Camera Calibration based on Circular Markers

Liang Li, Hanming Chen
School of Software
Beijing Institute of Technology
Beijing, China
Corresponding Author Email: hanming@bit.edu.cn

Guixiang Wang
School of Aerospace Engineering
Beijing Institute of Technology
Beijing, China

Abstract—In this paper, we focus on camera calibration problem of circular markers. We propose the iterative method to calculate the coordinates of tangency points, and then get the camera parameters due to the imaging model. To reduce the error, we preprocess the original image using LoG edge Detection. At last, we simulate and test this method. The results show that the method has a higher accuracy.

Keywords-Camera calibration; Circular marker; LoG algorithm; Simulation

I. INTRODUCTION

Three-dimensional reconstruction has a wide range of applications in machine vision [1], virtual reality and artificial intelligence [2]. And the key of three-dimensional reconstruction lies in that the two-dimensional positions correspond to three [3]. The correspondence is decided by the imaging model as well as the parameters of the camera itself. Generally, imaging model is fixed, so the key is how to get the parameters, namely, camera calibration.

The camera calibration method has been deeply studied for the past decades in the photogrammetry [4] and machine vision [1]. Many researchers put forward a variety of calibration methods [5]. Generally, camera calibration can be divided into two categories: traditional methods of camera calibration [6] and camera self-calibration [7] [8]. Because of its high calibration precision, the traditional calibration method is widely used in the situation, of which the extrinsic and intrinsic parameters are relatively fixed.

Generally, this method uses fixed square markers, which may not make full use of the image information. Moreover it's not conducive to the further optimization when processing the image [9].

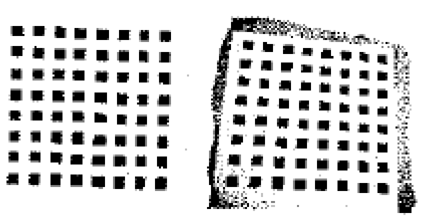

Figure 1. General markers (left) and its image (right)

In this paper, we consider circular markers as calibration objects. First we process the image to obtain the common tangents for every two circles, and then according to these points of tangency, we establish imaging model and calculate the camera parameters. For any two circles, four common tangents can be found at most. For n circles, we can get $4*C_n^2=2*n*(n-1)$ at most. And using **iterative algorithm,** we propose, we can obtain these common tangents and increase their accuracy due to the characteristics of circles.

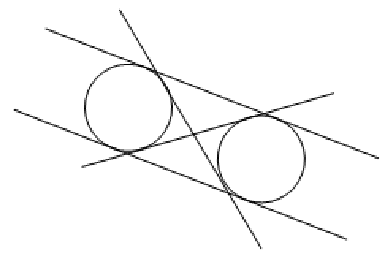

Figure 2. Circular markers

II. MODELLING

A. Problem Analysis and Basic Approach

According to the direct linear transformation [4] and imaging model [5] we have

$$(x_i', y_i', w_i') = (x_i, y_i, w)\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \tag{1}$$

where

$p_i=(x_i, y_i)$ denotes the coordinates of reference points in the image plane,

$p_i^{'}=(u_i, v_i)$ denotes the coordinates of reference points in the target plane,

$V=\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$ is transformation matrix,

with $u_i=\dfrac{x_i'}{w_i'}$, $v_i=\dfrac{y_i'}{w_i'}$, $a_{13}\neq 0$, $a_{23}\neq 0$, $w=1$.

Then we can get

$$u_i=\frac{a_{11}x_i+a_{21}y_i+a_{31}}{a_{13}x_i+a_{23}y_i+a_{33}} \tag{2}$$

$$v_i = \frac{a_{12}x_i + a_{22}y_i + a_{32}}{a_{13}x_i + a_{23}y_i + a_{33}} \tag{3}$$

If we have the coordinates of reference points and substitute them into the equations (2) and (3), we can get the transformation matrix V. If the points are enough, we can establish overdetermined equations and get many matrices. And then we can obtain the optimum solution through least square fitting, which can be more precise. So we should get reference points first.

B. *Iterative Algorithm*

In this paper, we consider tangency points as reference points. In an image that contains n circles, we can get $4*n*(n-1)$ tangency points at most. Here we propose **iterative algorithm** to obtain the coordinates of these points.

Algorithm Description:

1) Pick points A, B respectively from Circle a and b;

2) Connect A, B to create a line and to divide both Circle a and b into two parts;

3) Calculate the two parts of Circle a, and get the midpoint A' of the shorter arc in order to update A, as shown in Fig. 3;

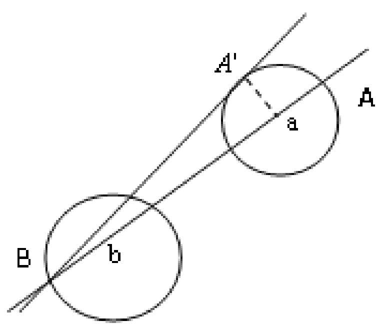

Figure 3. Update A

4) Process Circle b in the same way, and update B;

5) Test whether it meets the requirements of tangent (e.g. calculate the area of each part respectively); if so, return tangency points A, B, or jump to 2).

Here in *5)* we can set up requirements of different levels for different applications. It's easy to control.

III. MODEL OPTIMIZATION

A. *Accuracy Analysis*

Before calculating the points of tangency, we need to know the precise edge of the graph. The more precise the edge is, the more accurate the result will be.

There may be serious noises in the area close to the edge. It will affect, even submerge the original edge. Besides, the exterior disturbance exists at times. It will react to the image pixel value, bringing error to the pixel extraction at last.

Therefore, as image pre-processing, we use the Gaussian filter to eliminate noise interference, and then use the LoG algorithm to extract the edge contour precisely.

B. *Gaussian Filter*

We use Gaussian filter as preprocessing. The Gaussian function is

$$g(i,j) = e^{-\frac{(i^2+j^2)}{2\sigma^2}} \tag{4}$$

where σ^2 denotes variance, i and j denote the point coordinates of the image plane.

The key of filter design is to produce an appropriate convolution template. According to Gaussian function, we can calculates the template's weight

$$\frac{g'(i,j)}{c} = e^{-\frac{(i^2+j^2)}{2\sigma^2}} \tag{5}$$

Choose the appropriate σ^2 value, then we can obtain the template.

Suppose that $h(i,j)$ denotes the pixel value after convolution and $g(x,y)$ denotes Gaussian function. We have

$$h(i,j) = \frac{1}{N}(f(i,j) * g(i,j)) \tag{6}$$

where $N = \sum_{i=-m}^{m}\sum_{j=-m}^{m} g(i,j)$.

C. *LoG Edge Detection*

LoG is an algorithm that combines Gaussian filter and Laplacian edge detection. Here LoG operator's output $h(i,j)$ can be obtained from the convolution operation:

$$h(i,j) = \nabla^2(g(x,y) * f(x,y)) \tag{7}$$

According to convolution derivation principle, (7) becomes:

$$h(i,j) = (\nabla^2 g(x,y)) * f(x,y) \tag{8}$$

where $\nabla^2 g(x,y) = (\frac{x^2+y^2-2\sigma^2}{\sigma^4})e^{-\frac{x^2+y^2}{2\sigma^2}}$, $f(i,j)$ denotes the original pixel value and $g(x,y)$ denotes Gaussian function. Select an appropriate σ^2, and we can get a suitable filtering operator template $\nabla^2 g(x,y)$.

IV. SIMULATION

A. *Problem Description*

We use the target shown in Fig. 4 to test our model. The length of the square is 100 mm. A, C, D and E are vertices of the square and the center of the circles respectively. B is also the center of that circle, which is 30 mm from A in the line AC. The radii of these circles are 12 mm.

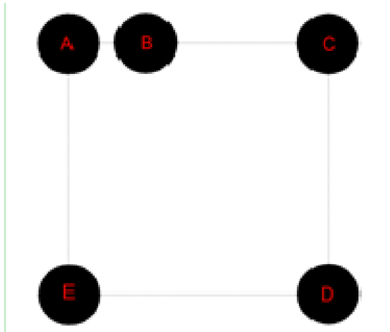

Figure 4. The target

We photograph the calibration object with some camera and get the image, as shown in Fig. 5. The resolution of this camera is 1024×768 (3.78 pixels/mm).

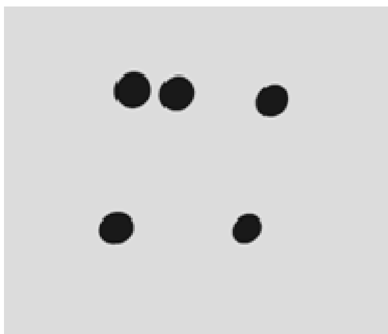

Figure 5. The image

B. *Preliminary Result*

We take the circles' 4 external common tangents and their points of tangency in the image plane, as shown in Fig. 6.

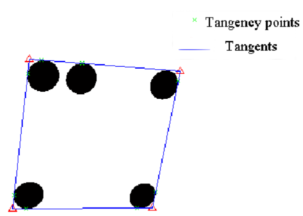

Figure 6.

Fig. 7 is the tangent obtained by the iterative algorithm.

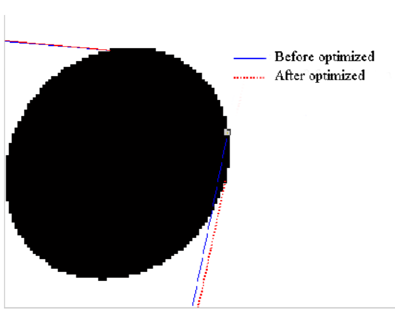

Figure 7. Iterative effect

Using the transformation functions (2) and (3), we get the transformation matrix

$$V = \begin{pmatrix} 0.9756 & 0.0979 & 0.0007 \\ -0.0451 & 0.9429 & 0.0008 \\ 75.69 & 38.78 & 1.0 \end{pmatrix}$$

We also get the coordinates for A_o, C_o, D_o and E_o, as shown in Table I.

TABLE I. COORDINATES

Points	A_o	C_o	D_o	E_o	B_o
X/mm	-50.13	33.52	18.52	-60.25	-23.80
Y/mm	51.22	45.64	-31.37	-31.34	49.47
Z/mm	417.20	417.20	417.20	417.20	417.20

C. *Optimization*

To increase the computational accuracy, we carry on Gauss filter processing as well as the LoG edge detection on the image at the cost of time. For Gaussian filter, we set $\sigma^2 = 2, m = 3$, and obtain 7×7 convolution template:

$$\begin{pmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{pmatrix}$$

Optimization results are shown in Fig. 8 and Fig. 9.

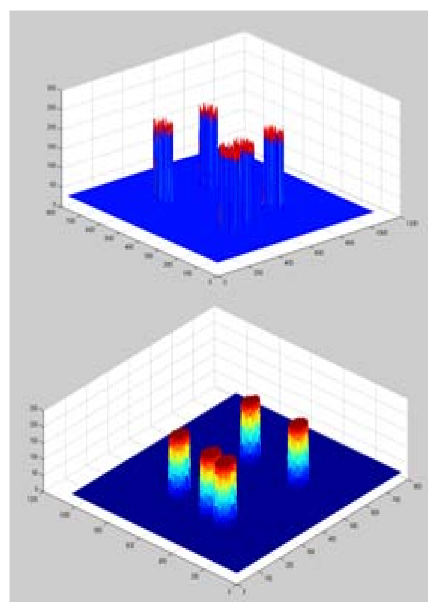

Figure 8. The effect of Gaussian filter

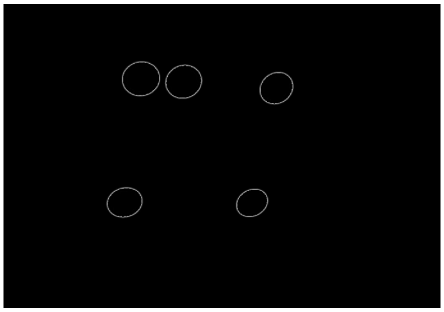

Figure 9. Edge detection by LoG

At last we get the transformation matrix

$$V = \begin{pmatrix} 1.005 & 0.1183 & 0.0009 \\ -0.0407 & 0.9588 & 0.0008 \\ 75.77 & 38.14 & 1.0 \end{pmatrix}$$

D. Model Test

We use another 4 inner common tangents to test the accuracy of our model, as shown in Fig. 10.

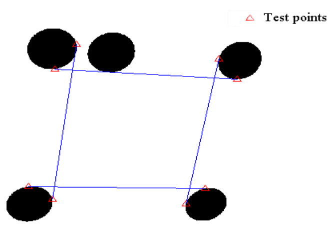

Figure 10. Testing

Suppose that (u_i, v_i) denotes the coordinates of tangency points and (u_i', v_i') denotes their coordinates obtained by the matrix, then we get error β_i:

$$\beta_i = \sqrt{(u_i - u_i')^2 + (v_i - v_i')^2} \tag{9}$$

The test results are shown in Fig. 11.

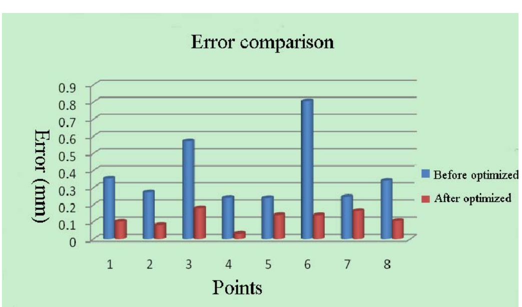

Figure 11. Error comparison

From the test results, we can see that the error has been reduced after the optimization. It falls from the original millimeter-level to the pixel-level.

Noting that we only make use of 4 tangents to calculate the transformation matrix in the simulation above, we can increase the tangents and the points of tangency. Then we can establish overdetermined equations and get many matrices. We can obtain the optimum solution through least square fitting, the accuracy of which may be 0.01 pixels level or even higher.

V. Conclusion

In conclusion, this paper analyzes the traditional calibration methods, most of which use square markers. However, we make some innovation and propose a calibration method based on circular markers. We propose iterative algorithm in order to get common tangents precisely. Therefore we can obtain the camera parameters by the imaging model.

In order to obtain the desired accuracy, we adopt a Gaussian filter as well as LoG algorithm on the original image as preprocessing to obtain accurate edges. If reference points are enough, we can establish overdetermined equations and get many matrices. Therefore we can obtain the optimum solution by using least square fitting, which can be more precise.

At last, we set up a simulation to test the method. The results show that this method has high accuracy.

References

[1] R. Y. Tsai, An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, USA, 1986.

[2] O. Faugeras, Three-Dimensional Computer Vision: a Geometric Viewpoint. MIT Press, 1993.

[3] T. Moons, L. V. Gool, M. Proesmans. Affine reconstruction from perspective image pairs with a relative object-camera translation in between, IEEE PAMI, 1996.

[4] Y. I. Abdel-Aziz, H. M. Karara, Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. Proceedings of the Symposium on Close-Range Photogrammetry. 1971.

[5] M. Qiu, S. Ma, Y. Li, Overview of camera calibration for computer vision. Acta Automatica Sinica. 2000,26(1).

[6] J. Heikkilä, O. Silvén, A four-step camera calibration procedure with implicit image correction, in Proc. IEEE Comput. Soc. Conf. Computer Vision and Pattern Recognition (CVPR'97), 1997.

[7] S. D. Ma, A self-calibration technique for active vision system. IEEE Trans on Robotics and Automation, 1996, 12(1).

[8] F. Du, M. Brady, Self-calibration of the intrinsic parameters of camera for active vision system. In: Proc IEEE Conference on Computer Vision and Pattern Recognition, 1993.

[9] P. A. Beardsley, D. W. Murray, A. Zisserman, Camera calibration using multiple images, Proc. Second European Conf. Computer Vision, 1992.