Detection and identification of a pattern for camera calibration

Franci Suni Lopez
Universidad Católica San Pablo-Arequipa Peru
franci.suni@ucsp.edu.pe

***Abstract*—Nowadays,**

***Keywords*-camera calibration,**

I. Introduction

The calibration of a camera is the process that allows, within the field of artificial vision, the obtaining of the parameters that define the conditions of formation of the image, including the internal geometry and the optics of the camera, as well as its position and orientation with respect to a reference object or calibration standard. In short, calibration is a procedure that seeks to know how a camera projects a 3D object in the image plane so that it can extract metric information from the images.

The use of calibrated cameras can be useful to solve a series of problems related to obtaining the 3D position of objects in space from their images or for their three-dimensional reconstruction. This can allow, among other tasks, the realization of maps of the environment of the camera, the tracking of a specific object or the obtaining of the position of the camera with respect to objects that surround it. It can also facilitate navigating around a mobile robot, avoiding obstacles, addressing specific objects or facilitating the definition of the most appropriate path to reach your destination.

The calibration of a camera is a complex problem to solve since many are the parameters to be solved and many are the factors that influence the results. In part, this complexity is limited by the methods of calibration using camera models that are, in fact, ideal or simplified models of their physical equivalents. With these simplifications, quite acceptable results are obtained (although not exact), but it causes many parameters not to be parametrized. The pin-hole model, which is the one used by most calibration methods, does not parametrize optical aspects such as focus distance, depth of field, aperture or possible misalignment between the Flat image and lens. Even an effect as important as the distortion of images is modelled, in most cases, in a very simplified way, and not in all cases is parametrized.

Other important factors are the possible sources of noise that influence the process of image formation since the image obtained by a camera introduces quantization errors, or the precision in the actual location of the elements (pattern points) used for performing the calibration is not exempt from imprecision. Due to these issues the calibration process of a camera can have multiple solutions, where the errors are compensated between the degrees of freedom (the parameters of the model), being able to obtain as a result of the calibration a parametrization that, although it minimizes an objective function final, it is unreal from the physical point of view.

II. Procedure

According to the video source (i.e. live or recorded) we change the setting for collecting each frame from video. Also we define the number of ring that our pattern had, this information will be useful when we will the selection of the correct ellipses. Figure 1 shows the pattern of 20 ring that is used for our experiments.

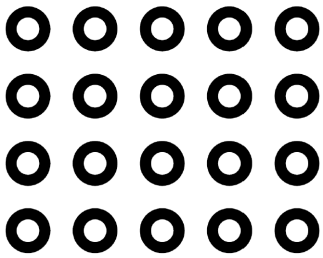


Fig. 1. Used calibration pattern with 20 rings.

Therefore our procedure to identify the calibration pattern is as follow:

- Convert each frame to gray scale, to convert an image in RGB to another one in gray scale it is enough to add the 3 components (red, green and blue) and divide by 3. In our experiments we use the function of OpenCV $cv :: cvtColor$, Figure 2 shows an example of gray scale conversion.

$$CV\_BGR2GRAY$$

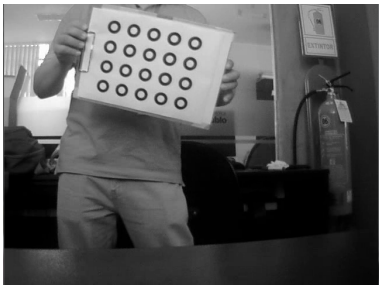


Fig. 2. Gray scale image.

- Apply the Gaussian blur of size 5x5, to reduce the detection of non-existent ellipses. :

$$cv :: GaussianBlur(...Size(5,5)...)$$

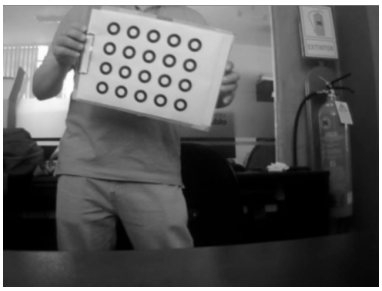


Fig. 3. Gaussian Blur applied on gray image.

- Edge detection is a widely used technique that allows us to isolate objects and separate them from the background. For that reason first we use a Canny edge detection:

$$cv :: Canny(...)$$

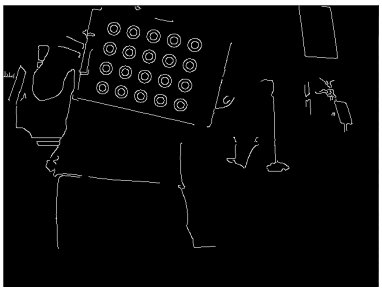


Fig. 4. Canny edge detection.

- Once the edges are obtained, the only thing that we would need is to detect the different contours in order to count the objects. For our experiments we use $CV\_RETR\_EXTERNAL$ as parameter for the function findContours, because we work with one ellipse for each ring.

$$findContours(..., CV\_RETR\_EXTERNAL, ...)$$

- After to compute the contours we find the contours that appear to a ellipse with the following OpenCV function:

$$fitEllipse(...);$$

- We select the ellipses where the difference between the contour area and the area of the generated ellipse is $\leq 2\%$.
- We calculate the centroid of the ellipses with the following formula:

$$(\frac{x_1 + x_2 + ... + x_n}{n}, \frac{y_1 + y_2 + ... + y_n}{n})$$

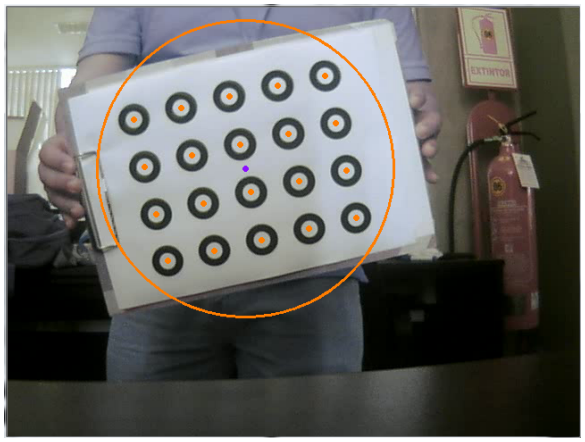
- Then we calculate the distances between the centroid with the center of each recognize ellipse.
- Also we compute the standard deviation of the computed distances

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(d_i - \overline{d})^2}$$

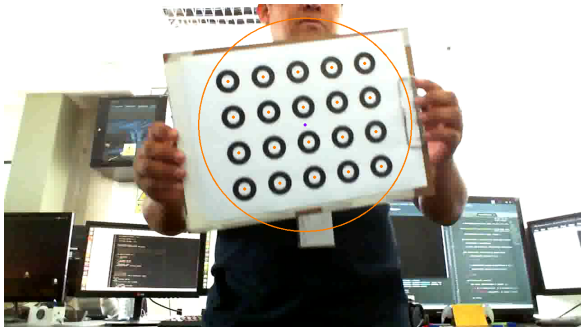
- Finally, we set a radio $r = \overline{d} + \frac{s}{1.3}$, to define a cluster of the most points that corresponds to the rings of the pattern.

III. Results

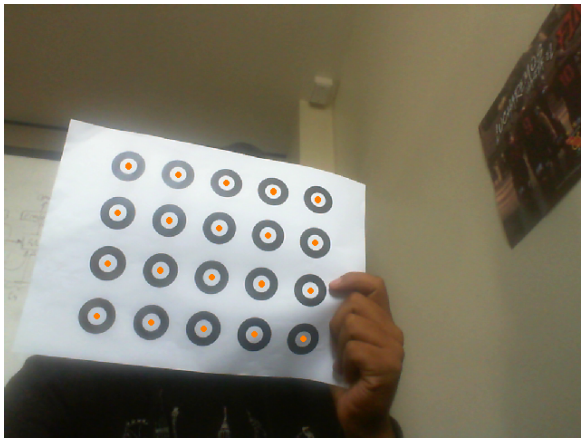
In the following images we present the results of our method in different videos:



(PS3)



(Life Camera)



(Real time)

Fig. 5. Results