint equation has previously been exploited in [11, 7], where a passive approach to the motion estimation is considered.

3.2. Position

Translations of the robot hand will not give any information about the position of the camera, i.e. the translation between the robot hand and the focal point of the camera, as Ma also pointed out in [9]. To obtain this translation we instead need to use motions containing also a rotational part. The instantaneous velocity of a point in the camera coordinate system under rotation can be written as

$$\begin{cases} \dot{X} = -\Omega_Y(Z - T_Z) + \Omega_Z(Y - T_Y) \\ \dot{Y} = -\Omega_Z(X - T_X) + \Omega_X(Z - T_Z) \\ \dot{Z} = -\Omega_X(Y - T_Y) + \Omega_Y(X - T_X) \end{cases} , \tag{8}$$

where $\Omega = (\Omega_X, \Omega_Y, \Omega_Z)$ is the direction of the rotation axis. The rotation axis will not cross the origin of the camera coordinate system but instead cross a point translated with the vector $T = (T_X, T_Y, T_Z)$ from the origin. We will choose this point as the origin of the robot hand coordinate system, which makes T exactly the unknown translation of the hand-eye calibration that we want to compute. Using the first part of (4) together with (8) gives the motion field

$$\begin{cases} u = U + x\Omega_Y \frac{T_X}{Z} - (x\Omega_X + \Omega_Z)\frac{T_Y}{Z} + \Omega_Y \frac{T_Z}{Z} \\ v = V + (\Omega_Z + y\Omega_Z)\frac{T_X}{Z} - y\Omega_X \frac{T_Y}{Z} - \Omega_X \frac{T_Z}{Z} \end{cases} , \tag{9}$$

where

$$\begin{cases} U = \Omega_X xy - (1 + x^2)\Omega_Y + \Omega_Z y \\ V = \Omega_X(1 + y^2) - \Omega_Y xy - \Omega_Z x \end{cases} . \tag{10}$$

This field will be plugged into the optical flow constraint equation (2). Since we know the axis of rotation, U and V are known. Introduce the notation

$$E'_t = E_t + E_x U + E_y V \ . \tag{11}$$

E'_t is a known quantity and the motion field (9) plugged into (2) can be written as (after multiplication with Z)

$$AT_X + BT_Y + CT_Z + ZE'_t = 0 \ , \tag{12}$$

where

$$\begin{cases} A = E_x\Omega_Y x + E_y(\Omega_Z + y\Omega_Y) \\ B = -E_x(x\Omega_X + \Omega_Z) - E_y\Omega_X y \\ C = E_x\Omega_Y - E_y\Omega_X \end{cases} . \tag{13}$$

Now we want to eliminate the depth Z, as when calculating the orientation, but because we are interested in the length of T we do not want a homogeneous system of equations. By moving the center of the rotation a known distance $e = (e_X, e_Y, e_Z)$ away from the robot hand and choosing a new rotation axis $\bar{\Omega}$ we get the new equation

$$\bar{A}(T_X + e_X) + \bar{B}(T_Y + e_Y) + \bar{C}(T_Z + e_Z) + Z\bar{E'}_t = 0 \ . \tag{14}$$

Note that $\bar{\Omega}$ can be chosen equal to $\bar{\Omega}$. By subtracting (12) and (14) we eliminate the depth and receive

$$(\bar{E}'_t A - E'_t \bar{A})T_X + (\bar{E}'_t B - E'_t \bar{B})T_Y + (\bar{E}'_t C - E'_t \bar{C})T_Z = E'_t(\bar{A}e_X + \bar{B}e_Y + \bar{C}e_Z) \ . \tag{15}$$

Now we have a system with N equations and 3 unknowns. By solving this system we get the translational part of the hand-eye transformation expressed in the camera coordinate system. The coordinates for the translation vector in robot hand coordinate system could easily be obtained if wanted.

4. Depth Reconstruction

After calculating the orientation of the camera, the obtained directions of translations, D and $\bar{D}$, can be used to calculate a depth map of the current scene. By solving equation (5) for Z we get

$$Z = \frac{E_x D_X + E_y D_Y - (E_x x + E_y y)D_Z}{E_t} \ . \tag{16}$$

An obvious difficulty with this equation is that if the temporal derivative E_t is zero or near zero in some pixel, the depth will approach infinity and the depth reconstruction becomes unstable and unreliable. Possible causes for $E_t \approx 0$ can of course be that the point to be reconstructed is indeed located far away and that the movement of the camera compared to the distance to the point is very small, or that the point to be reconstructed lies near the focus of expansion.

The most serious aspect of this problem, however, probably is that the depth in large homogeneous areas will be impossible to obtain. The difference of intensity due to the movement of the camera will be near zero if the point is an interior point of such an area. Note that also the spatial derivatives, E_x and E_y, also will be zero in this type of area and that usage of the optical flow constraint equation will fail altogether. The problem also occurs in textured areas which contains lines of similar grey-value and periodic textures as will be seen in the experiments section.

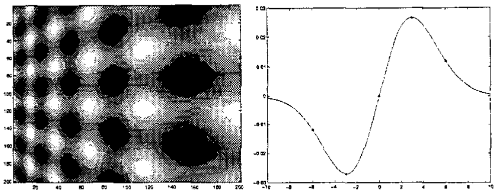

Figure 1. An image from the computer generated sequence and a Gaussian used to calculate the temporal derivatives.

$t = 0.1$	$t = 0.2$	$t = 0.3$	$t = 0.5$
0.6665	0.6660	0.6650	0.6677
0.3371	0.3487	0.3568	0.3861
0.6650	0.6594	0.6561	0.6365
0.7057	0.7018	0.6976	0.6847
0.7085	0.7124	0.7164	0.7287
0.0015	0.0051	0.0082	0.0388

Table 1. Results of motion estimations using the synthetic image data.

$\sigma_n = 0.01$	$\sigma_n = 0.02$	$\sigma_n = 0.05$	$\sigma_n = 0.1$
0.6662	0.6651	0.6561	0.6328
0.3491	0.3494	0.3611	0.3931
0.6590	0.6599	0.6627	0.6671
0.7015	0.7000	0.6976	0.6879
0.7126	0.7141	0.7165	0.7258
0.0050	0.0071	0.0046	0.0023

Table 2. Results of motion estimations using the synthetic image data with added noise.

5. The Calculation of the Derivatives

The spatial derivatives in the equations can be calculated by convolution with derivatives of a 2D Gaussian kernel of width σ. That is, $E_x = E * G_x$ and $E_y = E * G_y$ where

$$G_x = -\frac{1}{2\pi\sigma^4} x e^{-\frac{x^2+y^2}{2\sigma^2}}, \quad G_y = -\frac{1}{2\pi\sigma^4} y e^{-\frac{x^2+y^2}{2\sigma^2}} \quad . \tag{17}$$

On the synthetic image sequence used in the experiments in the next section, a higher value of σ almost always gave a better result. For real images, with more variation in depth and finer detail, this will probably not hold.

In [10] the temporal derivatives were calculated using only two subsequent images for each motion. The approximation of the derivatives are improved by using more images for each motion, especially in noisy image sequences. We construct a difference approximation of the derivatives by weighting the images in the motion sequence by numbers taken from the derivative of a 1D Gaussian kernel. In Figure 1 (right) the Gaussian, with indicated equidistant weights, for a motion sequence with five subsequent images, is shown.

6. Experiments

Experiments has been carried out on a synthetic image sequence. The scene consists of the plane $Z + \frac{X}{2} = 10$ in camera coordinate system. The texture on the plane is described by the function $I(x, y) = \sin(x) + \sin(y)$ in a coordinate system of the plane with origin at $O = (0, 0, 10)$ in the camera coordinate system, see Figure 1. Results of motion estimations to find the orientation of the camera coordinate system is shown in Table 1. The actual translation vectors are: $D = (0.6667, 0.3333, 0.6667)$, and $\bar{D} = (0.7071, 0.7071, 0)$. The width of the Gaussian kernels (both spatial and temporal) was here set to $\sigma = 1$ for the translations of length $t = 0.1$ and $t = 0.2$, $\sigma = 2$ for $t = 0.3$ and $\sigma = 3$ for $t = 0.5$. The unit of t is the unit of the camera coordinate system. That is, the distance in this unit to the plane is equal to 10. A translation of $t = 0.15$ approximately amounts to an apparent movement of one pixel in the image sequence. To test the noise sensitivity of the algorithm we have added different amounts of Gaussian distributed noise with standard deviation σ_n to the image sequences. The length of the translation was set to $t = 0.2$ and the width of the Gaussian kernels to $\sigma = 1$. The intensity span in the images is from -2 to 2, so a standard deviation of $\sigma = 0.05$ corresponds to approximately 3 grey-levels in an image with 256 grey-levels. The results are presented in Table 2.

Experiments on the synthetic image sequence to test the position part of the algorithm shows that it performs well for rotations around axes parallel to the optical axis. That is, we set $\Omega = \bar{\Omega} = (0, 0, 1)$. This can be done since we from the orientation part know the direction of the optical axis. In this way we can obtain good estimates of T_X and T_Y. For example, when rotating $\pi/240$ radians and using $T = (1, 1, 1)$ and $e = (3, 2, 1)$, we get $T_X = 1.0011$ and $T_Y = 1.0254$. By instead using $T = (7, 9, 5)$ and $e = (3, 1, 0)$ we get $T_X = 7.0805$ and $T_Y = 9.1768$. The influence of e on the result has not yet been evaluated. T_Z, on the other hand, seems rather hard to obtain accurately. An iterative search scheme may be necessary to apply along the optical axis to obtain a good result for this component.

Experiments on depth reconstruction was performed on the synthetic image sequence using three translations, D, $\bar{D}$ and $\bar{\bar{D}}$. The three different depth maps, Z, $\bar{Z}$ and $\bar{\bar{Z}}$,obtained from these translations contained lines and curves of large errors, since the the temporal derivative was approximately zero in those areas, due to the specific texture. These errors can be effectively reduced by calculating a weighted mean of the three depth maps, where the weights are functions of the corresponding temporal derivatives.

$$Z_m = \frac{Zw(E_t) + \bar{Z}w(\bar{E}_t) + \bar{\bar{Z}}w(\bar{\bar{E}}_t)}{w(E_t) + w(\bar{E}_t) + w(\bar{\bar{E}}_t)} \ . \tag{18}$$

Figure 2 shows, to the left, the resulting depth map Z_m. This map still contains some outliers. In the middle image, Z_m has been convolved with a Gaussian kernel with a rather small width ($\sigma = 2$). This can be motivated by the assumption that the depth changes smoothly in the image. Of course, more sophisticated methods to remove the outliers could be applied. To the right, the theoretical depth map is presented.

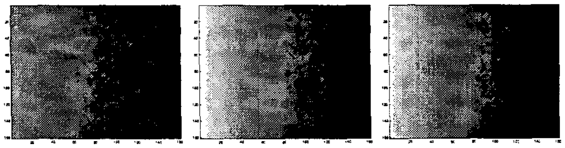

Figure 2. 2D depth maps of the synthetic scene. See text for details.

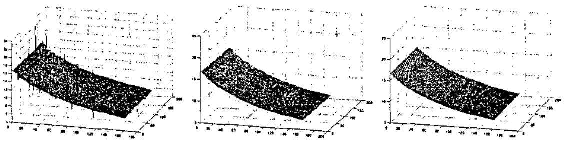

Figure 3. 3D depth maps of the synthetic scene. See text for details.

7. Conclusions

We have presented a direct method for hand-eye calibration that only uses the intensity derivatives in each pixel in an image sequence. No feature extraction or calibration object is needed using this approach. The method uses, in the minimal case, two translational motions and two rotational motions. As a by-product we could calculate an approximate depth map of the scene.

Results of experiments on a computer generated image sequence shows that the method works quite well on smooth images. By using information from multiple images for each motion when approximating the temporal derivatives, the algorithm can handle noise quite well. It remains to test the algorithm thoroughly on real images and to improve the calculation of the component along the optical axis of the translational part of the hand-eye transformation.

References

[1] T. Brodsky, C. Fernmuller, and Y. Aloimonos. Self-calibration from image derivatives. In *Proc. Int. Conf. on Computer Vision*, pages 83–89. IEEE Computer Society Press, 1998.

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

[3] F. Dornaika and R. Horaud. Simultaneous robot-world and hand-eye calibration. *IEEE Trans. Robotics and Automation*, 14(4):617–622, 1998.

[4] R. Horaud and F. Dornaika. Hand-eye calibration. In *Proc. Workshop on Computer Vision for Space Applications, Antibes*, pages 369–379, 1993.

[5] B. K. P. Horn. *Robot Vision.* MIT Press, Cambridge, Mass, USA, 1986.

[6] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[7] B. K. P. Horn and E. J. Weldon Jr. Direct methods for recovering motion. *Int. Journal of Computer Vision*, 2(1):51–76, June 1988.

[8] H. C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. In *Proc. Royal Society of London B*, volume 208, pages 385–397, 1980.

[9] S. D. Ma. A self-calibration technique for active vision systems. *IEEE Trans. Robotics and Automation*, 12(1):114–120, February 1996.

[10] H. Malm and A. Heyden. Hand-eye calibration from image derivatives. In *Proc. European Conf. Computer Vision, Dublin, June 2000.*

[11] S. Negahdaripour and B. K. P. Horn. Direct passive navigation. *IEEE Trans. Pattern Analysis Machine Intelligence*, 9(1):168–176, January 1987.

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

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

[14] H. Zhuang, Z. S. Roth, and R. Sudhakar. Simultaneous robot/world and tool/flange calibration by solving homogeneous transformation equation of the form $ax = yb$. *IEEE Trans. Robotics and Automation*, 10(4):549–554, 1994.

[15] H. Zhuang, K. Wang, and Z. S. Roth. Simultaneous calibration of a robot and a hand-mounted camera. *IEEE Trans. Robotics and Automation*, 11(5):649–660, 1995.