Assignment - 2
CSE344 : Computer Vision

Aakarsh Jain
2021507

March 2024

1

(a) Let the matrix for the rotations be R_1 and R_2 in the same order, and for translation be T. Then,

$$R_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$$
$$R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

using the standard form for rotation along the x and y-axis. Thus, multiplying the 2 matrices,

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

Now, writing the translation matrix using homogenous coordinates

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

Now, to find the overall transformation matrix, we can shift the rotation matrix to homogenous coordinate system, and find the product of 2.

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

(b) Multiplying the above matrix with the given point,

$$\begin{bmatrix} 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 3 \\ 0 & -1 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 5 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ -3 \\ 1 \end{bmatrix}$$

Thus, mapping to $(0, 1, 3)$. To find the mapping for origin,

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

Which is the same as purely translation as all axis of rotation pass through origin.

(c) As discussed in the lecture notes,

$$\theta = cos^{-1}(\frac{trace(R) - 1}{2}) = cos^{-1}(\frac{0 - 1}{2}) = \frac{2\pi}{3}$$

Then, we can find the axis of rotation as :-

$$n = \frac{1}{2\sin\theta} \begin{bmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{bmatrix} = \frac{1}{\sqrt{3}} \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$$

Thus, we rotate along the axis $\frac{1}{\sqrt{3}}(-x + y - z)$ by an angle of $\frac{2\pi}{3}$ in anticlockwise direction.

(d) Rodrigue's formula can be expressed in matrix notation as :-

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

where $N = \begin{bmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{bmatrix}$. Thus, for the first rotation, $N_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$, and $\theta_1 = \frac{\pi}{2}$

$$R_1 = \begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{bmatrix} + \begin{bmatrix}0 & 0 & 1\\0 & 0 & 0\\-1 & 0 & 0\end{bmatrix} + \begin{bmatrix}-1 & 0 & 0\\0 & 0 & 0\\0 & 0 & -1\end{bmatrix} = \begin{bmatrix}0 & 0 & 1\\0 & 1 & 0\\-1 & 0 & 0\end{bmatrix}$$

Similarly, for the second rotation, $N_2 = \begin{bmatrix}0 & 0 & 0\\0 & 0 & -1\\0 & 1 & 0\end{bmatrix}$, and $\theta_2 = \frac{-\pi}{2}$

$$R_2 = \begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{bmatrix} - \begin{bmatrix}0 & 0 & 0\\0 & 0 & -1\\0 & 1 & 0\end{bmatrix} + \begin{bmatrix}0 & 0 & 0\\0 & -1 & 0\\0 & 0 & -1\end{bmatrix} = \begin{bmatrix}1 & 0 & 0\\0 & 0 & 1\\0 & -1 & 0\end{bmatrix}$$

Thus, if we sequentially apply both matrices,

$$R_2R_1 = \begin{bmatrix}0 & 0 & 1\\-1 & 0 & 0\\0 & -1 & 0\end{bmatrix}$$

which is the same as we found previously.

2

First, we can express $x = x_\perp + x_\parallel$, w.r.t to the axis of rotation u.

Now, $x_\parallel = (u \cdot x)u$ by the definition of dot prodcut, and thus $x_\perp = x - (u \cdot x)u$.

Let x' be the rotated vector, then $x'_\parallel = x_\parallel$ and we can express $x'_\perp = \cos(\theta)x_\perp + \sin(\theta)(u \times x)$

Thus, now recombining,

$$x' = x'_\parallel + x'_\perp = x_\parallel + \cos(\theta)x_\perp + \sin(\theta)(u \times x)$$

Now, substituting $x_\perp$ and $x_\parallel$, we get,

$$\begin{aligned} x' &= (u \cdot x)u + \cos(\theta)(x - (u \cdot x)u) + \sin(\theta)(u \times x) \\ &= \cos(\theta)x + \sin(\theta)(u \times x) + (1 - \cos(\theta))(u \cdot x)u \end{aligned}$$

Expressing in the matrix form, $x' = Rx$ and $u \cdot x$ is equivalent to $u^T x$

$$Rx = \cos(\theta)x + \sin(\theta)(u \times x) + (1 - \cos(\theta))(u^T x)u$$

3

Since for the first camera, the world and camera coordinates are the same, we can get rid of all the extrinsic parameters. Thus,

$$x_1 = K_1 X$$

Hence, $X = K_1^{-1} x_1$

Now, since we need to apply the rotation R to obtain camera 2's reference from camera 1's reference, and the world coordinate is same as camera 1's coordinate,

$$x_2 = K_2[R|t]X$$

Where, $t = 0$

To ensure that the $[R|t]$ matrix is invertible, we will only use the homogenous coordinates. Thus, the matrix can be represented as :-

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

, which we know is invertible as it is a square matrix. Thus, $X = R'^{-1} K_2^{-1} x_2$

Hence, now comparing the 2 values of X,

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

Since we homogenised the coordinates of world and extrinsic parameters, K_i will now be a matrix of 3×4 size, with the extra row being filled with 0 marix

Thus, $H = K_1 R'^{-1} K_2^{-1}$. Also, since H is product of 3 invertible matrices, H is also invertible, and is a square matrix of size 3.

4

The clicked images can be found in the **chessboard** folder. There are total of 28 images clicked of a 7×5 chessboard. **OpenCV** was then used for all image processing, including detecting the corners, calibrating the camera and undistorting the image.

(a) The intrinsic parameters as calculated after camera calibration are (in mm) :-

- $f_x = 2805.45$
- $f_y = 2831.51$
- $c_x = 947.08$
- $c_y = 341.63$
- skew $= 0$

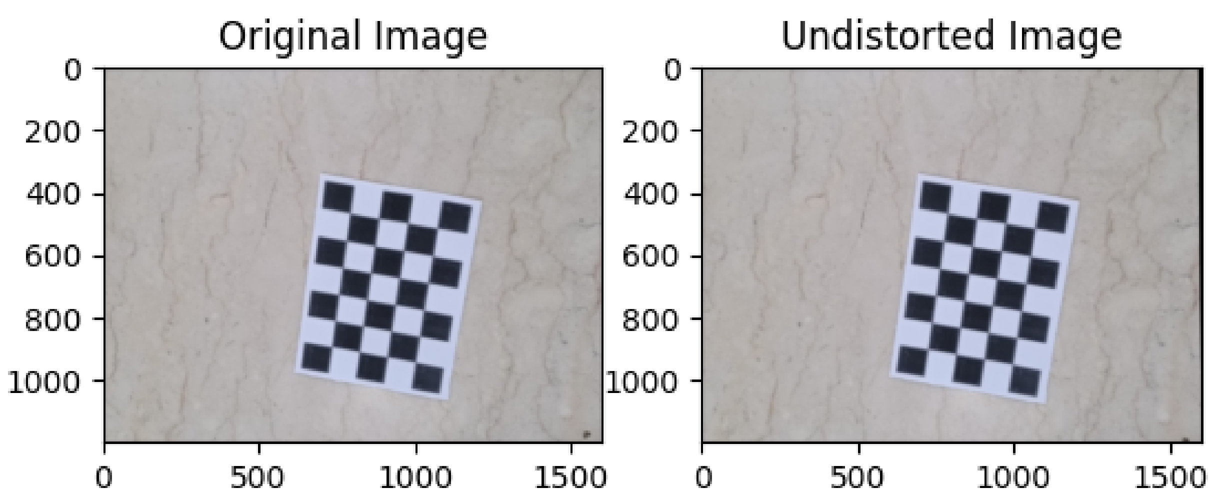
(b) The extrinsic parameters for each of the image have been calculated and can be found in the attached **q4_2021507.ipynb** file. The parameters for the first image are as follows :-

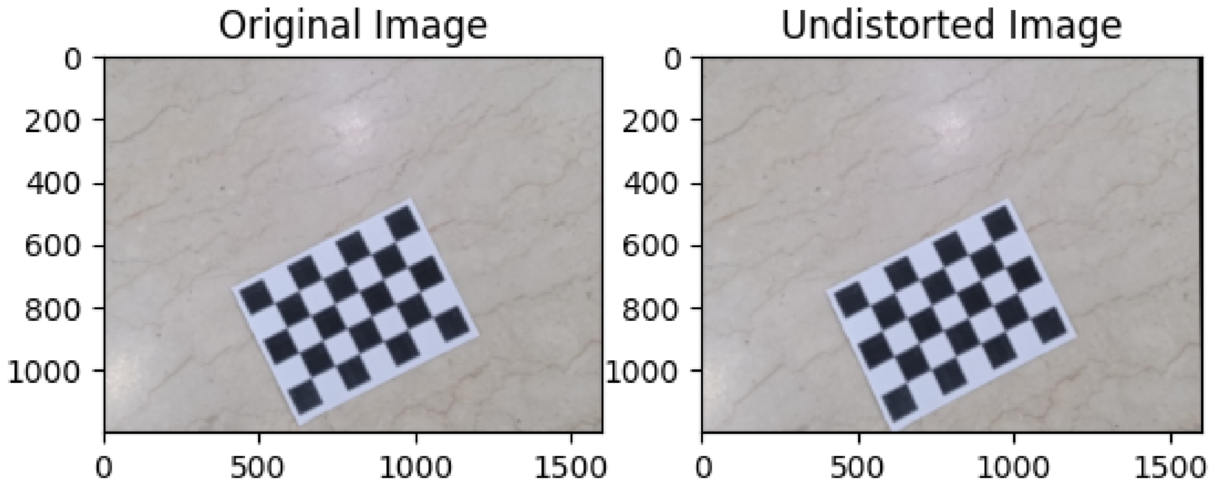
$$\text{Rotation matrix} = \begin{bmatrix} 0.89118822 & 0.44802773 & 0.07109648 \\ -0.45350291 & 0.88368473 & 0.11591551 \\ -0.01089351 & -0.135545 & 0.9907113 \end{bmatrix}$$

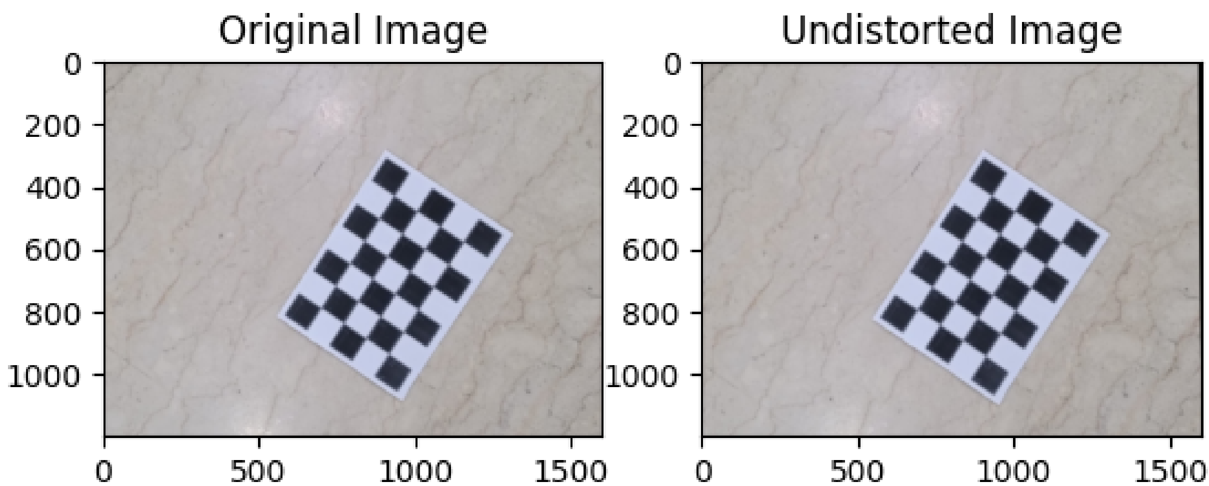
$$\text{Translation vector} = \begin{bmatrix} -4.59670877 \\ 5.15206066 \\ 31.91840931 \end{bmatrix}$$

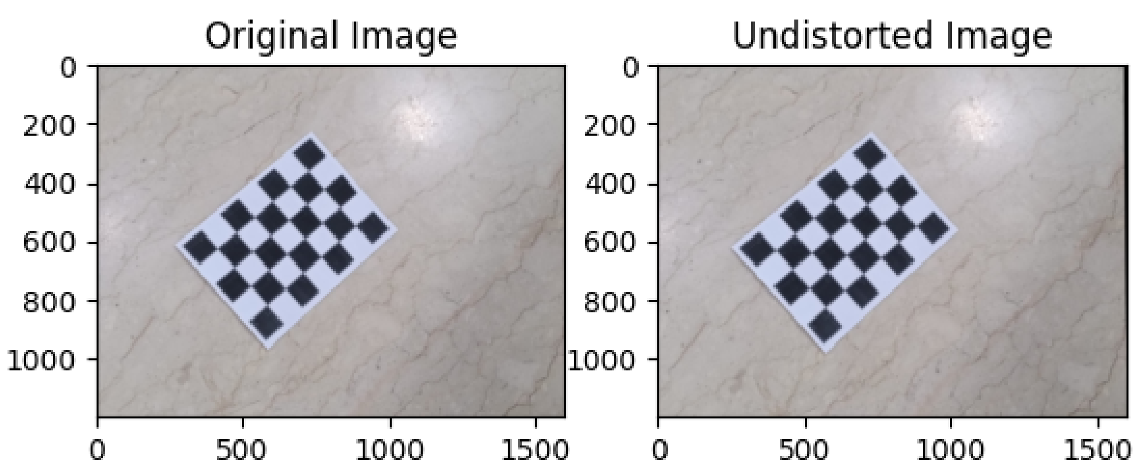
(c) The distortion coefficients are as follows:-

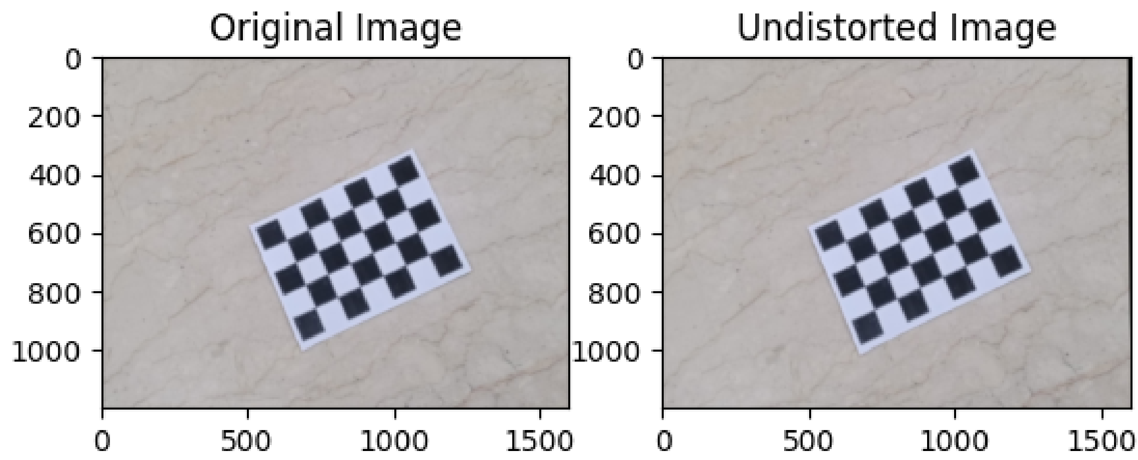
$$\text{Distortion coefficients} = \begin{bmatrix} 0.06846814 & -2.0937341 & -0.01717461 & 0.01765484 & 6.28917351 \end{bmatrix}$$




Original Image
Undistorted Image
0
200
400
600
800
1000
0
500
1000
1500

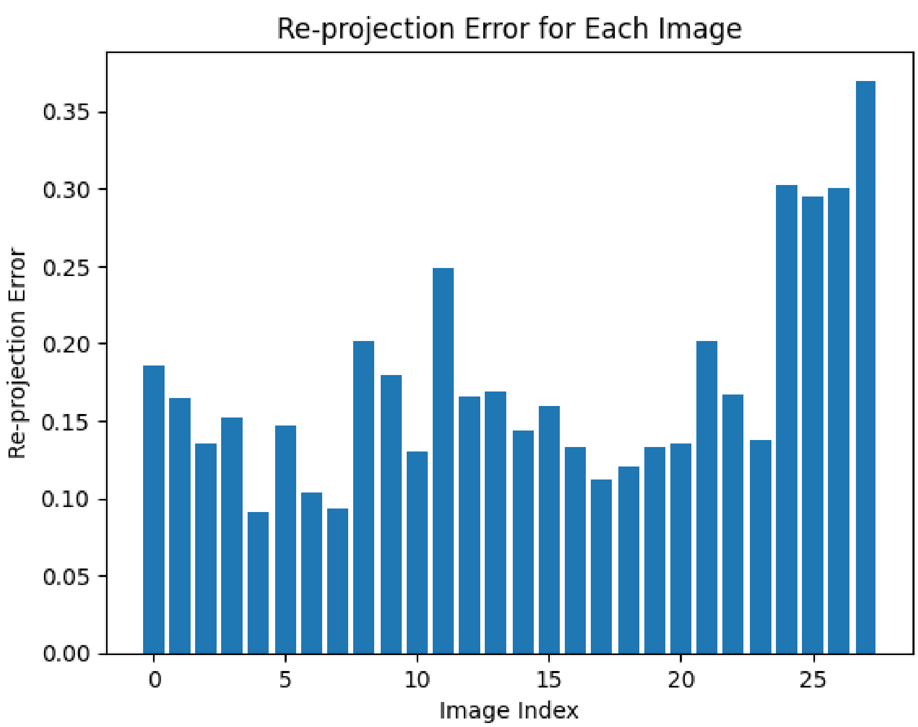

Original Image
Undistorted Image
0
200
400
600
800
1000
0
500
1000
1500


Original Image
Undistorted Image
0
200
400
600
800
1000
0
500
1000
1500

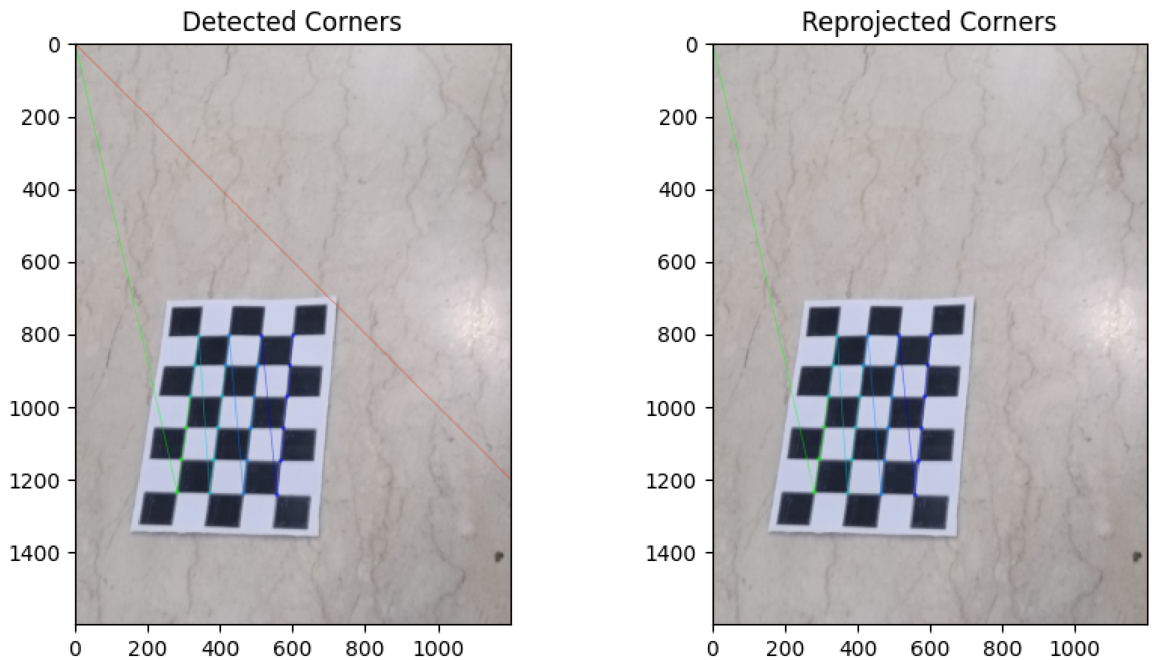


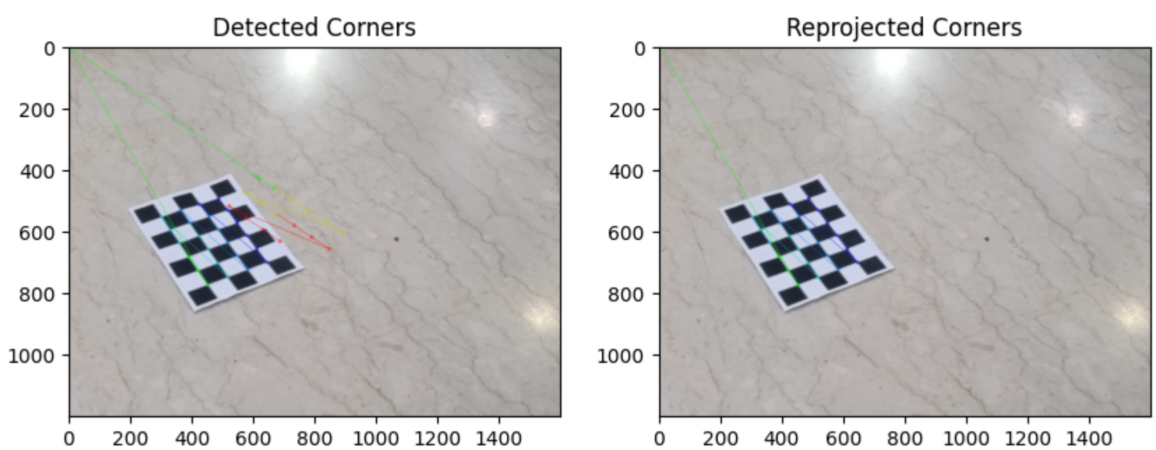
As we can see that the boundaries of the chessboard have been entirely straightened in the undistorted images. Since the border of chessboard was not a straight line in some cases, the same has also been straightened.

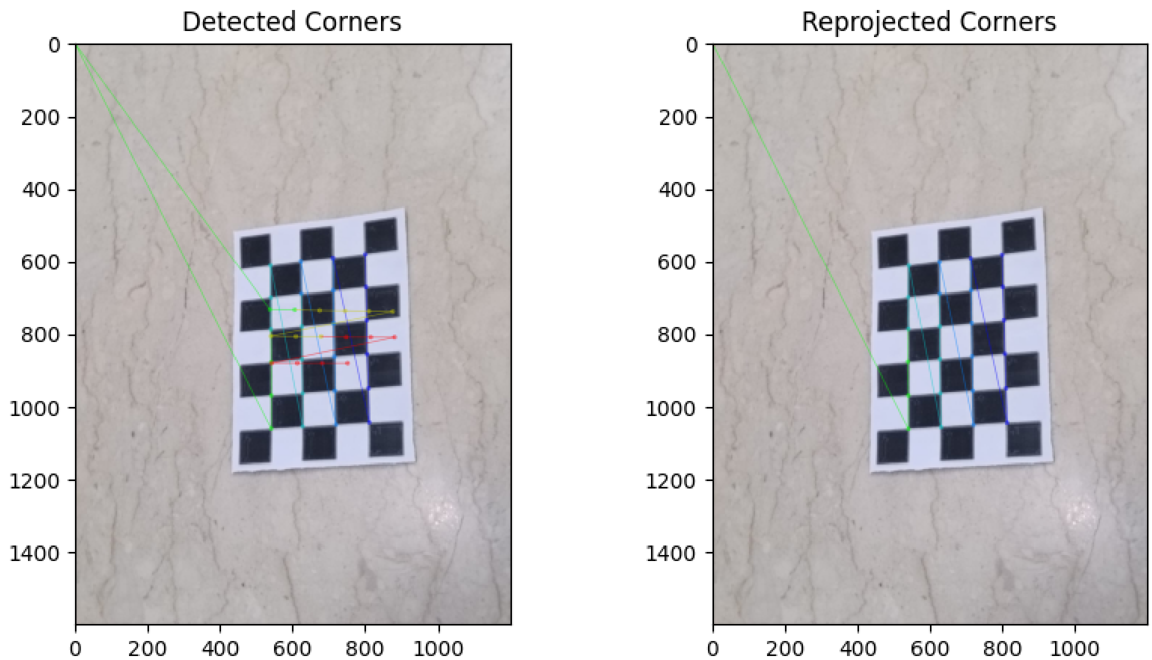
(d) The mean reprojection error is 0.17 and the standard deviation is 0.06. The above was calculated using the **L2 Norm** of the 2 images, standardized by the number of pixels in the image. The plot of the projection error is as follows:-



(e) After reprojecting the corners, the results are better and more consistent than before. 3 images have been attached here for reference, and the remaining can be found in the **q4_2021507.ipynb** file.


Detected Corners
Reprojected Corners
0
200
400
600
800
1000
1200
1400
0
200
400
600
800
1000


Detected Corners
Reprojected Corners
0
200
400
600
800
1000
0
200
400
600
800
1000
1200
1400



(f) Once again, we compute the normals of the plane in the reference frame of the image. We can do the same by using the last column of the rotation matrix, as the plane is entirely represented using 2 vectors, and thus, the third vector is simply normal to the plane. The values can be found in the same file.

5

(a) The normals for all 25 images have been found in the **q5_2021507.ipynb** file attached, using the SVD decomposition for the same.

(b) As per the section 5 of the provided thesis, we can calculate the rotation matrix R and translation vector T as follows:-

$$T = (\theta_c^T \theta_c)^{-1} \theta_c^T (\alpha_c - \alpha_l)$$
$$R = VU^T, \text{ where } USV^T = \theta_l \theta_c^T$$

Here θ_c and θ_l are the normal vectors of all the 25 images of the shape 25×3. Similarly, α_c and α_l are the offset vectors of all the 25 images of the shape 25×1.

We have computed the normals for the lidar scans in the previous part, and we can use the provided translation vector and image normal to calculate the offset for camera images.

The results for the same are :-

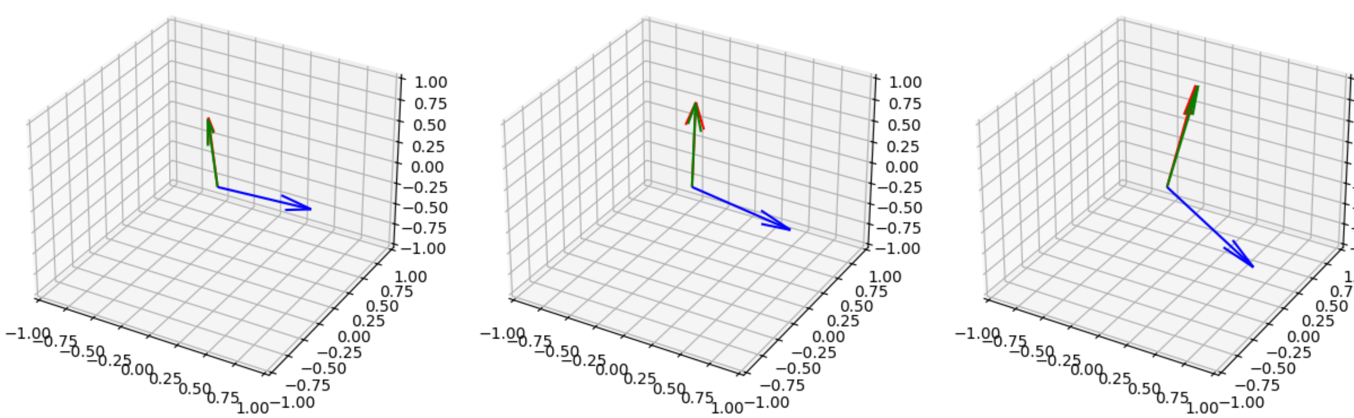
$$R = \begin{bmatrix} -0.168425664 & -0.985714233 & -0.000496095705 \\ 0.0139847753 & -0.00188629595 & -0.999900429 \\ 0.985615149 & -0.168415831 & 0.0141026927 \end{bmatrix}$$

$$T = \begin{bmatrix} 0.04163243 \\ -0.31097725 \\ -0.59496447 \end{bmatrix}$$

(c) The implementation of the function can be found in the attached **q5_2021507.ipynb** file. The determinant of the resultant matrix is indeed 1.

(d) All the checkerboards points are not in the boundary, but are within a small margin of the same for each image. All the images can be found by executing the attached code i the **q5_2021507.ipynb** file. An example is as follows :-

(e) The normal vectors can be plotted as follows (blue represents the normal for lidar plane, while red is the normal for camera plane and green is the projected normal):-



The remaining graphs can be found in the **q5_2021507.ipynb** file. The mean cosine similarity is 0.999339 and the std deviation is 0.00275.