Computer Vision Assignment 4 (2018/19)

Pedro Santos, *Universidade de Aveiro, 76532* Silvério Pereira, *Universidade de Aveiro, 76505*

***Index Terms*—OpenCv,c++,assigment_3**

I. Introduction

THIS report has a description and explications about the experiences done in the lesson number four and the topic covered is Camera calibration.

II. Exercises

A. Exercise 1

In the first exercise it is necessary to compile and test the file chessboard.cpp. This code is detecting the corners in a chessboard pattern using openCV functions and shows the results of the detection for a series of images. The next thing that it was necessary to do was to calibrate the camera using the function calibrateCamera. To save the results of the camera calibration process it is necessary to create four variables that are translation_vectors, rotaction_vectors, distCoeffs and intrinsic. This variables will be passe in the function calibrateCamera to save there the result. The other two arguments used are object_points, image_points and the image size. The variable object_points has the teorical value of the coordinates of each corner and the image_points has the value of the coordinates of the corners detected by the function FindAndDisplayChessboard. Lastly, the values of the results of the camera calibration are printed. and the values of the intrinsic and distortion matrices are saved in an xml file.

If the pattern was different to get the distance correctly evaluated it would be necessary to put in the vector the coordinates 3D of the new pattern.

B. Exercise 2

In the second exercise was implemented code to project an orthogonal line (normal) in the provided images. To do that for each image the image is read, then it is used the function projectPoints that receives the results of the camera calibration, the coordinates of the points to project and it is also passed the vector where it will be saved the corresponding coordinates of the points in the image. The, it is used the function line from openCV to project the orthogonal line in the image. The arguments of this function are the image, the two points that were calculated using the function project points, the color, the thickness and the type of the line. The last step was to show the final result. The final result of the first 5 images can be seen in the figure **??**, **??**, **??**, **??** and **??**. The other resulst are not presented in the report because they do not add value and to the report do not become so unnecessarily extensive.

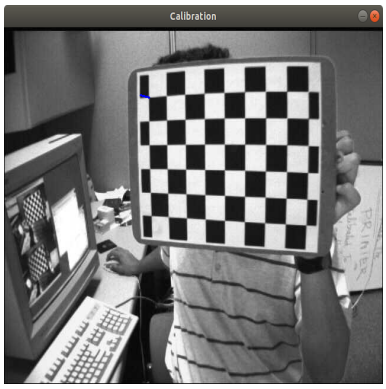

Fig. 1. orthogonal line in image 1

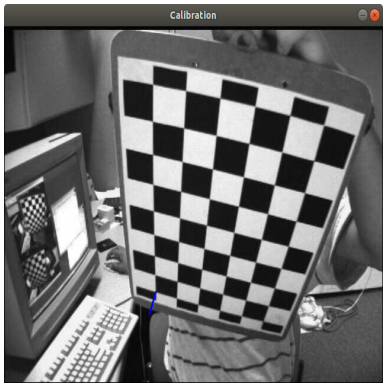

Fig. 2. orthogonal line in image 1

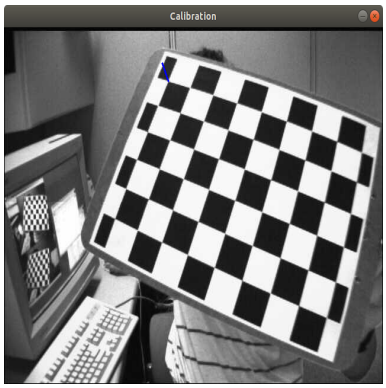

Fig. 3. orthogonal line in image 1

C. Exercise 3

- **src-** Input single-channel 8-bit or floating-point image.
- **dst-** Image to store the Harris detector responses.
- **blockSize-** Neighborhood size.
- **ksize-** Aperture parameter for the Sobel() operator.
- **k-** Harris detector free parameter.

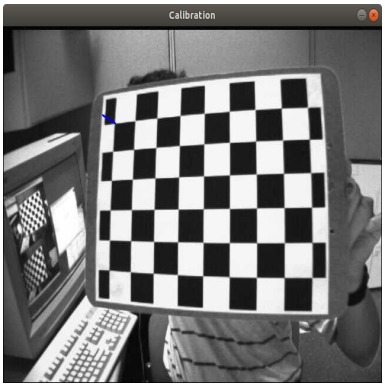

Fig. 4. orthogonal line in image 1

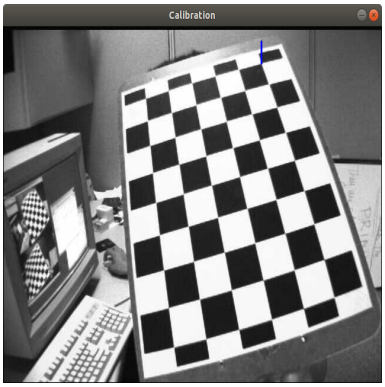

Fig. 5. orthogonal line in image 1

- **borderType-** Pixel extrapolation method.

D. Exercise 4