Camera calibration using adaptive segmentation and ellipse fitting

Raúl Romaní Flores * **Paul Alonzo, Quio Añamuro** **

* *Universidad Católica San Pablo (e-mail: raul.romani@ucsp.edu.pe).*
** *Universidad Católica San Pablo (e-mail: paul.quio@ucsp.edu.pe).*

Abstract: We describe an algorithm that Detects the control points in the raw images for camera calibration for ring calibration patterns. The algorithm was implemented in c++ 11 and using opencv library. The proposed algorithm for finding the control points consist of four parts, Apply a mask that encloses the control points in order to reduce the work area, Apply adaptive threshold in order segment the image, find contours and find ellipses. We also implemented an algorithm for drawing lines(independent of rotation), in a specific order, starting from the top-left to the top-right and going down. We have conducted an set of experiments with recorded videos and with two cameras in real time.

Keywords: Camera calibration, threshold, integral image, contours, ellipse.

1. INTRODUCTION

In many machine vision applications, a crucial step is to accurately determine the relation between the image of the object and its physical dimension by performing a calibration process. Over time, various calibration techniques have been developed. Nevertheless, the existing methods cannot satisfy the ever-increasing demands for higher accuracy performance. In this paper, we propose an algorithm for control points detection that is used in the first stages of camera calibration techniques. We use a ring calibration patterns (Fig. 1). The proposed algorithm for finding the control points consist of four parts, Apply a mask that encloses the control points in order to reduce the work area, Apply adaptive threshold in order segment the image, find contours and find ellipses. We also implemented an algorithm for drawing lines(independent of rotation), in a specific order, starting from the top-left to the top-right and going down.

We run our algorithm in 4 videos recorded with 5 different cameras (kinnect v2, mslifecam, ps3eyecam and real sense) and two of them was used to test in real time. In order to implement this algorithm we read some theory about Zhang [2000], Datta et al. [2009], and taken Prakash and Karam [2012] as a base for the creating the algorithm. Additionally we made some test of real time pattern detection using the Ps3EyeCam and MsLifeCam getting an average of 48 pfs.

2. ALGORITHM

The camera calibration algorithm for ring calibration patterns implemented in c++ 11 and opencv library. The proposed algorithm for finding the control points consist of four parts, Apply a mask that encloses the control points in order to reduce the work area, Apply adaptive threshold in order segment the image, find contours and find ellipses. We also implemented an algorithm for drawing lines(independent of rotation), in a specific order, starting from the top-left to the top-right and going down. Those ordered control points will be used later to calibrate the camera. We will describe both algorithms in detail in the next two subsections.

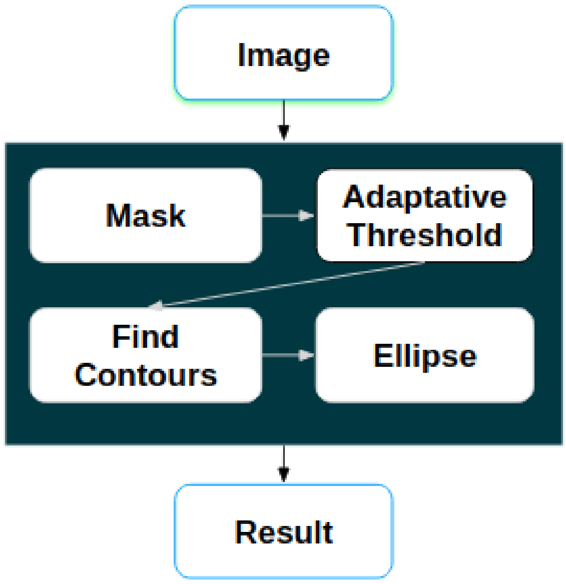

Fig. 1. Pipeline for finding control points

2.1 Mask

Apply a mask that encloses the control points in order to reduce the work area. This mask is computed and applied to the input image after we have found 20 control points, we computed this mask using minAreaRect(opencv function), this function return a rotated rectangle, we use this a mask.

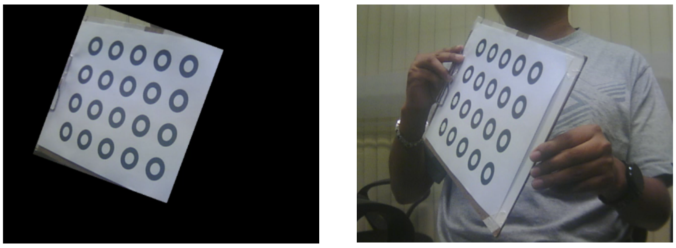

Fig. 2. At left we can see a good masking after found the 20 pattern points. If we didnt find the 20 pattern points, open the mask(right).

2.2 Adaptative Threshold

Preprocess the input image/video by usign adaptive thresholding using the integral image based on Bradley and Roth [2007].

In the paper cited above doesn't say what we should do with the image borders so we usse the opencv threshold to complete that information.

Additionally we calculate the adaptive threshold using opencv to complement the information of the edges with the obtained by the threshold implemented previously.

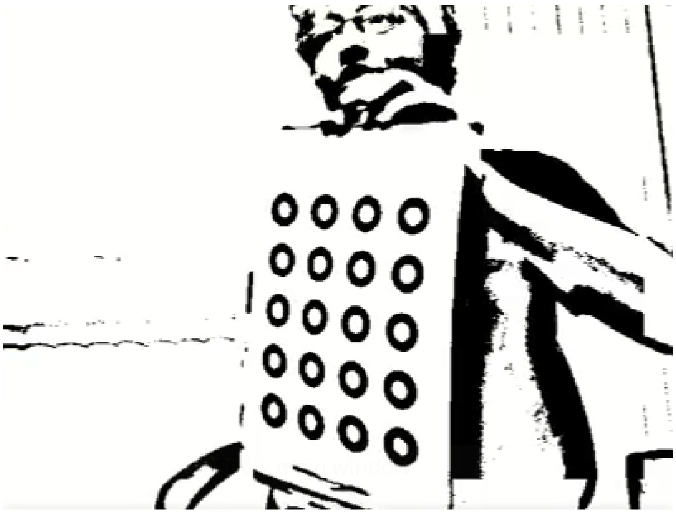

Fig. 3. Thresholding keeping good image information.

We have some troubles when the pattern is moved fast or the camera doesn't focus the pattern correctly.

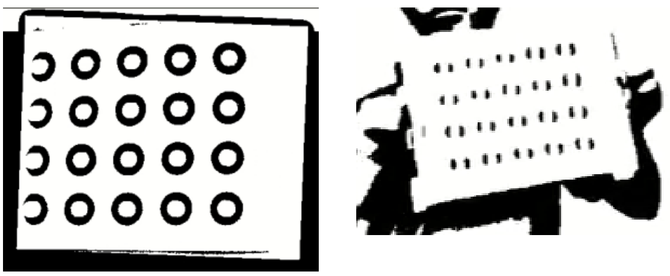

Fig. 4. At left we can see a thresholding error cause by fast movement. At right we can see a thresholding error cause by focus error.

2.3 Find contours

We use the function of openCV findContours to get all the contours from the segmented image in a hierarchy of contours, which will allow us to filter the ellipses in the next step.

Its an important step because if we don't found all pattern ellipses the next step will fail.

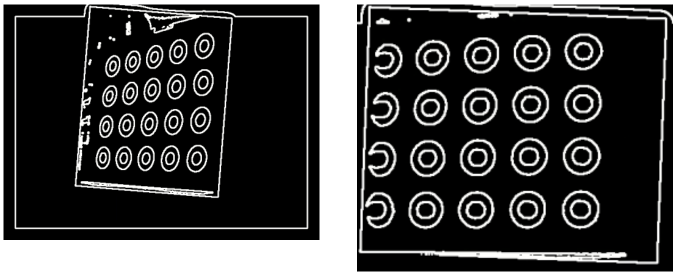

Fig. 5. At left we can see a good contours improved by thresholding. At right we can see a bad contours caused by a fast movement.

2.4 Find ellipse

- From the found contours we look for all the ellipses that have a father and a son, besides that the center of the son is very close to his own.
- From the set of ellipses found we verified that we have at least 2 ellipses near us at a distance no greater than 5 times the radius of the ellipse with which we make the comparison.
- If we have more than 20 ellipses we look for mode based on the hierarchy of the father of each ellipse and we eliminate all those who have a different father.

Fig. 6. Comparison between two ellipses to check if the ellipse c1 belong to the control point. the small Yellow line is the radio of the ellipse c2, the large Yellow line is five times the radio of the ellipse c2. We can see that condition distance(c1, c2) $<$ 5 * radio C2, does not hold, therefore the ellipse c1 does not belong to the control points.

2.5 Line representation

After we have found the rings of the pattern in the image we start to build the lines representation of the pattern.
If we lost the pattern we need to order the points again using the next algorithm and if we doesn't lost the pattern

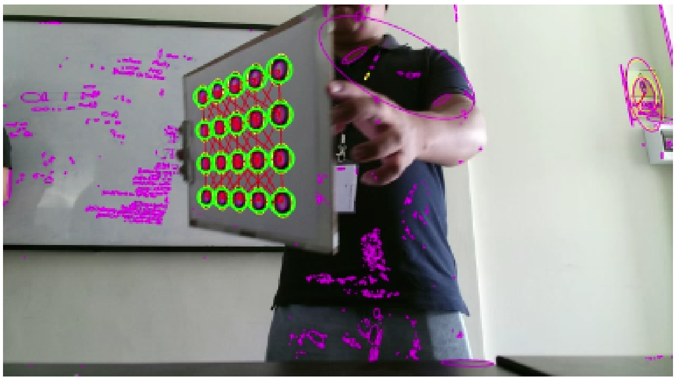

Fig. 7. Discard false positive using the father mode.

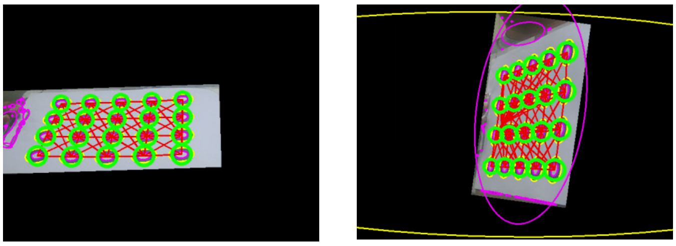

Fig. 8. Good patter points recognition.

Fig. 9. Ellipse not detected correctly cause by a poor image definition.

we perform the tracking process described in the next subsection.

- Using each pair of points we make a line and determine the line equation.
- Using the line equation count the number of points near to the line and store these points in a aux vector.
- If there are 5 points in the aux vector, order these points using the x coordinate and check the distance between the first and last elements if the distance is lower than the radio of circles then sort the aux vector again using the y coordinate
- Draw a line between the first and last elements of the aux vector.
- Keep the last point got from the process described before.
- Remove the points and continue searching process to find the next line until we found the 4 lines.

Whit the ordered points we only need to draw lines from the 0 point to 4 point, 5 point to 9 point etc.

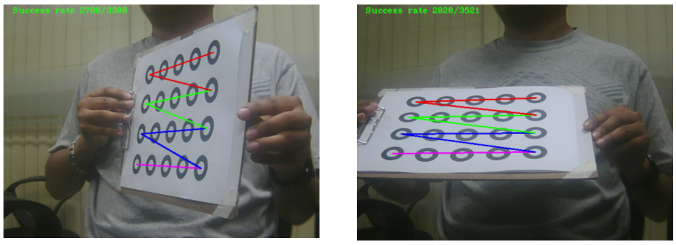

Fig. 10. Results after applying our algorithm on ps3eyecam video.

2.6 Tracking

To keep the order obtained in the previous process we perform a tracking process. From the points found in the previous frame we search in the new points some one that is nearly to these one using the radio as boundary. When the 20 points was found we check if any one has distance bigger than the radio to its near point if this is true we discard old point and consider the pattern was lost.

Fig. 11. Tracking process, the yellow circles are the area to search the new position for each pattern point.

2.7 Camera calibration

To perform the camera calibration we select some frames where the pattern was found and use these points to perform the calibration, in our experiments we have found good results for the ps3 video using every frame multiple of 60, and using 20 frames getting a reprojection error of 0.1346 and for the lifecam video using every frame multiple of 30, and using 20 frames getting a reprojection error of 0.1362.

The Figure 12 show the distribution of points taked for the calibration process.

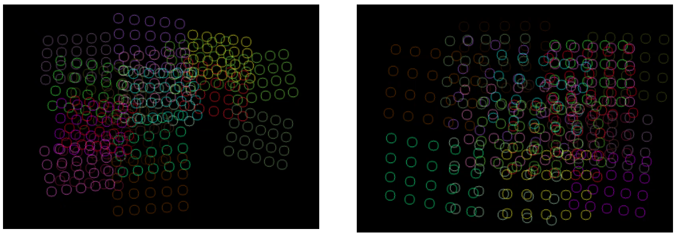

Fig. 12. At left we can see the points used for the calibration process of the LifeCam camera. At right we can see the points used for the calibration process of the PS3 camera.

Average of collinearity: In addition to the reprojection error we evaluate the distance from every inner line point to its main line to get the average of collinearity value and we made some comparisons whith the original and undistorted video, the results are shown in the next table.

	Average of collinearity	
Center point	Original video	Undistorted video
PS3	0.2946	0.09
LifeCam	0.1193	0.1063

Table 1. Average of collinearity

2.8 Undistort image

Using the camera matrix and the distortion coefficients we can transform the image input to correct the distortion.

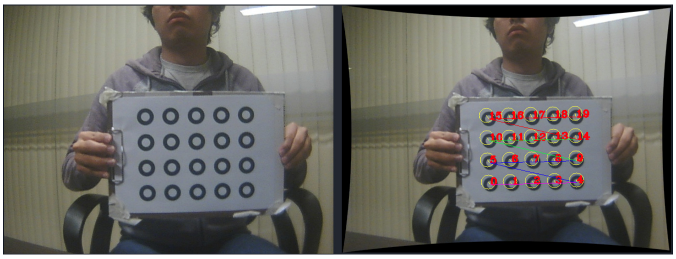

Fig. 13. Image with no distortion using the camera parameters and the distortion coefficients

3. RESULTS

We use the camera calibration algorithm provided by the OpenCV library to perform a comparison with the method implemented in this paper.We disable the flags FixAspectRatio, AssumeZeroTangentialDistortion and FixPrincipalPointAtTheCenter. See Table 2 and 3.

Parameter	Chessboard	Circles	Rings
Reprojection e	0.8346	0.1707	0.1362
Focal length x	611.1363	637.0749	618.3848
Focal length y	612.7416	640.6737	621.9344
Optical center x	318.4211	351.7642	341.9354
Optical center y	225.0789	229.1633	231.4488

Table 2. Comparisons of results obtained with chessboard, circles and rings pattern, on the LifeCam camera.

Parameter	Chessboard	Circles	Rings
Reprojection e	0.23547	0.1250	0.1346
Focal length x	847.5653	914.8484	842.2776
Focal length y	848.2543	918.5734	845.9872
Optical center x	332.6489	325.1919	306.8028
Optical center y	249.0016	233.0003	258.6146

Table 3. Comparisons of results obtained with chessboard, circles and rings pattern, on the PS3 camera.

4. CONCLUSIONS

- It is important to clean irrelevant information in the videos in order to capture relevant characteristics.
- The algorithm needs to be further improved to get better performance on other videos.
- The algorithm depends on how well the ellipses of the calibration pattern are detect. Depends basically on fitEllipse method of opencv.
- For the camera calibration process is important to perform a good tracking algorithm to keep the order of the points.
- The frames selected for the calibration process should cover the four quadrants of the image plane in similar proportions and have good quality to improve the results.

REFERENCES

Bradley, D. and Roth, G. (2007). Adaptive thresholding using the integral image. *Journal of graphics tools*, 12(2), 13–21.

Datta, A., Kim, J.S., and Kanade, T. (2009). Accurate camera calibration using iterative refinement of control points. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, 1201–1208. IEEE.

Prakash, C.D. and Karam, L.J. (2012). Camera calibration using adaptive segmentation and ellipse fitting for localizing control points. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, 341–344. IEEE.

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11), 1330–1334.