Camera Calibration with Concentric Rings Pattern Using OpenCV

Erick Tornero Tenorio
Universidad Catlica San Pablo
Walter Zuniga Herrera
Universidad Catlica San Pablo

Abstract—Camera calibration is an important subject for machine vision, 3D reconstruction, in this paper concentric rings patter is used as an hybrid method for camera calibration.

Index Terms—Pattern, Calibration, Machine Vision, Filter

1 Introduction

Camera calibration is one of the main steps in computer vision, it is useful to get information about the world from images. The are several ways to calibrate the Camera, in this work a method with a well known pattern and some images in different perspectives is used, this method was proposed by Zhengyou[1]

There are several ways to perform a detection of patterns, since the selection of what kind of pattern to use, it could be a mesh of squares, a grid of circles, or what disposition to use (symmetric or asymmetric), in this work a grid of concentric rings in symmetric disposition is used, the selection of filter and its parameters is other problem, is necessary to balance the accuracy and execution time.

The purpose of this paper is to get the internal parameters of a camera using *OpenCV*.
January 9, 2019

2 Steps

2.1 Space color

The raw image is converted to grayscale for two reasons, to reduce number of computations, and because the color in most situations does not provide any relevant information, in this particular scene to use the following filters is necessary work in just a single channel.

2.2 Preprocessing

Filters are used in order to remove unnecessary information or remove noise from the image or just to get relevant information depending on the application.

In order to remove noise for high frequency, a Gaussian filter is applied, See figure 1 this filter will help when a threshold filter is applied.

The strategy to solve the problem is to use an algorithm that has a special treatment with brightness, 2 algorithms were tested to obtain a better result, Integral Threshold[3] and Adaptive Threshold.

2.2.1 Integral Threshold

This algorithm has a good performance in the treatment of brightness in the environment, See figure 2, but the computational cost is quite high, 16.716 milliseconds, given that it has to travel two times each frame, which

causes its use in a real-time environment to be unmanageable, it does not provide enough time for other calculations.

However this is not the only reason, it does not provide a completely correct solution for the problem of intensity of brightness, failing in common positions, for this reason this option is not chosen to perform Thresholding.

2.2.2 Adaptive Threshold

Whereas the conventional thresholding operator uses a global threshold for all pixels, adaptive thresholding, see figure 3, changes the threshold dynamically over the image. This more sophisticated version of thresholding can accommodate changing lighting conditions in the image, those occurring as a result of a strong illumination gradient or shadows.

This satisfactorily solve the problem of changes and lighting, obtaining a degree of error, but being able to clearly differentiate the pattern in the image, and a low computational cost of 10.8667 milliseconds.

All the noise obtained after applying the filters will be processed by another heuristic to recognize only the searched patterns.

2.3 Ellipses Detection

After properly applied the filters, function of *OpenCV, findContours* is applied, this function returns all contours in the image and provides an hierarchy, the problem is that any kind of contour is retrieved by this function, objects that are not elements of the pattern is considering noise. Hierarchy will help to remove that noise.

This hierarchy have the information of child and its father contour. A contour that does not have neither father nor child, can not be a element of pattern, because concentric rings are used.

Since concentrinc rings are used, for each element pattern must exists two concentric ellipses sharing similar centers, *OpenCV* provides the function *fitEllipse* and this heuiristic is applied to remove other contour that is not an element of pattern.

Another heuristic is used, it is base on the weighted size of ellipses of previous frame, in the new frame an ellipse that meet the following restriction will be ignore: $(Ra + Rb)/2 > R + \epsilon$, where ϵ means the bigger the new ellipse could be in respect to the previous size.

Finally, Bounding box of is used to reduce the noise that appear out of the bounding box of the previous mesh.

2.4 Ordering Center Pattern

Since each center must have an specific index, a bi-linear transformation is proposed as a main step to identify each center. The idea of use this transformation is to maps from a quadrilateral (shape of patterns in the image) to a rectangle (easy to identify).

2.4.1 Find corners of Pattern

Bi-linear transformation is solved mapping 4 points from space A to space B, this four points will be the corners of the quadrilateral mapped into the corners of the rectangle, so find the corners of quadrilateral is necessary. In the case that the other Points is not found, compute the diagonal previously obtained, then the other 2 points will be the points that is further from the diagonal.

First an approximation to a rotated rectangle is performed using the function of *OpenCV: cv::minAreaRect(Points)* see blue rectangle in **Fig. 1**, with this approximation, at least one diagonal is find a simple minimal euclidean distance is applied using an ϵ.

If there are corners that are not identified **Fig. 1 (b)**, the diagonal previously obtained is computed, then the other points are that points that have the greatest distance from the diagonal line, see Cian line in **Fig. 1 (b)** .

After get the four quadrilateral corners, a basic labeled must be do it, started from point that have

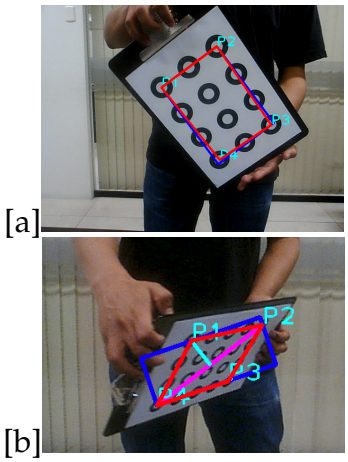

Fig. 1. (a), (b) Blue rectangle: Aproximation with *cv::minAreaRect*, Red quadrilateral: real shape of pattern. (a) Show cases when there when the corners of aproximate rectangle is near to some center, (b) shows when just the elements of diagonal can be aproximate.

2.4.2 *Bilinear Transformation*

After identified the four quadrilateral corners, apply a bilinear transformation in such way that a rectangle is get with proportions of the grid pattern.

Once the set of points have form of rectangle, is easy to identify what will be the index of each center. See **Fig. 2**

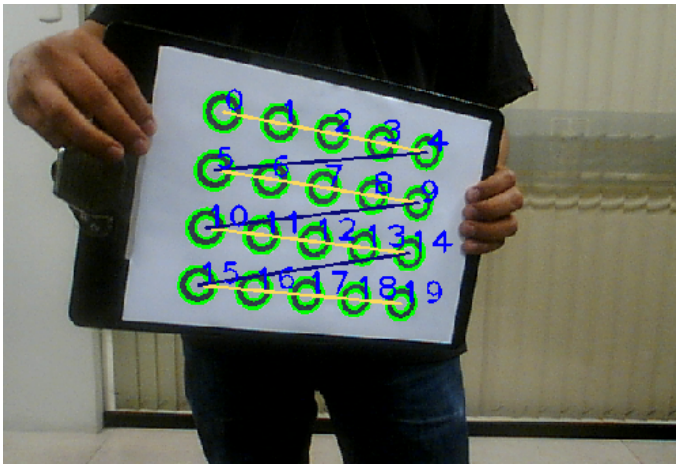

Fig. 2. Ordered Centers

2.5 Identification and tracking of pattern elements

A simple heuristic of tracking is used to do this task. First a unique identification is assign to each pattern element, then for the previous set of *Centers* the new location of centers is computed by applying euclidean distance to each point.

2.6 Camera Calibration

This section is doing using the function of *OpenCV calibrateCamera* that receive as main parameters the grid of center in real scale, the grid of centers computed, and returns the intrinsic parameters, distortion rotation and translation.

The images to calibrate is selected from video with some time of difference to ensure that different perspectives is used

3 Conclusion

Heuristics to find corners show good results, just 1 fail of this heuristic was detected in a video of more than 5000 frames.

Bilinear transformation shown good results even when the image is rotated, problems could happen when the pattern is rotated more than 90, this problem must be treated in future works.

References

[1] Zhengyou Zhang, *A Flexible New Technique for Camera Calibration*, 1998

[2] Shubham Rohan Asthana, *Enhanced Camera Calibration for Machine Vision using OpenCV*. In International Journal of Artificial Intelligence, Septiembre 2014.

[3] Dereck Bratley, *Adaptative thresholding using the integral image*. In Journal of Graphics, GPU, and Video Game Tools, August 2014.