CSE344: Computer Vision Assignment 2 - Report

Name: Prakhar Gupta
Roll No: 2021550

1.

1. (a)

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\theta = \pi/2$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$\theta = -\pi/2$

$$\vec{t} = \begin{bmatrix} -1 \\ 3 \\ 2 \end{bmatrix} \qquad T = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_{overall} = R_x(\theta_1) R_y(\theta_2) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = T \cdot R_x(\theta_1) \cdot R_y(\theta_2)$$

$$= \begin{bmatrix} 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 3 \\ 0 & -1 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(b) Point (2,5,1):

$$\begin{bmatrix} 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 3 \\ 0 & -1 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 5 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ -3 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 0/1 \\ 1/1 \\ -3/1 \end{bmatrix} = \underbrace{(0,1,-3)}_{\text{new coordinate}}$$

Origin:

$$\begin{bmatrix} 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 3 \\ 0 & -1 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} -1 \\ 3 \\ 2 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} -1 \\ 3 \\ 2 \end{bmatrix} = \underbrace{(-1,3,2)}_{\text{new coordinate of origin}}$$

(c)

We can get the axis of combined rotation in the original frame of reference by finding eigenvectors →

$$\begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} v = \lambda v$$

$$\Rightarrow \begin{vmatrix} -\lambda & 0 & 1 \\ -1 & -\lambda & 0 \\ 0 & -1 & -\lambda \end{vmatrix} = 0 \Rightarrow 1 - \lambda^3 = 0$$

$$\lambda^3 = 1$$

$$\boxed{\lambda = 1}$$

$$\begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} v = v \quad \& \quad \bar{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

$$\begin{bmatrix} v_3 \\ -v_1 \\ -v_2 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

$$\Rightarrow \quad \underline{\underline{v_1 = 1}}, \quad \underline{\underline{v_3 = 1}}, \quad \underline{\underline{v_2 = -1}}$$

So, $\bar{v} = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} \Rightarrow \hat{v} = \begin{bmatrix} 1/\sqrt{3} \\ -1/\sqrt{3} \\ 1/\sqrt{3} \end{bmatrix} \longrightarrow$ axis of rotation

<u>Angle of Rotation</u> :

$$\theta = \cos^{-1}\left(\frac{\text{trace}(R) - 1}{2}\right) \Rightarrow \cos^{-1}\left(\frac{0-1}{2}\right) = \cos^{-1}\left(\frac{-1}{2}\right)$$

$$\boxed{\theta = 2\pi/3} \qquad \Rightarrow 2\pi/3$$

(d) By Rodrigues formula,

$$R = I + (\sin\theta) N + (1 - \cos\theta) N^2$$

$$n = \frac{1}{2\sin\theta}\begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \Rightarrow \frac{1}{\cancel{2}(\sqrt{3}/\cancel{2})}\begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -1/\sqrt{3} \\ 1/\sqrt{3} \\ -1/\sqrt{3} \end{bmatrix}$$

$$N = \begin{bmatrix} 0 & 1/\sqrt{3} & 1/\sqrt{3} \\ -1/\sqrt{3} & 0 & 1/\sqrt{3} \\ -1/\sqrt{3} & -1/\sqrt{3} & 0 \end{bmatrix} = \frac{1}{\sqrt{3}}\begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}$$

Now,

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \frac{\sqrt{3}}{2} \cdot \frac{1}{\sqrt{3}} \cdot \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix} +$$

$$\frac{3}{2} \cdot \frac{1}{3} \cdot \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$$

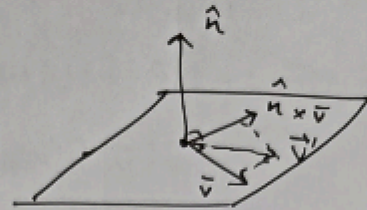
$$\Rightarrow R = \begin{bmatrix} 1 & 1/2 & 1/2 \\ -1/2 & 1 & 1/2 \\ -1/2 & -1/2 & 1 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} -2 & -1 & 1 \\ -1 & -2 & -1 \\ 1 & -1 & -2 \end{bmatrix}$$

$$R = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

R is the same as it was after the product of the rotation matrices.

2.

2. Consider a plane with normal vector $\hat{n}$ and a vec. lying in the plane $\bar{v}$ such that $\bar{v}$ lies completely in the plane, and hence $\hat{n} \perp \bar{v}$.



Here, $\hat{n} \times \bar{v}$ will also lie in the plane of the normal vector $\hat{n}$, where $\bar{v}$ lies completely in plane.

Let the rotated vector be $\bar{v}'$.

$\bar{v}'$ is divided into two components $\rightarrow (\cos\theta)\bar{v}$ in the direct^n of $\bar{v}$, and $(\sin\theta)(\hat{n} \times \bar{v})$ in the direct^n of $\hat{n} \times \bar{v}$.

So, the rotated vector is $\rightarrow$

$$\boxed{\vec{v}' = (\cos\theta)\bar{v} + (\sin\theta)(\hat{n} \times \bar{v})}$$

$\rightarrow$ when $\hat{n} \perp \bar{v}$.

Now, for a general vector $\bar{v}$, where $\bar{v} \not\perp \hat{n}$, we divide $\bar{v}$ into $\bar{v}_{\parallel}$ & $\bar{v}_{\perp}$.

$\Rightarrow \bar{v}_{\parallel} = (\hat{n} \cdot \bar{v})\hat{n}$, $\rightarrow$ in the direct^n of $\hat{n}$.

and $\bar{v} = \bar{v}_{\parallel} + \bar{v}_{\perp}$

So, $\boxed{\bar{v}_{\perp} = \bar{v} - \bar{v}_{\parallel}}$

for the vector $\bar{v}_{\parallel}$, rotat^n by the axis of $\hat{n}$ will have no effect as they are parallel.

However, since $\hat{n} \perp \bar{v}_{\perp}$, it will be rotated given by $\rightarrow$

$$\bar{v}_{\perp}' = (\cos\theta)\bar{v}_{\perp} + (\sin\theta)(\hat{n} \times \bar{v}_{\perp})$$

& the rotated vector is $\bar{v}' = \bar{v}_\perp' + \bar{v}_\parallel'$

Here, $\bar{v}_\parallel' = \bar{v}_\parallel \Rightarrow (\hat{n}.\bar{v})\,\hat{n}$

$$\bar{v}_\perp' = (\cos\theta)(\bar{v} - \bar{v}_\parallel) + (\sin\theta)\left(\hat{n} \times (\bar{v} - \bar{v}_\parallel)\right)$$

$$\bar{v}_\perp' = (\cos\theta)\,\bar{v} - (\cos\theta)(\hat{n}.\bar{v})\hat{n} + (\sin\theta)(\hat{n} \times \bar{v}) - \cancelto{0}{(\sin\theta)(\hat{n} \times \bar{v}_\parallel)} \quad (1)$$

$$\bar{v}_\parallel' = (\hat{n}.\bar{v})\,\hat{n}$$

Now, $\bar{v}' = \bar{v}_\perp' + \bar{v}_\parallel'$

$$\Rightarrow \bar{v}' = (\cos\theta)\,\bar{v} - (\cos\theta)(\bar{v}.\hat{n})\hat{n} + (\sin\theta)(\hat{n} \times \bar{v}) \cancel{- \sin\theta(\hat{n} \times (\hat{n} \times \bar{v}} + (\hat{n}.\bar{v})\,\hat{n}$$

$$\Rightarrow \bar{v}' = (\cos\theta)\,\bar{v} - (\cos\theta)(\hat{n}.\bar{v})\hat{n} + (\sin\theta)(\hat{n} \times \bar{v}) + (\hat{n}.\bar{v})\hat{n}$$

$$\Rightarrow \boxed{\bar{v}' = (1 - \cos\theta)(\hat{n}.\bar{v})\,\hat{n} + (\sin\theta)(\hat{n} \times \bar{v}) + (\cos\theta)(\bar{v})}$$

Hence proved.

3.

3 For cameras C_1 & C_2, & intrinsic parameters K_1 & K_2, the image pts are x_1 & x_2.

Second frame = R. (First frame)

Since x_1 is image pt,

x_1 in camera C_1 frame → $K_1^{-1} x_1$.

Now, ~~x_1 in~~ the pt in camera C_2's frame

$\hookrightarrow \; R.(K_1^{-1} x_1) \Rightarrow R K_1^{-1} x_1$

Now, transforming the pt from camera C_2's frame to image pt x_2 → $x_2 = K_2 (R K_1^{-1} x_1)$

So, $x_2 = K_2 R K_1^{-1} x_1$

$$\Rightarrow x_1 = (K_2 R K_1^{-1})^{-1} x_2$$

$$\Rightarrow x_1 = K_1 R^T K_2^{-1} x_2$$

$$\boxed{x_1 = K_1 R^T K_2^{-1} x_2}$$

So, $\boxed{H = K_1 R^T K_2^{-1}}$

4.

(1) Camera intrinsic parameters:

Camera Intrinsic Matrix:

```
[485.40629819   0.         262.5051723 ]
[  0.         487.04652074 262.54720001]
[  0.           0.           1.        ]
```

Focal Length (fx, fy): (485.406298185692, 487.0465207409381)
Principal Point (cx, cy): (262.5051722983334, 262.54720001202196)
Skew Coefficient: 0.0
Distortion Coefficients: [2.95175621e-01 -2.30374358e+00 1.72739831e-03
3.41344935e-03
5.58612187e+00]
Mean Reprojection Error: 1.0201390429906425

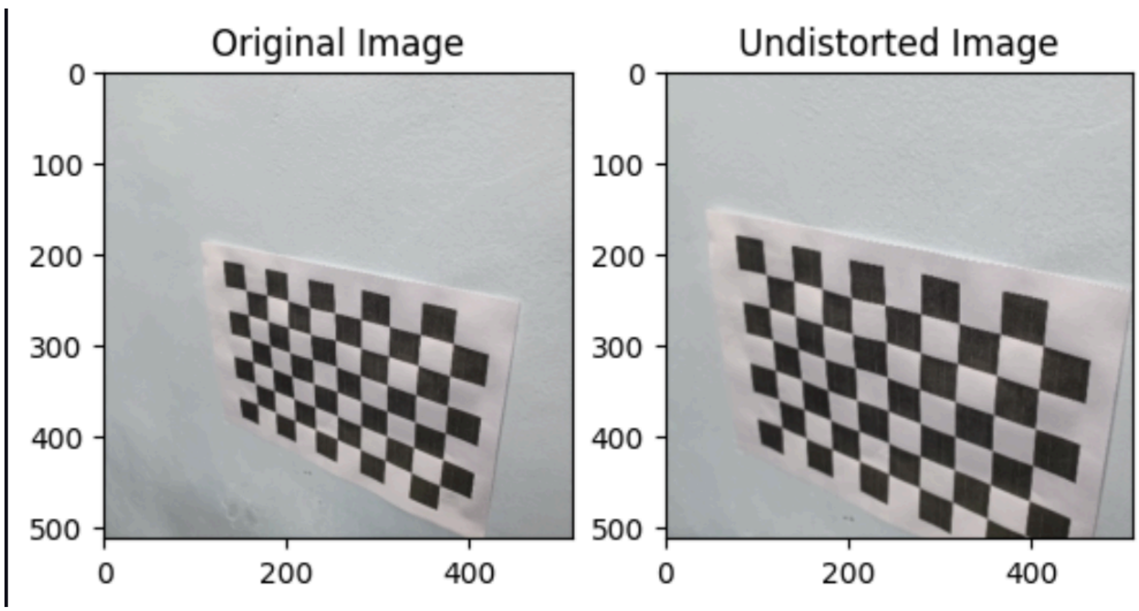
(2) Camera extrinsic parameters:

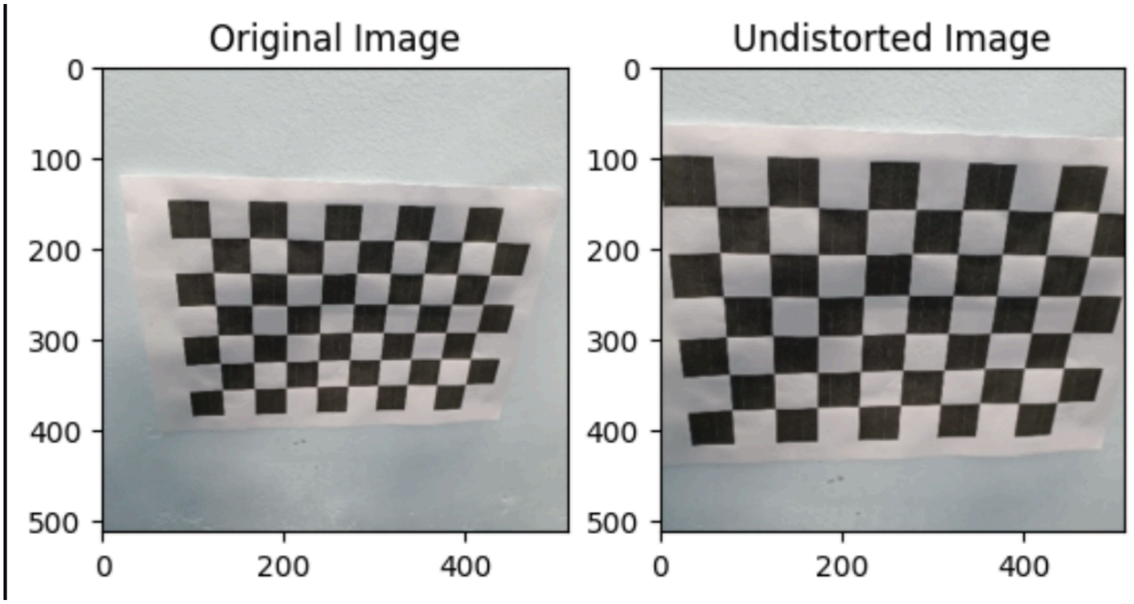
Image 0
Rotation Vector: [0.09560065 0.07129398 -1.57219482]
Translation Vector: [-3.98521945 2.15749286 12.67382097]
Image 1
Rotation Vector: [0.86920417 -0.06668269 -1.51535309]
Translation Vector: [-2.81548126 3.53493994 12.08760974]
Image 2
Rotation Vector: [0.0946836 0.11150948 -1.55285579]
Translation Vector: [-4.33033001 2.89817386 12.49048132]
Image 3
Rotation Vector: [0.36390107 0.01982289 -1.54643838]
Translation Vector: [-3.15682772 3.6522204 11.7287868]
Image 4
Rotation Vector: [0.17140174 -0.2963308 -1.47982473]
Translation Vector: [-3.48859834 2.55248936 10.64611649]
Image 5
Rotation Vector: [-0.23149146 -0.08731194 -1.54061736]
Translation Vector: [-4.36328633 1.84546473 11.27818543]
Image 6
Rotation Vector: [0.18524243 0.25792458 -1.53091255]
Translation Vector: [-4.15596307 4.18196673 13.4763175]
Image 7
Rotation Vector: [0.08700205 0.01405954 -1.5586453]
Translation Vector: [-3.89847577 3.29187977 12.25943124]
Image 8
Rotation Vector: [0.06611028 0.83431398 -1.29687929]
Translation Vector: [-4.49759535 1.84109808 15.71161448]
Image 9
Rotation Vector: [0.85655786 -0.10198963 -1.53627836]
Translation Vector: [-2.18105933 3.19590317 10.54677057]
Image 10

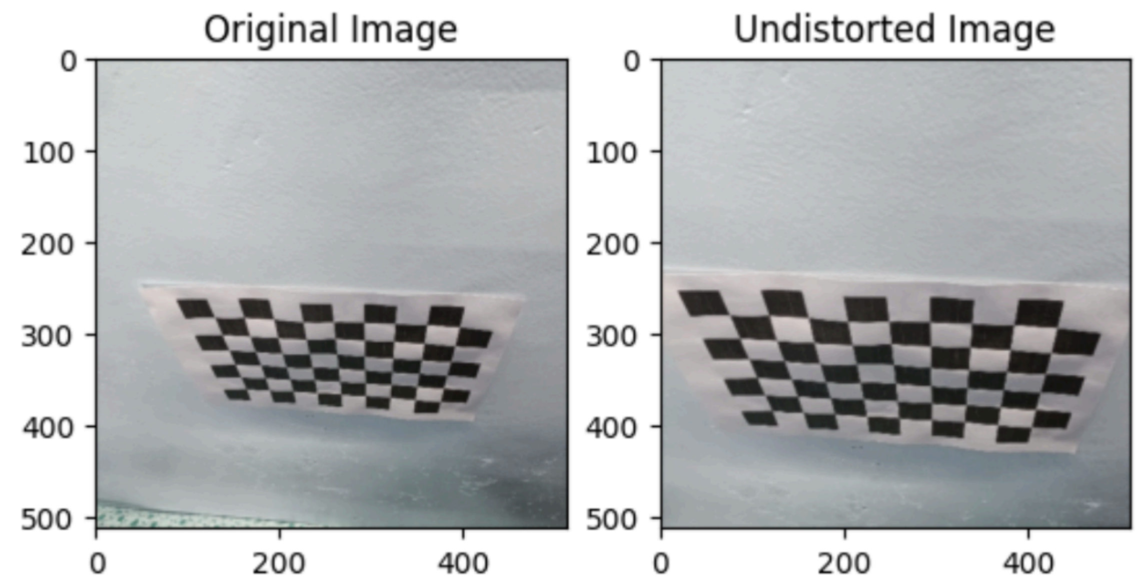
Rotation Vector: [0.28687428 -0.86685503 -1.28460793]
Translation Vector: [-0.81762008 1.5956738   8.86597777]
Image 11
Rotation Vector: [-0.2747488 -0.31069828 -1.54844291]
Translation Vector: [-3.79791027 1.41577081 12.13499871]
Image 12
Rotation Vector: [-0.34433488 -0.4761719 -1.54301168]
Translation Vector: [-3.59722694 1.87991281 12.07846616]
Image 13
Rotation Vector: [-0.51084834 -0.68316735 -1.52178989]
Translation Vector: [-3.9817211   0.97782949 11.61896575]
Image 14
Rotation Vector: [1.02533297 -0.15437411 -1.34663332]
Translation Vector: [-2.78328205 3.11822975 9.66558563]
Image 15
Rotation Vector: [0.11088738 0.90303425 -1.75018526]
Translation Vector: [-2.95693686 5.60158009 16.49400126]
Image 16
Rotation Vector: [-0.75804451 0.19570874 -1.58852445]
Translation Vector: [-4.39947822 2.33920013 13.26030606]
Image 17
Rotation Vector: [-0.70816719 0.76393987 -1.39459754]
Translation Vector: [-5.74316741 2.00064669 16.23893989]
Image 18
Rotation Vector: [0.60533977 0.77614333 -1.47097565]
Translation Vector: [-3.80400492 3.8211445 18.96451414]
Image 19
Rotation Vector: [-0.35661686 -0.56272975 -1.54465701]
Translation Vector: [-3.97226185 -1.91192177 15.14737054]
Image 20
Rotation Vector: [0.06247476 0.63122839 -1.53260608]
Translation Vector: [-4.46956242 3.45907127 19.00503028]
Image 21
Rotation Vector: [0.46906367 0.26866348 -1.50816213]
Translation Vector: [-3.73392423 2.68640801 14.10375971]
Image 22
Rotation Vector: [-0.24960159 0.61793447 -1.49411068]
Translation Vector: [-3.95064903 2.47329157 15.87057428]
Image 23
Rotation Vector: [-0.1558652   0.75459862 -1.39335783]

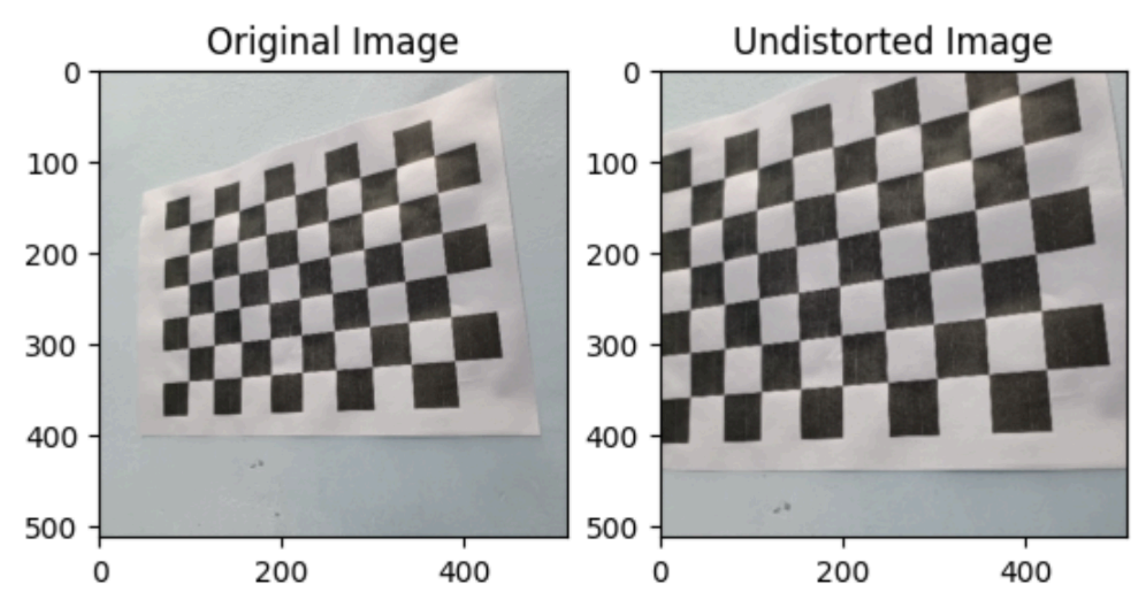
Translation Vector: [-3.55000262 3.8785531 18.49719229]
Image 24
Rotation Vector: [0.13086283 0.7860942 -1.24791513]
Translation Vector: [-4.95734517 1.9748631 15.91691491]
Image 25
Rotation Vector: [-0.48796188 0.22965357 -1.63336193]
Translation Vector: [-4.54583753 2.19857463 13.54587906]
Image 26
Rotation Vector: [-0.6610983 0.11174666 -1.53731249]
Translation Vector: [-4.07772839 2.14795688 13.43916167]
Image 27
Rotation Vector: [-0.35895607 -0.24800904 -1.47912371]
Translation Vector: [-4.22462101 0.97318688 12.7688806]

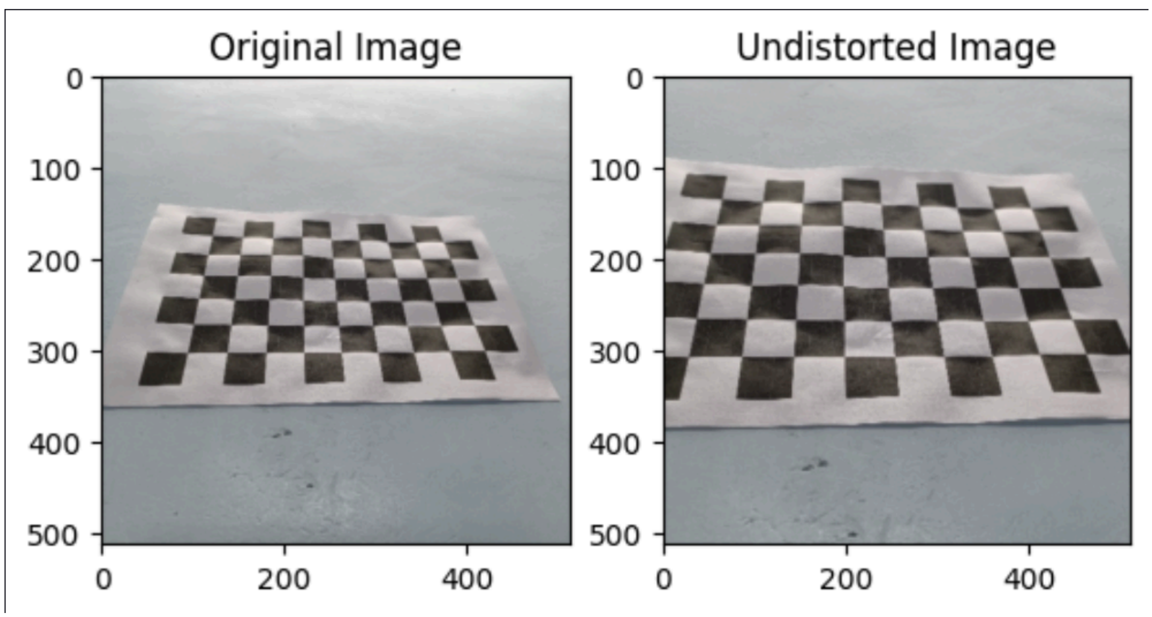
(3) **Radial distortion coefficients**:
Radial Distortion Coefficients: [[2.95175621e-01 -2.30374358e+00 1.72739831e-03 3.41344935e-03
5.58612187e+00]]




Original Image
Undistorted Image
0
100
200
300
400
500
0
200
400


Original Image
Undistorted Image
0
100
200
300
400
500
0
200
400



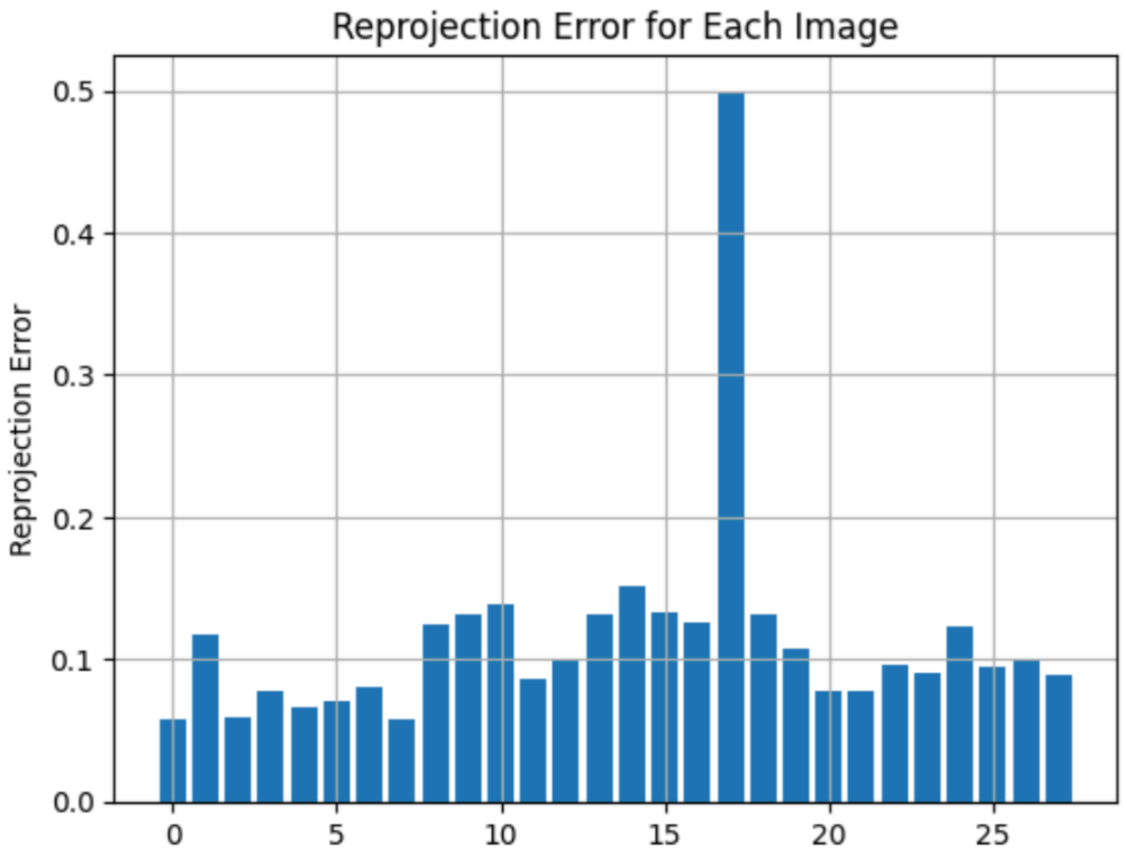


Applying radial distortion coefficients to undistort images alters the appearance of straight lines at the image corners. Initially distorted due to lens imperfections, these lines become straighter and more accurately represented after correction. This demonstrates the corrective effect of distortion compensation, essential for preserving geometric fidelity in tasks like image analysis and computer vision.

(4) Reprojection error per image:

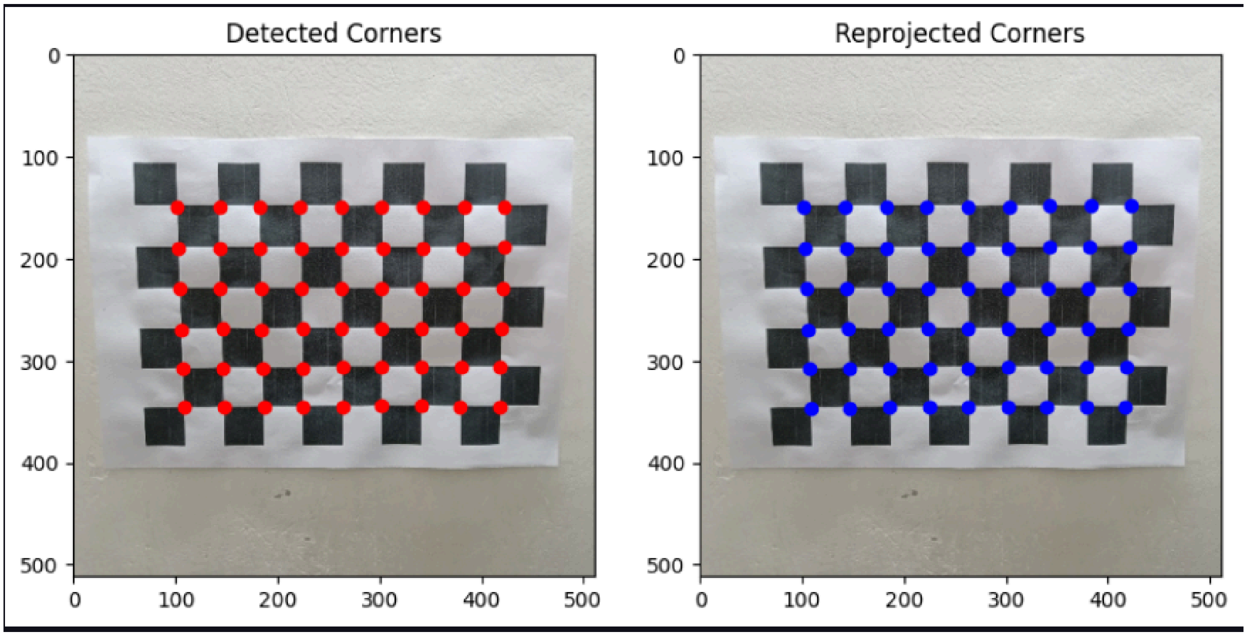
Reprojection Error for Image 0 : 0.05840403190094309
Reprojection Error for Image 1 : 0.11689671199278649
Reprojection Error for Image 2 : 0.05868370202225279
Reprojection Error for Image 3 : 0.07738909831699038
Reprojection Error for Image 4 : 0.06607172535422497

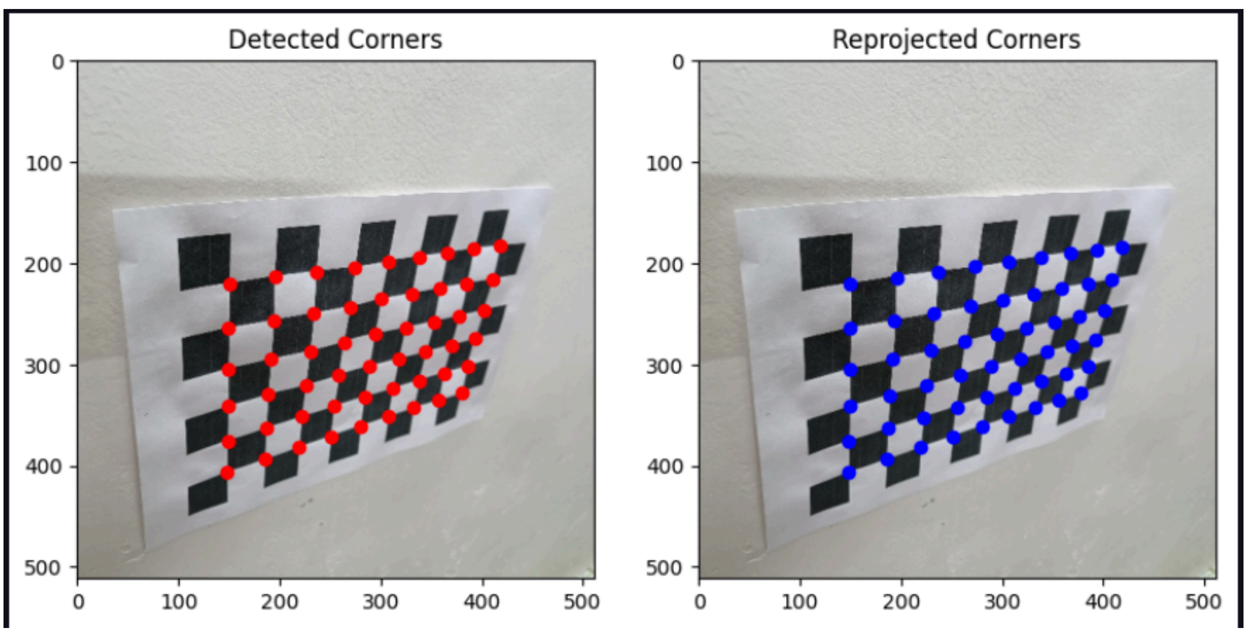
Reprojection Error for Image 5 : 0.07103060623981218
Reprojection Error for Image 6 : 0.08103985892621121
Reprojection Error for Image 7 : 0.05835918981508332
Reprojection Error for Image 8 : 0.1251031858314579
Reprojection Error for Image 9 : 0.131698485076388
Reprojection Error for Image 10 : 0.13938404579038924
Reprojection Error for Image 11 : 0.0864790686903139
Reprojection Error for Image 12 : 0.0988447416942954
Reprojection Error for Image 13 : 0.13227185535604305
Reprojection Error for Image 14 : 0.1513022848315769
Reprojection Error for Image 15 : 0.13374579822568314
Reprojection Error for Image 16 : 0.12560198690876934
Reprojection Error for Image 17 : 0.4997915466248572
Reprojection Error for Image 18 : 0.1311857171729557
Reprojection Error for Image 19 : 0.10723147968781452
Reprojection Error for Image 20 : 0.07838432232473122
Reprojection Error for Image 21 : 0.0774735618777967
Reprojection Error for Image 22 : 0.09600641066775956
Reprojection Error for Image 23 : 0.09008681054427724
Reprojection Error for Image 24 : 0.12318111739506209
Reprojection Error for Image 25 : 0.09409616870214074
Reprojection Error for Image 26 : 0.09999316900221938
Reprojection Error for Image 27 : 0.08939491486205048

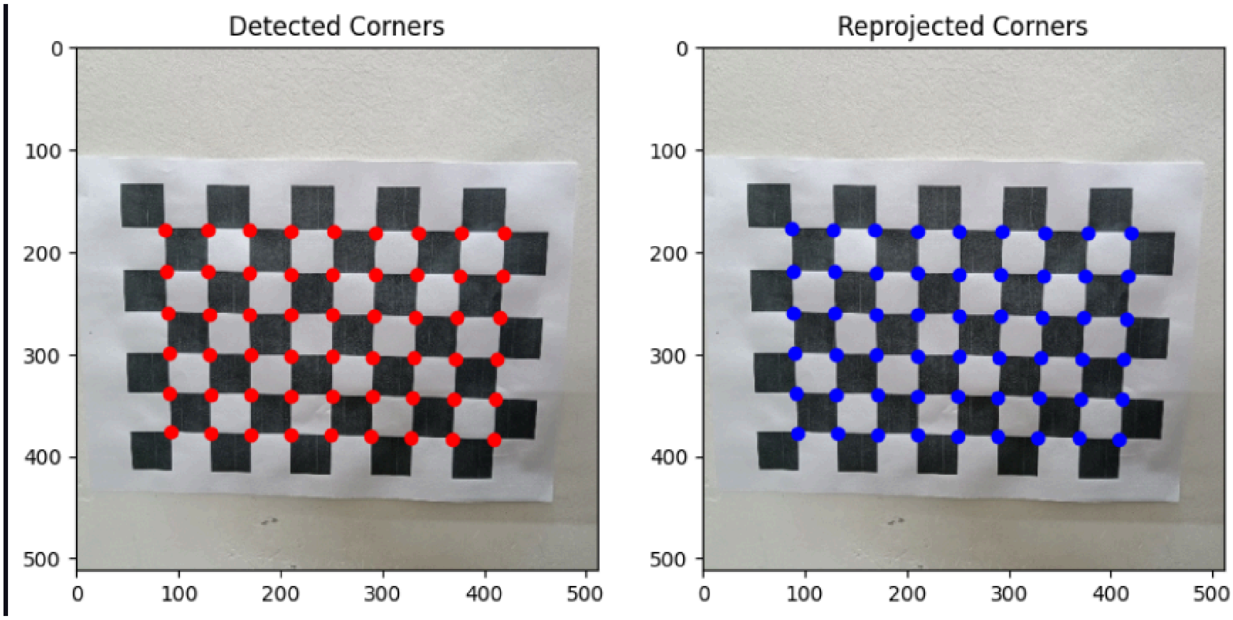


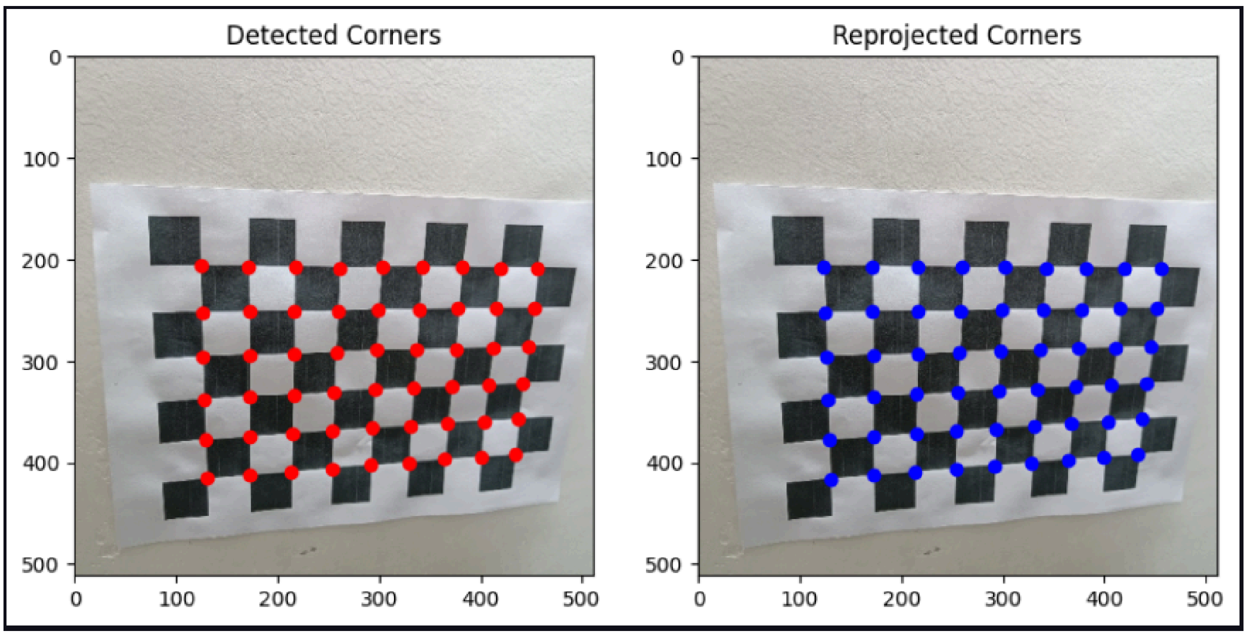
The reprojection errors across the 25 selected images vary, indicating differences in calibration quality and scene complexity. It is observed that image 17 stands out with a significantly higher reprojection error, suggesting potential issues with calibration or feature detection specific to that image. Further investigation into outliers like this can help identify and address underlying calibration challenges.

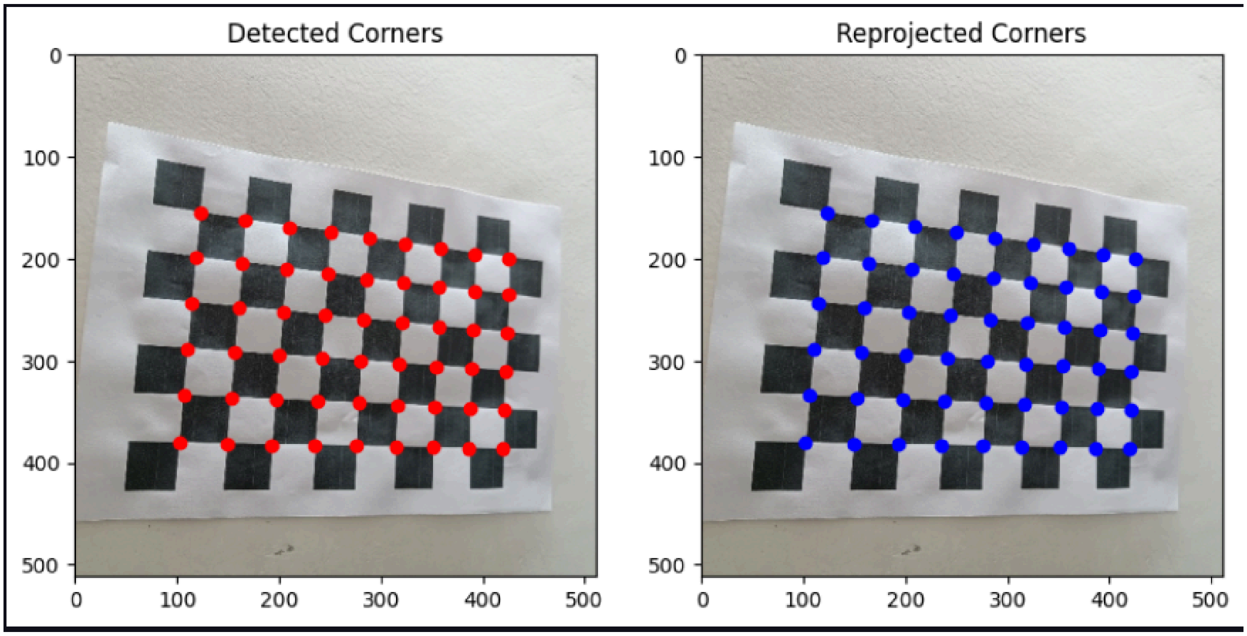
(5)

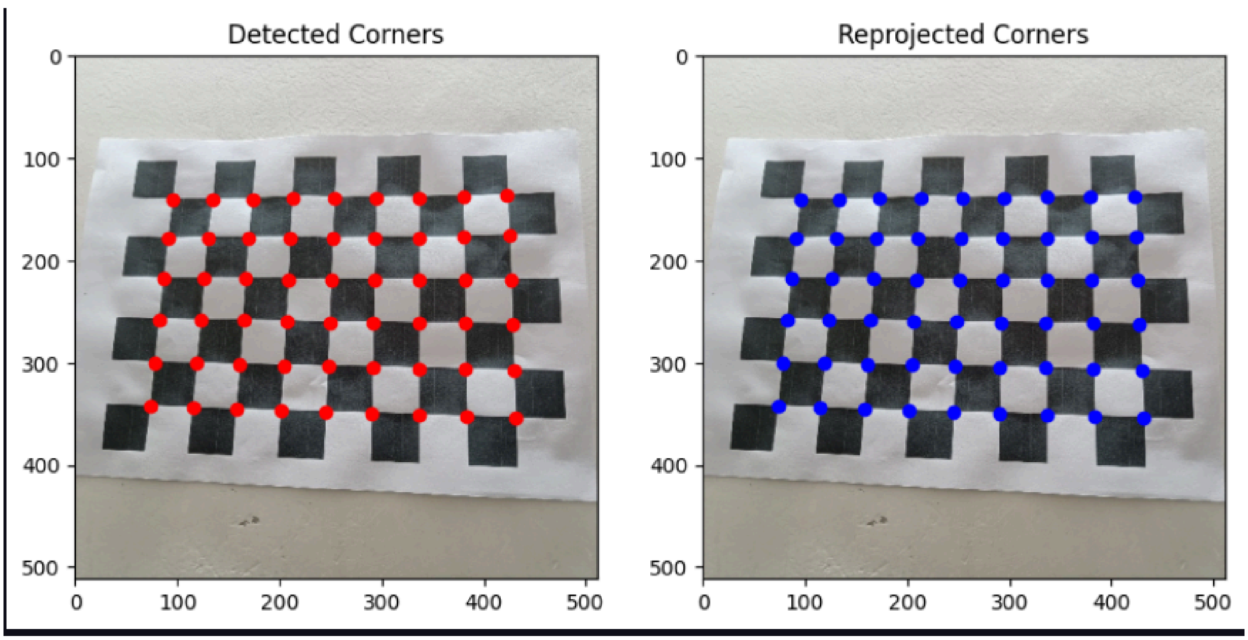

Detected Corners
Reprojected Corners
0
100
200
300
400
500
0
100
200
300
400
500

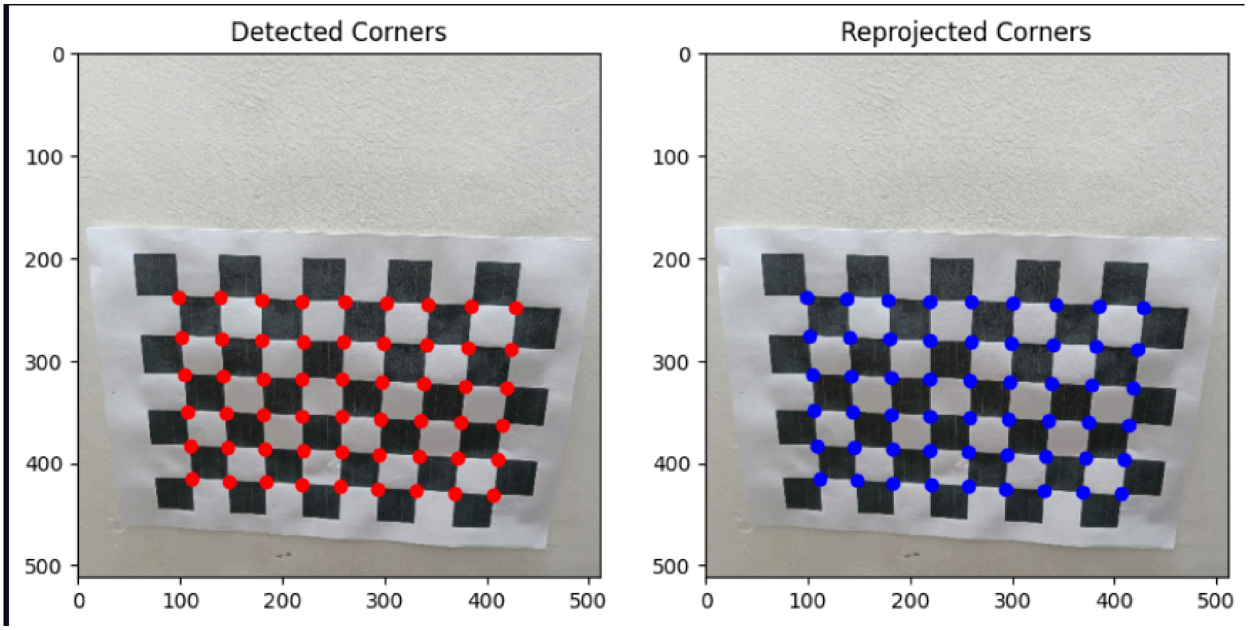

Detected Corners
Reprojected Corners
0
100
200
300
400
500
0
100
200
300
400
500

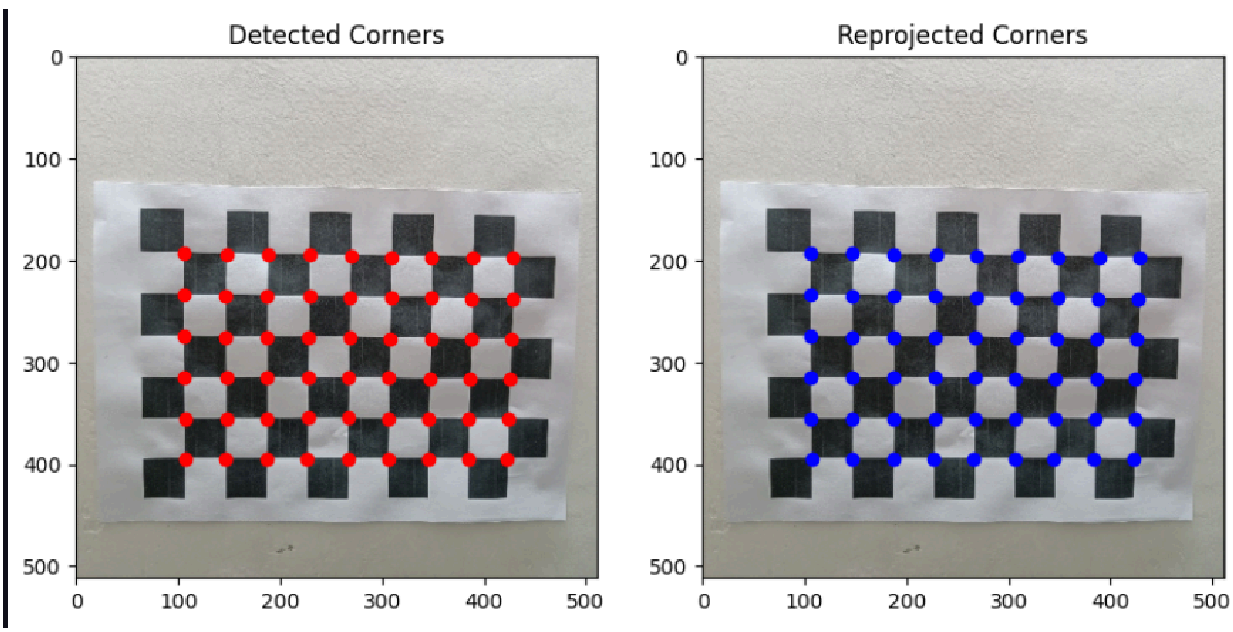

Detected Corners
Reprojected Corners
0
100
200
300
400
500

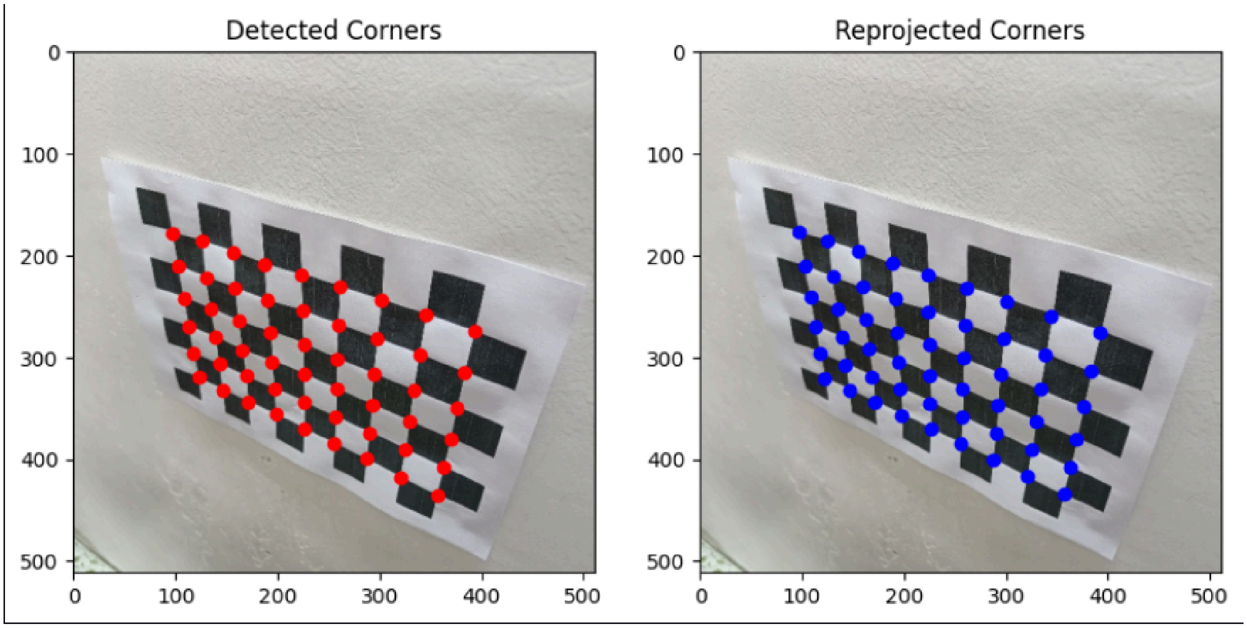

Detected Corners
Reprojected Corners
0
100
200
300
400
500

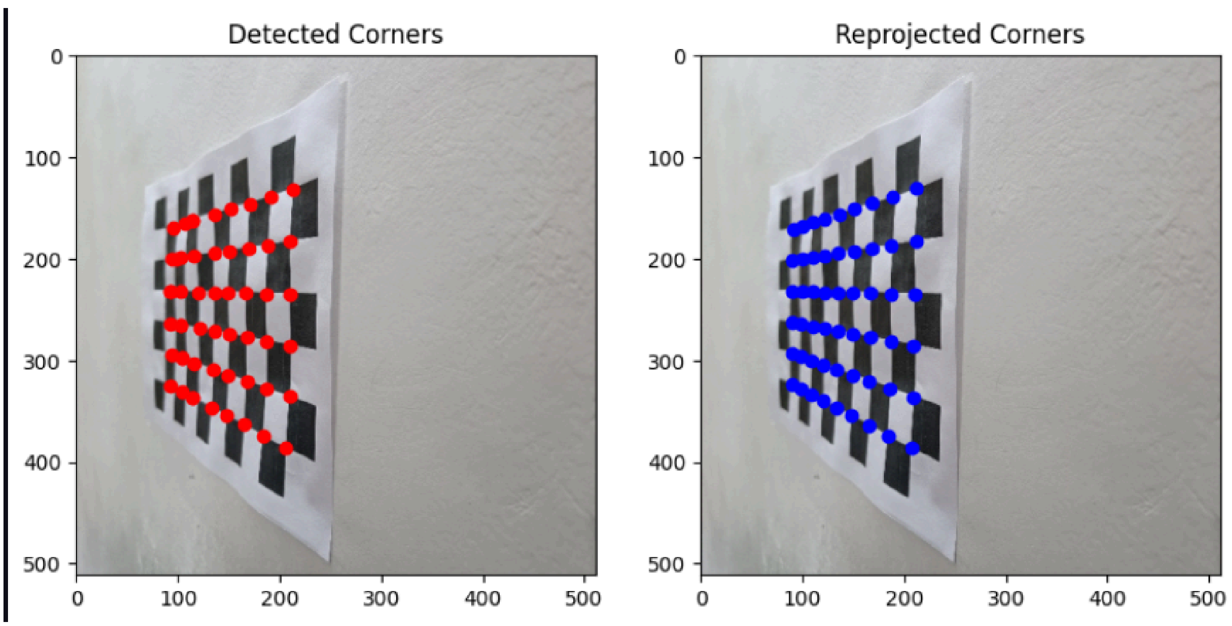

Detected Corners
Reprojected Corners
0
100
200
300
400
500
0
100
200
300
400
500


Detected Corners
Reprojected Corners
0
100
200
300
400
500
0
100
200
300
400
500


Detected Corners
Reprojected Corners
0
100
200
300
400
500


Detected Corners
Reprojected Corners
0
100
200
300
400
500





(6) Checkerboard Plane Normals:
Image 0: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.01560078 -0.10598594 0.99424524]
Image 1: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.54457481 -0.45049254 0.70745654]
Image 2: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.01171964 -0.13109141 0.99130101]
Image 3: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.21447336 -0.24136679 0.94643714]
Image 4: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.29907212 0.06752283 0.9518385]

Image 5: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.0889987 0.2030217 0.97512123]
Image 6: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.04979827 -0.27882541 0.9590498]
Image 7: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.04609801 -0.06463214 0.99684385]
Image 8: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.50533174 -0.4845893   0.71401193]
Image 9: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.5612525 -0.41724945 0.71477166]
Image 10: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.69914887 0.26864529 0.66258627]
Image 11: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.02282628 0.36455796 0.93090089]
Image 12: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.07693458 0.49900957 0.86317467]
Image 13: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.08610594 0.68906238 0.71956849]
Image 14: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.62947184 -0.51708066 0.57999379]
Image 15: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.35197029 -0.61676915 0.70406869]
Image 16: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.56858762 0.30065685 0.76571116]
Image 17: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.81360542 -0.01008935 0.58132988]
Image 18: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.08937947 -0.77180282 0.62954883]
Image 19: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.11614401 0.55199699 0.8257178]
Image 20: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.34123381 -0.42001739 0.84091907]
Image 21: Checkerboard Plane Normal (in Camera Coordinate Frame):
[-0.11662774 -0.45543769 0.88259531]
Image 22: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.52532142 -0.21518413 0.82324552]
Image 23: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.561321   -0.32575207 0.76079191]
Image 24: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.46096833 -0.4951995   0.73640047]

Image 25: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.44147484 0.13488983 0.88707649]
Image 26: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.46591897 0.32439299 0.8232185]
Image 27: Checkerboard Plane Normal (in Camera Coordinate Frame):
[0.05610097 0.38250129 0.92225021]

Q5.

(1)

Given a set of N points $\{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_N\}$ in a point cloud, where each point $\mathbf{p}_i$ is represented as a three-dimensional vector $[x_i, y_i, z_i]$, the algorithm proceeds as follows:

1. **Centering the Points**: Compute the centroid of the points $\mathbf{c} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{p}_i$ and center the points by subtracting the centroid from each point: $\mathbf{q}_i = \mathbf{p}_i - \mathbf{c}$.

2. **Covariance Matrix**: Construct the covariance matrix $\mathbf{C}$ of the centered points $\mathbf{q}_i$, given by $\mathbf{C} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{q}_i\mathbf{q}_i^T$.

3. **Singular Value Decomposition (SVD)**: Perform SVD on the covariance matrix $\mathbf{C} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices and $\mathbf{\Sigma}$ is a diagonal matrix containing the singular values.

4. **Extracting Normal Vector**: The normal vector of the plane is given by the last column of $\mathbf{V}$, denoted as $\mathbf{n}$. Normalize the normal vector to obtain a unit normal: $\mathbf{n} = \frac{\mathbf{v}_3}{\|\mathbf{v}_3\|}$.

5. **Computing Offset**: Compute the offset d of the plane from the origin by taking the dot product of the normalized normal vector $\mathbf{n}$ and the centroid $\mathbf{c}$: $d = \mathbf{n} \cdot \mathbf{c}$.

6. **Correction for Negative Offset**: If the computed offset is negative, reverse the direction of the normal vector and adjust the offset accordingly.

The algorithm iterates over all the point cloud files in the provided directory, computes the plane normals and offsets using the described procedure, and stores the results in lists. Finally, the function returns the arrays containing the computed plane normals and offsets.

List of all computed normals and offsets:

Normal: [0.63693437 -0.76499839 0.09535237], Offset: 4.993811298096004

```
Normal: [ 0.70169742 -0.70118932  0.1263102 ], Offset: 4.720779659728762
Normal: [ 0.93809241 -0.22958973  0.25936688], Offset: 5.204305546806318
Normal: [ 0.72352642 -0.68812782 -0.05467752], Offset: 4.79350162621247
Normal: [ 0.83333533 -0.55208241  0.02751785], Offset: 5.220803337324183
Normal: [ 0.91935323 -0.3607026   0.15710911], Offset: 5.702524575286329
Normal: [0.94151039 0.18949708 0.27865578], Offset: 6.323572215248568
Normal: [ 0.81878735  0.50763718 -0.26812639], Offset: 6.445439719171263
Normal: [ 0.93288206 -0.2440387   0.26490788], Offset: 5.6742584632412605
Normal: [ 0.95207118 -0.1539658   0.26430097], Offset: 5.743694294895789
Normal: [ 0.60407281 -0.77412903  0.18926246], Offset: 5.062352211986537
Normal: [ 0.75932657 -0.56254883  0.32705042], Offset: 4.954015451590529
Normal: [ 0.47346333 -0.7599202  -0.44536924], Offset: 5.2549842264287
Normal: [ 0.52528321 -0.84450165 -0.1043768 ], Offset: 6.103645571922049
Normal: [ 0.40409023 -0.88184713  0.24301591], Offset: 5.436720512190625
Normal: [ 0.96334688 -0.25648046 -0.07861659], Offset: 8.213132600834639
Normal: [ 0.87718352 -0.41582556  0.24007951], Offset: 7.435546784352128
Normal: [ 0.9065703  -0.30866042 -0.28785247], Offset: 8.020993504372287
Normal: [0.94902101 0.13112718 0.28664401], Offset: 9.499025451119042
Normal: [ 0.98745566  0.06712949 -0.14291586], Offset: 10.309979614425067
Normal: [ 0.90149712  0.39908052 -0.16744454], Offset: 8.140582366167768
Normal: [ 0.95936444 -0.00899235 -0.2820266 ], Offset: 8.724272669222731
Normal: [ 0.9493562   0.22833398 -0.21583881], Offset: 7.271752317637931
Normal: [ 0.43460444 -0.89601842  0.09093933], Offset: 6.595058223525191
Normal: [ 0.76477664 -0.64422625 -0.00944619], Offset: 6.987835947389349
Normal: [ 0.61872738 -0.78293781  0.06469016], Offset: 6.589564593223998
Normal: [ 0.97607976 -0.21597961  0.02492227], Offset: 4.772067004354305
Normal: [ 0.60371533 -0.79239849  0.08736381], Offset: 3.703875621094408
Normal: [ 0.73737765 -0.5739658  -0.35614247], Offset: 3.971542150381968
Normal: [ 0.89093965  0.24536379 -0.38212975], Offset: 7.618700179547205
Normal: [ 0.99121488 -0.12226682 -0.05043704], Offset: 6.093678679246951
Normal: [ 0.825289   -0.10756967 -0.55437067], Offset: 5.978131037863135
Normal: [ 0.93223066 -0.19842704 -0.30260984], Offset: 6.265358115913809
Normal: [ 0.98936009 -0.13850217 -0.04453952], Offset: 7.325197139472717
Normal: [ 0.96691202 -0.15794554 -0.2003356 ], Offset: 8.493221054907725
Normal: [ 0.99427448 -0.1042191   0.02359328], Offset: 8.846168921169088
Normal: [ 0.662242   -0.74563698  0.07389876], Offset: 7.749464153135403
Normal: [ 0.73370584 -0.67940958  0.00885226], Offset: 6.969188884408079
```
(2)

4.2)

Suppose the plane normals for the camera & lidar space are given by $\bar{\theta}_c$ and $\bar{\theta}_l$ respectively.

Here, $\bar{\theta}_c = [\theta_{1,c}\ \theta_{2,c} \dots \theta_{n,c}]$

$\bar{\theta}_l = [\theta_{1,l}\ \theta_{2,l} \dots \theta_{n,l}]$

where $n \rightarrow$ no. of pts

The distance b/w the origin and planes wrt the coordinate system of the camera is ~~the~~ given by α_c & α_l.

And, $\alpha_c = [\alpha_{c,1}\ \alpha_{c,2} \dots \alpha_{c,n}]^T$,

$\alpha_l = [\alpha_{l,1}\ \alpha_{l,2} \dots \alpha_{l,n}]^T$

and the closed form solⁿ given by $\rightarrow$

$$\boxed{t_1 = (\theta_c \theta_c^T)^{-1} \theta_c (\alpha_c - \alpha_l)} \rightarrow \text{Translation vector}$$

This vector t_1 is used for LIDAR-to-camera translation as ${}^{c}\vec{t}_L$.

the closed form solⁿ of R is given by $\rightarrow$

$$\boxed{R = VU^T} \text{ where, } \boxed{USV^T = \theta_l \theta_c^T}, \text{ obtained by SVD.}$$

This R is used for rotation as ${}^{c}R_L$.

The combined transformation matrix $\rightarrow {}^{c}T_L = [{}^{c}R_L \mid {}^{c}t_L]$

$${}^{c}T_L = \begin{bmatrix} {}^{c}R_L & {}^{c}t_L \\ 0 & 1 \end{bmatrix}, \boxed{{}^{c}t_L = (\theta_c \theta_c^T)^{-1} \theta_c (\alpha_c - \alpha_l)}, \boxed{{}^{c}R_L = VU^T}$$

(3)

Transformation matrix:

```
[[-0.17934806 -0.98377389 -0.00481772  0.15017004]
 [ 0.01810312  0.00159606 -0.99983485 -0.41198374]
 [ 0.98361911 -0.17940566  0.01752313 -0.60030909]
 [ 0.          0.          0.          1.        ]]
```

Determinant of rotation matrix: 0.9999999999999999

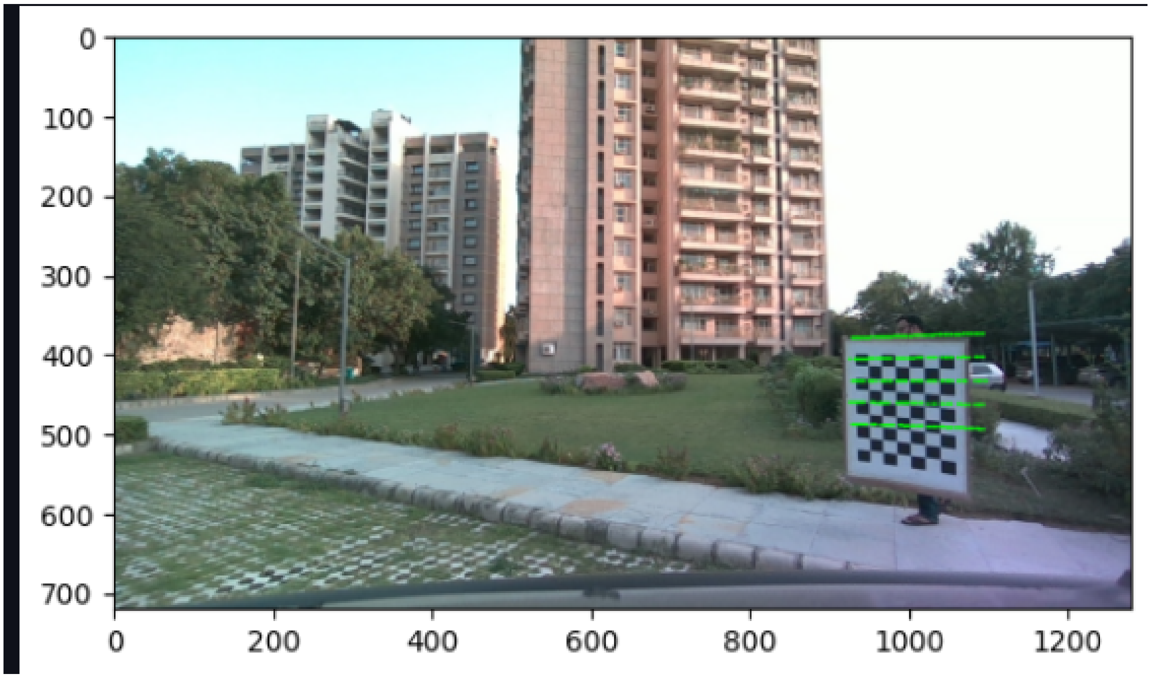
Rotation matrix:

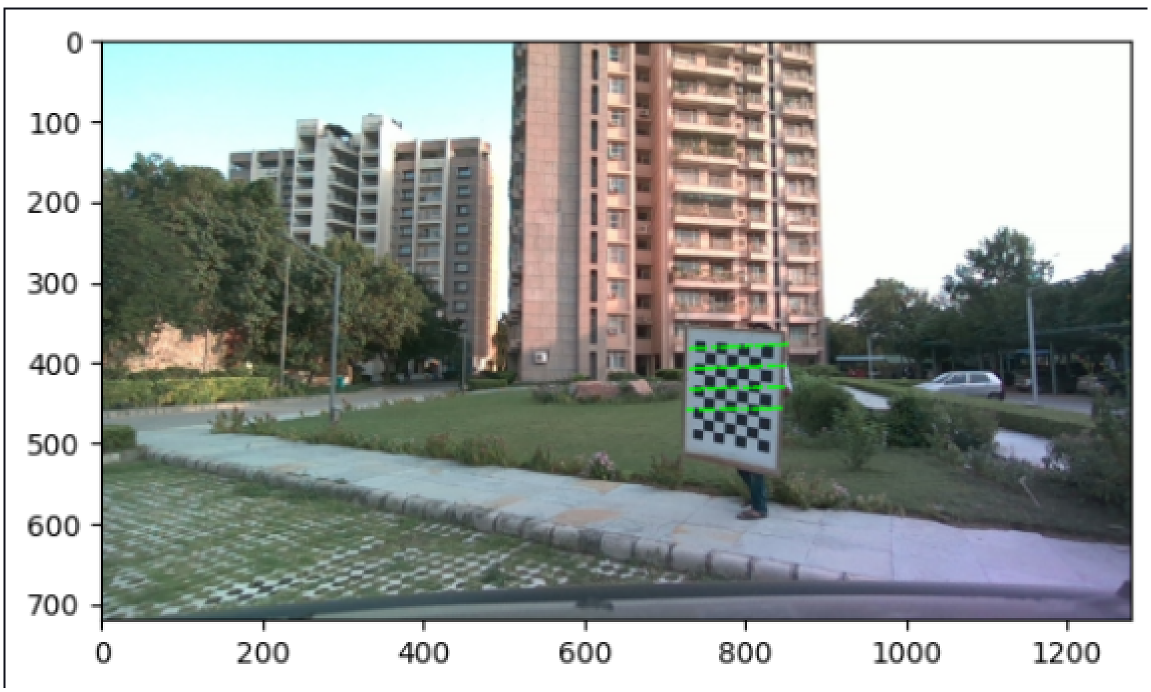
```
[[-0.17934806 -0.98377389 -0.00481772]
 [ 0.01810312  0.00159606 -0.99983485]
 [ 0.98361911 -0.17940566  0.01752313]]
```

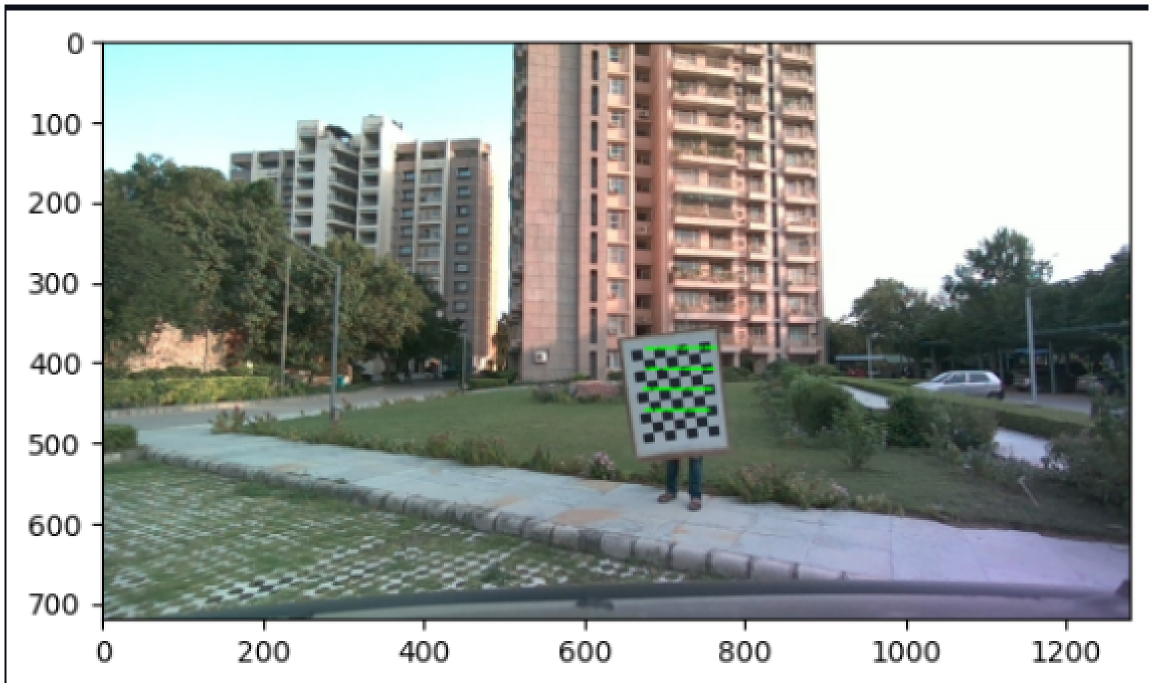
Translation matrix: [[0.15017004 -0.41198374 -0.60030909]]

(4)

Most of the points are within the checkerboard’s boundary. Few images of the LIDAR cloud points superimposed on the camera’s image:


0
100
200
300
400
500
600
700
0 200 400 600 800 1000 1200


0
100
200
300
400
500
600
700
0 200 400 600 800 1000 1200

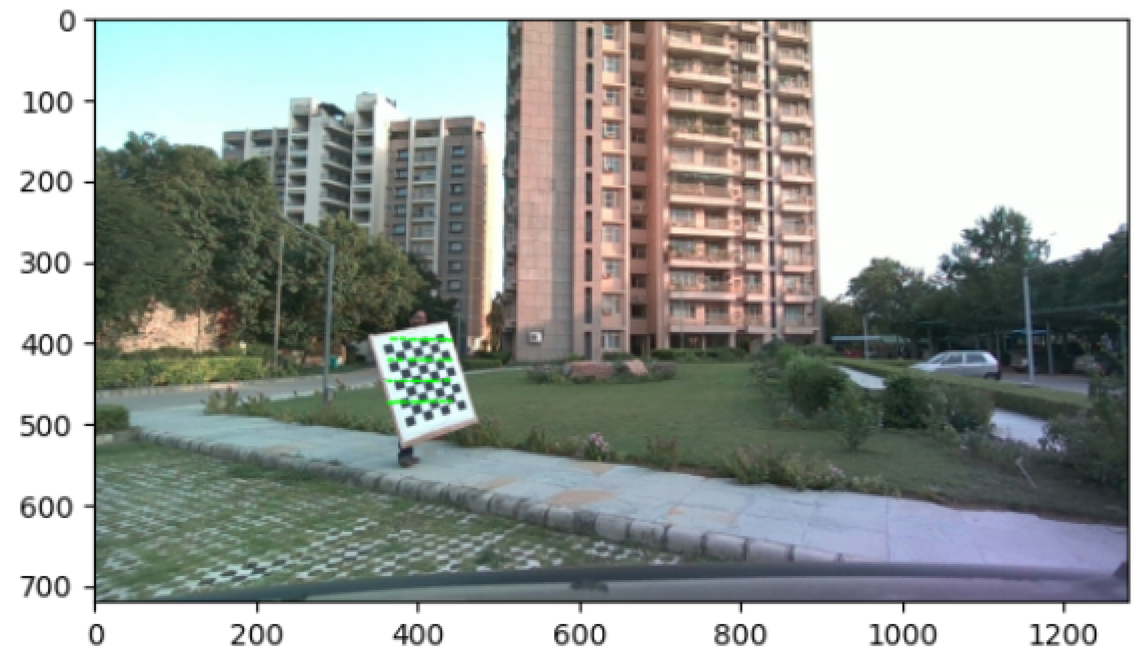

0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200

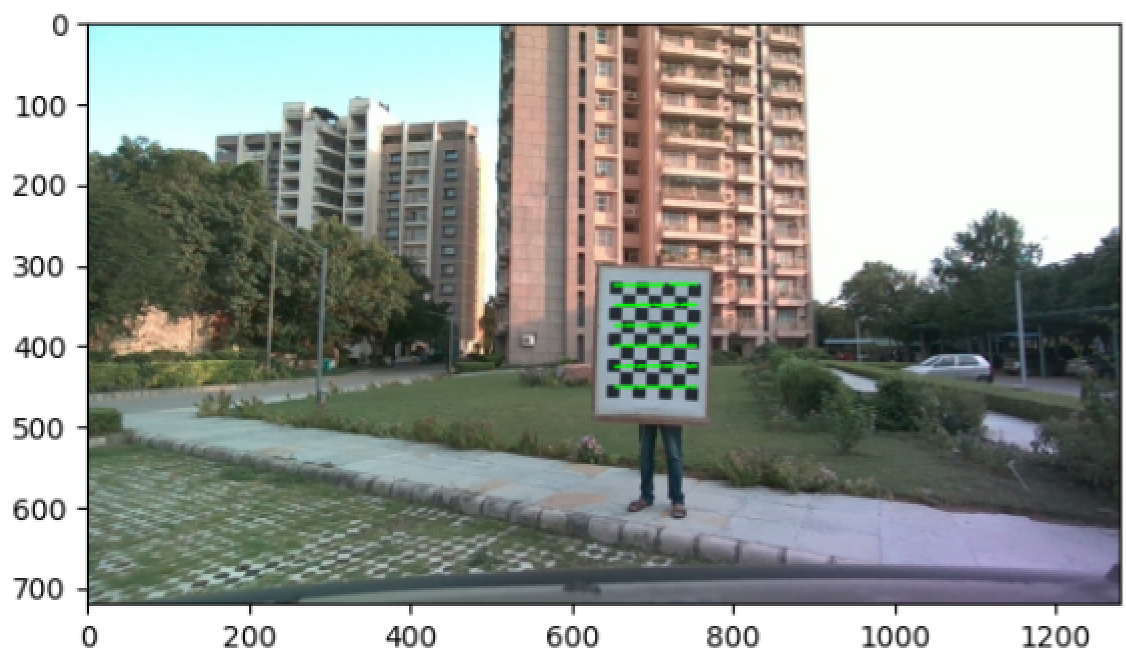

0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200

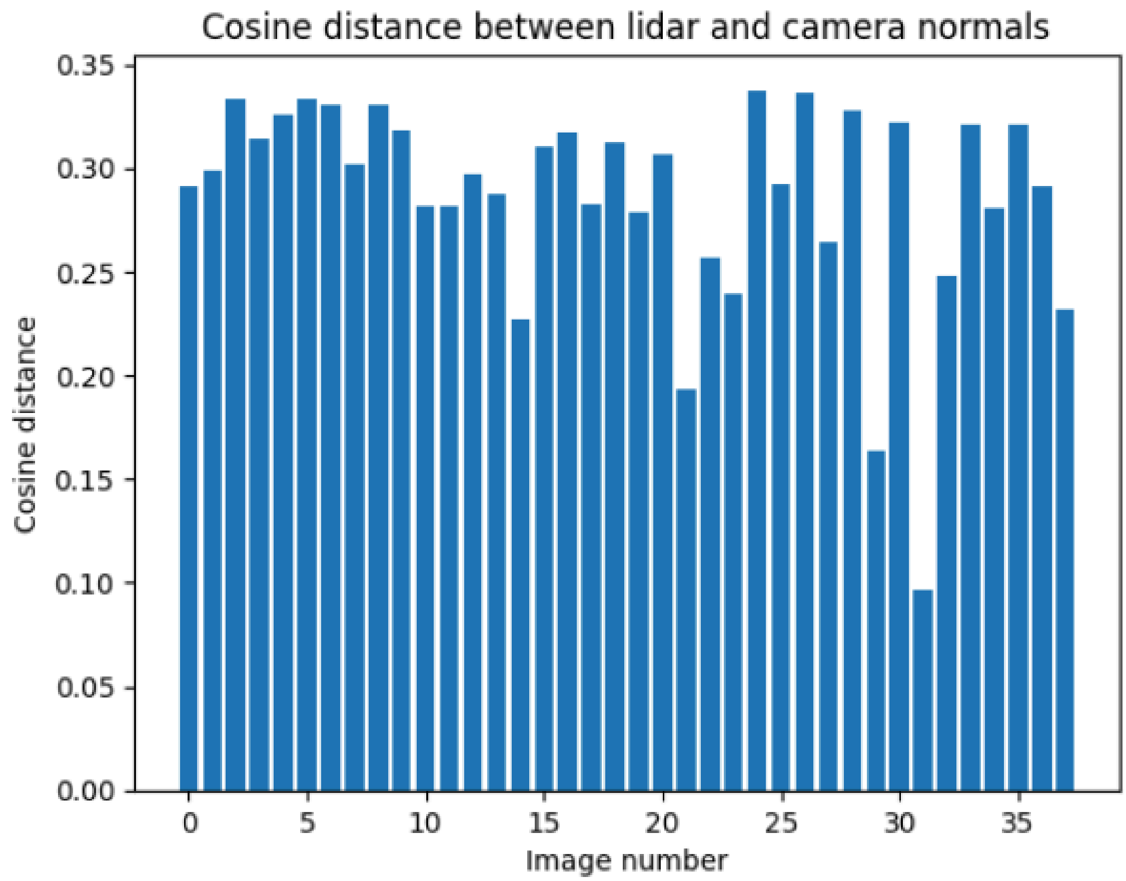

0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


0
100
200
300
400
500
600
700
0 200 400 600 800 1000 1200


0
100
200
300
400
500
600
700
0 200 400 600 800 1000 1200



(5) Histogram plot between cosine distance between the LIDAR and Camera normals:



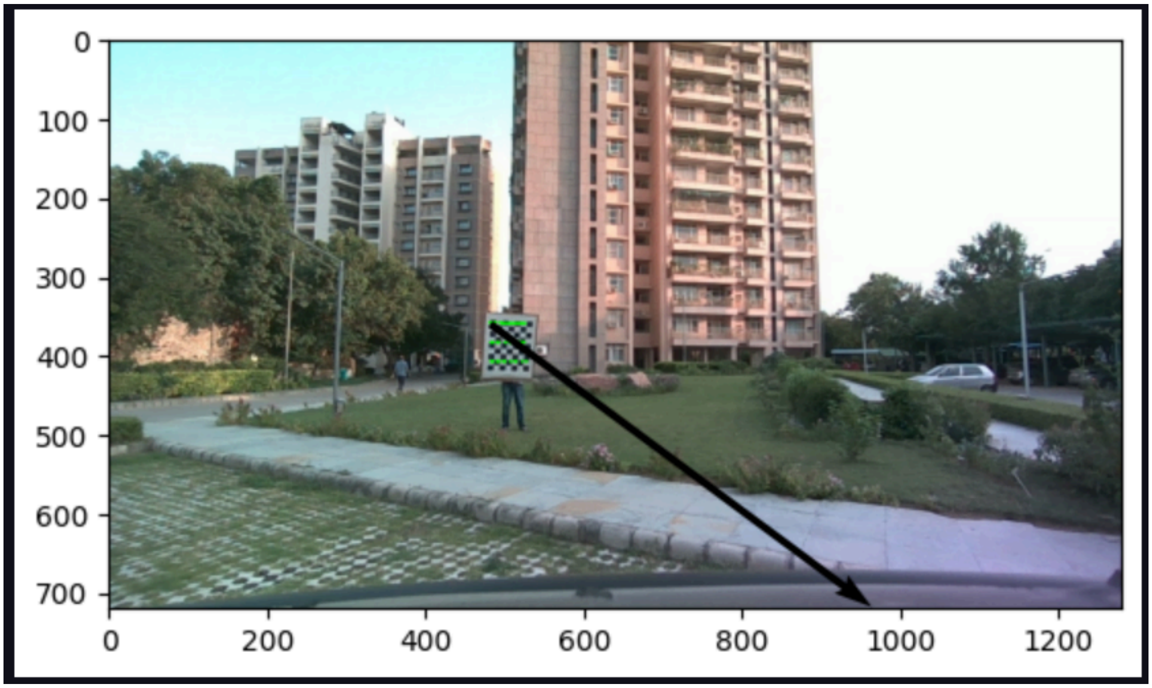
Mean cosine distance: 0.2863614261059832

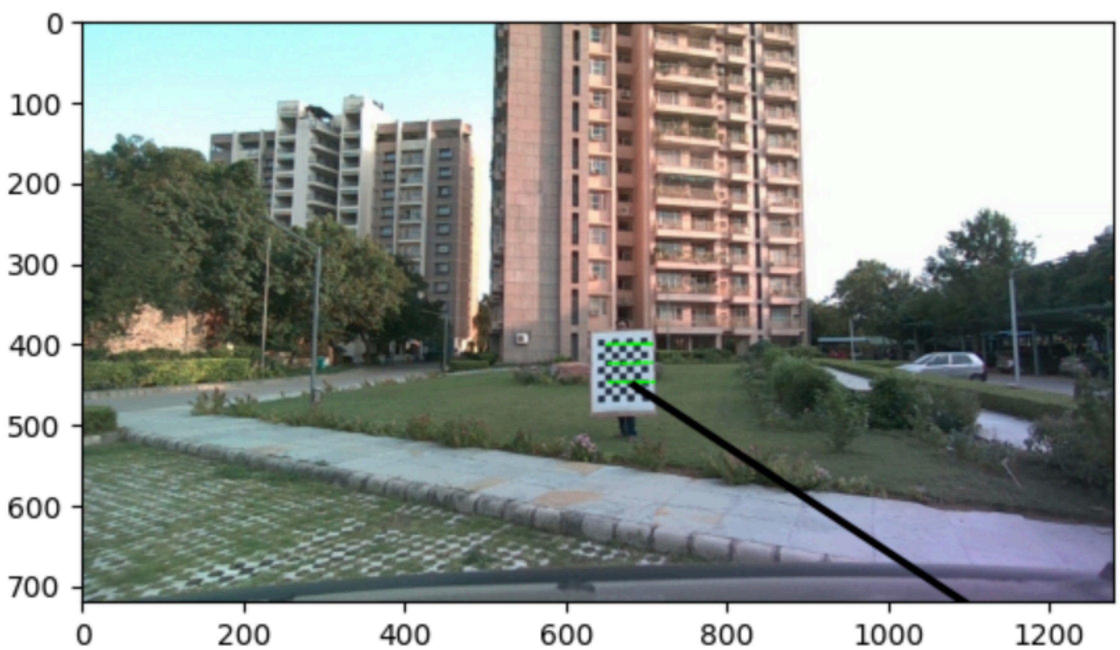
Standard deviation of cosine distance: 0.05050556201956534

Average cosine distance: 0.2863614261059832

Error distance (MSE): 0.002550811794912161

LIDAR Normals in camera space:


0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200