UNIVERSITÀ DI TRENTO

Combined calibration of two camera systems

Computer Vision Project
Sebastiano Chiari - 220527

1 Introduction

This report describes the work of the Computer Vision project. The aim of the project was to calibrate two camera systems (one having RGB cameras and another one having Motion Capture cameras) within a studio theater. The purpose of this project was to align the two systems creating a reference system or coordinate space where these two sets of cameras can be used as a single entity.

2 Methodology

2.1 Hardware

The studio on which the cameras are placed is a square. The origin is placed in the center and has already been calibrated as the $(0, 0)$ point for the cameras in the Motion Capture system. The Motion Capture cameras are placed along the sides of the square (in the center) and at each corner. Next to each Motion Capture camera on the perimeter (not on the corners) an RGB camera is also placed. The theatre schema can be found in Figure 1.

Since the RGB camera #11 does not frame in its field of view the origin of the system, nor other points common to all other cameras (this was a technical liability in order to perform the PnP estimation, see Chapter 2.3), it was decided not to consider it in the calibration, as well as its corresponding Motion Capture camera #7.

2.1.1 Image/Video extraction

RGB cameras. We can directly access the cameras via IP address. Thus, you can record the video feed via `ffmpeg` using the `rstp` protocol with the following line of bash code.

```
ffmpeg -i "rtsp://IP ADDRESS" -vcodec copy -acodec copy "output.mp4"
```

Motion Capture cameras. To extract a video feed from the motion capture cameras, we can go directly from their proprietary software, Motive. We need to export a video in MJPEG format using the **grayscale** view made available. It is necessary to later convert this video from the MJPEG codec to an H264 codec, otherwise OpenCV could encounter some difficulties while processing it (causing frame skipping and misinterpretation of length and frames in the video).

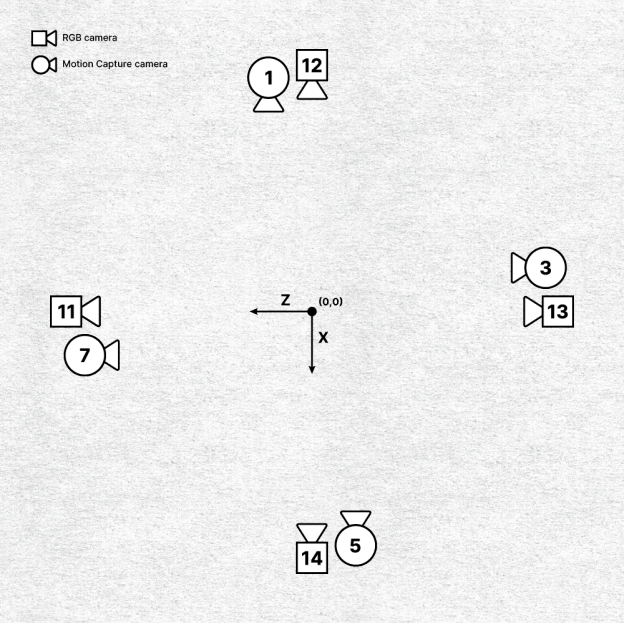


Figure 1: Theatre schema, with camera numbers attached to camera positions

2.2 Intrinsic camera calibration

First, we needed to calibrate the cameras in order to retrieve their own intrinsic parameters, which are specific to each camera. They include information like focal length (fx, fy) and optical centers (cx, cy) and they allow to create a camera matrix, which can be used to remove distortion due to the lenses of a specific camera.

Calibration is performed through OpenCV `cv.calibrateCamera` function, using a set of images where it can be detected a checkerboard pattern. To speed up the process, the videos where I show the checkerboard in favor of the camera were preprocessed: every 30 frames (empirically chosen value) in which the pattern is detected, a .jpg file is exported to be later used for calibration.

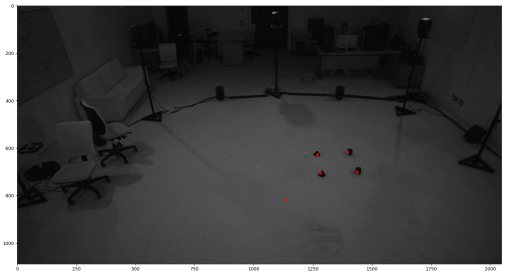
2.3 Extrinsic camera calibration

Once we retrieved the camera matrix and the distortion parameters for all the cameras, we have to retrieve the extrinsic camera parameters, namely the *rvec* vector and *tvec* vector, which represent the rotation vector and translation vector, respectively. These two vectors describes the position of the camera in the space.

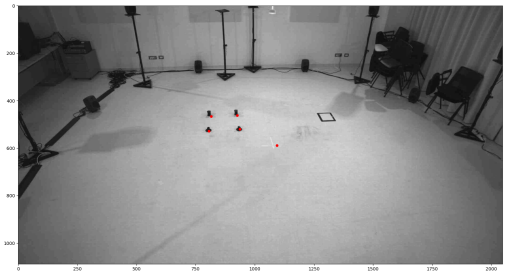
For this step, **Perspective-n-Point (PnP) pose computation** was used, which involves providing at least 4 points whose world coordinates and position in pixels are known in the respective camera to mathematically extract the *rvec* and *tvec* vectors.

Since the space around the existing reference system on the floor of the stage was small in some cameras, I had to make a new one by placing markers on the floor and calculating their coordinates in the world reference system.

Once this step is completed, we have finished the camera calibration and the system is ready to be used.

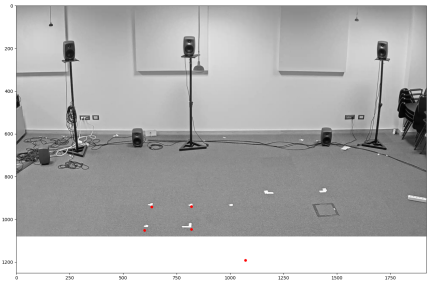


(a) Motion Capture camera #1



(b) Motion Capture camera #3

(c) RGB camera #12

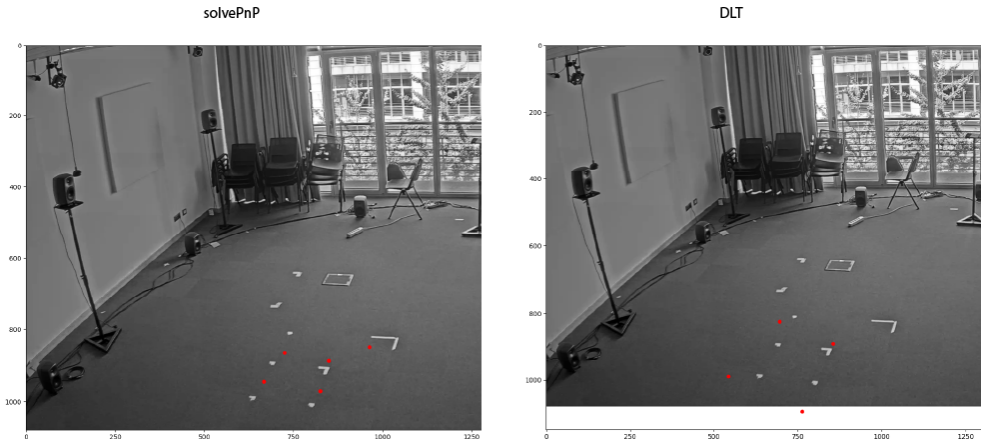


(d) RGB camera #13

Figure 2: Pixel reprojection to check calibration correctness

2.3.1 DLT

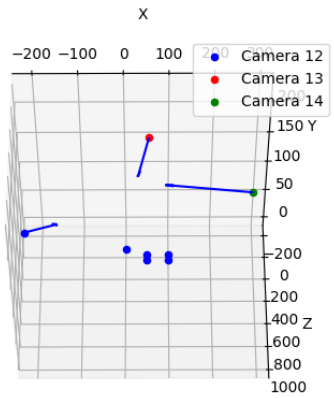
DLT (Direct Linear Transform) algorithm was also implemented (can be found in the extra folder) to validate the efficiency of the PnP method. Generally, DLT produced less accurate results than PnP: in addition to failing to represent distances correctly, the distance and positioning relationships between points are also heavily offset. However, this could also be due to the number of coplanar points used to determine correspondences between world coordinates and image pixels: the algorithm may struggle to converge these points.



2.4 Points reprojection and world consistency

In order to visualize the correctness of the extracted matrices and thus the calibration of the chambers, a script was implemented to perform point reprojection to verify the correspondence between known points in the real world and their respective pixel positions.

In addition, a method has been implemented to display the position of the cameras in relation to the origin of the world coordinate system. This is done to provide a quick overview and to check the consistency of the positions in relation to the actual placement of the cameras inside the studio.



3 Conclusions

In conclusion, this project is a starting point for building a more complex and functional calibration system for calibrating multiple camera systems. The basic functionalities are properly functioning and allow the RGB cameras to be used in conjunction with the Motion Capture cameras.

Certainly, a first improvement to be made could be the introduction of a ground marker system to be recognized by edge detection algorithms to improve the correspondence between real-world measurements and pixels. In addition, it would be great to be able to improve the visualization of the position of the chambers relative to the origin for even clearer and more immediate visual feedback.