Lydia Qiao
ID: 45837573

Progress Update

Milestones

1. Calibrate cameras
2. Reconstruct the object after a color decode function that stores the object's RGB and produces a color mask to sort out the background.
3. Clean up the mesh
4. Generate meshes for different views
5. Aligned scan data and produce final mesh

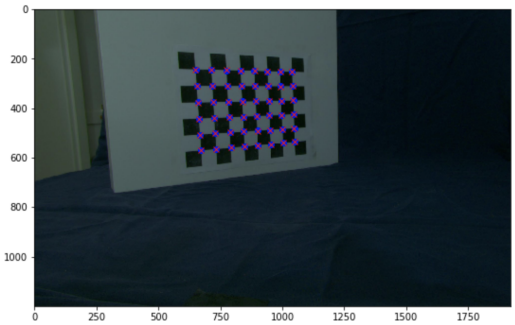
Milestones 1: Calibrate cameras (Completed)

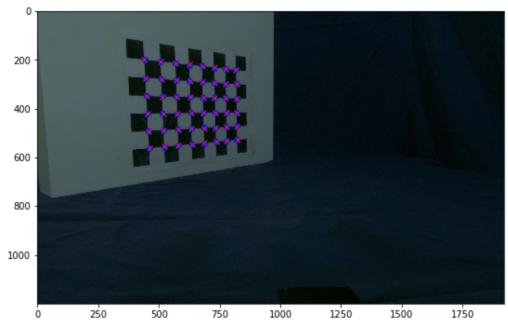
First, I import the calibrate.py to get the intrinsic parameter for two cameras by reading all the images in the calib_jpg_u folder and save the result to the calibration.pickle for future use. Then, I used the calibratePose function to calibrate my cameras.

The camera parameter I got is

```
Left Camera :
 f=1404.600966175714
 c=[[962.16736926 590.91595681]]
 R=[[ 0.04237413  0.99207514  0.11828508]
 [ 0.87061203 -0.09474566  0.48276076]
 [ 0.49014194  0.08252384 -0.86772732]]
 t = [[  8.15884708 -18.70825726  59.51710062]]
Right Camera :
 f=1404.600966175714
 c=[[962.16736926 590.91595681]]
 R=[[ 0.00217571  0.99082383  0.13514213]
 [ 0.65889089 -0.10307958  0.74514253]
 [ 0.75223538  0.0874227  -0.65306907]]
 t = [[  7.63972937 -28.46393402  51.4418431 ]]
```

Finally, I visualized the camera calibration to make sure those are correct.





Lydia Qiao
ID: 45837573

This milestone is easy to complete, but I was struggling about which picture is for Left and which is for Right camera. However, I realized that it was not important if I paired the parameter with the camera correctly.

Milestones 2: Reconstruct the object after a color decode function that stores the object's RGB and produces a color mask to sort out the background. (Completed)

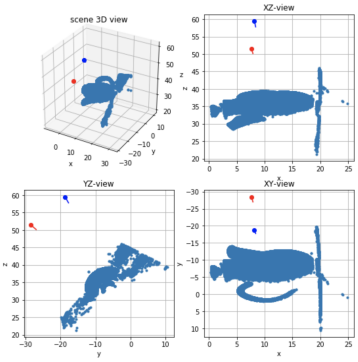
I wrote a function color_decode to return the RGB value for the picture and mask for both cameras to map out the background.

```
def color_decode(pts2L,pts2R,imprefix,start,threshold):
    mask_t=[]
    colorL = plt.imread(imprefix+"C0_00.png")
    colorFL= plt.imread(imprefix+"C0_01.png")
    colorR= plt.imread(imprefix+"C1_00.png")
    colorFR= plt.imread(imprefix+"C1_01.png")
    cL_list=[]
    cR_list=[]
    for i in range(pts2L.shape[1]):
        cL_list.append(colorFL[pts2L[1][i]][pts2L[0][i]])
        cR_list.append(colorFR[pts2R[1][i]][pts2R[0][i]])
    cL=np.array(cL_list).T
    cR=np.array(cR_list).T
    rbg=(cL+cR)/2
    colorL= colorL-colorFL
    colorR= colorR-colorFR
    mask_t= np.asarray(mask_t)
    mask_t=mask_t.astype(int)
    maskL= np.zeros((colorL.shape[0],colorL.shape[1]))
    maskR= np.zeros((colorR.shape[0],colorR.shape[1]))
    undecodableL=np.where(np.abs(colorL) >threshold )
    undecodableR=np.where(np.abs(colorR) >threshold )
    for i in range(undecodableL[1].size):
        maskL[undecodableL[0][i]][undecodableL[1][i]]=1
    for i in range(undecodableR[1].size):
        maskR[undecodableR[0][i]][undecodableR[1][i]]=1

    return rbg,maskL,maskR
```

Then used this function and reconstruct function to compute the points and color

```
imprefixC0 = 'teapot/grab_0_u/frame_C0_'
imprefixC1 = 'teapot/grab_0_u/frame_C1_'
threshold = 0.02

pts2L,pts2R,pts3 = reconstruct(imprefixC0,imprefixC1,threshold,camL,camR)
color,maskLc,maskRc = color_decode(pts2L,pts2R,'teapot/grab_0_u/color_',0,threshold)
```

The grab_0_u view result after reconstructing.

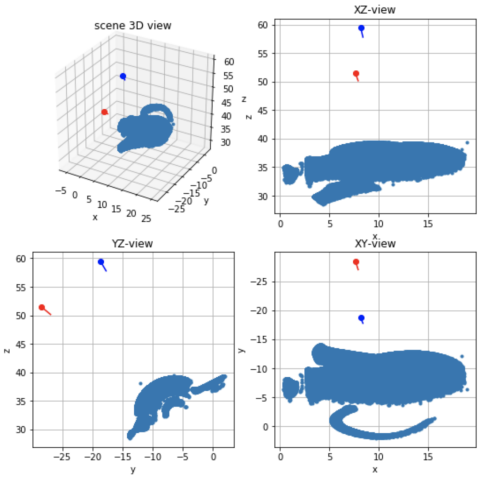


Lydia Qiao
ID: 45837573

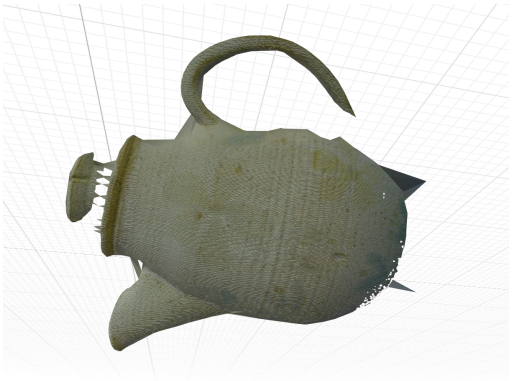
The challenge here is to figure out how to store the RGB values properly so that I can use it in writeply function in the future milestones.

Milestone 3: Clean up the mesh (Completed)

In this milestone, I did pruning which deletes the redundant point which is out of the box limit and also the point that is too far away to produce a triangle. The points seem more reasonable to produce the object.



Then I used writeply to create a mesh using those points and RGB value and stores it in grab0.py



The challenge here is to find the proper value for pruning. Also, the pruning step might need to be changed according to the future steps which stitches all the meshes together.

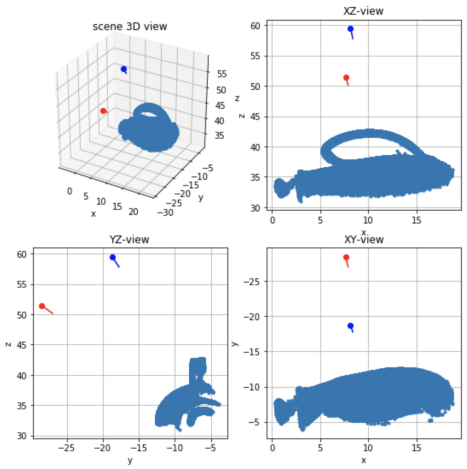
Milestone 4: Generate meshes for different views (In Progress)

This milestone generates all the meshes for every single view. Basically, repeating the previous steps. Each mesh is stored in its .ply file. The challenge here is to make sure each mesh is pruned correctly and has some points that overlap with other meshes.

I think the pruning method might need to be removed or changed to make the next milestone work. Just looking at the handle part, I don't think I can stitch them together and create a smooth object.

Lydia Qiao
ID: 45837573

```
imprefixC0 = 'teapot/grab_1_u/frame_C0_'
imprefixC1 = 'teapot/grab_1_u/frame_C1_'
threshold = 0.02

pts2L,pts2R,pts3 = reconstruct(imprefixC0,imprefixC1,threshold,camL,camR)
color,maskLc,maskRc = color_decode(pts2L,pts2R,'teapot/grab_1_u/color_',0,threshold)
boxlimits = np.array([0,19,-16,10,0,50])
trithresh = 2
pts2L,pts2R,pts3,color, tri = clean_up(boxlimits,trithresh,pts2L,pts2R,pts3,color)
writeply(pts3,color,tri,'grab1.ply')
vis_scene(camL,camR,pts3,looklength=2)
```



Milestone 5: Aligned scan data and produce final mesh (Not Started)

I am using meshlab to align the scan data. The problem that I think will face is the mesh will have a lack of overlapping points and will not correctly align with each other. I might need to change my plan from pruning for each mesh to align all the meshes then do the pruning.