**Research on implementing an augmented reality viewer**

**Group member:**

**Abstract**

This project is about 3D reconstruction. First, COLMAP software is used to generate a 3D point cloud from a set of images. The original result contains some outliers which need being removed. The RANSAC-based plane fitting routine algorithm is utilized to reserve the effective points. Then some geometric manipulations are implemented to complete 3D to 2D camera projection. Besides, a virtual object (3D box) is created to put in the scene and displayed.

1. **COLMAP**

COLMAP should be the current state-of-art incremental SFM solution, which can easily reconstruct a series of two-dimensional pictures in three dimensions. No need to calibrate the camera, only need to shoot the reconstructed scene or object from different angles to get a series of images as input. Click the menu Reconstruction, select start Reconstruction to automatically rebuild, and get the sparse point cloud model. Our experimental picture collection is a set of photos of bottles. The set contains 108 images of different angles. After sparse reconstruction, three text files were produced: camera.txt, images.txt and points3D.txt. The cameras.txt describes the intrinsic parameters of camera, the pose and key points of all reconstructed images are stored in images.txt. The points3D.txt stores coordinates of 3D point cloud. The result of sparse reconstruction and dense reconstruction are shown as following.

|  |
| --- |
| A screenshot of a cell phone  Description automatically generated |
|  |
| **Figure 1. sparse and dense reconstruction results.** |

1. **RANSAC**

The basic assumptions of RANSAC are:

(1) The data consists of "internal points", for example: the distribution of data can be explained by some model parameters;

(2) "outside points" are data that cannot be adapted to the model; (3) other data belong to noise. The causes of outliers are extreme values of noise, wrong measurement methods and wrong assumptions about data.

RANSAC also makes assumptions that given a set of (usually small) intra-office points, there is a process that can estimate the model parameters; and the model can explain or apply to intra-office points. RANSAC is to look for data that appears in a certain model (in this project is a plane) under strong interference.

The flow of RANSAC is as follows:

1. Randomly select a plane in 3D space and consider that all points within the threshold distance from the plane are inlier points.
2. Use this model to test all points in the data set. If a certain point is suitable for this model, it is temporarily considered as an in-house point.
3. If enough points are classified as this model, the model is considered reasonable.
4. Re-evaluate the model with all the intra-office points currently included in the model (how? For example, through these intra-office points and the overall error of the model). Record this error.
5. Back to 1, iterate a fixed number of times and find the model corresponding to the smallest error.

The plane and points’ distance can be described as following.

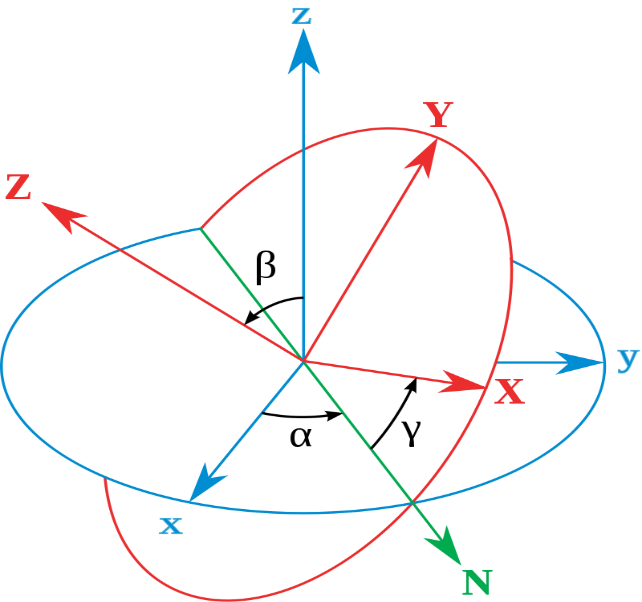


The results before and after RANSAC is shown in figure 2. The plane is generated in the plane\_fitting.py, after applying the RANSAC to the point cloud, we can get a plane which contains most of the points among the total 108770 points we have. We take 70% percent of these points valid and 30% of them segment out, which the number of valid points is around 70000. The red points are the original points and the blue surface is the plane we made after applying RANSAC.

|  |
| --- |
| A close up of a piece of paper  Description automatically generated |
| **Figure 2. The results before and after RANSAC** |

1. **Euclidean transformation**

For a reference system in three-dimensional space, the orientation of any coordinate system can be represented by three Euler angles. The reference system, also known as the laboratory reference system, is stationary. The coordinate system is fixed to the rigid body and rotates with the rotation of the rigid body. There are three parameters: α，β and γ. The following figure can describe them.

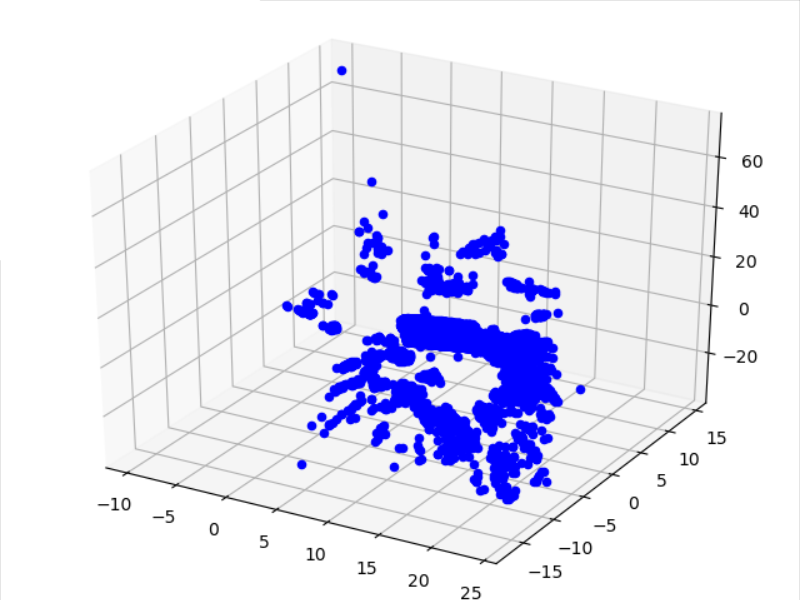


**Figure 3. three Euler angles.**

α is the angle between the x-axis and the line of intersection, β is the angle between the z-axis and the z-axis and γ is the angle between the intersection line and the X-axis. The transformation can be represented as following matrix.



In this project, the dominant plane should be transformed to z = 0 plane and the original system’s local origin lies roughly in the middle of the set of inlier points. Therefore, the system needs being firstly translated to make (0, 0, z) in the middle of the set. Then the rotation matrix is set as α =-60，β =10 and γ =0. The result after translation and rotation is shown as following figure.



**Figure 4.** result after transformation**.**

1. **Create a virtual object.**

A 3D box is created in this project. The center of box is set as (0, 0, 0). The length of box edges is 20. The process of creating box is: 1. Generate four edges of top four edges. 2. Generate four bottom edges. 3. Generate four edges for supporting edges. The result is shown in figure 5.

**Figure 5.** result after adding 3D box**.**

1. **Make projection from 3D to 2D**

In this project, the result points of previous steps can be taken from a set of 3D (X, Y, Z) points into a set of 2D(COL, ROW) pixel locations. This is a process of transforming world 3D coordinates to image coordinates. At first, the world coordinate system needs to be transformed to the camera coordinate system. The objects will not deform. The transformation formula can be represented as following.



Where X, Y and Z are coordinates in camera system and U, V and W are coordinates in world system. R is the rotation matrix and C is the translation matrix. The R and C in our project are calculated in quat2rot.py. This py file first turn the quaternion into rotation matrix, saved in a txt file. Then we read the txt file and apply the R, C matrix one-by-one to each of the world system. Therefore, we can get the camera system points.

Then the intrinsic parameters of camera are utilized to convert the camera coordinates to 2D image coordinates. The perspective projection matrix and film plane to pixels matrix can be represented as following.



Where f is the focal length. Sx and Sy are scale factors.  are skew parameter (essentially zero). Therefore, the whole transformation can be described as formula 1.5.



The intrinsic parameters can be gained from COLMAP. And the results of different projections is shown in figure 6. Notice that there should be 108 images output where I pick some of the images that can obviously observe the shape of tables or other bottles. The whole 108 images are contained in the file too, which can have a clearer look on how the camera moves.

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated |  |
| A screenshot of a cell phone  Description automatically generated | A screenshot of a cell phone  Description automatically generated |
| A screenshot of a cell phone  Description automatically generated | A screenshot of a cell phone  Description automatically generated |
|  | A screenshot of a cell phone  Description automatically generated |
| **Figure 6.** results of different projections | |