Instituto Superior de Engenharia de Lisboa

Computer Vision and Mixed Reality

Report

Marker Based Augmented Reality

Mestrado em Engenharia Informática e Multimédia

Rodrigo Dias, 45881

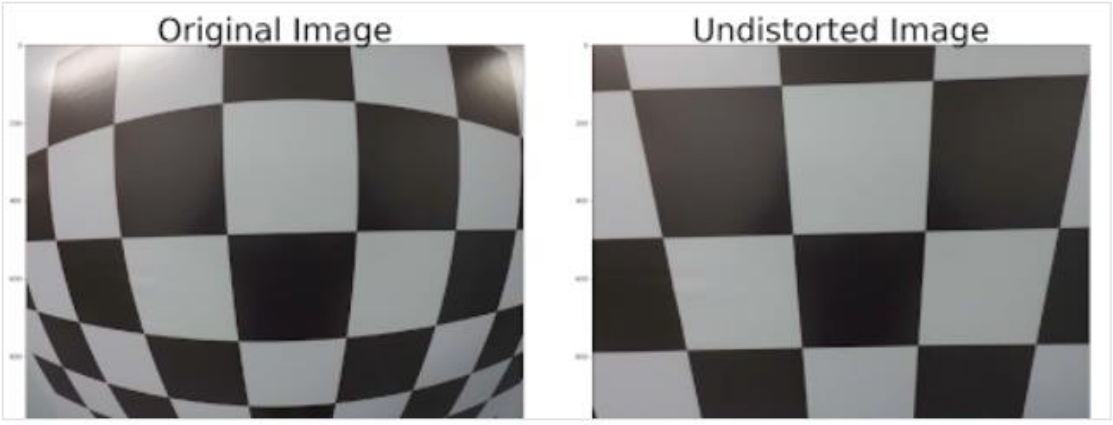
Semestre de Verão, 2021/2022

Introduction

In this project, we're going to develop a computer vision application which implements marker based augmented reality that allows the inclusion of virtual elements aligned with real fiducial markers.

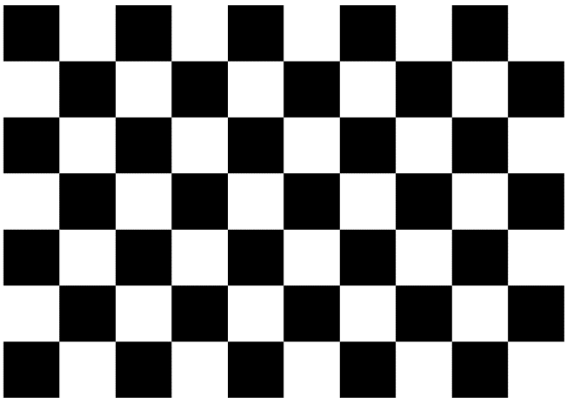
Camera calibration

The first step we must calibrate the camera, to correct for any camera distortion caused by the lens. Imagining that we have a fisheye lens, the idea is to undistort like it shows the image below.

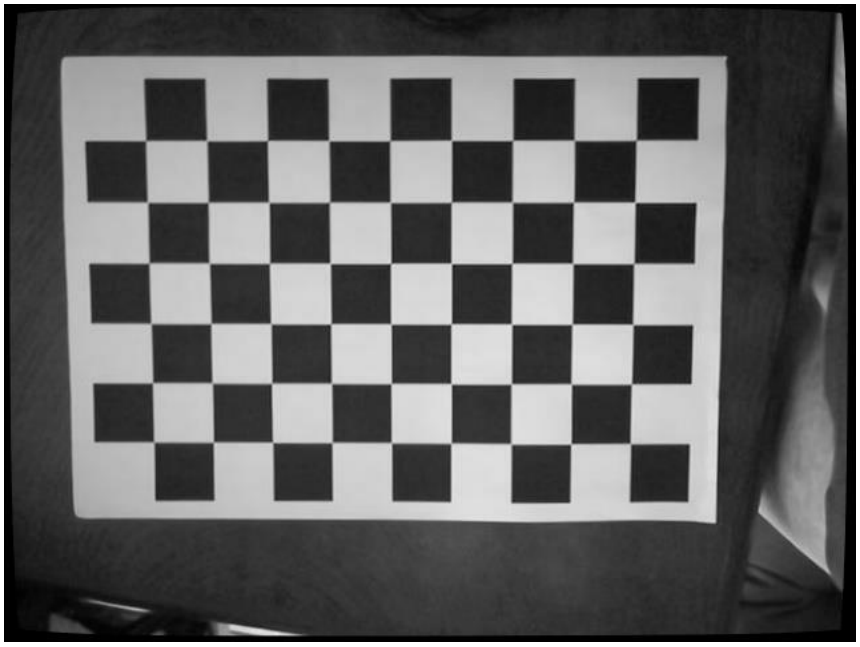


This calibration process is to calculate the intrinsic parameters of the camera, which can be different for every camera, especially if they don't have fixed lens.

For that we use a chess grid to calibrate using OpenCV libraries, indicating the real size of each square and the height and width of the grid. The one used is shown below.



We calculated the intrinsic parameters of the camera and undistorted the image. The picture below shows the image undistorted using the phone. We can see some black borders compensating for the distortion. They are not that noticing because the distortion is not severe.

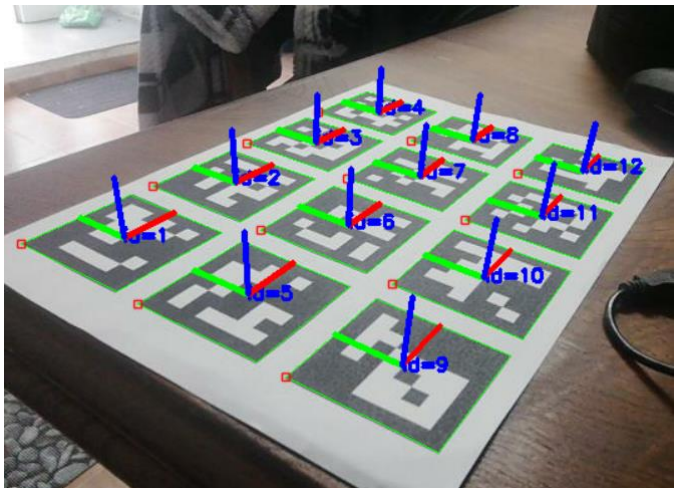


For the code, it was used functions to save and load the coefficients, other to get the distortion and other to undistort the image. It was implemented a main function to calculate the distortion using video, and when clicking ‘p’ it with would calibrate and save the parameters along with the image undistorted.

Detection and camera pose estimation with ArUco library

In this phase we load a dictionary with the aruco markers information and call a function to detect the markers and their ids. We can then draw the detected markers to verify if the code is working.

After verifying everything is working fine, it’s used the estimatePoseSingleMarkers to calculate the rotation vectors and translation vectors using the camera matrix and distortion coefficients loaded from camera calibration. Then it’s used the drawAxis function to draw the axis in the center of each marker, compensating for the camera distortion.



Registration of virtual objects

In this phase the idea is to mix virtual objects into the image, using different objects for each marker. For this it's used the warpPerspective to warp the images loaded to the corners of each marker. The image below shows the output for this exercise.

Conclusion

It was possible to see how to calibrate a camera to compensate for lens distortion, to detect aruco markers and how to mix the real world with the virtual world for augmented reality.