Vision for multiple and moving cameras
Lab Evaluation

Bolutife Atoki (CV6NBH)
bolutife.atoki@estudiante.uam.es
Universidad Autonóma de Madrid Escuela Politécnica Superior

Abstract

This paper contains my solution and implementation of the requirements for the Final Project (Lab Evaluation) Assignment,resources used are listed at the end of the paper.

1 Introduction

The solution in this report aims to achieve a 3D reconstruction of an object by capturing images from my phone's camera, calibrating it, and extracting feature points from multiple views. The obtained feature points were used to compute the fundamental matrix between views. The final step involved creating a 3D point cloud reconstruction and representing the object's geometric elements using this point cloud.

2 Section 1: Obtaining the intrinsic parameters of the camera of a mobile phone

The objective of this section was to calibrate my mobile phone's camera (obtaining its internal and external parameters), while ensuring that there were no changes in the focal distance nor changes in the aspect ratio during the calibration process.
Zhang's method was used to obtain the parameters of the camera from mxn correspondences $m_{ij} \longleftrightarrow M_{ij}$, (i=1,...,n), (j=1,...,m, m≥3), with n points from m images of a planar pattern. The pattern and point markings are shown below:

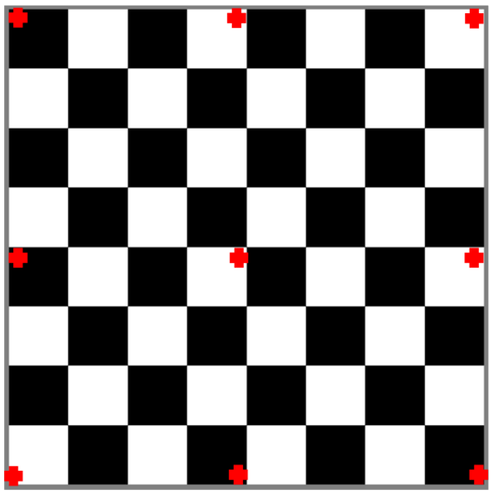


Figure 1: Checkerboard Planar Pattern

2.1 Implementing Zhang's calibration using the provided checkerboard pattern

The planar pattern used was the famous checkerboard pattern provided (the 1080p resolution copy), which was displayed on the monitor at the classroom. Such that 6 images were taken and marked in the same way as shown in Figure 2. A montage of the 6 images are shown below:

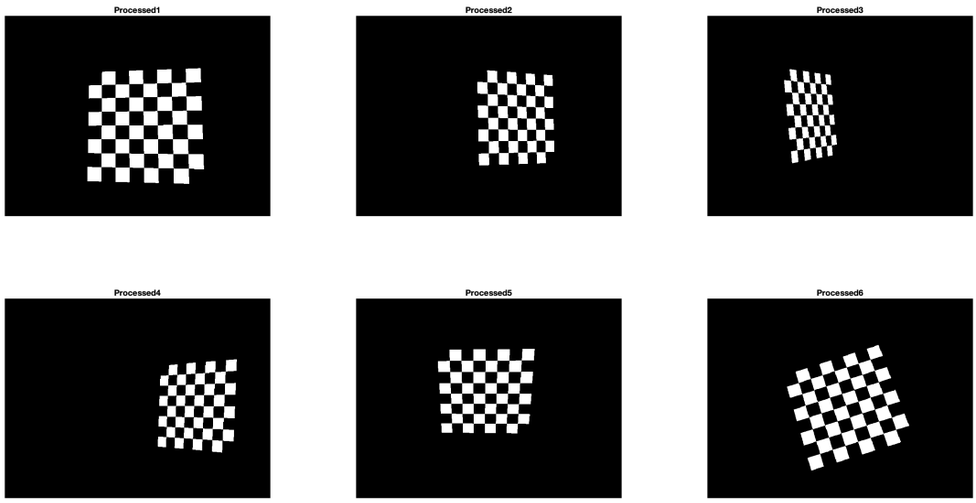


Figure 2: Montage of Processed screen checkerboard images used for calibration.

The information obtained during this process are presented below:

1. The measured size of the checkerboard on the screen was **260mm**
2. Since the same focal distance and aspect ratio was used for all images, the resolution for each was **4032** $\times$ **3024** pixels
3. The obtained matrix of internal parmeters is

$$\begin{bmatrix} 6075.2 & 17.1392 & 1458.2 \\ 0 & 6131.4 & 2020.6 \\ 0 & 0 & 1 \end{bmatrix}$$

4. The squareness of the pixels was evaluated via the closeness of the focal lengths f_x and f_y to each other, and also by the pixel aspect ratio i.e the closeness of the ratio of f_x and f_y to 1. Given $f_x = 6075.2$ and $f_y = 6131.4$, we see that they are close to each other and also that the ratio of them is 0.99, which is close to 1, therefore the pixels can be said to be square.
5. The degree of coincidence between the principal point and the center of the image plane given principal point $C_x = 1458.2$ and $C_y = 2020.6$, is obtained with the equation

$$\sqrt{((C_x - 0)^2 + (C_y - 0)^2)}$$

$$\sqrt{((1458.2 - 0)^2 + (2020.6 - 0)^2)}$$

$$\text{distance} = 2488.1971$$

6. the orthogonality of the axes of the image plane is determined by examining the skew parameter in the camera matrix which is 17.13.

2.2 Implementing Zhang's calibration using self created pattern

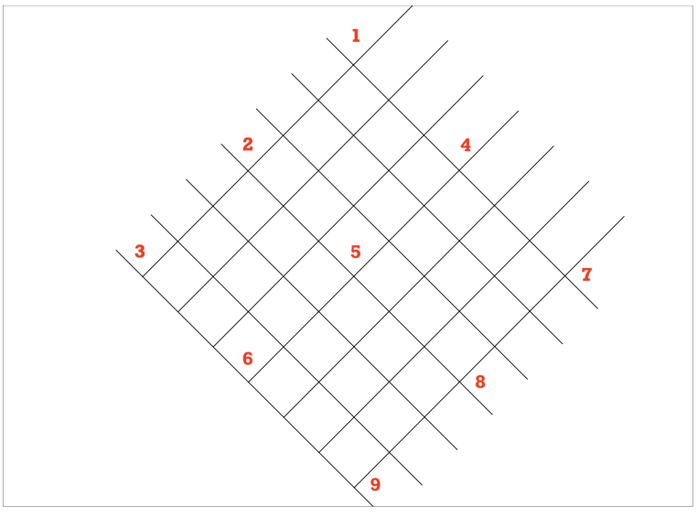


Figure 3: Checkerboard Planar Pattern

The figure 3 above shows the planar pattern that was used. It was created by intersecting parallel line at 45 deg. It was constructed on AutoCAD and exported, after which it was displayed on the screen of the 55inch television at home, and pictures were taken for the calibration. 9 points were chosen at the intersections of these parallel lines and then a function for obtaining the origin coordinates for these points was written and implemented. As with the previous calibration, 6 images were also taken and marked.

For obtaining the origin coordinates of the points, the function **get_real_points_parrallel_board_vmmc** was created. Then the same steps from the previous exercise were performed to obtain the calibration.
A montage of the 6 images are shown below:

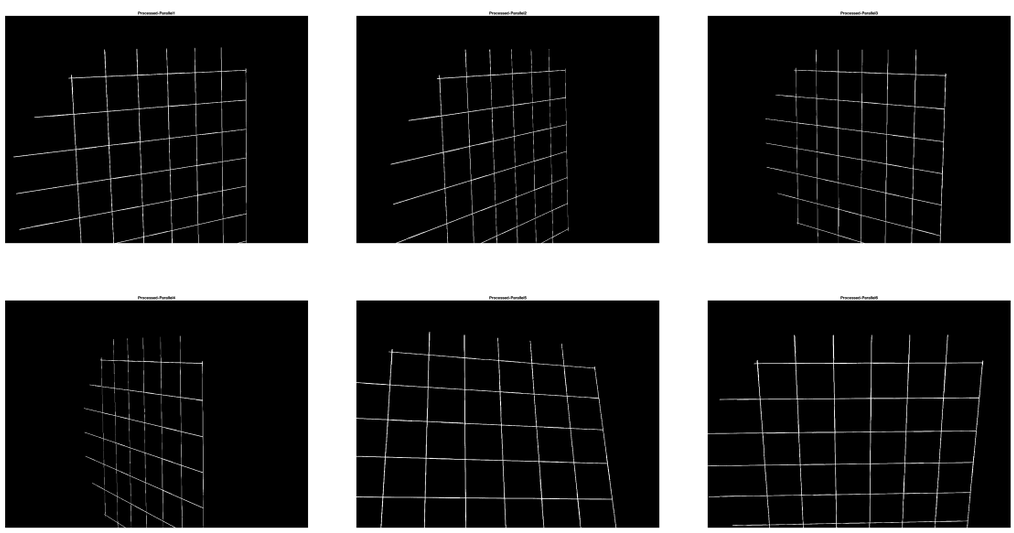


Figure 4: Montage of created pattern images used for calibration.

The information obtained during this process are presented below:

1. The measured size of the patten on the TV screen was **388mm**
2. Since the same focal distance and aspect ratio was used for all images, the resolution for each was **4032** $\times$ **3024** pixels
3. The obtained matrix of internal parmeters is

$$\begin{bmatrix} 6197.2 & 46.6 & 2032.6 \\ 0 & 6153.3 & 1290.6 \\ 0 & 0 & 1 \end{bmatrix}$$

4. The squareness of the pixels was evaluated via the closeness of the focal lengths $f_x and f_y$ to each other, and also by the pixel aspect ratio i.e the closeness of the ratio of $f_x and f_y$ to 1. Given $f_x = 6197.2$ and $f_y = 6153.3$, we see that they are close to each other and also that the ratio of them is 1.007, therefore the pixels can be said to be square.

5. The degree of coincidence between the principal point and the center of the image plane given principal point $C_x = 1458.2$ and $C_y = 2020.6$, is obtained with the equation

$$\sqrt{((C_x - 0)^2 + (C_y - 0)^2)}$$

$$\sqrt{((2032.6 - 0)^2 + (1290.6 - 0)^2)}$$

$$\text{distance} = 2407.72$$

6. the orthogonality of the axes of the image plane is determined by examining the skew parameter in the camera matrix which is 46.63

2.3 Conclusion

From the examination of both A matrices, the values obtained are similar with almost exact values obtained for squareness and The degree of coincidence, theoretically however since the same camera and settings (focal distance, aspect ratio, etc) were used, the values of both Matrices should be the exact same. In practice however due to the fact errors from marking the points, using an unconventional planar pattern, etc slight variations and noise were introduced which led to the deviation between the two matrices. Hence the camera parameters obtained from calibrating with the given checkerboard were used in subsequent tasks.

3 Section 2: Finding local matches between several views of an object.

The objective of this section was to set up a scene, use the already calibrated camera to capture several views of the scene. Then, for pairs of views, to detect, describe, and match feature points (Keypoints) using combinations of detectors and descriptors, and finally to create a new detector.

3.1 Object scene and Capturing

The scene used consists of 5 carved wooden owl dolls placed at angles and positions from each other to simulate depth to be observed in 3D point cloud reconstructions. The dolls were selected as they are highly textured and have sharp and rectilinear contours, which will increase the likelihood and number of interesting points (Keypoints).

With the calibrated mobile camera, the scene was moved around and different views were captured, by varying tilt, roll, pitch and distance of the camera. 14 pictures were taken to get better 3D point cloud reconstruction, while noting that it might result in additional difficulty in effectively matching the obtained Keypoints.

Possible Challenges noted, avoided or made robust-to during scene setup include:

1. No object parts were hidden from the camera's view, also Occlusions between objects was avoided, while still ensuring depth is simulated.
2. Reflective or transparent surfaces that may cause reflections or refractions and inaccurate depth measurement were avoided.
3. Objects used did not pose any Depth ambiguity.

The images captured of the scene are given in a montage below:

Figure 5: Montage of Scene from different viewpoints

3.2 Description of Implemented Detector: Harris Laplace

The implemented detector was the Harris-Laplace detector which combines the Harris corner detector with the Laplacian of Gaussian (LoG) scale-space representation to identify and localize interest points in an image based on their spatial and scale properties. The steps involved in the implementation include:

1. **Scale-space construction**, where the image is convolved with a series of Gaussian filters at different scales to create a scale-space representation.
2. **Harris corner detection**, such that at each scale in the scale-space pyramid, the Harris corner detection algorithm is applied to identify potential interest points by analyzing local intensity variations and identifying regions with significant changes in intensity.
3. **Laplacian response calculation**, where the Laplacian operator is applied to each pixel in the scale-space pyramid to calculate the Laplacian response.
4. **Non-maximum suppression**, to eliminate weak or redundant responses and retain only the strongest and most distinctive features.
5. **Scale selection:**, finally interest points are assigned scales corresponding to the scale at which they were detected in the scale-space pyramid.

Due to the many convolutions being performed and on multiple scales, its performance was really slow and it was not applied in the detector-descriptor combinations. However an example image showing detected points is shown below:

Figure 6: Keypoints detected by Implemented Harris-Laplace detector

3.3 Detection, description and matching of feature points

Picking image pair 1 and 4, different detector and descriptor combinations were tried out while extracting and describing feature points, matching these points between both views, estimating the homography transformation between both views, and finally the fundamental matrix between both views. In order to ensure consistency in the comparison of the different combinations, the detector parameters were kept constant are are given below: params.nscales = 12, params.noctaves = 5, params.sigma0 = 1.6, params.npoints = 500, params.th = 0.001
The following combinations were tested out as per the requirements: DoH + SIFT, SURF + SURF, KAZE + KAZE, SIFT + DSP-SIFT, SURF + KAZE. For the Quantitative comparison between the different combinations, the following values were estimated for each combination and presented in a table:

1. The reprojection error obtained by using the estimated fundamental matrix to calculate the Projection matrix and then the respective 3D points.
2. The number of inliers from homography estimation between the two views.
3. The number of inliers, from estimated fundamental matrix between the two views
4. The total number of matched points.

Metric	SURF + SURF	KAZE + KAZE	SIFT + DSP-SIFT	KAZE + SURF	SURF + KAZE
Reprojection error obtained by using the estimated fundamental matrix	0	0	0	0	0
Number of inliers from homography estimation between the two views	22	102	26	50	1
Number of inliers, from estimated fundamental matrix between the two views	288	848	116	900	1
The total number of matched points	576	1695	232	1799	1

From the comparative table above of the different combinations, it is observed that the KAZE + SURF combination has the most number of inliers from the fundamental matrix (900) and the most number of matched points (1799), hence this combination was used to evaluate other image pairs, also because it had the fast computation speed amongst all combinations with the DoH + SIFT being the slowest, due to the DoH detector used.
Again, Quantitative evaluation is performed for different view pairing using the same metrics as before and represented in a table below:

Metric	Im1 & Im5	Im2 & Im7	Im4 & Im6	Im6 & Im10	Im3 & Im8	Im5 & Im7	Im8 & Im10	Im1 & Im4
Reprojection error obtained by using the estimated fundamental matrix	0	0	0	0	0	0	0	0
Number of inliers from homography estimation between the two views	63	35	102	113	35	257	203	61
Number of inliers, from estimated fundamental matrix between the two views	779	526	1205	657	568	1520	999	904
The total number of matched points	1558	1051	2409	1313	1135	3040	1998	1807

3.4 Individual capabilities of Detectors and Descriptors

Detectors:

1. **SIFT:** They are robust to viewpoint changes, are invariant to changes in lighting conditions and partial occlusions, and finally perform well in scenes with rich and distinct textures.
2. **SURF:** They are fast, robust to changes in scale and rotation, handle partial occlusions and scenes with clutter, and are relatively illumination invariant, however don't have as much performance in scenes with very repetitive patterns or in images with low-texture regions.
3. **KAZE:** They are designed to handle challenging environments with significant changes in scale, viewpoint, and lighting conditions, as well as effectively detect keypoints in textureless regions or areas with low contrast.

Descriptors:

1. **DSP-SIFT:** They provide faster feature extraction and matching, making them suitable for real-time applications where quick computation is required, or in resource-constrained environments.
2. **KAZE:** They are designed to be robust to changes in illumination and viewpoint,
3. **SURF:** They are efficient in handling large-scale image datasets due to their speed and scalability. They can describe local image structures across a large number of images, making them useful for tasks like image retrieval, image stitching, and content-based image analysis.

3.5 For the best combination

The pair of images with the correspondences on them:

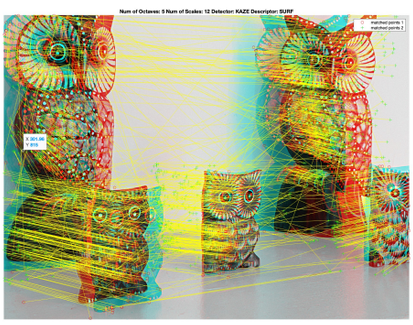


Figure 7: The pair of images with the correspondences overlaid on them

The warped images given below:

(a) Transformation from Image 1 to Image 2

(b) Transformation from Image 2 to Image 1

Figure 8: The warped images

The estimated Homography matrices are given below:

$$T_{12} = \begin{bmatrix} 0.67 & -0.085 & -9.41e-05 \\ -0.043 & 0.826 & 3.95e-06 \\ 0 & 0 & 1 \end{bmatrix} T_{21} = \begin{bmatrix} 1.64 & 0.207 & 0.0002 \\ -0.122 & 1.27 & -7.00e-05 \\ -350.17 & -271.48 & 1 \end{bmatrix}$$

The estimated Fundamental matrix is given below:

$$F = \begin{bmatrix} -7.41402028578191e-08 & 1.00436708431826e-06 & -0.000137570937895065 \\ -2.32008086601429e-07 & 1.05663177438271e-07 & -0.00693100944517441 \\ -0.000338587130935425 & 0.00534757144502485 & 0.999961615468767 \end{bmatrix}$$

The Screen captures of the vgg_gui_F.m GUI are given below

(a) Epipolar line from Image 1 to Image 2

(b) Epipolar line from image 2 to 1

Figure 9: The Screen captures of the vgg_gui_F.m GUI

4 Section 3: 3D reconstruction and calibration

This section involved using the intrinsic parameters matrix (K) of the camera, obtained in Section 1 and the Keypoints for each image obtained in Section 2, to obtain a 3D reconstruction of your object/scene (a 3D point cloud). The following steps were implemented as suggested with results for each step attached per step:

4.1 Computing consistent point matches among N views

This was done using the **n_view_matching** by passing it the detected keypoints, locations and images (for visualization) to obtain those keypoints matched for all images. These obtained points then had to be homogenized by adding the third coordinate of 1 to each point. The matched points in all images, for each image is shown in the monatage below:

Figure 10: Monatage of matched points in each image for all images

4.2 Computing the Fundamental matrix and an initial projective reconstruction

This was done using the cameras at the extreme ends i.e views from both corners (2 cameras), and the same steps as performed in the lab were carried out:

1. Using the **MatFunProjectiveCalib** function to obtain the Fundamental and Projective Matrices using the matched points from the first and last cameras alone
2. Using the 2d points and the triangulted 3D points to obtain the reprojection error due to the Projection matrix

The two images used for the estimation of the fundamental and projection matrices are given below:

(a) Image from first camera

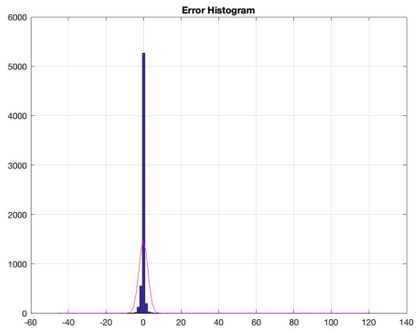
(b) Image from last camera

Figure 11: Images used for estimating Fundamental and Projection matrix using just two camera views

The reprojection error and the Reprojection error histogram are given as a screenshot from matlab and an image respetively below:

```
Minimum singular value = 0.33367
Resudual reprojection error. 8 point algorithm   = 5.4567
Pixel error:          mean = [  0.02174   -0.18579]
Pixel error:          std  = [  0.43252    3.27030]
Cost EIG = 2.1877e-11
Cost EIG = 7.1648e-11
Cost EIG = 1.2882e-10
```

(a) Reprojection error



(b) Reprojection error histogram

Figure 12: Reprojection error and Reprojection error histogram

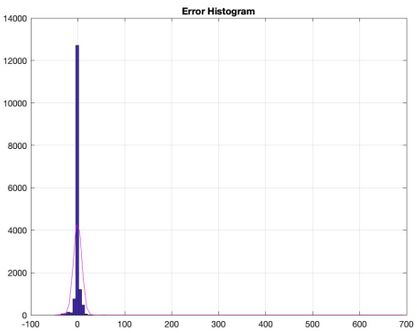
4.3 Improving the initial reconstruction by Resectioning and Projective Bundle Adjustment using all selected views

This was done by initially resectioning, which involved using the already obtained projection matrices for both cameras to obtain the projection matrices of the rest of cameras via the **PDLT_NA** function. This was followed by Projective Bundle Adjustment which used the recently estimated projection matrices for all cameras as well as their corresponding 2D and 3D points to refine the Projection matrices and points and obtain those having the least reprojection error. The reprojection error and the Reprojection error histogram after each of the two steps (resectioning and Projective Bundle Adjustment) were recorded and provided.

Reprojection Error and Histogram after Resectioning:

```
Residual reprojection error, After resectioning  = 74.0234
Pixel error:          mean = [  0.03910   -0.61113]
Pixel error:          std  = [  11.70620    3.26429]
```

(a) Reprojection error



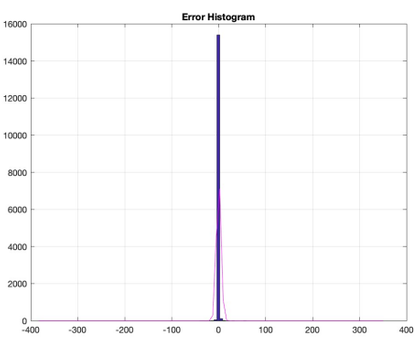
(b) Reprojection error histogram

Figure 13: Reprojection error and Reprojection error histogram

Reprojection Error and Histogram after Projective Bundle Adjustment:

```
Reprojection error, After Bundle Adjustment  = 23.836
Pixel error:          mean = [  0.00000    0.00000]
Pixel error:          std  = [  6.67297    1.77471]
Cost EIG = 3.1818e-06
Cost EIG = 3.859e-06
Cost EIG = 2.8983e-06
```

(a) Reprojection error



(b) Reprojection error histogram

Figure 14: Reprojection error and Reprojection error histogram

From the observations above we see that there is a decrease in the reprojection error after after resectioning and after Bundle Adjustment because:

1. Projective bundle adjustment optimizes the 3D coordinates of the observed points, by adjust-

ing the 3D positions of the points in the scene until it finds a configuration that minimizes the discrepancy between the observed image points and their corresponding reprojected points. This refinement ensures that the 3D points align more accurately with their corresponding image measurements.

2. Consistency across multiple views: Projective bundle adjustment optimizes the projection matrix and 3D points jointly, taking into account the information from multiple views. By considering the relationships and constraints between the views, the algorithm seeks a solution that is consistent across all observations. This helps in reducing the overall reprojection error and achieving a more coherent and accurate reconstruction.

4.4 Euclidean reconstruction of the scene

1. First a new Fundamental matrix was obtained using the refined projective matrices of the first and last cameras (i.e the ones obtained after Bundle Adjustment) via the **vgg_F_from_P** function.

2. Then the matrix of internal parameters computed in the first section was called and scaled (using the same scales the images were downsampled with).

3. Both matrices were used to obtain the Essential matrix (which is basically a fundamental matrix that accounts for the Matrix of internal parameters of the camera).

4. Next the Rotational and Translational matrices for both cameras were obtained and then Euclidean projection matrices for both were also obtained

5. Then resectioning was used to extend these Euclidean projection matrices obtained for both cameras to all the cameras

6. Finally the aggregated mean re-projection error and the reprojection error histogram for all cameras were calculated.

The different solutions for the reconstructed scene are given below:

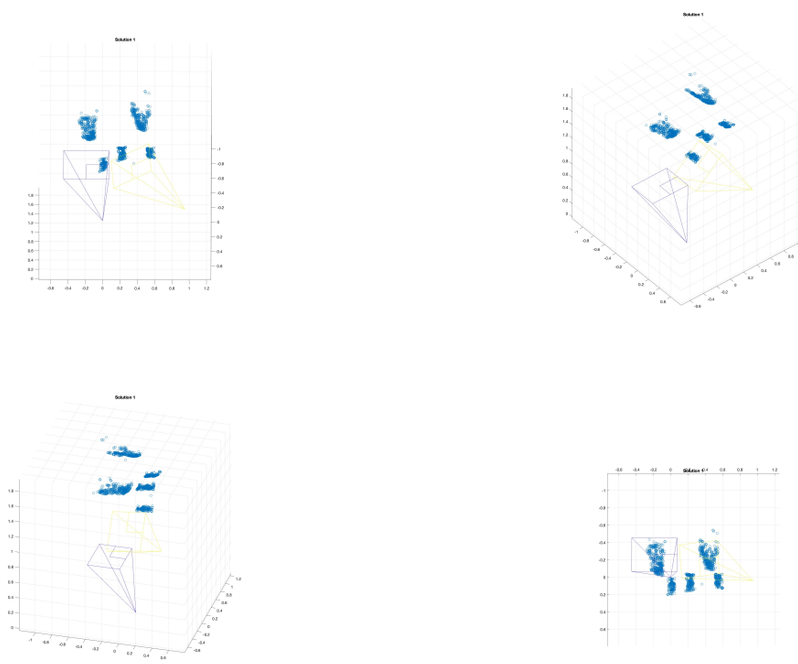
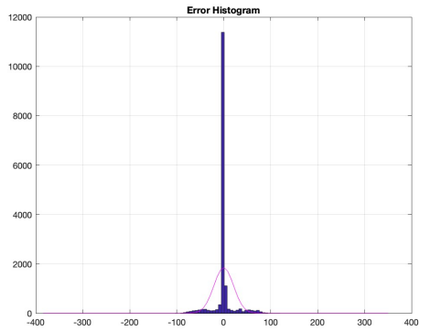


Figure 15: Different viewpoints of 3D point cloud reconstruction

The mean re-projection error and the reprojection error histogram aggregated for all cameras are given below:

```
Cost EIG = 3.9599e-06
Cost EIG = 8.6893e-06
Cost EIG = 2.7499e-05
Residual reprojection error, After final resectioning for all cameras = 174.6627
Pixel error:          mean = [ -0.33823    0.56287]
Pixel error:          std  = [  6.87096   28.73646]
************************************ END
```

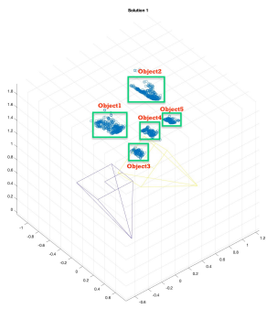
(a) Reprojection error



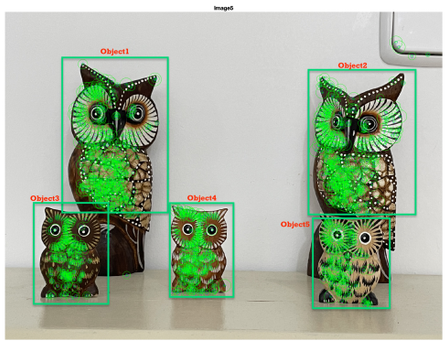
(b) Reprojection error histogram

Figure 16: Aggregated Reprojection error and Reprojection error histogram for all cameras

The visualization of the scene in the 3D world is given below, with objects in the scene and their corresponding point cloud reconstructions:



(a) Point cloud



(b) 3D scene

Figure 17: Comparison between 3D scene and obtained point cloud