Lab 04, Morphological Operations

João Santos, *MRSI, 76912*

***Index Terms*—OpenCV, Computer Vision, MRSI, UA, DETI, LaTeX.**

I. Introduction

THIS report is intended to be used alongside the Python3 code developed for this Lab.

The Lab #05 is a introduction class to camera geometry and calibration, OpenCV and Python 3.

This report was written using LaTeX .

II. Exercises

Lets analyse the resolution of the proposed exercises.

A. Ex. 5.1: Chessboard calibration

As stated on section I, the goal of this class was to learn, understand, and implement a camera calibration. To do so, a chessboard pattern is required so the algorithms can obtain real-world data from the taken image and compute the best matching intrinsic parameters from the camera.

By inputting the count of the (inner) corners of the chessboard pattern and the size between each square (for now, the units are not important, so we can set it to one square) the *cv2.calibrateCamera()* method can output the intrinsic parameters of the camera being calibrated.

Fig. 1 shows the chessboard corners found with *cv2.findChessboardCorners()* and displayed with *cv2.drawChessboardCorners()*.

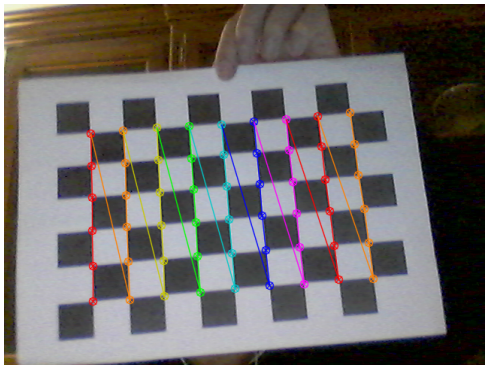

Fig. 1. Chessboard corners found on the pattern.

If, for some reason, we decided to use another size for the chessboard pattern, we would need to also update the chessboard size on the *objp* array. Also, if we wanted to use some kind of metric units, we should build this same array with the real-world unit. For example, lets say that each square in the pattern had 20 mm of side. Then, in the *objp*, each computed point should have a distance of 20 all the closest neighbors on the vertical and horizontal axis.

B. Ex. 5.2: Projection of 3D points in the image

In this exercise, it was requested that the author drawn a axes or wireframe cube on top chessboard in the provided images. Yet, the author diverged from this and chose to do this on the live feed from its camera.

This was only possible after the calibration of the laptop camera used. Fig. 2 demonstrates the output.

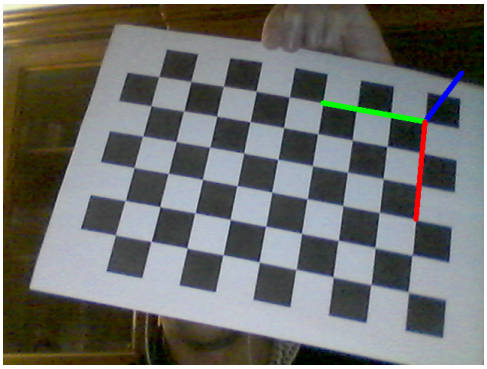

Fig. 2. Axes drawn on top of the chessboard pattern.

C. Ex. 5.3: External calibration

For the final exercise, the author should calibrate the camera on its laptop and save the parameters to a .npz file. This process makes use of the *cv2.solvePnp()* method to compute the rotation and translation vectors for each of the captured images.

These vectors are the ones used on the previous exercise in order to draw the axes system on top of the image when the calibration has already been performed.