TTK4255 Robotic Vision

Assignment 5

Written Spring 2022 by A. Fanebust and Ø. Solbø

Part 1: The epipolar constraint

Task 1.1)

It is given that $\mathbf{l} = [\cos(\theta), \sin(\theta), -\rho]^T$, and that $\mathbf{x}^T\mathbf{l} = 0$. By multiplying $\mathbf{l}$ with a scalar factor k, one gets $\mathbf{x}^T(k\mathbf{l}) = k(\mathbf{x}^T\mathbf{l}) = 0$. Thus representing the same line.

Given a random line, $\mathbf{l} = [a, b, c]^T$, one could use the properties for $\cos(\cdot)$ and $\sin(\cdot)$, where $k\cos^2(\cdot) + k\sin^2(\cdot) = k$. One could therefore remove the multiplied factor k from $\mathbf{l}$ by using this property. The normalized equations are therefore given as

$$\begin{aligned}\cos(\theta) &= \frac{a}{\sqrt{a^2+b^2}} \\ \sin(\theta) &= \frac{b}{\sqrt{a^2+b^2}} \\ -\rho &= \frac{c}{\sqrt{a^2+b^2}}\end{aligned}$$

Task 1.2)

Using that

$$\begin{aligned}\mathbf{x_a} &= [x_a, y_a, 1]^T \\ \mathbf{x_b} &= [x_b, y_b, 1]^T\end{aligned}$$

the cross-product is given as

$$\mathbf{x_a} \times \mathbf{x_b} = \begin{vmatrix} x & y & z \\ x_a & y_a & 1 \\ x_b & y_b & 1 \end{vmatrix} = \begin{bmatrix} y_a - y_b \\ -x_a + x_b \\ x_a y_b - x_b y_a \end{bmatrix}$$

The points, $\mathbf{x_a}$ and $\mathbf{x_b}$, lies on the calculated line if the set of equations

$$\begin{aligned}(y_a - y_b)x_a + (-x_a + x_b)y_a + (x_a y_b - x_b y_a)c_a = 0 \\ (y_a - y_b)x_b + (-x_a + x_b)y_b + (x_a y_b - x_b y_a)c_b = 0\end{aligned}$$

holds. Under the assumption that $c_a = c_b$, which is valid due to homogenization, it results in the following equation

$$(y_a - y_b)(x_a - x_b) + (-x_a + x_b)(y_a - y_b) + (x_a y_b - x_b y_a)(c_a - c_b) = 0$$

As the equation hold true, it means the points lie on the line calculated using the cross-product above.

Task 1.3)

λ is understood to represent the distance from the camera origin of the camera to an object. This understanding, gives the following limits:

$\lambda = 0$: The point is here fixed at camera 1's origin. The observed point for camera 2, is therefore $x_2 = t$, which is on the epipolar line of camera 1.

$\lambda = \infty$: This results in the point being infinity far away from both cameras. Both cameras will observe the object at a similar camera coordinate, meaning the translation between the cameras has theoretically no impact.

Task 1.4)

From previous, it is known that a line could be calculated using the cross-product. Setting the two points as, $\mathbf{p_0} = \mathbf{t}$ and $\mathbf{p_1} = \lambda \mathbf{R x_1} + \mathbf{t}$, the equation for the line is given as

$$\begin{aligned} \mathbf{l} &= \mathbf{p_0} \times \mathbf{p_1} \\ &= \mathbf{S}(\mathbf{t})(\lambda \mathbf{R x_1} + \mathbf{t}) \\ &= \mathbf{S}(\mathbf{t})(\lambda \mathbf{R x_1}) \\ &= \lambda \mathbf{E x_1} \end{aligned}$$

where $\mathbf{S}(\cdot)$ is the skew-symmetric matrix, and $\mathbf{E}$ is the essential matrix.

Task 1.5)

The $\mathbf{E}$ matrix is only dependant on $\mathbf{R}$ and $\mathbf{t}$, which again is only dependent on the 2 cameras relative position and orientation. If their relative position and orientation remains static, the $\mathbf{E}$ matrix should also continue to be static.

Part 2: The 8-point algorithm

Task 2.1)

The corresponding points with the epiporal lines is shown in figure 1. One can see that the epiporal lines matches reasonable well. No large discrepancy is observed.

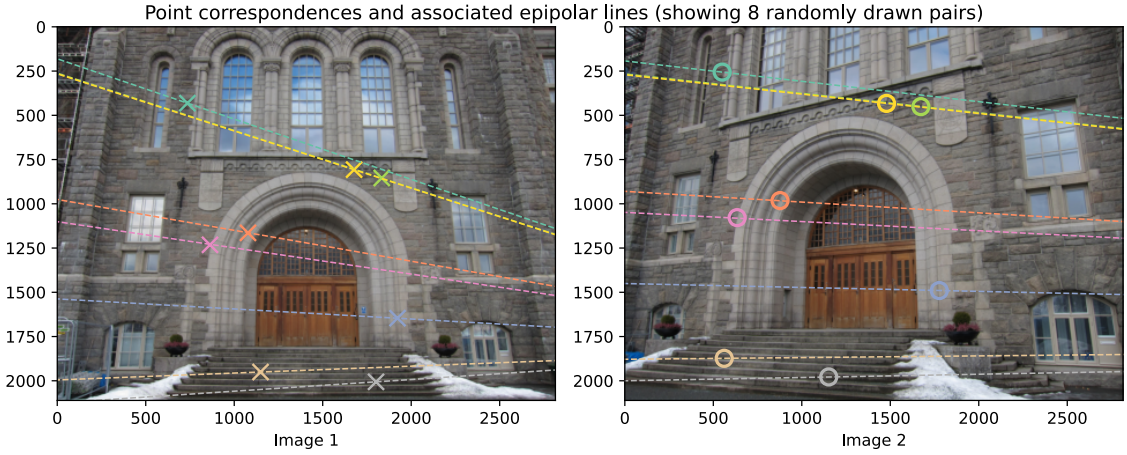

Figure 1: Point correspondences and epiporal lines for task 2.1).

Part 3:

Task 3.1)

As a theoretical example, it is postulated that the cameras are positioned 1 unit apart in x-direction, and have the same orientation. The matrix, $\mathbf{A}$, given in equation (9) in the assignment, is therefore found to be

$$\mathbf{A} = \begin{bmatrix} -1 & 0 & x_1 & 0 \\ 0 & -1 & y_1 & 0 \\ -1 & 0 & x_2 & -1 \\ 0 & -1 & y_2 & 0 \end{bmatrix},$$

where x_i, y_i are the pixel-coordinates for camera i. It is theorized that the point/object moves on a the ray from camera 1, such that x_1, y_1 are constant. Based on the matrix and the object's

movement, only x_2 will change. When the object is ∞ units away from the cameras, x_2 will have converged. However, as the distance has increased to ∞ units, it is required for W to be nonzero.

Task 3.2)

The correspondences between the images are shown in figure 2. It was developed with the world frame fixed to camera 1, and such only the projection matrix for camera 2 was approximated.

Both of the cameras are angled upwards compared to the ground. This causes the building to angle backwards in figure 2.

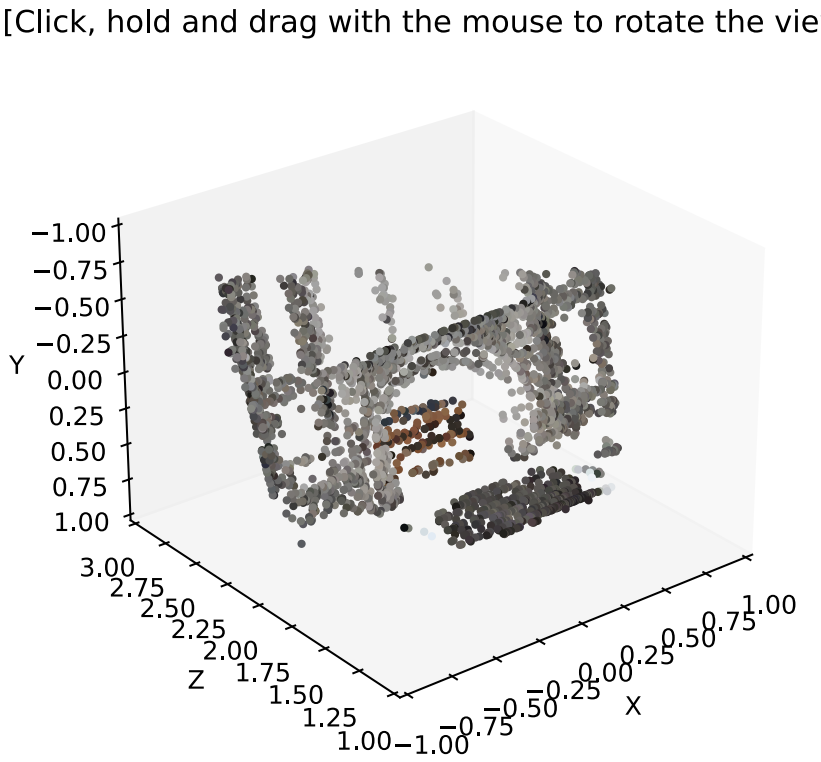

Figure 2: The 3D point-cloud generated on the correspondences for task 3.2).

Part 4:

Task 4.1)

The histograms showing the residual errors, are shown in figure 3. One can see that the residual errors are quite large, where it in some cases exceed several thousand. This implies the eight-point algorithm is not sufficient when the dataset also contains outlying correspondences.

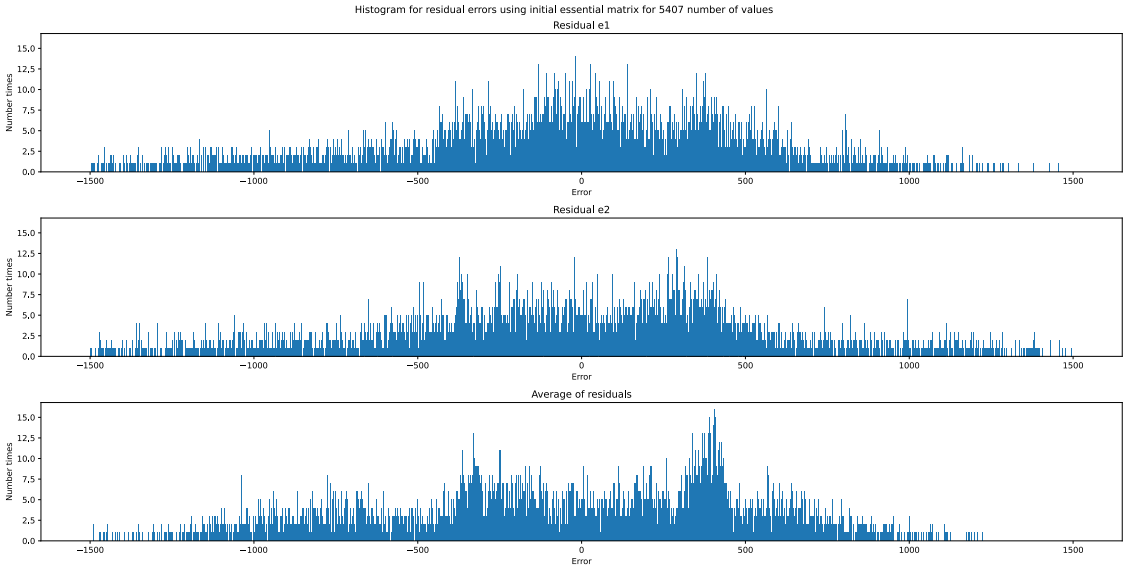

Figure 3: The histogram generated on the task4-matches data. The residuals are shown on the horizontal axis, with number of occurrences along the vertical axis.

Task 4.2)

As RANSAC searches randomly, there are multiple different sets of inliers that could be detected. Three examples are shown in figures 4 to 6 which have 1535, 1706 and 2174 number of inliers respectively.

[Click, hold and drag with the mouse to rotate the view]

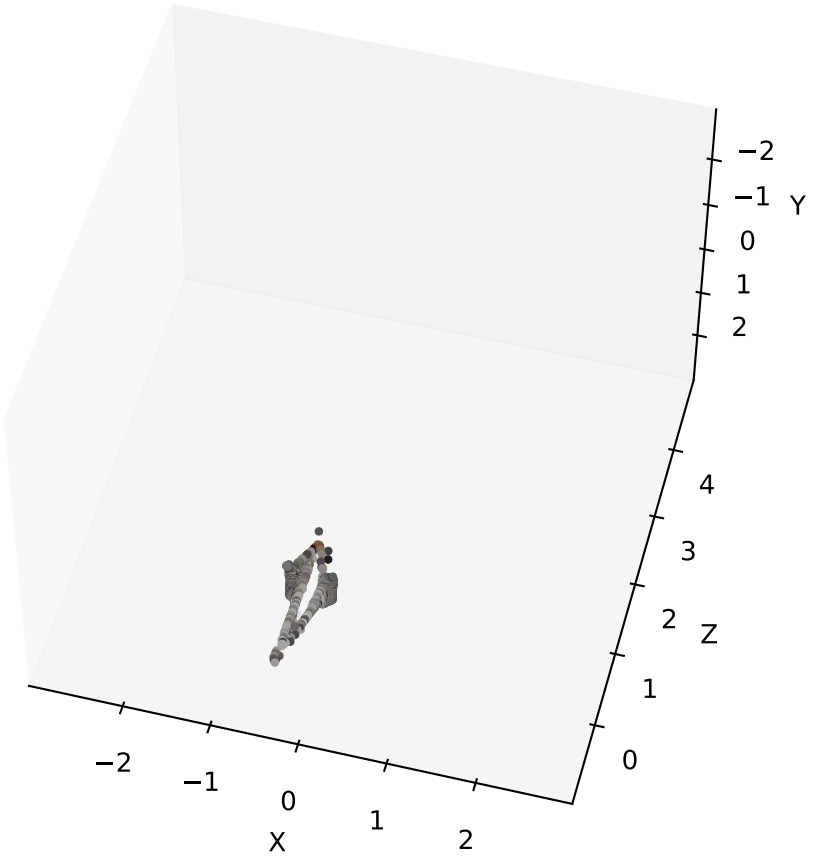

Figure 4: Detected inlier points, where RANSAC detected 1535 inliers.

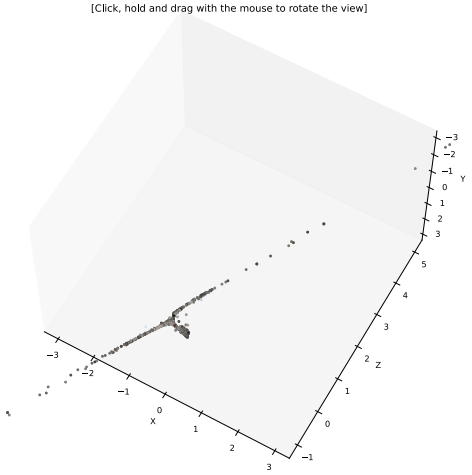

Figure 5: Detected inlier points, where RANSAC detected 1706 inliers.

[Click, hold and drag with the mouse to rotate the view]

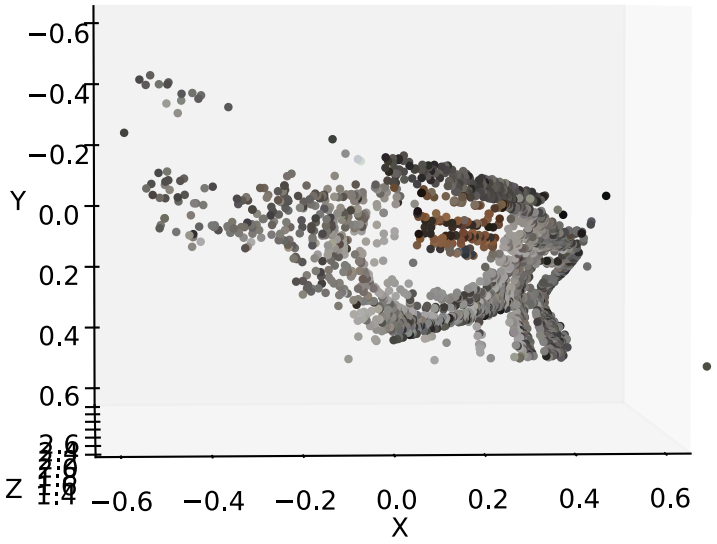

Figure 6: Detected inlier points, where RANSAC detected 2174 inliers.

Task 4.3)

During the development of RANSAC, it was assumed to average the residuals.

It is hypothesised the averaging of e1 and e2 could produce outliers. For example if their respective magnitudes are outside of the accepted threshold, but e1 and e2 have opposite signs, they might cancel each other out in the average process. To eliminate this issue, the norm could be used instead of a simple average.

Task 4.4)

Using equation (6.30) in [1, p. 319], it was found to require minimum 1177 iterations to obtain a probability of 99 % for success. However, no comparatively difference was found when running the modified algorithm. It converged onto clearly suboptimal point-correspondences multiple times. An example of this, is shown in figure 7, which is clearly worse compared to 6. As such it would be an overstatement to expect it to find the largest inlier set 99 % of the time. This is substantiated by the use of a random function. It is well known that a computer is unable to generate a truly

random number, and as such the algorithm would be unable to guarantee the desired probability of success.

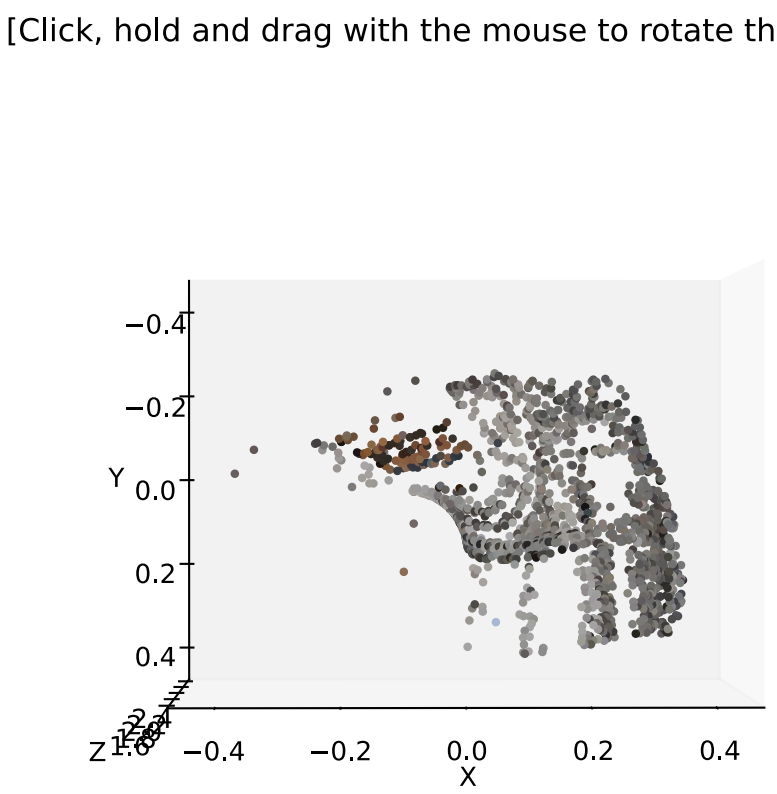

Figure 7: Detected inlier points, where RANSAC detected 1592 inliers.

[Click, hold and drag with the mouse to rotate the view]

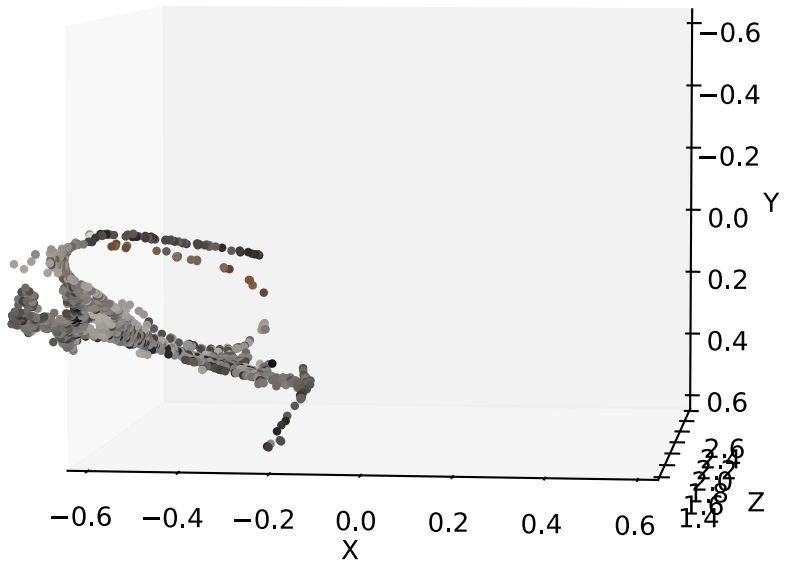

Figure 8: Detected inlier points, where RANSAC detected 1086 inliers.

References

[1] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2010.