3D SCENE RECONSTRUCTION
COMPUTER VISION

May 12, 2017

Partners: Evelyn Paiz & Nadile Nunes

Instructor: Alain Tremeau

OBJECTIVES

To integrate RGB data and depth data from 2 Kinect V1 devices to reconstruct a scene. This from a series of color and depth image pairs with the objective to reconstruct the scene using a point cloud (a way to represent the external surface of an object).

As part of this lab sessions, the following tasks were done sequentially:

- Image acquisition.
- Camera calibration.
- 3D-reconstruction.

INTRODUCTION

Vision begins with the detection of light from the world. That light begins as rays emanating from some source (e.g., a light bulb or the sun), which then travels through space until striking some object. When that light strikes the object, much of the light is absorbed, and what is not absorbed we perceive as the color of the light. Reflected light that makes its way to our eye (or our camera) is collected on our retina (or our imager). The geometry of this arrangement — particularly of the ray's travel from the object, through the lens in our eye or camera, and to the retina or imager — is of particular importance to practical computer vision.

We all are familiar with the stereo imaging capability that our eyes give us. To what degree can we emulate this capability in computational systems? Computers accomplish this task by finding correspondences between points that are seen by one imager and the same points as seen by the other imager. With such correspondences and a known baseline separation between cameras, we can compute the 3D location of the points. Although the search for corresponding points can be computationally expensive, we can use our knowledge of the geometry of the system to narrow down the search space as much as possible.

For this reason, camera calibration is important for relating camera measurements with measurements in the real, three-dimensional world. This is important because scenes are not only three-dimensional; they are also physical spaces with physical units. Hence, the relation between the camera's natural units (pixels) and the units of the physical world (e.g., meters) is a critical component in any attempt to reconstruct a three-dimensional scene.

In general, there is no reliable way to do calibration or to extract 3D information without multiple images. The most obvious case in which it is possible to use multiple images to reconstruct a three- dimensional scene is *stereo vision*. In stereo vision, features in two (or more) images taken at the same time from separate cameras are matched with the corresponding features in the other images, and the differences are analyzed to yield depth information. In this case we are primarily interested in *disparity effects* (triangulation) as a means of computing distance.

PRACTICAL SESSION

In this final practical session, the complete process for a 3D scene reconstruction was tried to be carried out. All started with the images acquisition, a calibration and a final 3D reconstruction. The following results were obtained:

Part 1: Image acquisition

The first part of the lab session was to obtain a set of images RGB and depth pairs from the Kinect stereo system. Figure 1 shows the basic arrangement of the system. It was composed of a general scene with around 5-7 simple objects. Two Kinect sensors were aligned as a stereo

system to be able to acquire two images per shot using the matlab scripts **Kinect_capture.m** and **Kinect_preview.m**.

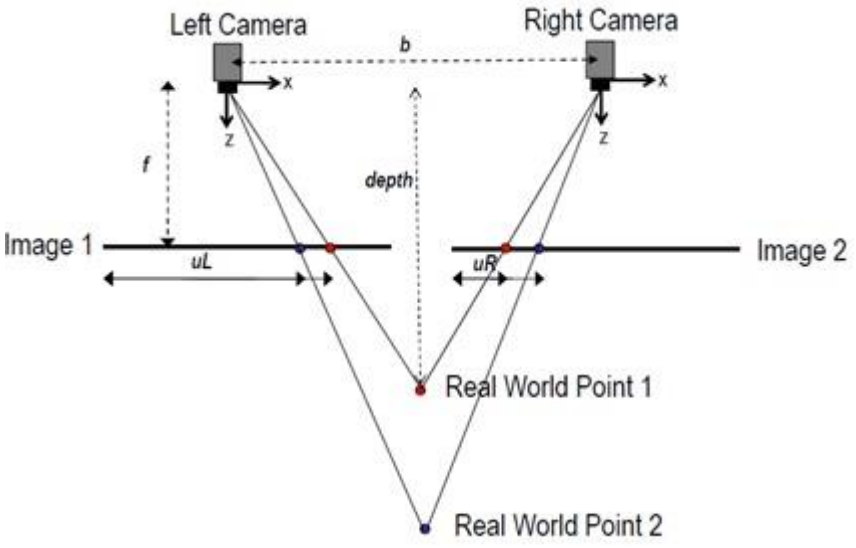

Figure 1: *System configuration of 2 Kinects cameras*

The first set of images acquired was for the calibration process. In total, it was 18 images from the left Kinect sensor and 18 from the right Kinect sensor. Apart from the RGB images, also the images from the depth sensor was also obtained. With the images from the calibration, the scene with the objects was also captured with one RGB image from each Kinect. Also for this was saved the images from the depth sensor.

Part 2: Camera calibration

Following the image acquisition, the camera calibration was performed for each camera and then for the stereo system. The aim of the camera calibration was to determine the distortion parameters k_1, k_2, k_3, p_1 and p_2. The method used was the one developed by CalTech is based on chessboard images (used in the lab session 2).

This method can also provide the information about the location and orientation of two cameras that were used to take the images of the scene, and therefore was a suitable stereovision system calibration.

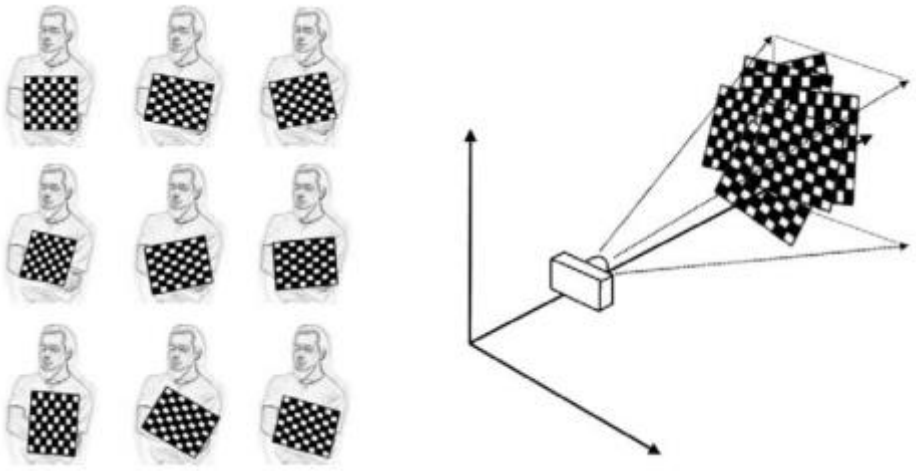

Figure 2: Calibration method based on chessboard images

1. *Left camera single calibration:*

First a single camera calibration was performed for the left Kinect RGB data.

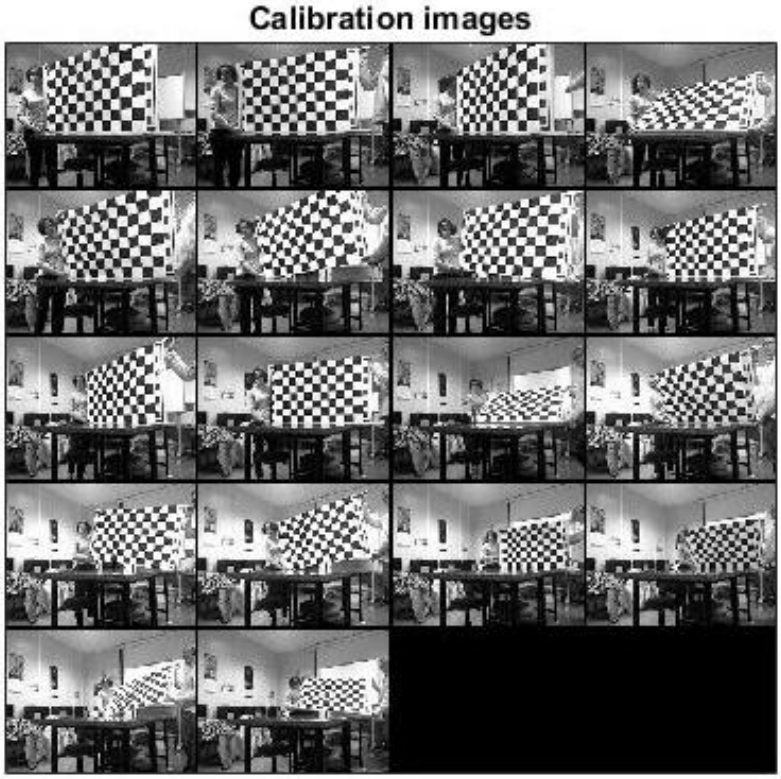

Figure 3: Image loading of 18 samples for single left camera calibration

The manual corner extraction was performed with a windows size of $(wintx = winty = 5)$ and $dX = dY = 30\ mm$.

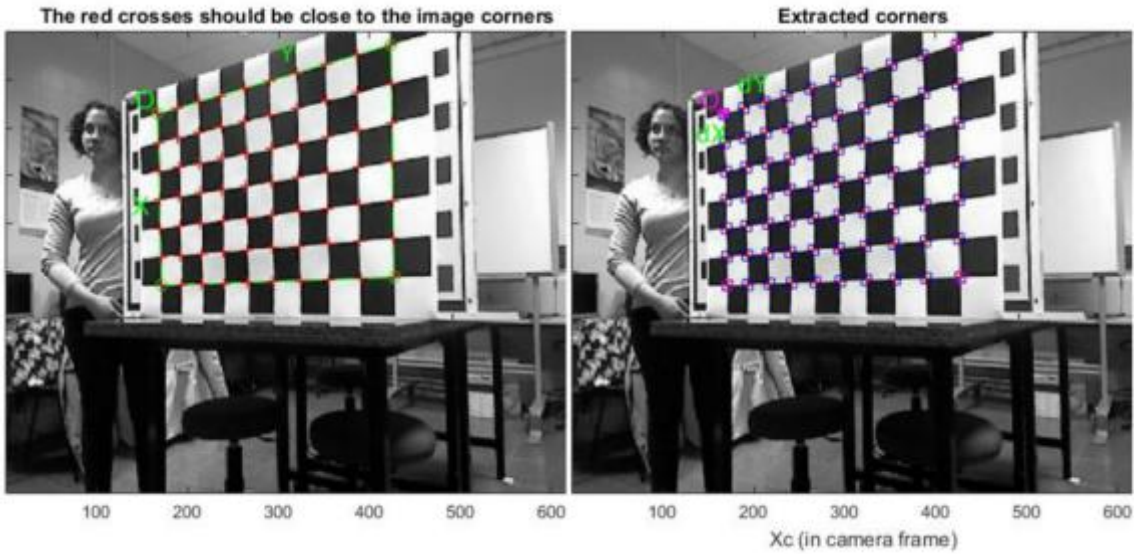

Figure 4: Corner extraction result for one of the left Kinect data

Followed the main calibration step was done (with the manual selection of the corners). Then, the corner extraction was computed automatically for all the images and the calibration was performed again.

Aspect ratio optimized (est_aspect_ratio = 1) -> both components of fc are estimated (DEFAULT).

Principal point optimized (center_optim=1) - (DEFAULT). To reject principal point, set center_optim=0

Skew not optimized (est_alpha=0) - (DEFAULT)

Distortion not fully estimated (defined by the variable est_dist):

Sixth order distortion not estimated (est_dist(5)=0) - (DEFAULT) .

Initialization of the principal point at the center of the image.

Initialization of the intrinsic parameters using the vanishing points of planar patterns.

Initialization of the intrinsic parameters - Number of images: 18

Calibration parameters after initialization:

Focal Length: fc = [510.34288 510.34288]

Principal point: cc = [319.50000 239.50000]

Skew: alpha_c = [0.00000] => angle of pixel = 90.00000 degrees

Distortion: kc = [0.00000 0.00000 0.00000 0.00000 0.00000]

Main calibration optimization procedure - Number of images: 18

Gradient descent iterations: 1...2...3...4...5...6...7...8...9...10...11...12...13...14...15...16...17...18...19...20...done

Estimation of uncertainties...done

Calibration results after optimization (with uncertainties):

Focal Length: fc = [530.82646 533.53115] +/- [5.13653 5.81081]

Principal point: cc = [300.55945 273.73426] +/- [6.82931 5.19285]

Skew: alpha_c = [0.00000] +/- [0.00000] => angle of pixel axes = 90.00000 +/- 0.00000 degrees

Distortion: kc = [0.22705 -0.33860 0.01336 -0.01282 0.00000] +/- [0.02195 0.05485 0.00441 0.00509 0.00000]

Pixel error: err = [0.40482 0.30180]

Note: The numerical errors are approximately three times the standard deviations (for reference).

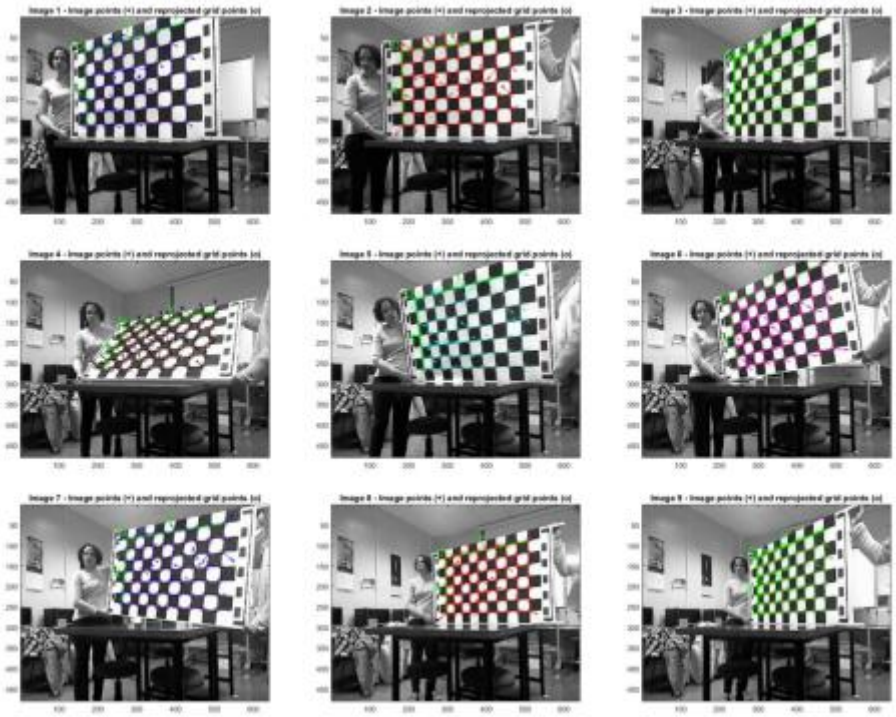

Figure 5: Reprojection (part 1) result for all images of left camera

Figure 6: Reprojection (part 2) result for all images of left camera

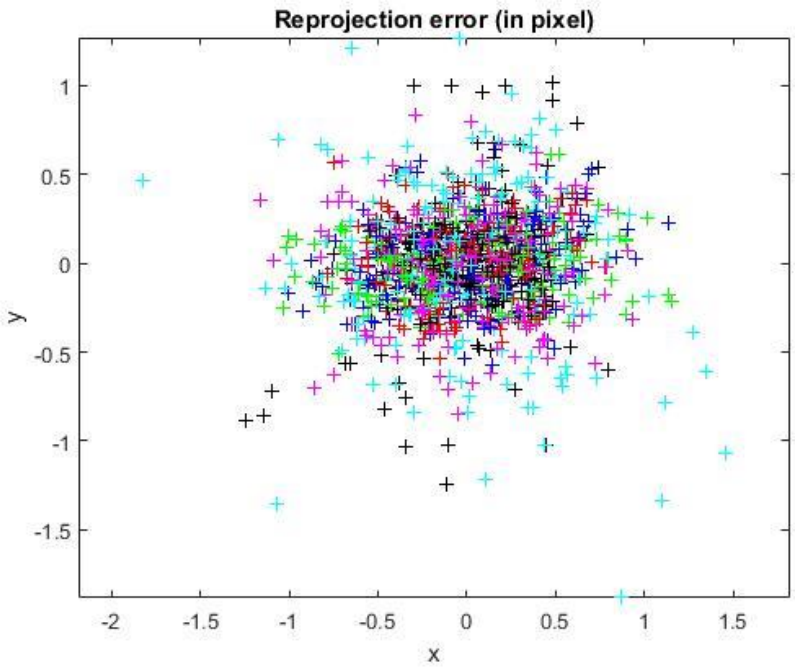

Figure 7: Reprojection error for left camera

Finally, the extrinsic parameters were showed as a 3D representation of relative positions of the grids with respect to the camera.

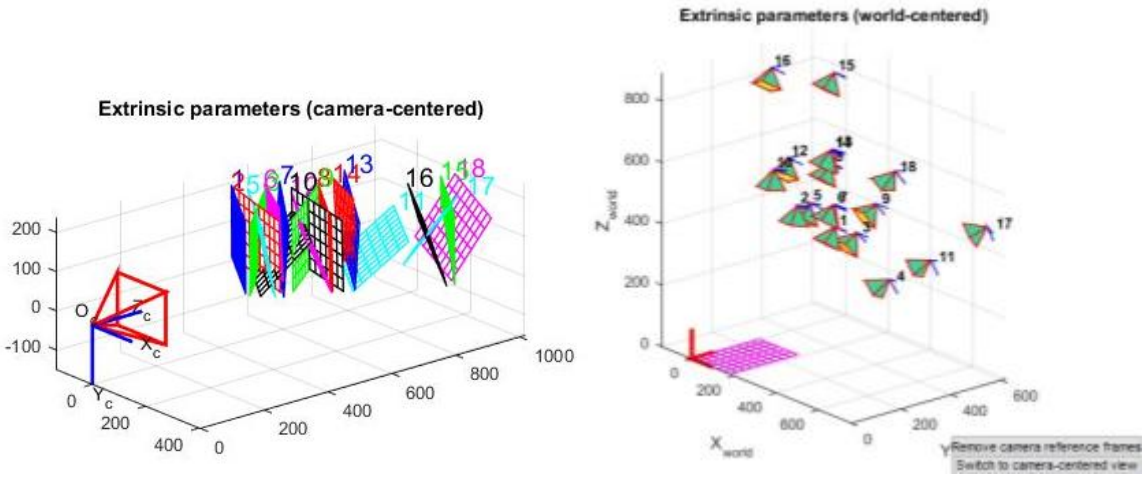

Figure 8: Extrinsic parameters of the left camera (3D plot)

2. *Right camera single calibration:*

Next the same process of the previous section (left camera) was performed on the right camera.

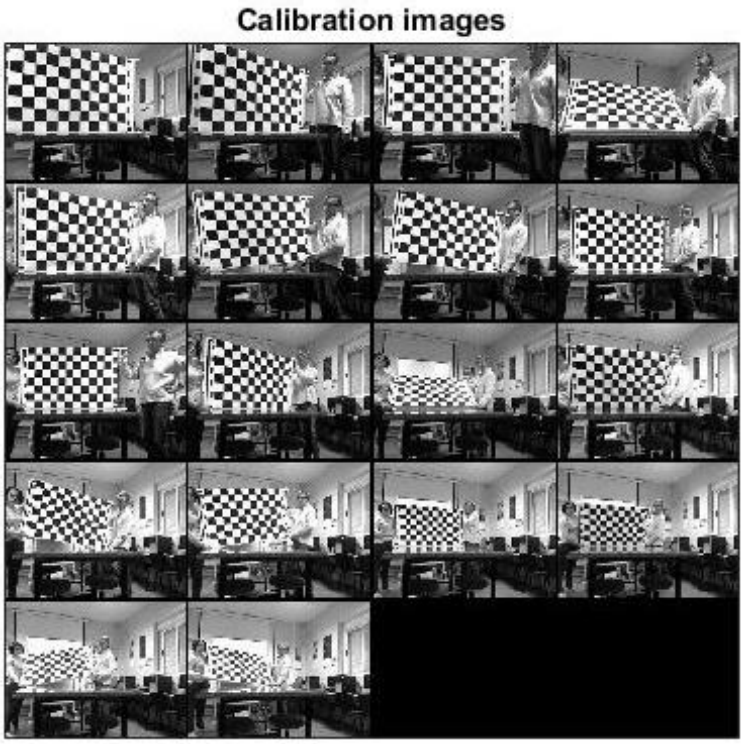

Figure 9: Image loading of 18 samples for single right camera calibration

The manual corner extraction was performed with a windows size of $(wintx = winty = 5)$ and $dX = dY = 30\ mm$.

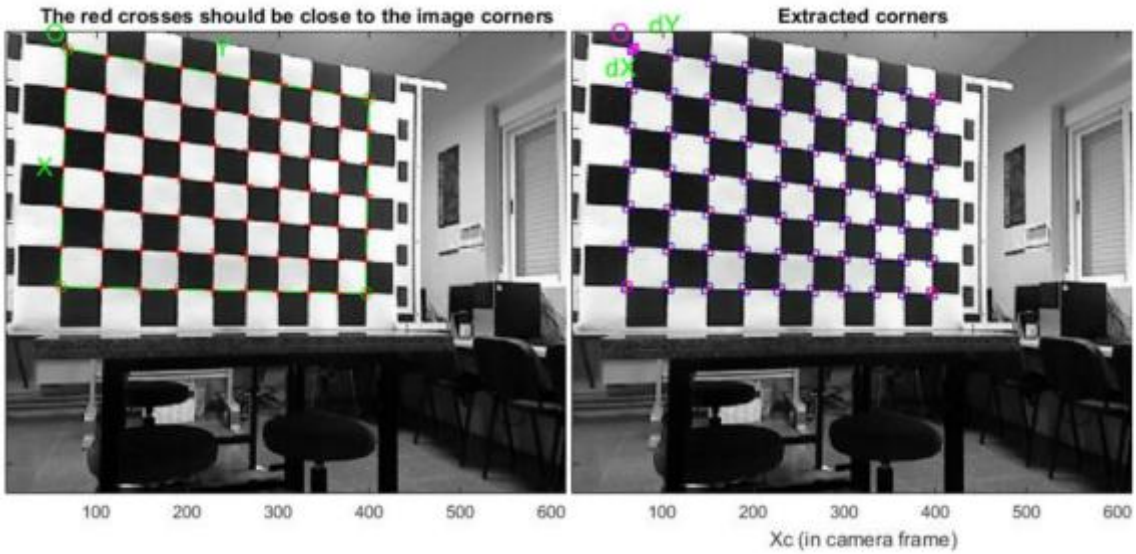

Figure 10: Corner extraction result for one of the right Kinect data

Followed the main calibration step was done (with the manual selection of the corners). Then, the corner extraction was computed automatically for all the images and the calibration was performed again.

Aspect ratio optimized (est_aspect_ratio = 1) -> both components of fc are estimated (DEFAULT).
Principal point optimized (center_optim=1) - (DEFAULT). To reject principal point, set center_optim=0
Skew not optimized (est_alpha=0) - (DEFAULT)
Distortion not fully estimated (defined by the variable est_dist):
Sixth order distortion not estimated (est_dist(5)=0) - (DEFAULT) .
Initialization of the principal point at the center of the image.
Initialization of the intrinsic parameters using the vanishing points of planar patterns.
Initialization of the intrinsic parameters - Number of images: 18
Calibration parameters after initialization:
Focal Length: fc = [540.37423 540.37423]
Principal point: cc = [319.50000 239.50000]
Skew: alpha_c = [0.00000] => angle of pixel = 90.00000 degrees
Distortion: kc = [0.00000 0.00000 0.00000 0.00000 0.00000]
Main calibration optimization procedure - Number of images: 18
Gradient descent iterations:
1...2...3...4...5...6...7...8...9...10...11...12...13...14...15...16...17...18...19...20...done
Estimation of uncertainties...done

Calibration results after optimization (with uncertainties):
Focal Length: fc = [537.17305 539.45765] +/- [4.88018 5.15538]
Principal point: cc = [328.37196 274.39770] +/- [4.89284 3.73384]
Skew: alpha_c = [0.00000] +/- [0.00000] => angle of pixel axes = 90.00000 +/- 0.00000 degrees
Distortion: kc = [0.16983 -0.23988 0.00477 0.01069 0.00000] +/- [0.01691 0.03029 0.00275 0.00366 0.00000]

Pixel error: err = [0.32978 0.28179]

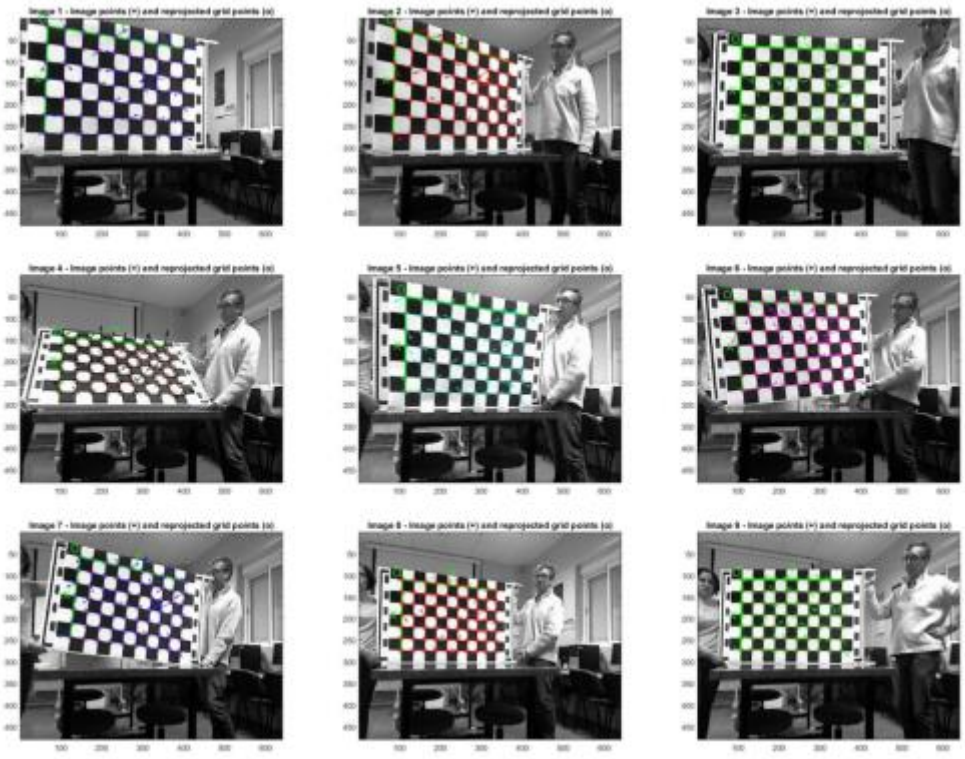

Figure 11: Reprojection (part 1) result for all images of right camera

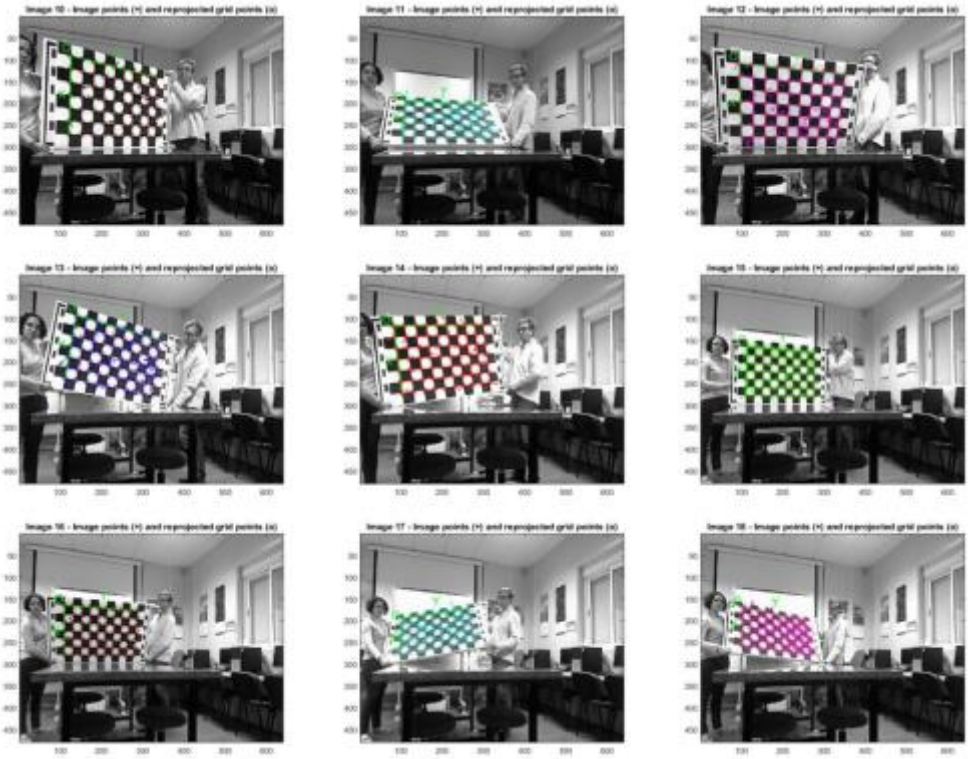

Figure 12: Reprojection (part 2) result for all images of right camera

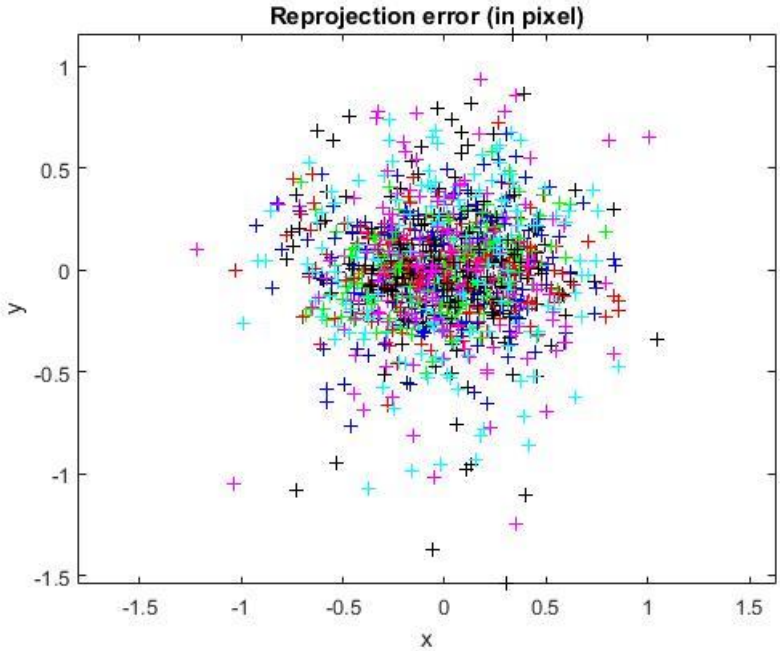

Figure 13: Reprojection error for right camera

Finally, the extrinsic parameters were showed as a 3D representation of relative positions of the grids with respect to the camera.

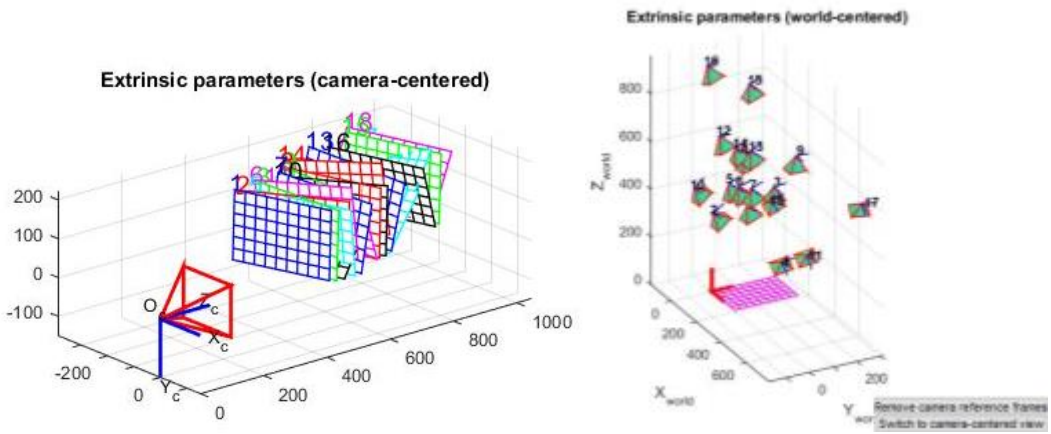

Figure 14: Extrinsic parameters of the right camera (3D plot)

3. *Stereo calibration:*

From results of the previous two sections (calibration of single left and right camera) the stereo calibration was performed. First, the calibration files (for each camera) were loaded:

Stereo calibration parameters after loading the individual calibration files:
Intrinsic parameters of left camera:
Focal Length: fc_left = [530.85490 533.56059] ± [5.13322 5.80701]
Principal point: cc_left = [300.52398 273.74748] ± [6.82469 5.18945]
Skew: alpha_c_left = [0.00000] ± [0.00000] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion: kc_left = [0.22710 -0.33859 0.01337 -0.01284 0.00000] ± [0.02193 0.05480 0.00441 0.00509 0.00000]

Intrinsic parameters of right camera:
Focal Length: fc_right = [537.18538 539.47036] ± [4.87933 5.15448]
Principal point: cc_right = [328.37726 274.40264] ± [4.89184 3.73319]

Skew: alpha_c_right = [0.00000] ± [0.00000] => angle of pixel axes = 90.00000 ± 0.00000 degrees

Distortion: kc_right = [0.16987 -0.23992 0.00477 0.01069 0.00000] ± [0.01691 0.03029 0.00275 0.00366 0.00000]

Extrinsic parameters (position of right camera wrt left camera):

Rotation vector: om = [-0.02366 -0.92063 0.03743]

Translation vector: T = [322.14597 6.52729 166.32499]

After that, the global stereo optimization procedure was done and the results saved. Figure 15 shows the spatial configuration of the two cameras.

Recomputation of the intrinsic parameters of the left camera
(recompute_intrinsic_left = 1)

Recomputation of the intrinsic parameters of the right camera
(recompute_intrinsic_right = 1)

Main stereo calibration optimization procedure - Number of pairs of images: 18
Gradient descent iterations: 1...2...3...4...done
Estimation of uncertainties...done

Stereo calibration parameters after optimization:

Intrinsic parameters of left camera:
Focal Length: fc_left = [525.86734 527.80597] ± [1.93172 2.24390]
Principal point: cc_left = [306.71831 267.07293] ± [4.26192 3.40349]
Skew: alpha_c_left = [0.00000] ± [0.00000] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion: kc_left = [0.23010 -0.38028 0.00838 -0.01009 0.00000] ± [0.01738 0.04695 0.00291 0.00371 0.00000]

Intrinsic parameters of right camera:
Focal Length: fc_right = [527.92155 530.36965] ± [2.12415 2.31646]
Principal point: cc_right = [321.72962 270.35185] ± [4.34163 3.36120]
Skew: alpha_c_right = [0.00000] ± [0.00000] => angle of pixel axes = 90.00000 ± 0.00000 degrees
Distortion: kc_right = [0.16402 -0.24480 0.00217 0.00700 0.00000] ± [0.01515 0.03009 0.00253 0.00352 0.00000]

Extrinsic parameters (position of right camera wrt left camera):
Rotation vector: om = [-0.01767 -0.89911 0.04436] ± [0.00820 0.01100 0.00358]

Translation vector: T = [320.68347 7.90718 156.95911] ± [1.50561 0.92538 2.84456]

Note: The numerical errors are approximately three times the standard deviations (for reference).

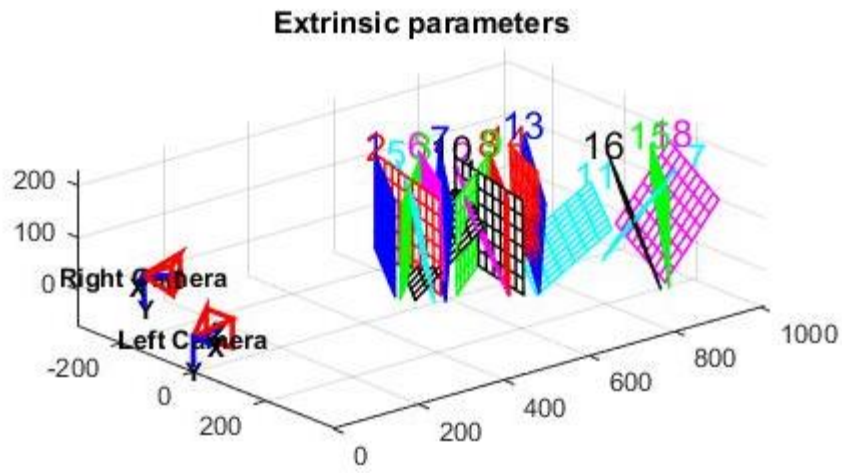

Figure 15: Extrinsic parameters of a stereo camera (3D)

The stereo calibration was followed by rectification of the stereo images used:

Calculating the rotation to be applied to the right and left images in order to bring the epipolar lines aligned with the horizontal scan lines, and in correspondence...

Saving the *NEW* set of intrinsic and extrinsic parameters corresponding to the images *AFTER* rectification under Calib_Results_stereo_rectified.mat...

Pre-computing the necessary data to quickly rectify the images (may take a while depending on the image resolution, but needs to be done only once - even for color images)...

Rectifying all the images (this should be fast)...

Loading image LeftRgbobjects1.jpg...
Image warping...
Saving image under LeftRgbobjects_rectified1.bmp...

Loading image RightRgbobjects1.jpg...
Image warping...
Saving image under RightRgbobjects_rectified1.bmp...

Loading image LeftRgbobjects2.jpg...

Image warping...
Saving image under LeftRgbobjects_rectified2.bmp...

Loading image RightRgbobjects2.jpg...
Image warping...
Saving image under RightRgbobjects_rectified2.bmp...
...
Loading image LeftRgbobjects17.jpg...
Image warping...
Saving image under LeftRgbobjects_rectified17.bmp...

Loading image RightRgbobjects17.jpg...
Image warping...
Saving image under RightRgbobjects_rectified17.bmp...

Loading image LeftRgbobjects18.jpg...
Image warping...
Saving image under LeftRgbobjects_rectified18.bmp...

Loading image RightRgbobjects18.jpg...
Image warping...
Saving image under RightRgbobjects_rectified18.bmp...

Figure 16: Rectification (part 1) result for all images of left camera

Figure 17: Rectification (part 2) result for all images of left camera

Figure 18: Rectification (part 1) result for all images of right camera

Figure 19: Rectification (part 2) result for all images of right camera

The 3D coordinates of the points of the left and right camera reference frames (Xc_1_left and Xc_1_right, respectively) and approximation of X_left_1 were also calculated. Figure 20 shows the Euclidian distance between the points and his approximated value.

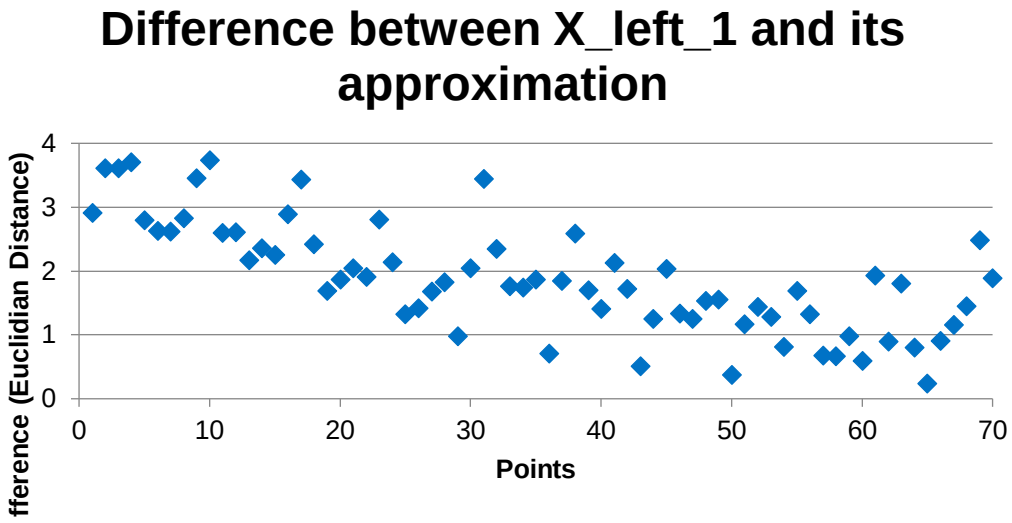

Figure 20: Euclidian distance between X_left_1 and X_left_approx_1

4. *Depth Kinect calibration:*

As the final calibration, also the data from the calibration of the Kinect sensors were obtained. The following results were found:

```
Kinect Depth camera parameters
fx_d =  5.7616540758591043e+02;
fy_d = 5.7375619782082447e+02;
cx_d = 3.2442516903961865e+02;
cy_d = 2.3584766381177013e+02;

Extrinsic parameters between RGB and Depth camera for Kinect V1
Rotation matrix
R = inv([ 9.9998579449446667e-01, 3.4203777687649762e-03, -
4.0880099301915437e-03;
  -3.4291385577729263e-03, 9.9999183503355726e-01, -2.1379604698021303e-03;
  4.0806639192662465e-03, 2.1519484514690057e-03, 9.9998935859330040e-
01]);

Translation vector.
T = -[  2.2142187053089738e-02, -1.4391632009665779e-04, -7.9356552371601212e-
03 ]';
```

Part 3: 3D reconstruction

For the last section, the topic was moved into three-dimensional vision, first with projections and then with multi-camera stereo depth perception. For this it was needed the camera intrinsic and extrinsic parameters calculated in the previous section.

1. *Mapping depth pixels with color pixels:*

Once calibration parameters were calculated, it was possible to unambiguously project points in the physical world to points in the image. This meant that, given a location in the three-dimensional physical coordinate frame attached to the camera, it was possible to compute where on the image, in pixel coordinates, an external 3D point should appear.

The images used for the reconstruction contains a scene, with objects of different size and shape. At the time of acquisition, it was verified that no object was being blocked by another one on either left or right camera. Figure 21 show the RGB and Depth images obtained on the left and right cameras.

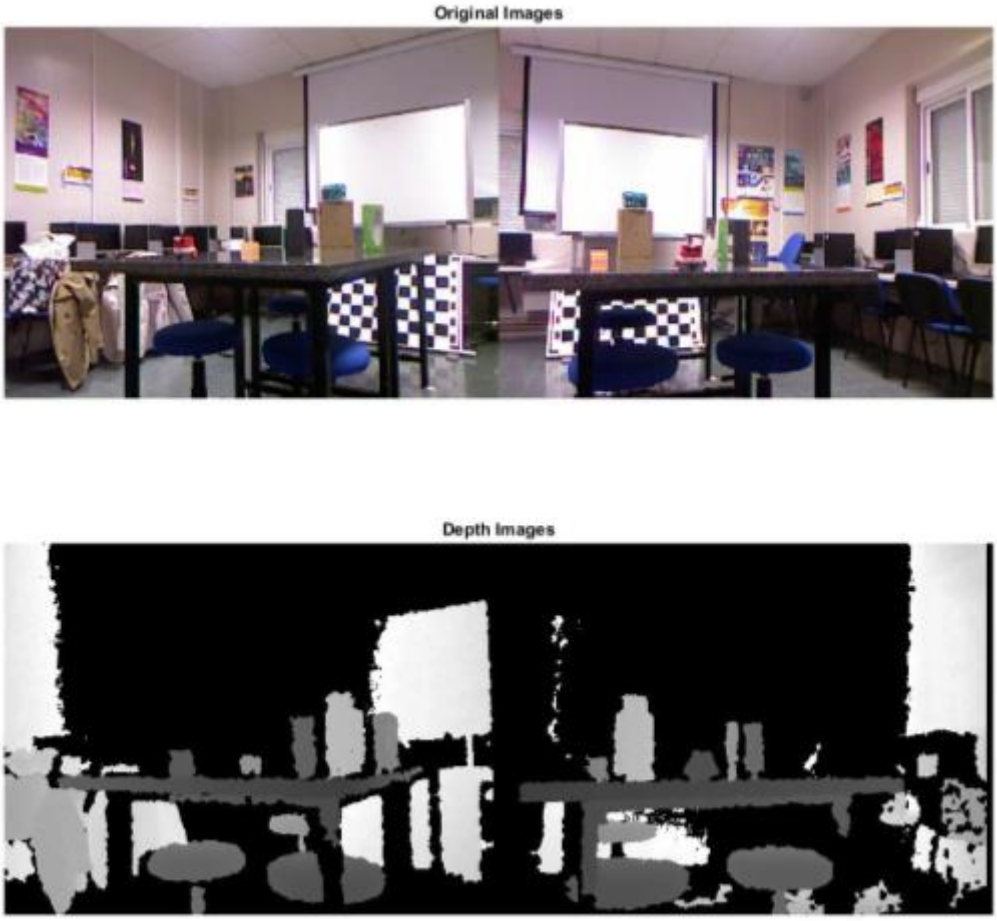

Figure 21: RGB and Depth images for the left and right cameras, respectively.

To project a 3D metric space, it was applied the intrinsic values of the depth camera to each pixel of a depth image using the following formulas:

$$P3D(x, y, 1) = (x - depth.cx_d) * \frac{D(x, y)}{depth.fx_d}$$

$$P3D(x, y, 2) = (y - depth.cy_d) * \frac{D(x, y)}{depth.fy_d}$$

$$P3D(x, y, 3) = D(x, y)$$

where $P3D$ was the projected 3D space, x and y were the position of each pixel, fx_d and fy_d were the focal point and cx_d and cy_d were the principal point. After these calculations, the values were plotted using the function $pcshow(xyzPoints, C)$, where $xyzPoints$ were replaced by $P3D$ and C was the specified color for these points. The final

operation result on the graphs that exhibit the image on a 3D environment. The results are shown on figure 22 (for the left images used as references) and figure 23 (for the images of the right camera).

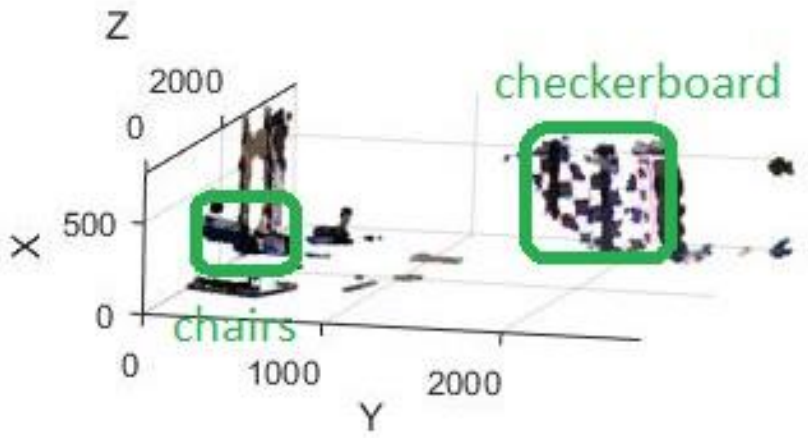

Figure 22: 3D projection for the image obtained on the left camera

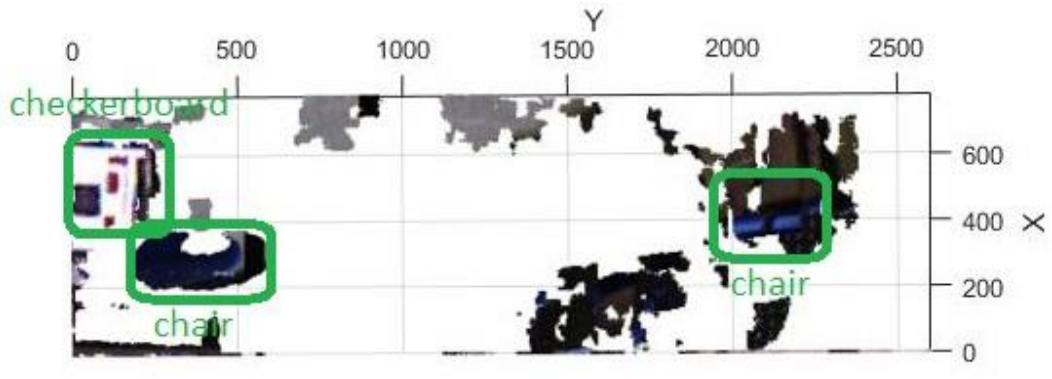

Figure 23: 3D projection for the image obtained on the right camera

Looking again for the depth images on figure 21, it's possible to observe that the upper part of the image was mainly black, which might be one of the reasons the reconstruction exhibited only the objects located on the bottom part (under the desk, like the checkerboard and the chairs found it).

Also, it was noted that not all the points were reconstructed and for that it took us some time to understand what the graph was really showing. Finally, because of the quality of our image, the 3D reconstruction using left and right images at the same time was not possible to test.

Part 4: Extra approach

As an extra approach, it was implemented also a reconstruction of a scene from a disparity map to compare the results from the previous section. The following steps were followed:

1. *Camera calibration:*

As the first step a calibration was performed. Instead of using the CalTech tool from the previous section, the Matlab checkboard detection tool was used. This procedure used a total of 8 images from the data set and started with an automatically corner extraction as seen on figure 24.

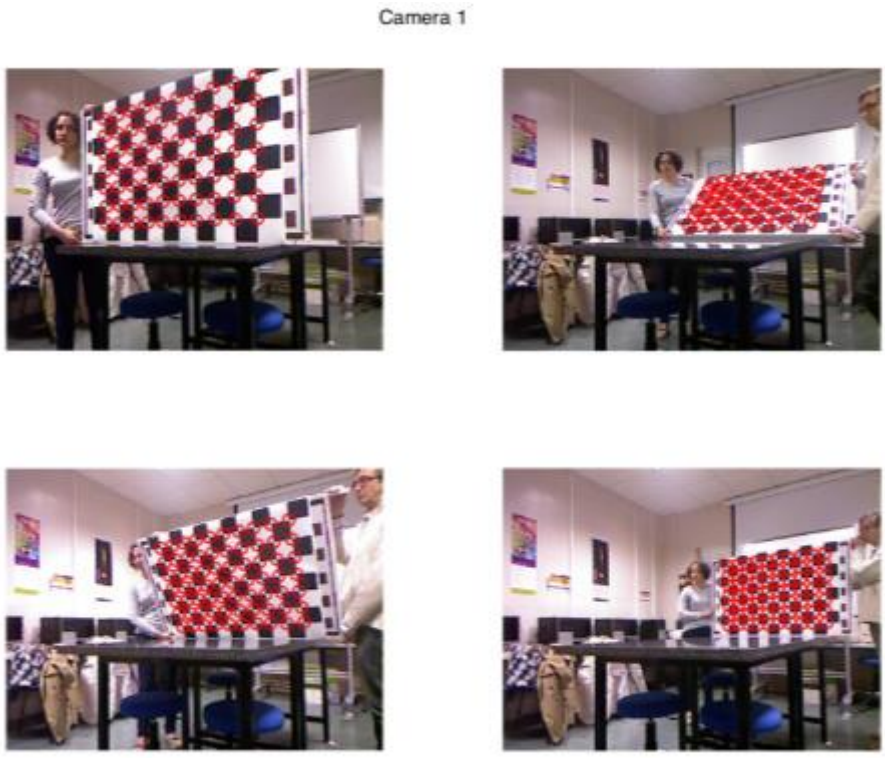

Figure 24: Corner extraction results for 4 of the left Kinect data

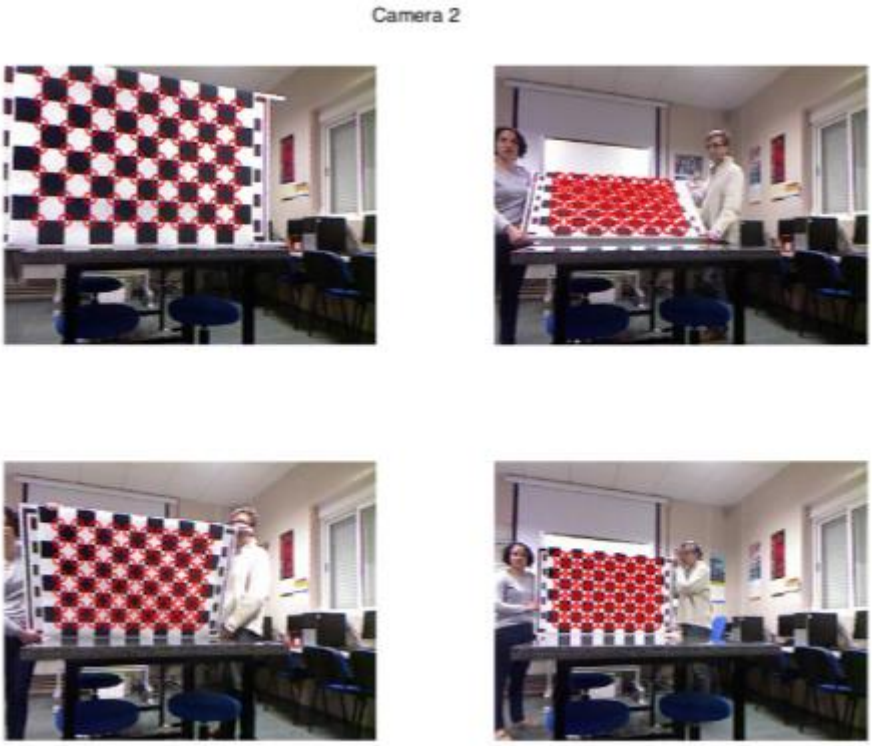

Figure 25: Corner extraction results for 4 of the right Kinect data

Followed the main calibration step was done for both cameras as a stereo vision system. Then the error was evaluated (mean value of 0.32) as is possible to been observe on figure 26.

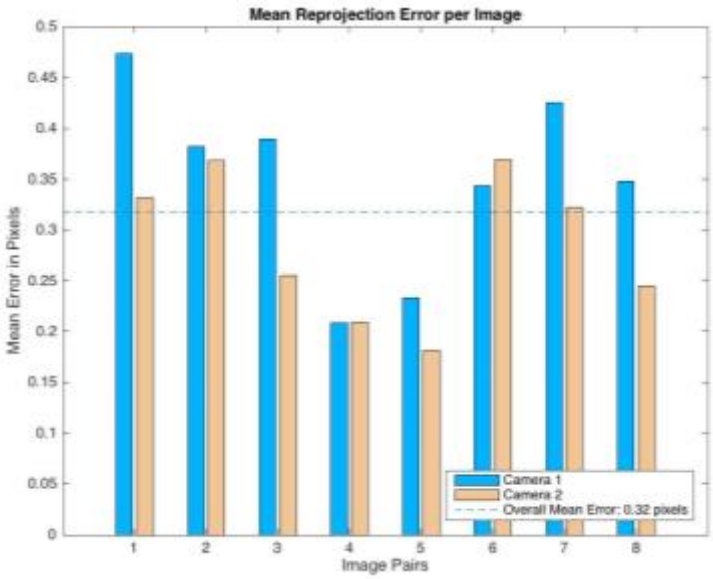

Figure 26: Error for calibration using Matlab toolbx

Finally, the extrinsic parameters were showed as a 3D representation of relative positions of the grids with respect to the camera.

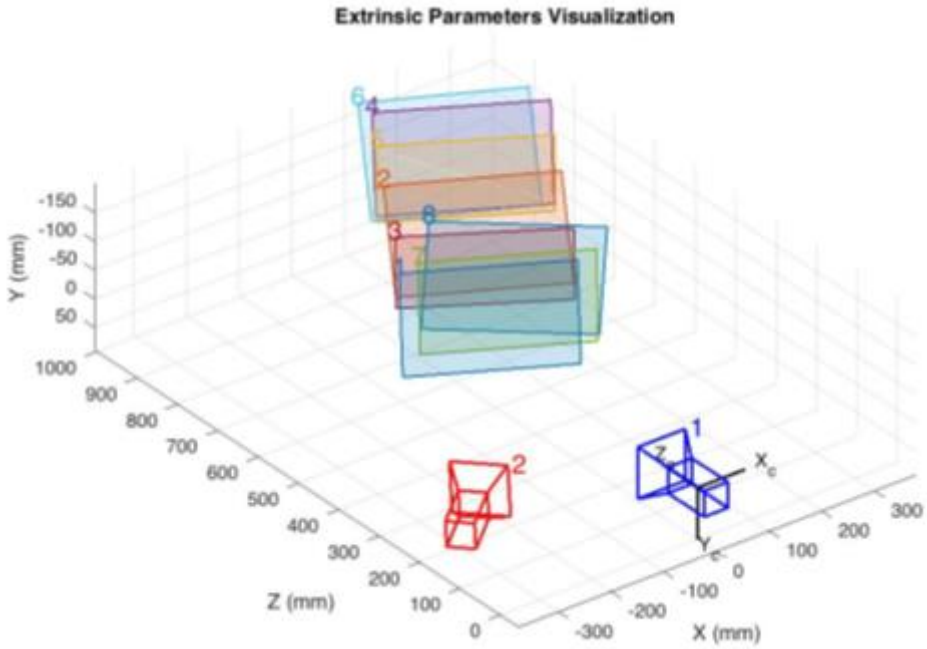

Figure 27: Extrinsic parameters of the stereo camera system (3D plot)

With these results, it was possible to load the two scene images (left and right) for the reconstruction.

2. *Remove distortions:*

Next the removal of the distortion from the two scene images was computed. Figure shows this result.

Figure 28: Distortion correction(down) for original images (up) of the stereo vision scene

3. *Rectification of images:*

With the result of the undistorted images from the previous section, the images were rectified to be able to create just one single image (figure).

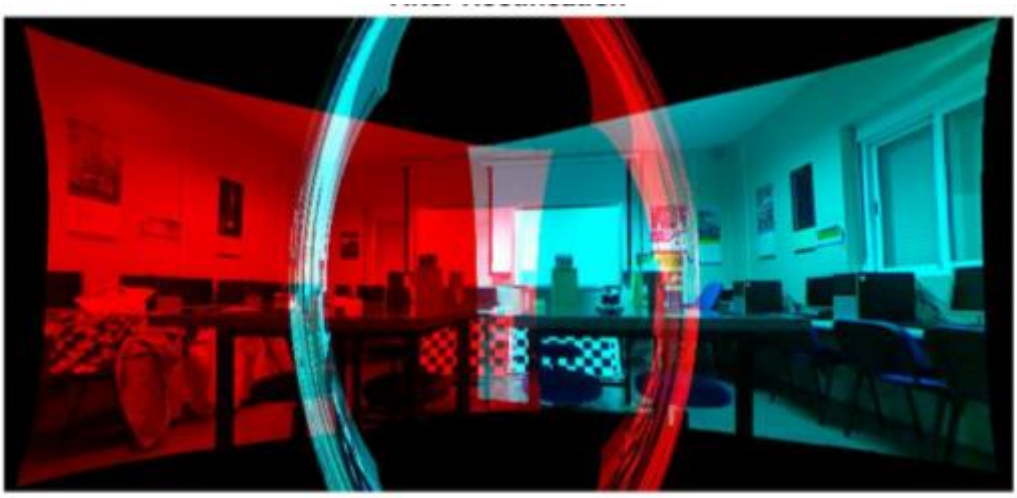

Figure 29: Rectification of the two scene images

4. *Disparity map:*

Finally, the disparity map was computed. With this information, it was possible to do also a cloud plot to do the reconstruction as the previous section.

Figure 30: Disparity map

CONCLUSIONS

The conclusions of the final lab session were the following:

- ✓ It was possible to follow all the flow of a 3D-reconstruction process.
- ✓ Even with good results for the stereo calibration, this does not assure that the related images are going to produce also good results for the reconstruction.
- ✓ The quality of the depth images has large influence on the results for the reconstruction, it is important to consider adding a human subject to improve these results because the Kinect sensor was design mostly for detection of humans.
- ✓ There exists various methods and tools to implement a stereo vision system. Nevertheless it is important to evaluate various options and choose the one with better result for the application.

REFERENCES

[1] Gary Bradski & Adrian Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*, O'Reilly Media, Inc., 2008.

[2] *Kinect calibration*, http://burrus.name/index.php/Research/KinectCalibration#tocLink7.

[3] Robert Collins, *Harris Corner Detector*, Penn State.