Assignment 2 Report Computer Vision

Kapuriya Janak (MT22032)

Question 1

Part 1

Camera intrinsic parameters

Focal length = (Fx, Fy) = (653.33, 654.89)

Skew parameter = 0

Principal Points = (Cx, Cy) = (331.32, 239.79)

Part 2 Camera Extrinsic parameters

Rotation Matrix

```
rotation_matrix : 1
[[-0.00912114 -0.9953674  -0.09571069]
 [ 0.99699956 -0.0016947  -0.07738862]
 [ 0.07686791 -0.09612939  0.99239632]]

rotation_matrix : 2
[[ 0.03876285 -0.99451655 -0.09713017]
 [ 0.99733125  0.03248685  0.0653833 ]
 [-0.06186932 -0.0994054   0.99312172]]

rotation_matrix : 3
[[ 0.01405033 -0.98708427 -0.15958458]
 [ 0.99609413  0.02773157 -0.0838298 ]
 [ 0.08717261 -0.15778343  0.98361849]]

rotation_matrix : 4
[[ 0.03433278 -0.99157304 -0.1249166 ]
 [ 0.99487183  0.04580723 -0.09017619]
 [ 0.09513836 -0.12118001  0.98806078]]

rotation_matrix : 5
[[ 0.00312775 -0.99983821  0.01771361]
 [ 0.99994076  0.00331177  0.01036888]
 [-0.01042586  0.01768013  0.99978933]]
```

```
rotation_matrix : 6
[[-0.56444154 -0.81686554 -0.11889672]
 [ 0.81908773 -0.57211581  0.04217567]
 [-0.10247455 -0.07358115  0.99201047]]

rotation_matrix : 7
[[ 0.85194682 -0.51451473  0.09726874]
 [ 0.52361937  0.83601886 -0.1639976 ]
 [ 0.00306068  0.19064904  0.98165349]]

rotation_matrix : 8
[[ 0.69854625 -0.05321062  0.71358375]
 [-0.07664531  0.98593035  0.14854915]
 [-0.71144827 -0.1584613   0.6846396 ]]

rotation_matrix : 9
[[-0.21694433 -0.96771477 -0.12830929]
 [ 0.91674044 -0.15679999 -0.36742447]
 [ 0.33544319 -0.19733697  0.92116013]]

rotation_matrix : 10
[[ 0.85682134 -0.40605801 -0.3177642 ]
 [ 0.29961367  0.89365475 -0.33408508]
 [ 0.41962941  0.19104473  0.88736299]]
```

```
rotation_matrix : 11
[[-0.38517468 -0.14641666 -0.91115456]
 [-0.04320919 -0.9833894   0.17629026]
 [-0.92183157  0.10727279  0.37245015]]

rotation_matrix : 12
[[-0.00610283 -0.99946397  0.03216424]
 [ 0.62772721 -0.02886688 -0.77789797]
 [ 0.77840947  0.01544298  0.6275669 ]]

rotation_matrix : 13
[[ 0.9611106   0.03916628 -0.2733723 ]
 [-0.22729964  0.67440118 -0.70250831]
 [ 0.15684796  0.73732561  0.6570766 ]]

rotation_matrix : 14
[[ 0.7717812  -0.63571388 -0.0148878 ]
 [ 0.62139267  0.74901066  0.22990037]
 [-0.13499973 -0.18668396  0.97310029]]

rotation_matrix : 15
[[ 0.76827206  0.48473277  0.41808155]
 [-0.31195493  0.8538384  -0.41670625]
 [-0.55896526  0.18972117  0.80719497]]
```

```
rotation_matrix : 16
[[-0.68809252  0.69398306 -0.21193441]
 [-0.69316014 -0.71503105 -0.09088246]
 [-0.21461057  0.08436895  0.97304891]]

rotation_matrix : 17
[[-0.7310111   0.59166926 -0.33992684]
 [-0.56794584 -0.80369288 -0.17752543]
 [-0.37823313  0.06328698  0.92354451]]

rotation_matrix : 18
[[-0.94890571 -0.31235685 -0.04484579]
 [ 0.30400809 -0.86679057 -0.39528876]
 [ 0.08459925 -0.38872524  0.91746153]]

rotation_matrix : 19
[[-0.09324682  0.99342779  0.06637956]
 [-0.8251877  -0.0398059  -0.5634543 ]
 [-0.55710886 -0.10731592  0.82347617]]

rotation_matrix : 20
[[ 0.00632507  0.99867939 -0.05098497]
 [-0.57117223 -0.03824236 -0.8199389 ]
 [-0.82080587  0.03430737  0.57017605]]
```

```
rotation_matrix : 21
[[-0.18503197 -0.98211514 -0.03482854]
 [ 0.97895131 -0.18109829 -0.09411561]
 [ 0.08612498 -0.05150984  0.99495187]]

rotation_matrix : 22
[[-0.39719707 -0.91773146  0.00185741]
 [ 0.91167929 -0.39434397  0.11547162]
 [-0.10523948  0.04755835  0.99330904]]

rotation_matrix : 23
[[-2.06597251e-02 -9.98865239e-01 -4.29116610e-02]
 [ 8.99284944e-01  1.89215668e-04 -4.37363183e-01]
 [ 4.36875000e-01 -4.76256138e-02  8.98260561e-01]]

rotation_matrix : 24
[[ 0.34632904 -0.92203939  0.17291489]
 [ 0.85454684  0.23402517 -0.46366142]
 [ 0.38704766  0.30834329  0.86897556]]

rotation_matrix : 25
[[ 0.23083924 -0.96332763 -0.13679594]
 [ 0.96094916  0.20366615  0.18734141]
 [-0.15261045 -0.1746997   0.97272302]]
```

Translation Vectors

```
translation vectors : 1
[[-5.90885981]
 [-3.24757797]
 [30.68418193]]

translation vectors : 2
[[ -5.36753366]
 [-10.18136487]
 [ 29.42343327]]

translation vectors : 3
[[  3.4309006 ]
 [-10.41957649]
 [ 32.28180187]]

translation vectors : 4
[[14.58763544]
 [-2.98534377]
 [33.66533595]]

translation vectors : 5
[[ 3.85690023]
 [ 1.17179767]
 [22.36660359]]

translation vectors : 6
[[ 2.37249433]
 [-1.21889651]
 [33.16738916]]

translation vectors : 7
[[ 3.24698236]
 [-6.85174162]
 [32.21357804]]

translation vectors : 8
[[ 1.4595038 ]
 [-2.97576975]
 [35.4643796 ]]

translation vectors : 9
[[ 2.67844916]
 [ 4.14088674]
 [29.02576096]]

translation vectors : 10
[[ 9.27197123]
 [ 0.10790289]
 [31.24956136]]

translation vectors : 11
[[-5.8992582 ]
 [ 2.84500834]
 [32.10582723]]

translation vectors : 12
[[ 4.28295203]
 [-5.15686681]
 [30.44308829]]

translation vectors : 13
[[ 4.122986  ]
 [-3.83860282]
 [28.83730512]]

translation vectors : 14
[[-7.53672233]
 [-0.64868673]
 [32.77529258]]

translation vectors : 15
[[-2.3913286 ]
 [-7.98588288]
 [37.70083724]]

translation vectors : 16
[[-4.82898123]
 [ 9.70771616]
 [30.47804247]]

translation vectors : 18
[[12.50356439]
 [ 4.06270264]
 [34.76717856]]

translation vectors : 19
[[-4.66737626]
 [-7.41500853]
 [45.14196052]]

translation vectors : 20
[[-5.54542629]
 [-9.42151888]
 [48.63370675]]

translation vectors : 21
[[ 4.8491594 ]
 [-0.52945575]
 [21.67976264]]

translation vectors : 22
[[ 5.08000463]
 [-1.1068221 ]
 [19.22800507]]

translation vectors : 23
[[ 4.43199299]
 [-2.98088035]
 [21.48137583]]

translation vectors : 24
[[ 5.3953505 ]
 [-7.80528251]
 [27.11092257]]

translation vectors : 25
[[ 5.2309205 ]
 [-3.89576839]
 [26.34840104]]
```

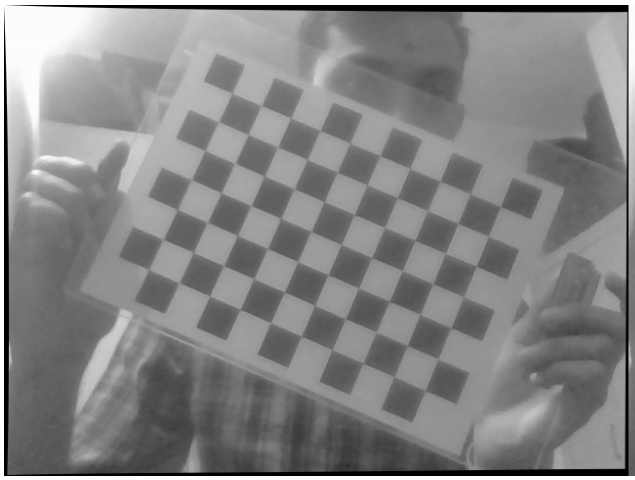
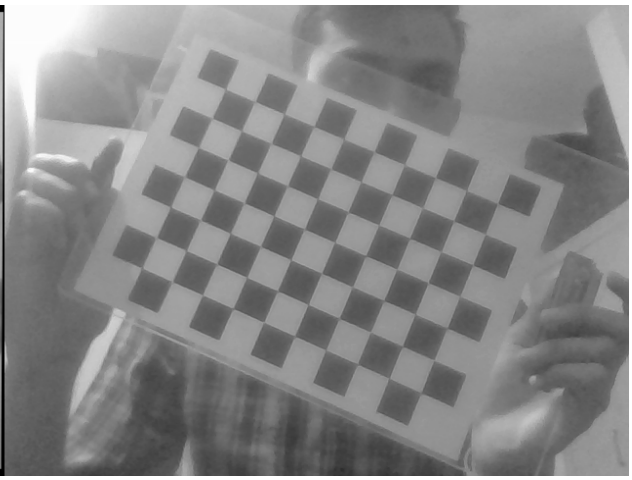
Part 3

Distortion coefficients: [0.02662453 -0.05833975 0.00582998 0.01269458, -0.07144689]

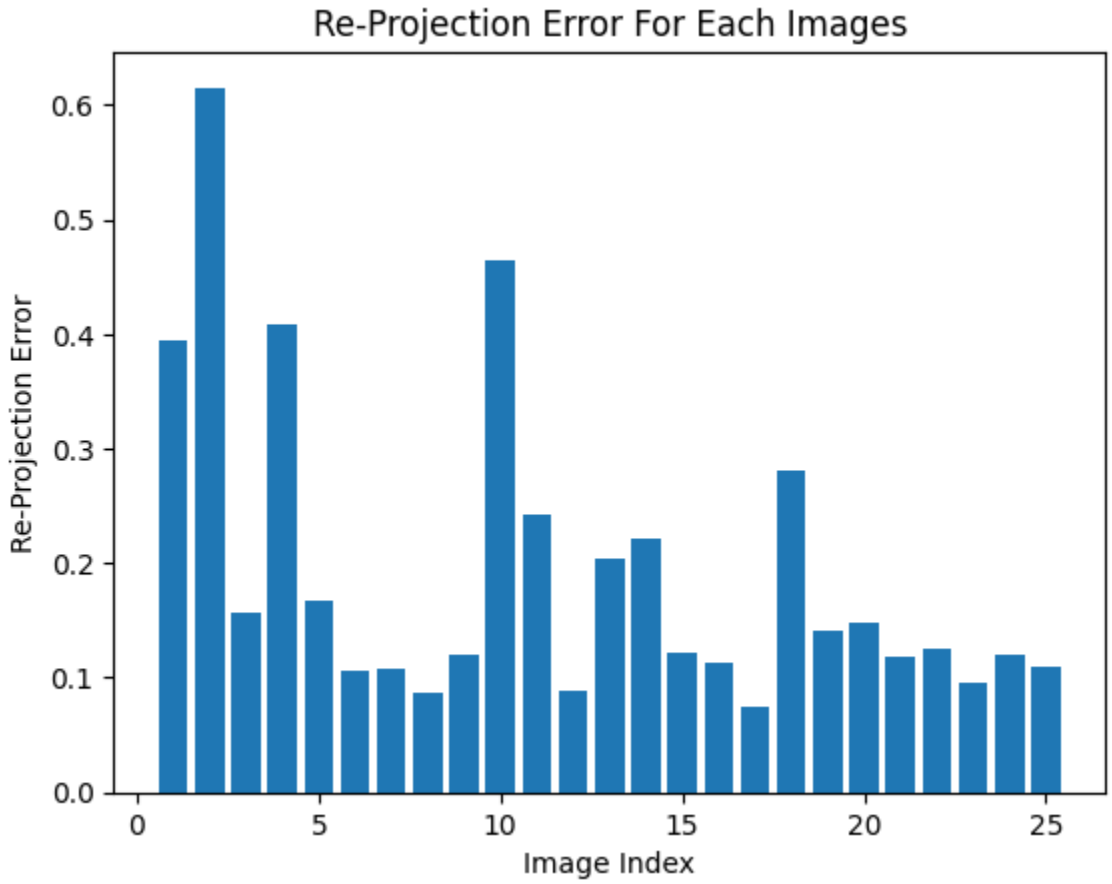
Distorted VS Undistorted Images

As we saw in the above and below images, the left images are distorted and can be seen on straight lines at image boundaries will be distorted where black pixels signify the radial distortion. After removing the radial distortion, image boundaries become straight.



Part 4 Reprojection Error

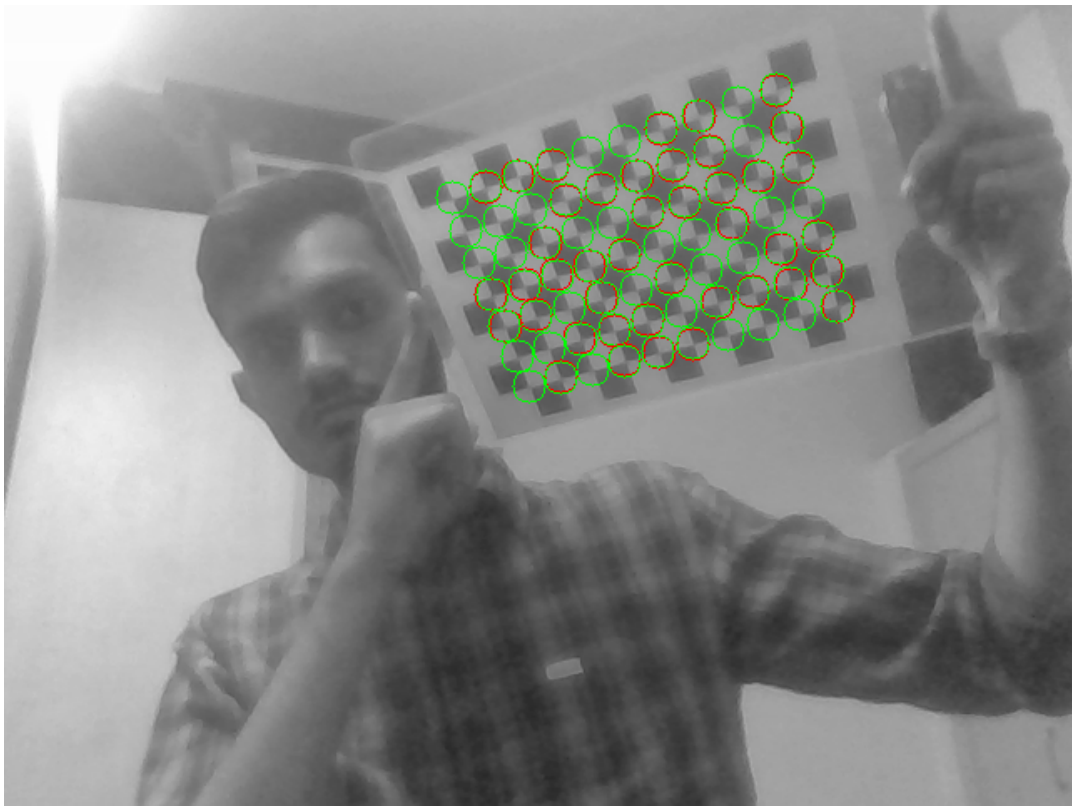


The mean of Reprojection errors = 0.193252

The standard deviation of Reprojection errors = 0.135173

Part 5 Example of error after Reprojection of corners

Reprojection error is computed as the absolute square difference between the original point and the points that are projected using translation and rotation matrices. Less the error, the better the estimation of camera parameters and vice versa.

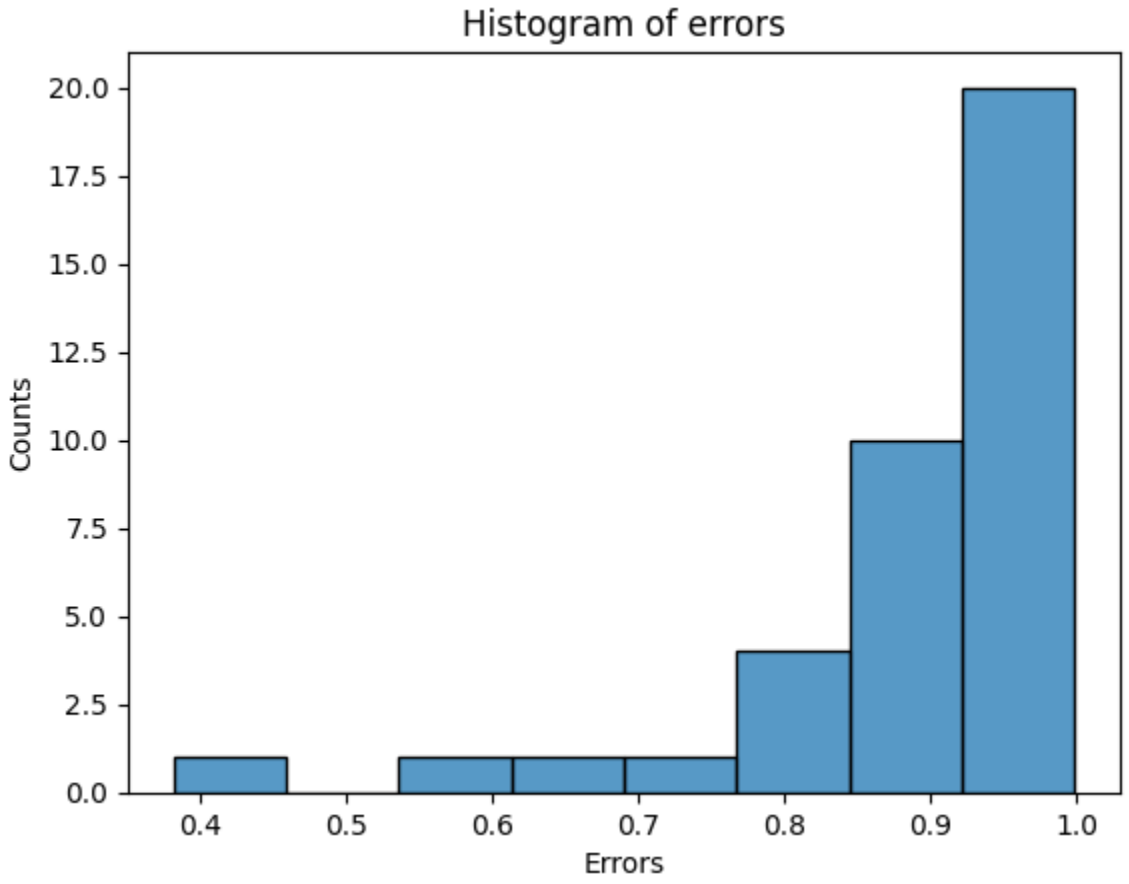


Question 2

Part 4

As we can see, all points are not projected exactly onto the checkerboard after the lidar-to-camera transformation. Projected points in some of the images are not projected on the checkerboard because of reprojection errors.

Part 5



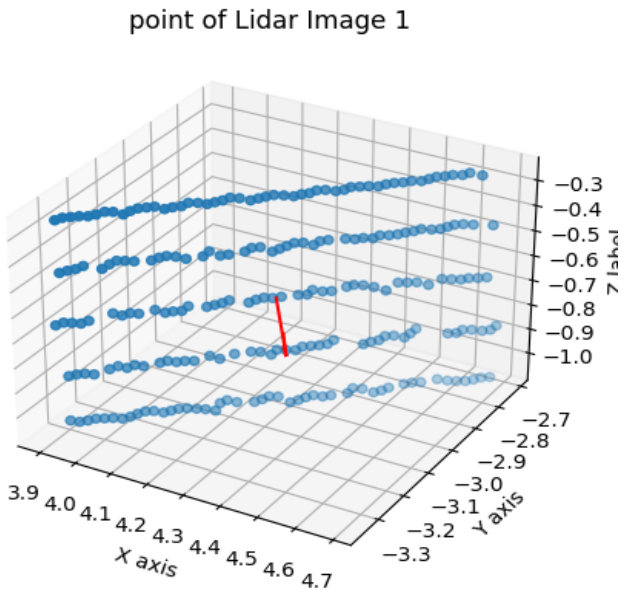
More images have significant errors because after optimization, we got Rotation and translation vectors that are not optimal.

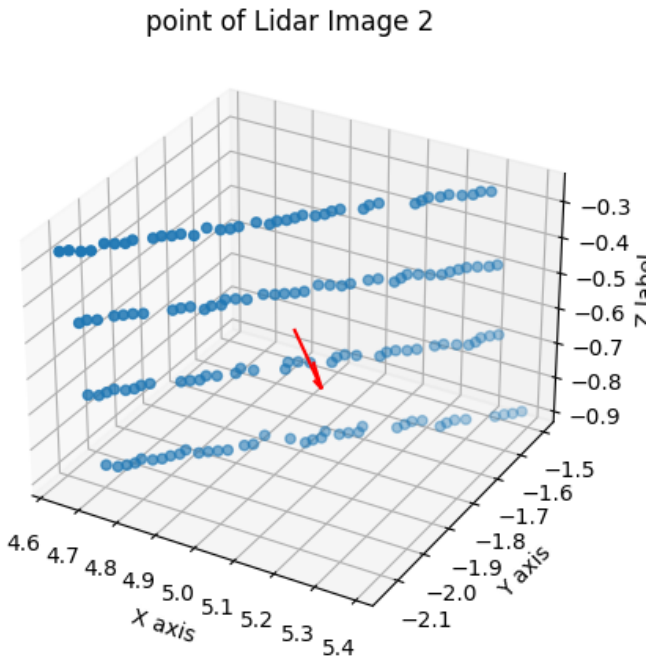
Mean of the errors = 0.895

The standard deviation of the errors = 0.129

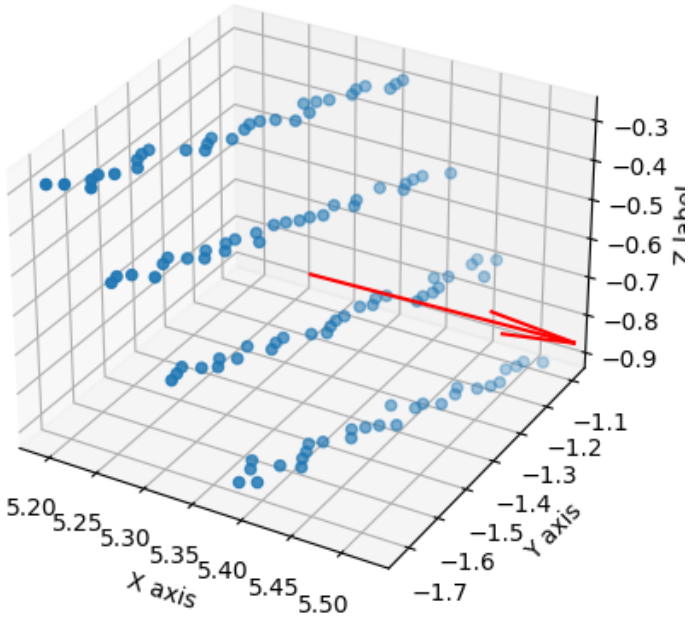
Plotting of camera and lidar normals on the checkerboard

Lidar normals on Lidar Image

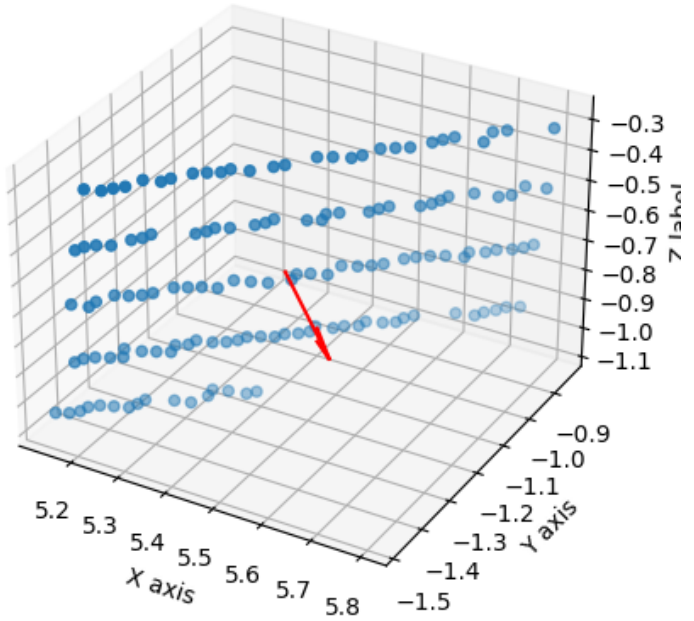


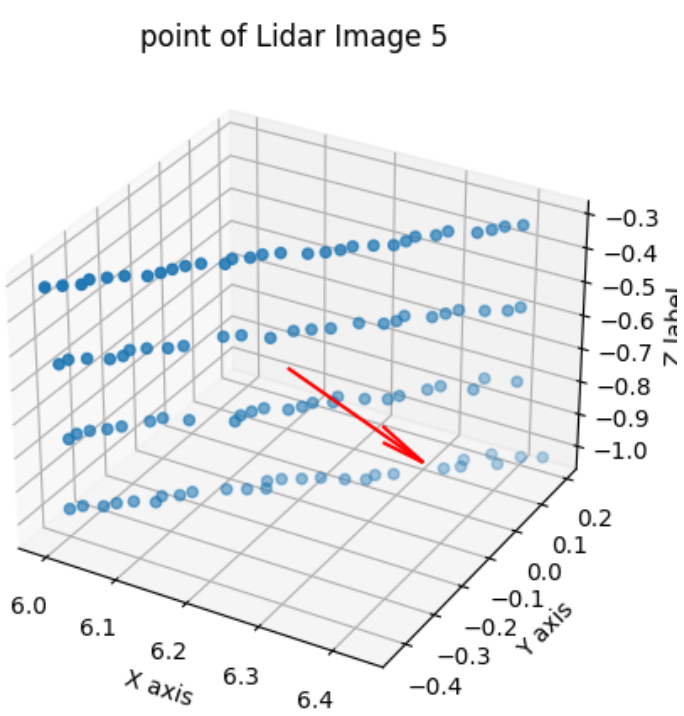


point of Lidar Image 3

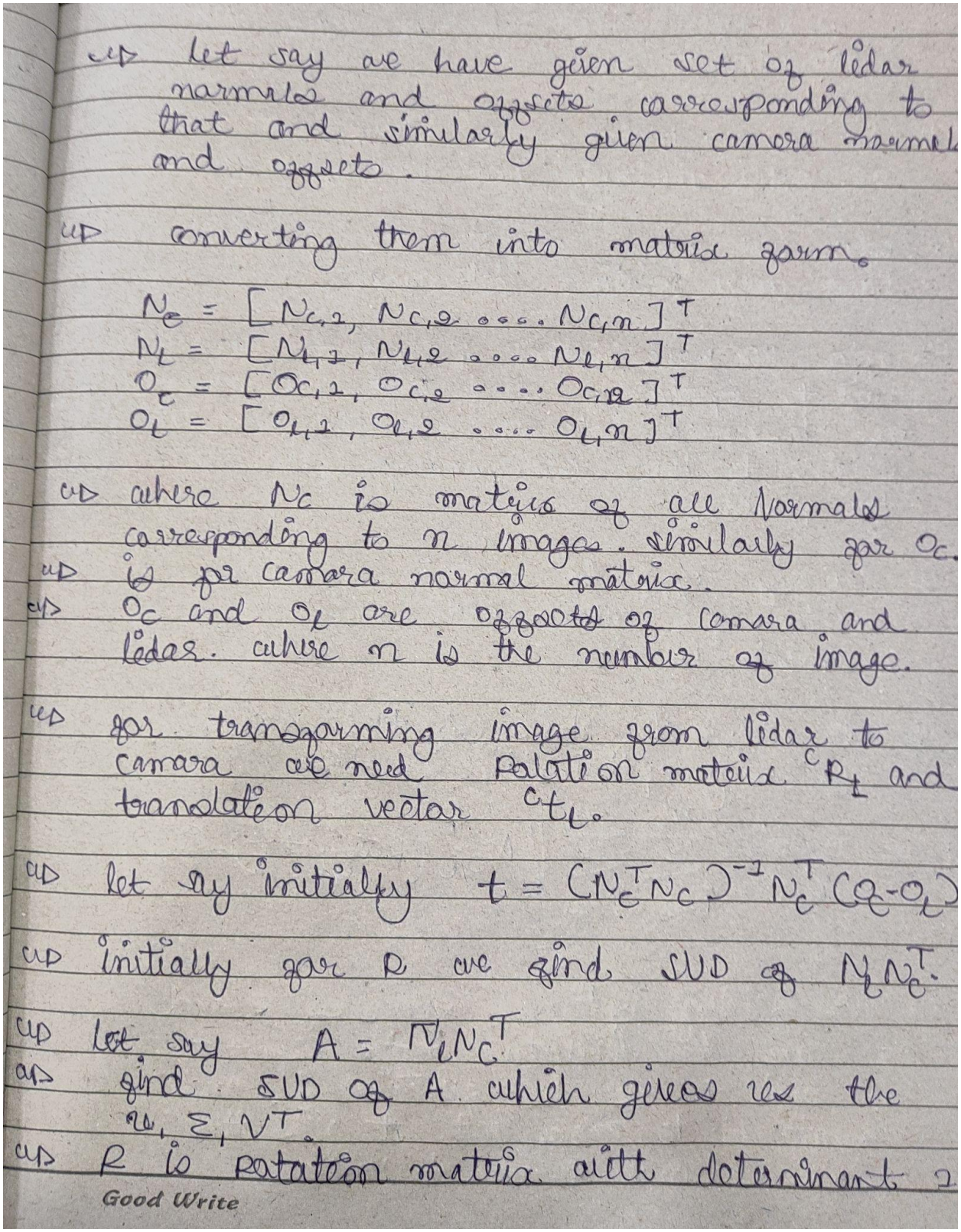


point of Lidar Image 4




point of Lidar Image 5
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
−0.9
−1.0
Z label
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.4
Y axis
6.0
6.1
6.2
6.3
6.4
X axis

Question 2.2 derivation



- Let say we have given set of lidar normals and objects corresponding to that and similarly given camera normals and objects.
- converting them into matrix form.

$$N_c = [N_{c,1}, N_{c,2} \ldots N_{c,n}]^T$$
$$N_l = [N_{l,1}, N_{l,2} \ldots N_{l,n}]^T$$
$$O_c = [O_{c,1}, O_{c,2} \ldots O_{c,n}]^T$$
$$O_l = [O_{l,1}, O_{l,2} \ldots O_{l,n}]^T$$

- where N_c is matrix of all Normals corresponding to n images. similarly for O_c.
- is for camera normal matrix.
- O_c and O_l are objects of camera and lidar. where n is the number of image.
- for transforming image from lidar to camera we need Rotation matrix ${}^cR_l$ and translation vector ${}^ct_l$.
- let say initially $t = (N_c^T N_c)^{-1} N_c^T (O_c - O_l)$
- initially for R we find SVD of $N_l N_c^T$.
- let say $A = N_l N_c^T$
- find SVD of A which gives us the U, Σ, V^T.
- R is rotation matrix with determinant 1

Good Write

- $R = VU^T$ which can not have the determinant +1.
- to ensure that R has determinant +1 we will multiply last row of V by -1.
- after that again compute the R which have now positive determinant.
- after we have initial R and t we need to optimize such that it minimize the difference between normal from the origin to the corresponding planes in two frames.
- let say $x_{i,i}$ be matrix of lidar points per which each image have i points in it.
- to achieve the above objective we use following optimization function.

$$L = \underset{R,t}{\operatorname{argmin}} \sum_{i=1}^{n} \sum_{j=1}^{m} \| N_{c,i}^{T} (R x_{i,i}^{(j)} + t) - O_{c,i} \|$$

- for finding optimal values of R and t we do derivative with respect to R and t of loss function L and equate it to zero. and it gives the final optimal R and t values.

Good Write

Optimization on this function is with constraint such that the restaurant rotation matrix must have a determinant equal to 1.

We will use R and t to transform the point in the lidar frame of reference to the camera frame of reference.

Scipy will do the constraint optimization as given above.