Computer Vision: Ex3

Or Tal

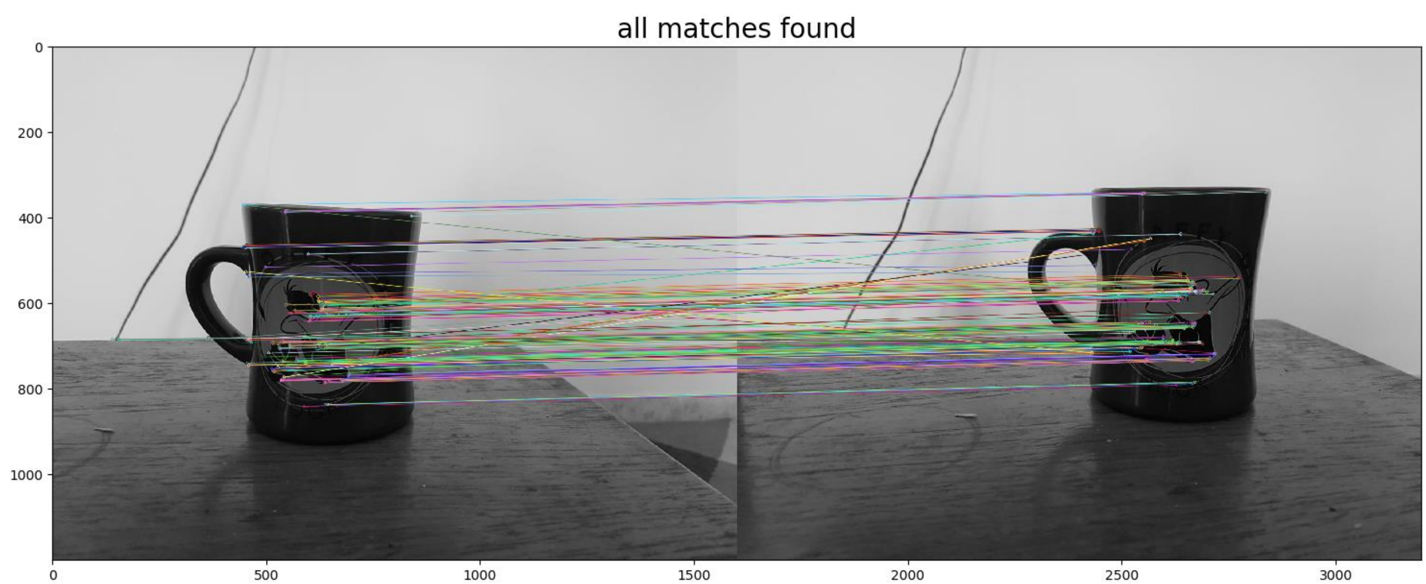
Part one

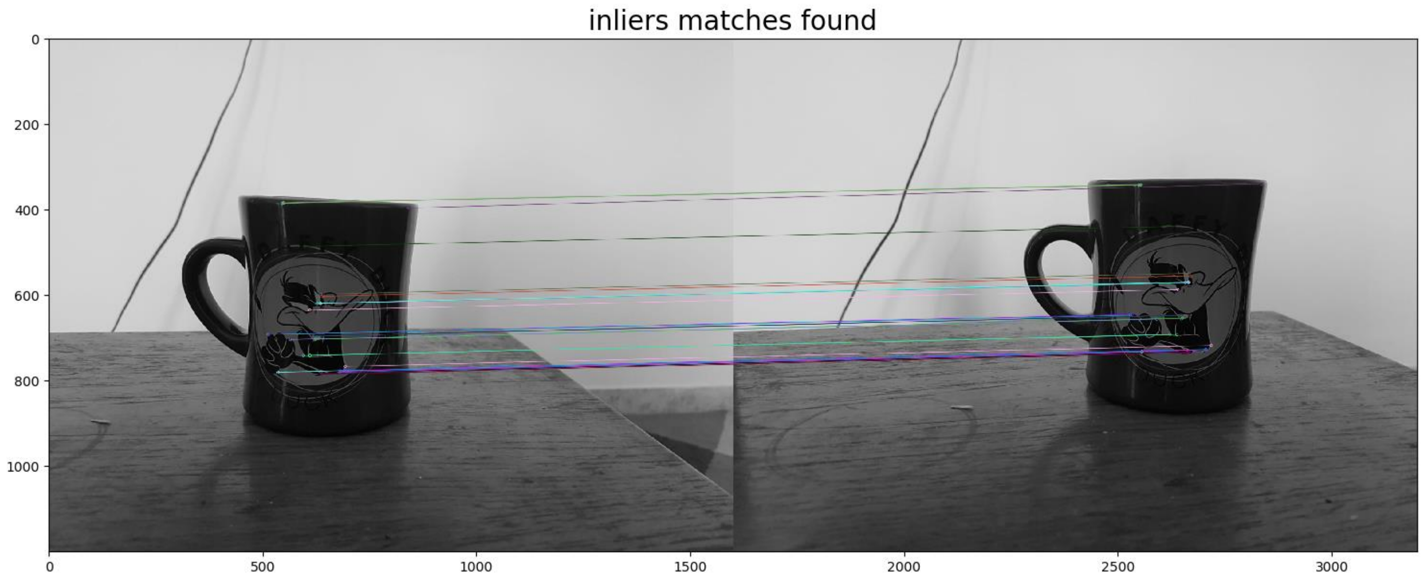
Compute the fundamental matrix using normalized 8 points algorithm, and draw the epipolar lines

Note: please see attached code files and sources,

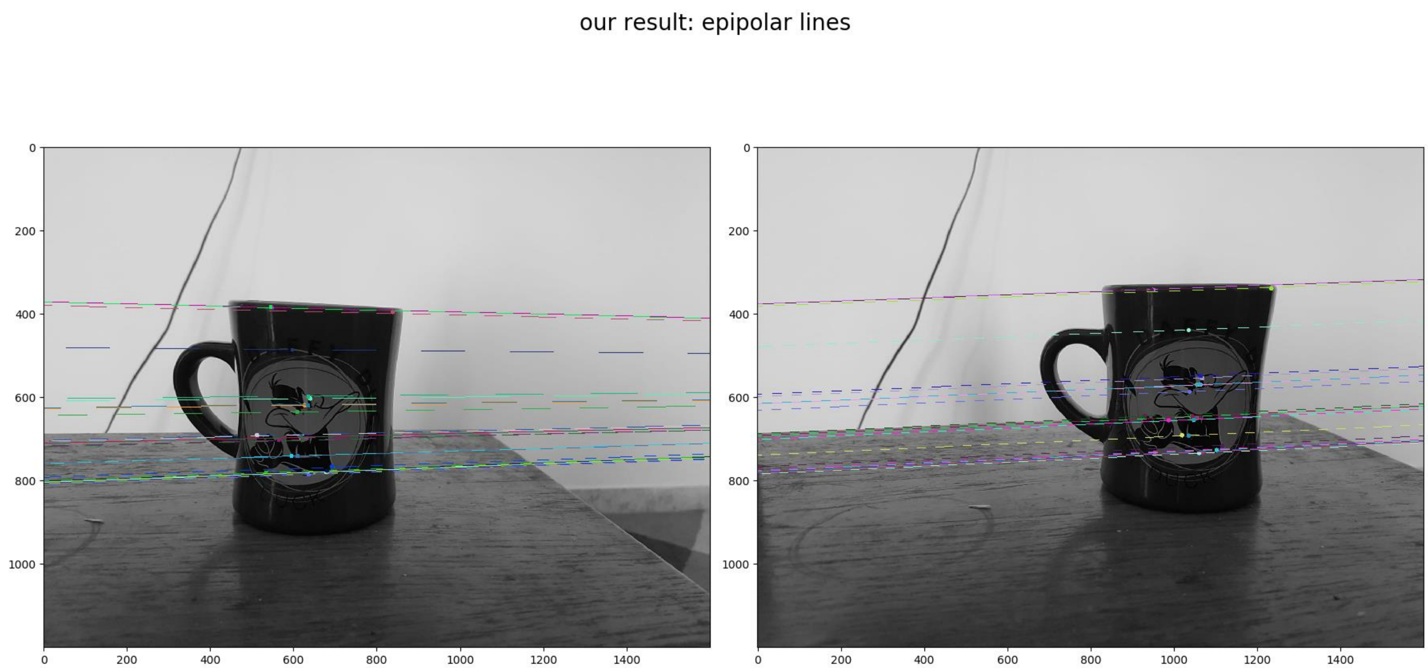
after unpacking simply run from shell: python3 eight_point_alg.py

input images:





inliers matches found



our result: epipolar lines

Part two

1. Show how the linear triangulation method extends to $n > 2$ images

The linear triangulation works for two images, hence in case of $n > 2$ images it should be about the same for the following point correspondence $\boldsymbol{x_1},\cdots,\boldsymbol{x_n}$ (indexed by image) with $P_1,\cdots,P_n$ according camera matrices we could then describe the matrix A as following:

$$for\ \boldsymbol{x_i} = (x_i, y_i)^T \rightarrow A = \begin{bmatrix} x_1\boldsymbol{p}_1^{3T} - \boldsymbol{p}_1^{1T} \\ y_1\boldsymbol{p}_1^{3T} - \boldsymbol{p}_1^{2T} \\ x_2\boldsymbol{p}_2^{3T} - \boldsymbol{p}_2^{1T} \\ y_2\boldsymbol{p}_2^{3T} - \boldsymbol{p}_2^{2T} \\ \vdots \\ x_n\boldsymbol{p}_n^{3T} - \boldsymbol{p}_n^{1T} \\ y_n\boldsymbol{p}_n^{3T} - \boldsymbol{p}_n^{2T} \end{bmatrix}_{2n\times 4}$$

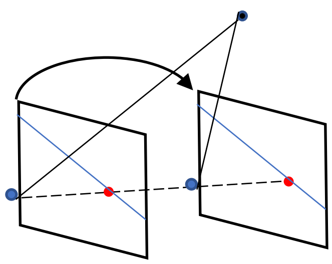
where $\boldsymbol{p}_i^{jT}$ is the j'th row of the i'th camera matrix.

The 3D homogeneous point $\boldsymbol{X}$ is the one satisfying $A\boldsymbol{X} = \boldsymbol{0}$

One way to find $\boldsymbol{X}$ would be computed by taking the last vector of V matrix in the SVD of A: $A = UDV^T$

2. Derive a method for triangulation in the case of pure translational motion of the cameras

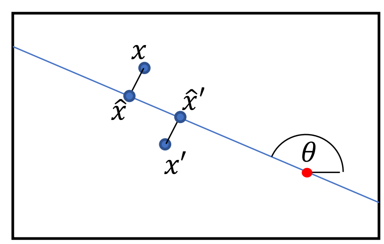
We had previously seen that in case of pure translation, the fundamental matrix would be $F = [e']_\times$



Due to that, we may conclude that the epipolar lines which correspond to the same point $\boldsymbol{X}$ in the world frame, should be the same, as the epipole remains at the same.

We may then conclude that we may parametrize the epipolar line by its relevant angle θ where the epipole is its axis, and hence calculate the min geometric distance

$$c(x, x') = d\big(x, l(\theta)\big)^2 + d(x', l(\theta))$$



After doing so, we would estimate $\hat{x}, \hat{x}'$ matching x, x' on the epipolar line we just found, hence assuming the translation is accurate and the only distortion is in the measurements, the lines from camera centers via $\hat{x}, \hat{x}'$ would then lie on a single plane, therefor they would intersect at an estimated 3D point $\widehat{\boldsymbol{X}}$

Purposed method for the triangulation in case of pure motion: (this assumes that the translation is known)

1. Assume that the first camera is positioned at the origin, and the second camera is a pure translation of it. Find the epipole in the image intersecting the line between camera centers (the translation) with the image plane. The 3D line would be represented by 4 parameters matching $ax + by + cz + d = 0$, and in this case, as it passes through the origin, we may assume that $d = 0$, moreover, we would know that $z = 1$ on the intersection plane, we could then have two constraints: $\begin{cases} ax + by + c = 0 \\ 1 + x^2 + y^2 = (ax)^2 + (by)^2 + c^2 \end{cases}$
   Solving these will define the epipole in the image frame.
2. For the corresponding pts x, x',find the line $l(\theta)$ that minimizes $d(x, l)^2 + d(x', l)^2$

The line equation would match: $tan\theta \cdot x - y + (e_y - e_x \cdot tan\theta)$; where $\boldsymbol{e} = \begin{bmatrix} e_x \\ e_y \\ 1 \end{bmatrix}$ is the epipole

Hence: $\boldsymbol{l} = \left(tan\theta, -1, (e_y - e_x \cdot tan\theta)\right)^T = (a, b, c)^T$

We could then compute the distance for $\boldsymbol{x} = (x_0, y_0, 1) \Rightarrow d(\boldsymbol{x}, \boldsymbol{l}(\theta))^2 = \frac{(l^T x)^2}{a^2+b^2}$ (same for $\boldsymbol{x}' = (x_1, y_1, 1)^T$)

So, we would need to solve the minimization problem: $\min_{\theta}\{d(\boldsymbol{x}, \boldsymbol{l}(\theta))^2 + d(\boldsymbol{x}', \boldsymbol{l}(\theta))^2\}$

A closed formula for this would be: (calculated using wolfram alpha for simplicity)

$$\theta \approx \tan^{-1}\left(\frac{x_0(y_0 - e_y) - e_x(y_0 + y_1) + x_1(y_1 - e_y) + 2e_x e_y}{x_0^2 - 2e_x(x_0 + x_1) + x_1^2 + 2e_x^2}\right) + \pi k; \;\; k \in \mathbb{Z}$$

3. $\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{x}}'$ could now be estimated, for $\widehat{\boldsymbol{x}} = (x, y)^T, \boldsymbol{x} = (x_0, y_0)^T, \boldsymbol{l} = (a, b, c)^T$: (same for $\widehat{\boldsymbol{x}}'$; $\boldsymbol{x}' = (x_1, y_1)^T$)

$$x = \frac{b(bx_0 - ay_0) - ac}{a^2 + b^2}, y = \frac{a(-bx_0 + ay_0) - bc}{a^2 + b^2}, z = 1$$

Shifting $\widehat{\boldsymbol{x}}'$ by the translation **t**, we could calculate the estimated world point $\widehat{\boldsymbol{X}}$ by building two 3D line equations, between the camera centers and $\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{x}}' + \boldsymbol{t}$ accordingly, and then calculate $\widehat{\boldsymbol{X}}$ by their intersection.