Computer Vision (7315)

2020 Computer Vision
Assignment 2: 3D Reconstruction

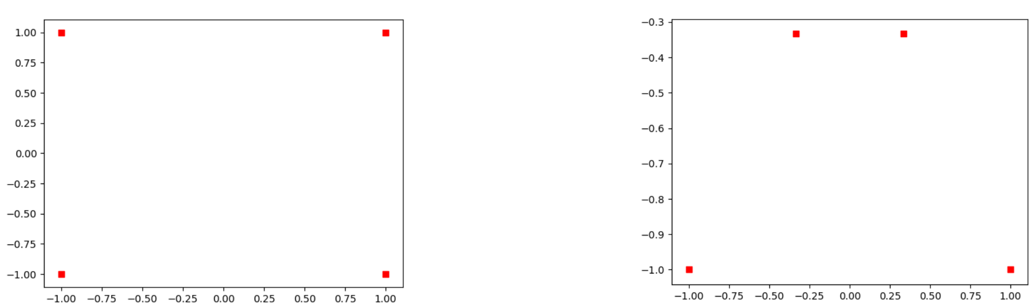
Vandit Jyotindra Gajjar
School of Computer Science, The University of Adelaide
vanditjyotindra.gajjar@student.adelaide.edu.au
Student ID: a1779153

Abstract

In Computer Vision, the process for retrieval of camera geometry and 3D reconstruction is a major problem. This assignment/report will mainly focus on those challenges in recovering 3D geometry. In general, we will work on the fundamentals of projective geometry which includes cartesian/heterogeneous coordinates, homogeneous coordinates, and certain transformations (i.e. rotational, translational, etc.). Also, using OpenCV's triangulation in-built function, we will try to get the idea behind errors in 3D space, when point locations are injected with some amount of noise. Finally, this report focuses to solve the assignment questions with detailed analysis and in-depth context relating to internal and external parameters for camera model, 3D reconstruction of the point cloud, error estimation in stereo vision when point locations are injected with noise, and essential matrix for pose estimation. To evaluate our results, we use random homogeneous arrays and point locations using the OpenCV library and Python language. The source code is publicly available at *https://github.com/Vanditg/COMP-SCI-7315---Computer-Vision*

Task 1
Answer (a)

Firstly, we use given points to project the scatter plot which is shown in Figure 1. The four homogeneous 3D points are x = ±1, y = ±1, z = 1, and w = 1. The result for these points is the square in the graph as coordinates are [±1/1, ±1/1] shown in Figure 1(a). Now, we change these points to the value as follows: x = ±1, y = -1, z = 2 ± 1, and w = 1 and plot these in scatter plot. As we can see that from the upper two points are shifted left and right respectively in Figure 1(b). The focal length here used in unit focal length.



(a) (b)

Figure 1. Scatter plot for given homogeneous points. (a) has the points x = ±1, y = ±1, z = 1, and w = 1, while (b) has the points x = ±1, y = -1, z = 2 ± 1, and w = 1. The focal length in this case is a unit focal length. (Best view in magnification and color.)

Now, we change the focal length progressively and see the difference in the scatter plot. For 3D points x = ±1, y = ±1, z = 1, and w = 1, Figure 2 shows the difference in projection while for 3D points x = ±1, y = -1, z = 2 ± 1, and w = 1, Figure 3 shows the difference in projection by progressively changing the focal length from 5, 15, 25, 50.

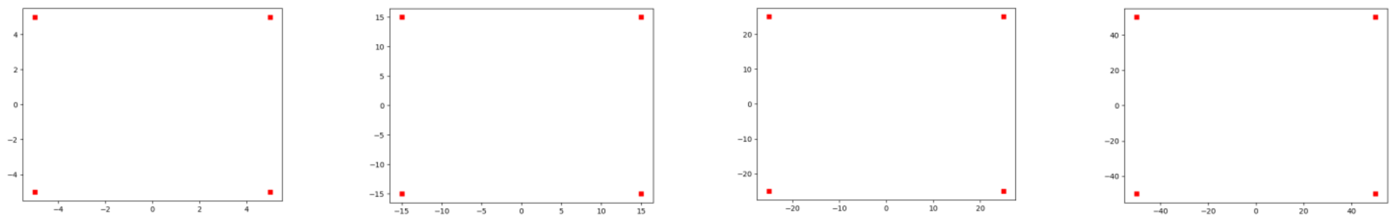


Figure 2. Progressively changing the focal length from 5, 15, 25, 50 for the 3D points x = ±1, y = ±1, z = 1, and w = 1. We can see that by changing the focal length the projection seems zoomed in. (Best view in magnification and color.)

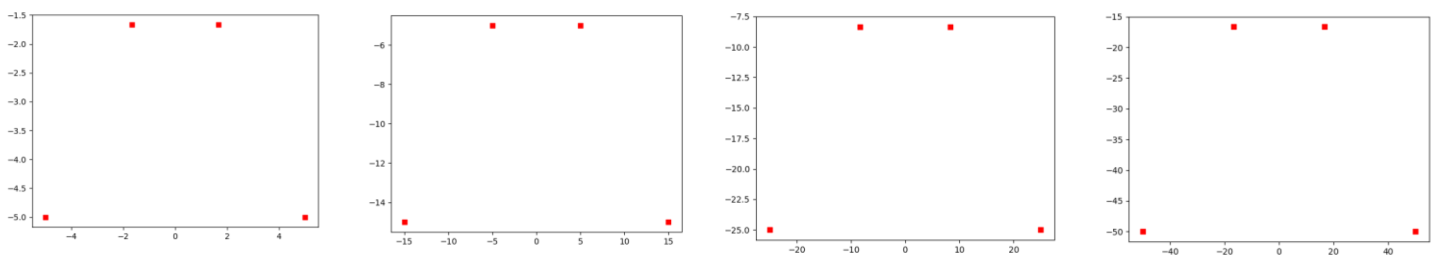


Figure 3 Progressively changing the focal length from 5, 15, 25, 50 for the 3D points x = ±1, y = -1, z = 2 ± 1, and w = 1. We can see that by changing the focal length the projection seems zoomed in. (Best view in magnification and color.)

By visualizing this projection using progressively changing the focal length from 5 to 50, we can see that the projection is somewhat zoomed in as appears exactly in the real camera.

- The focal length of an optical system (i.e. camera) is a measure of how strongly the systems converge/diverges light. Figure 4 shows the basic pinhole camera model. It is the inverse of the optical power. A positive focal length indicates that a system converges light, while a negative focal length indicates that the system diverges light. A system with shorted focal length bends the rays more sharply. In general, for a real camera, the focal length determines the magnification at which projects distinct objects. It is exactly the distance between the image plane and a pinhole that images distant objects that has the same size.

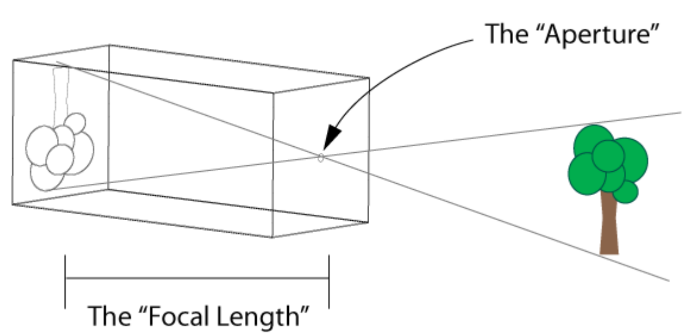


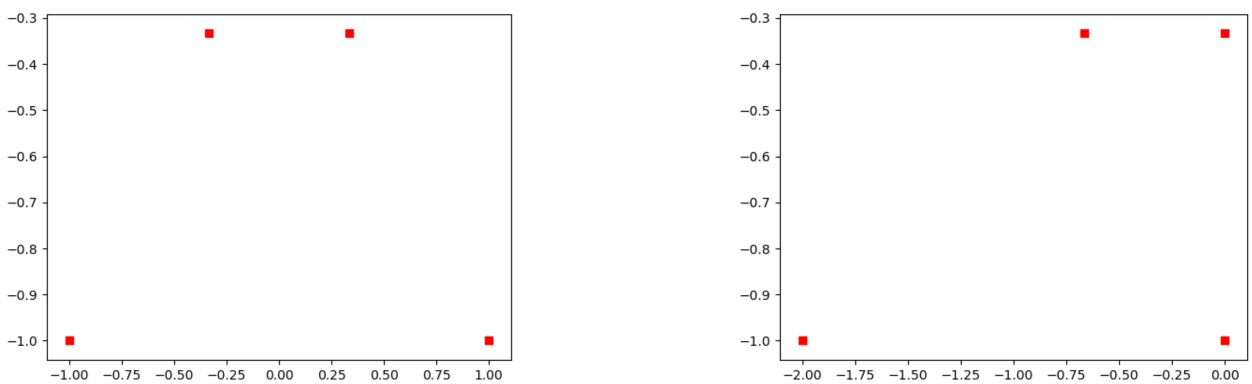
Figure 4. The pinhole camera model, focal length, and the aperture for the real camera model. (Best view in magnification and color.)

Answer (b)

In the last part, we have focused on the intrinsic parameters of the camera model, now we will work on extrinsic parameters such as rotational and translation transformations. Firstly, we will briefly see the translation part, the details are as follows.

$$\text{The translation matrix } T = \begin{bmatrix} I_{3x3} & t \\ 0^T & 1 \end{bmatrix}$$

In here $t = [-1, 0, 1]^T$ that translates the scene 1 unit to left. Using the same 3D points from part-1 the result will be shown in Figure 5.



(a) (b)

Figure 5.Scatter plot in (a) for 3D points x = ±1, y = -1, z = 2 ± 1, and w = 1, while after translation the scatter plot will looks like (b). (Best view in magnification and color.)

$$\text{The rotation matrix } R = \begin{bmatrix} \cos(t) & 0 & \sin(t) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(t) & 0 & \cos(t) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

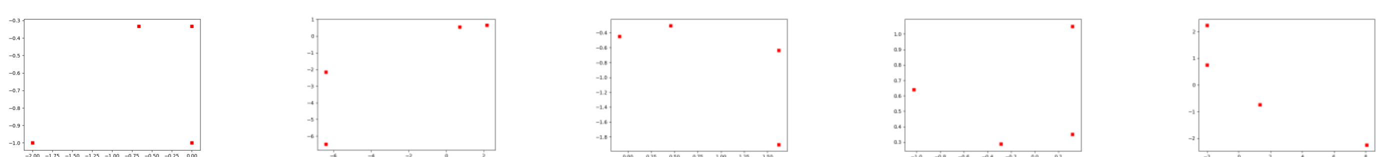


Figure 6. Transformation by M = TR in which the scene will be rotated about y-axis before it gets translated. From left to right the angles are progressively increasing from 0 to 90: 0, 30, 45, 60, 90. (Best view in magnification and color.)

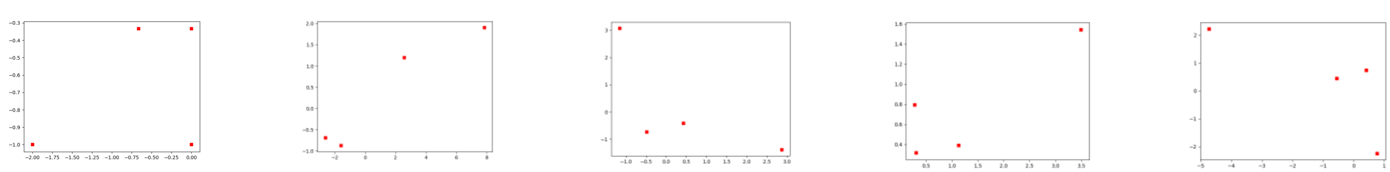


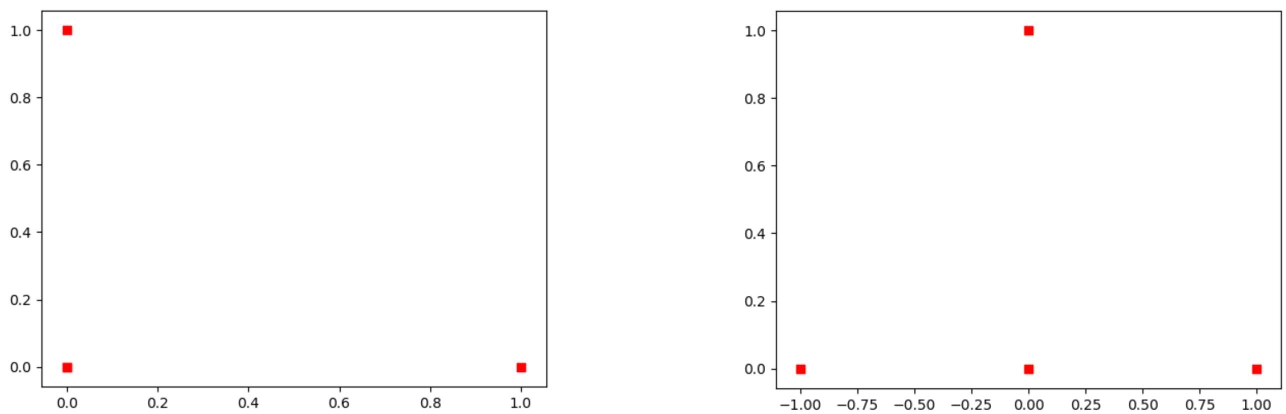
Figure 7. Transformation by M = RT in which this will transform the orbit from the camera around the origin. From left to right the angles are progressively increasing from 0 to 90: 0, 30, 45, 60, 90. (Best view in magnification and color.)

This rotation matrix is about rotating from x and z axes. We will first transform by M = TR in which first the scene will be rotated about y-axes before it is gets translated, in Figure 6, we will show the scatter plot with different angles. Then using M = RT, this will orbit the camera around the origin, and we will show the scatter plot in Figure 7 with different angles.

From Figure 6 and Figure 7 we can see that for the transformation of M = TR, the scene is being rotated about the y-axis before it gets translated. From 0 to 60 degrees we will be able to see that change, while for the transformation of M = RT, this will orbit the camera around the origin, and also this can be seen by the change of degree from 0 to 90. Also, for the translation transformation, we can shift the projection by x and y coordinates, as shown in Figure 5.

Task 2
Answer (a)

The task will focus on point cloud generation for given homogeneous arrays and reconstruction using OpenCV's traingulatePoints function. Also, we need to find the error rate when comparing to the original points and reconstructed points. Figure 8. shows the scatter plot for two different cameras placed at distance d. The focal length for the cameras is unit focal length.

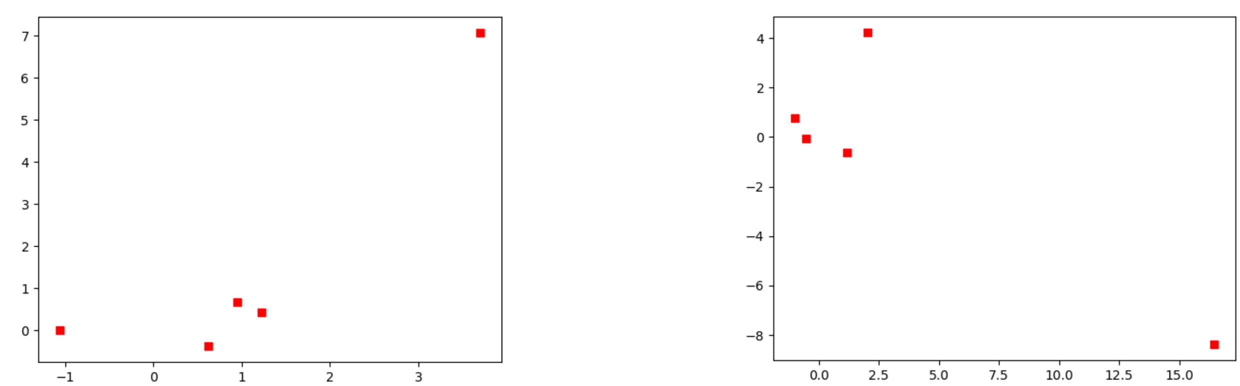


(a) (b)

Figure 8. Camera P1 and P2 projected on the plane with the distance of d which is a baseline distance from two cameras. Here the focal length for cameras is unit focal length. (Best view in magnification and color.)

Answer (b)

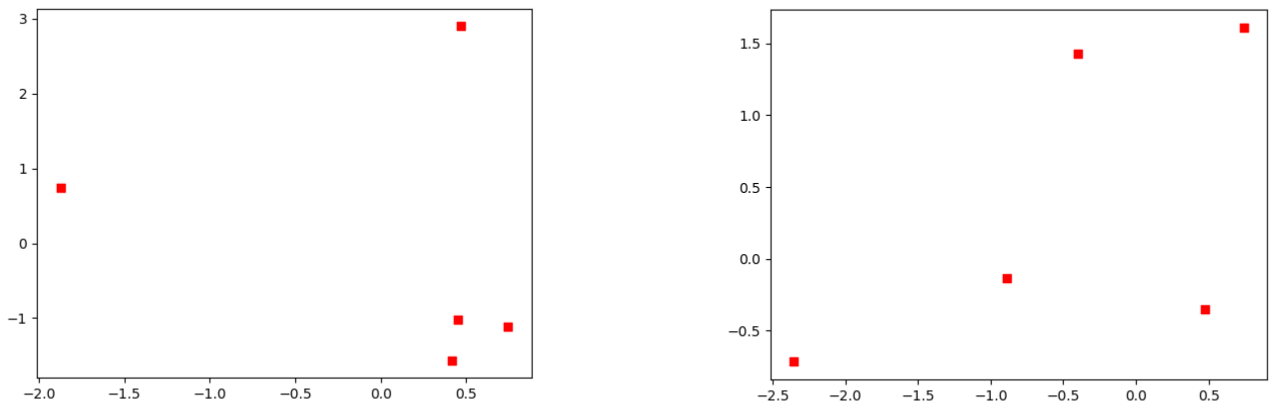
After that, we are projecting these 3D points into two different views and add some noise such as random Gaussian noise to see the effect. Figure 9. Shows these points which will be reconstructed after using traingulatePoints function in OpenCV.



(a) (b)

Figure 9. Projecting 3D points into two different views (a) and (b) and also by adding some amount of random Gaussian noise into point location. (Best view in magnification and color.)

Now we will use OpenCV's triangulation function to reconstruct the 3D points from the pair of image points and measuring the 3D distance between these points and the actual value. As per the experiments, the value comes out to be zero, as we are reversing the process of projection. Figure 10. shows the plots of 3D points reconstruction by using the triangulation function.



(a) (b)

Figure 10. Using OpenCV's triangulate function to reconstruct the 3D points that we have generated, in our case the number is 5 test points. While after reconstruction, we find the residual value which is zero, as we reverse the process of projection, but in our case, it will not because we have added the noise which is random Gaussian noise thus the (a) and (b) is different. (Best view in magnification and color.)

After getting reconstruction of the 3D points, we will tune certain parameters such as focal length, noise value, and distance value to check the error-rate. Figure 11. shows the change of focal length from 5, 15, 25, and 50.

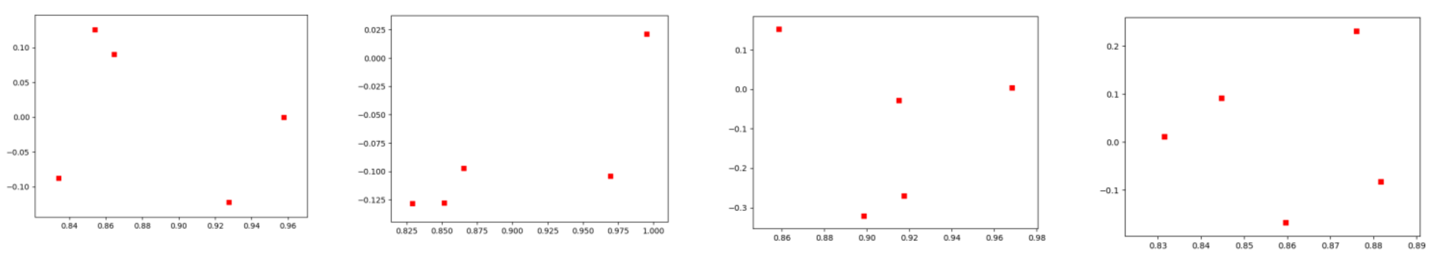


Figure 10. From left to right, we change the focal length from 5, 15, 25, and 50 to see how the 3D reconstruction works for the test points and check the error rate. (Best view in magnification and color.)

After working with the focal length, we will tune the parameter of random Gaussian noise to check the distortion in point locations and error rates. Figure 12. shows the change of noise level.

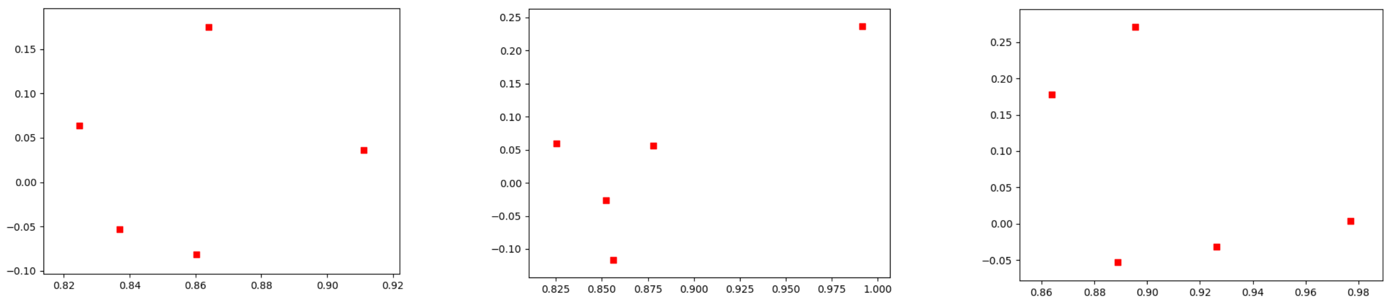
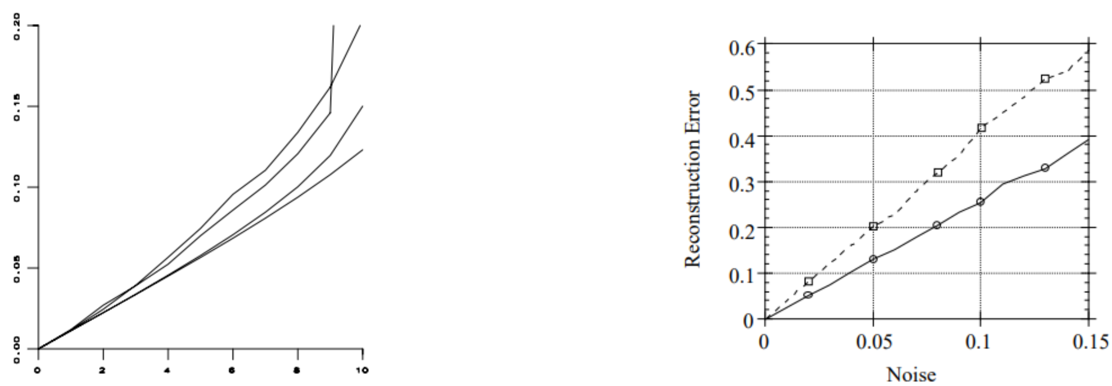


Figure 11. Change of noise level in the point locations, as we can see the graphs are looking weird as the random Gaussian noise changes significantly and so does the error rate. (Best view in magnification and color.)

Considering the random Gaussian noise is present, the two rays will not generally be met, in which case it is necessary to find the best point of intersection. This problem is critical in affine and projective reconstruction in which there is no meaningful metric information about the object space. It is indeed desirable to find a triangulation method that is invariant to projective transformations of space. It is common to assume that features in the images are subject to Gaussian noise which displaces the feature from its correct location in the image. Hereby we present graphs presenting the relationship between noise, focal length, and reconstruction error for reconstruction using the triangulation method in OpenCV. Figure 13 shows the graphs.



(a) (b)

Figure 13. Relationship between noise, focal length, and reconstruction error for the reconstruction of 3d points which we have used in our experiments. The left image shows the relationship between focal length and noise, while the right image shows the reconstruction and noise. (Best view in magnification and color.)

Computer Vision (7315)

Task 3

The task will focus on measuring the effect of varying the convergence angle on the reconstructed noisy point cloud projections. Here we change the angle progressively with a certain amount of translation to see the change in the plots. Figure 14. Shows the plots in which the angle changes progressively from 0, 30, 45, 60, and 90.

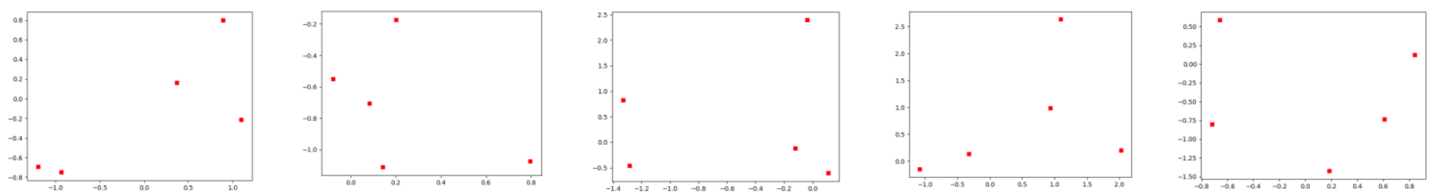


Figure 14. Change of convergence angle on the reconstructed noisy point cloud projections. From left to right, the converging angle has been changed to 0, 30, 45, 60, and 90. (Best view in magnification and color.)

Now for the comparison of error rate, the discussion is as follows: In our experiments, 5 points were chosen at random in the common field of view. For each of several noise levels, each point was reconstructed 50 times, with different instances of noise chosen from a random Gaussian variable with the given standard deviation (noise level). For each reconstructed point, both the 3D reconstruction error and the residual error were measured. The errors are the average errors. Median errors were also computed. To measure this the rotational and translation transformation was applied to each camera matrix. They were chosen so that one of the cameras matrices was of the form of (I | 0). This is a normalized form of a camera matrix used in reconstruction methods. Eventually, it represents significant distortion as the actual camera matrix was the form of (M | 0). In one of the cases, the points are at a distance of 0.15 units in front of the first camera and 0.55 units distance for the other case. The measured error is denoted as a 2D error or 3D error in distance. Figure 15 shows the graphs of the focal length vs error and noise vs reconstruction error.

Figure 15. Relationship between noise, focal length, and reconstruction error of 3d points after adding a convergence angle and after performing rotational and translation transformations. The left image shows the relationship between focal length and noise, while the right image shows the reconstruction and noise. (Best view in magnification and color.)

The goal of these experiments was to determine how the triangulation method affects the accuracy of reconstruction. Since it makes sense to measure the accuracy of reconstruction in a Euclidean frame where distance has a meaning, a close approximation to a correct Euclidean model for the object was estimated by eye and refined using the measured image locations of the corners of the dark squares. The Euclidean model so obtained was used as ground truth. We desired to measure how the accuracy of the reconstruction varies with noise. For this reason, the measured pixel locations were corrected to correspond exactly to the Euclidean model. At this stage, we had a model and a set of matched points corresponding exactly to the camera model. As we can see from the previous task the error and the graphs are somewhat different. The main reason behind the result are different is that the inclusion of extrinsic transformation and convergence angle and because of that the 3D reconstruction of points differs compared to this task.

Task 4

In this part, given a set of corresponding 2D points, we will estimate the fundamental matrix using OpenCV's recoverpose in-built method. Now that we know how to project a point from a 3D coordinate to a 2D coordinate, we will look at how to map corresponding 2D points from the same scene. You can think of the fundamental matrix is something that maps points from one view into a line in the other view. We get a line because a point in one image is only *a* projection to 2D, which means we cannot know the "depth" of that point. As such, from the viewpoint of the other camera, we can see the entire "line" that our first point could exist on. The fundamental matrix constraint between two points x_0 and x_1 in the left and right views, respectively, is given by the following equation:

$$x^T_0Fx_1=0$$

Hence to estimate FF we can minimize the line-to-point distance between the point x_0 and the line Fx_1, for all matching points (x_0,x_1). This makes estimating the fundamental matrix a least-squares problem. Now we have a function that can calculate the fundamental matrix F from matching pairs of points in two different sets of coordinates. However, having to manually extract the matching points is undesirable. Essential Matrix contains information about translation and rotation, which describe the location of the second camera relative to the first in global coordinates. But we prefer measurements to be done in pixel coordinates, right? Fundamental Matrix contains the same information as Essential Matrix in addition to the information about the intrinsic of both cameras so that we can relate the two cameras in pixel coordinates. In our case here, we use the point correspondence to recover the pose using the essential matrix. Figure 16 shows the orientation of the camera with the point cloud using OpenCV's build-in recover pose method.

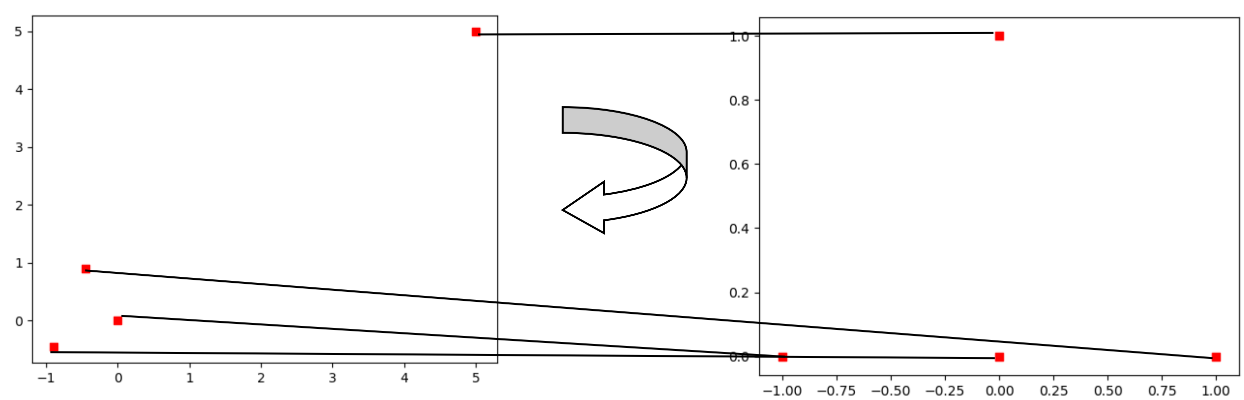


Figure 12. **The orientation of the camera with the point cloud** using OpenCV's built-in recoverpose method. (Best view in magnification and color.)

References:

[1] Anthony Dick, “Computer Vision Stereo Vision (7315)." School of Computer Science, The University of Adelaide, 2020.

[2] Bradski, Gary, and Adrian Kaehler. Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc.", 2008.

[3] Walt, Stéfan van der, S. Chris Colbert, and Gael Varoquaux. "The NumPy array: a structure for efficient numerical computation." Computing in Science & Engineering 13.2 (2011): 22-30.

[4] Zhang, Zhengyou. "Determining the epipolar geometry and its uncertainty: A review." *International journal of computer vision* 27.2 (1998): 161-195.

[5] Hartley, Richard I., and Peter Sturm. "Triangulation." *Computer vision and image understanding* 68.2 (1997): 146-157.