ECE 543 MP2

Nick, yyang192

University of Illinois at Urbana-Champaign

4/5/2021

Q1

3. **Putative Matches**

4. **Homography Estimation and RANSAC**

To implement RANSAC to estimate a homography mapping, we first set up the threshold and iteration number. The parameters I used are threshold = 0.9, iteration = 1000. In each iteration, we randomly choose four matches from all putative matches. And then we compute the homography from the stacked matches. After the SVD, we will get a homograph matrix. Then we calculate the inlinears based on the homography matrix and putative matches. In my implementation, if the residual of the match is less than 5, I will take the match as a inlinear. If the number of inlinears is more than threshold * number of all matches, break the loop and return the parameters.

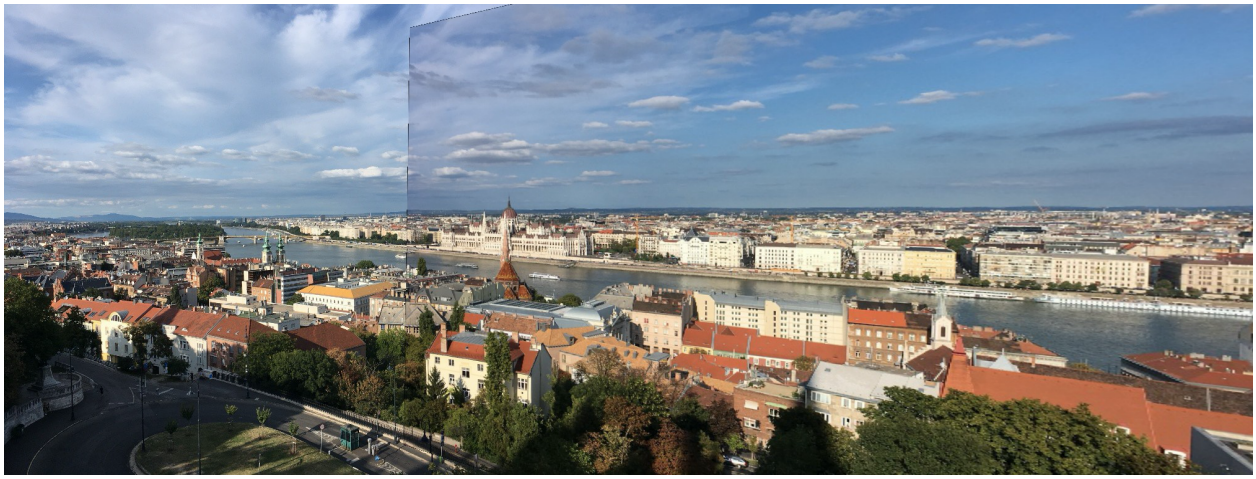
Best model:

[[3.14229528e+00 3.53304669e-02 -2.06759761e+03]
 [7.37889790e-01 2.70748120e+00 -5.96222353e+02]
 [2.16315396e-03 -4.52639160e-05 1.00000000e+00]]

Average residual: 41.02459338565225

Inliers: 375

5. **Image Warping**



Q2

1. **Fundamental Matrix Estimation**

Unnormalized method:

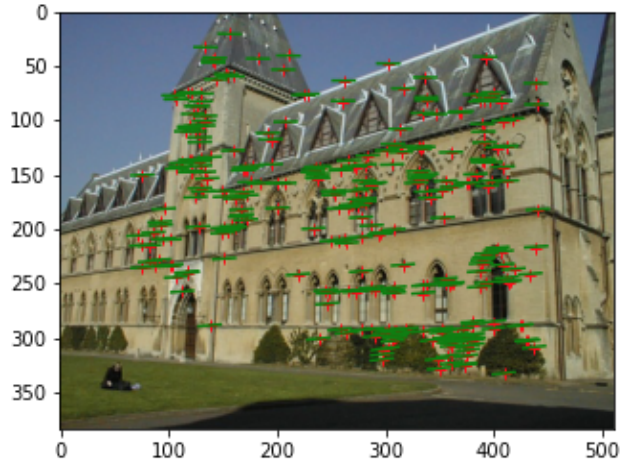
Estimated fundamental matrix:

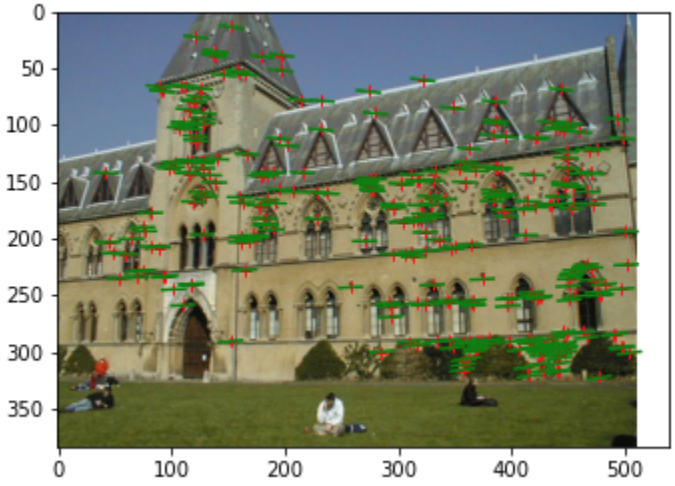
[[-1.32341616e-06 1.36640519e-05 -6.82803870e-04]
 [-2.88178174e-05 2.66440807e-07 4.09069255e-02]
 [5.62362952e-03 -3.72771609e-02 -9.98451273e-01]]

residual in frame 2: 0.17921336680315691

residual in frame 1: 0.14912309938770843

residual combined : 0.16416823309543266





Normalized method:

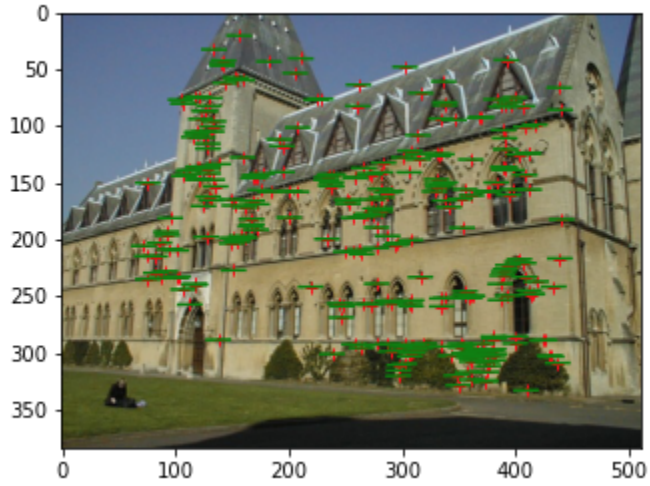
Estimated fundamental matrix:

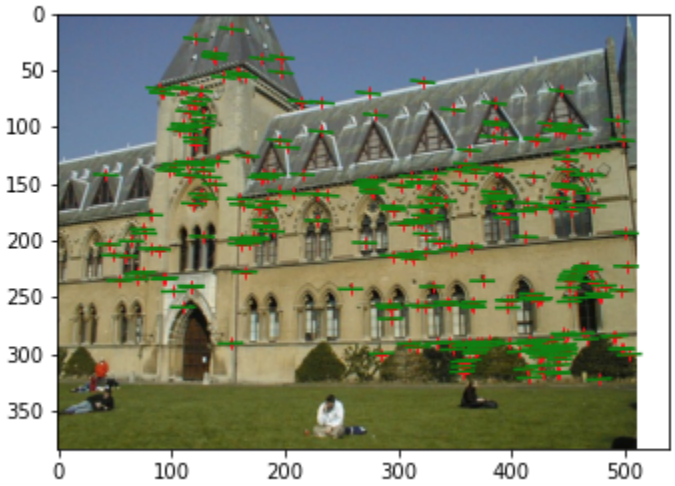
[[-3.44725739e-08 7.27167745e-07 -1.09292791e-04]
 [-4.37299224e-06 -4.44216115e-08 8.10999749e-03]
 [1.04291060e-03 -7.28410119e-03 -1.97254324e-01]]

residual in frame 2: 0.060251052582048846

residual in frame 1: 0.05481363409220714

residual combined: 0.05753234333712799





2. **Camera Calibration**

lab 1 camera projection
[[3.09963996e-03 1.46204548e-04 -4.48497465e-04 -9.78930678e-01]
 [3.07018252e-04 6.37193664e-04 -2.77356178e-03 -2.04144405e-01]
 [1.67933533e-06 2.74767684e-06 -6.83964827e-07 -1.32882928e-03]]
residual in lab1: 13.545832902721848

lab 2 camera projection
[[6.93154686e-03 -4.01684470e-03 -1.32602928e-03 -8.26700554e-01]
 [1.54768732e-03 1.02452760e-03 -7.27440714e-03 -5.62523256e-01]
 [7.60946050e-06 3.70953989e-06 -1.90203244e-06 -3.38807712e-03]]
residual in lab2: 15.544953451380152

library1 camera projection
[[-4.5250208e+01 4.8215478e+02 4.0948922e+02 3.4440464e+03]
 [4.8858466e+02 2.7346374e+02 -1.3977268e+02 4.8030231e+03]
 [-1.9787463e-01 8.8042214e-01 -4.3093212e-01 2.8032556e+01]]

library2 camera projection
[[-5.9593834e+01 5.5643970e+02 2.3093716e+02 3.5683545e+03]
 [4.6419679e+02 2.2628430e+02 -1.9605278e+02 4.8734171e+03]
 [-1.9116708e-01 7.2057697e-01 -6.6650130e-01 2.8015392e+01]]

3. **Camera Centers**

lab1 camera center [305.83276769 304.20103826 30.13699243]
lab2 camera center [303.10003925 307.18428016 30.42166874]
library1 camera center [7.28863053 -21.52118112 17.73503585]
library2 camera center [6.89405488 -15.39232716 23.41498687]

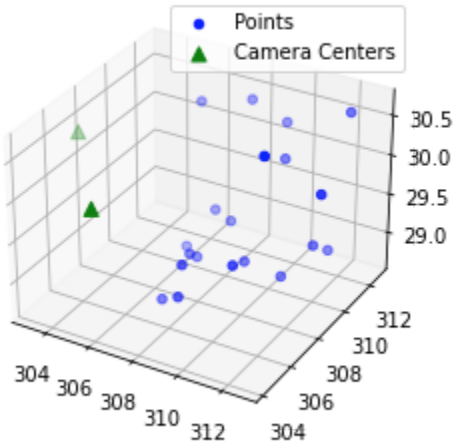
4. **Triangulation**

Mean 3D reconstruction error for the lab data: 0.02246

The Mean error here is the average euclidean distance between reconstructed 3d coordinates and real-world 3d coordinates.

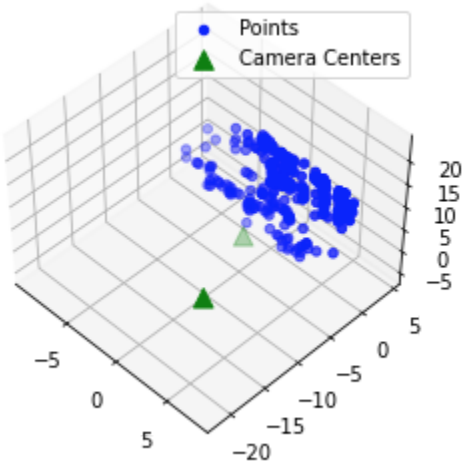
2D reprojection error for the lab 1 data: 24.31825304256491

2D reprojection error for the lab 2 data: 3.9232806980109385



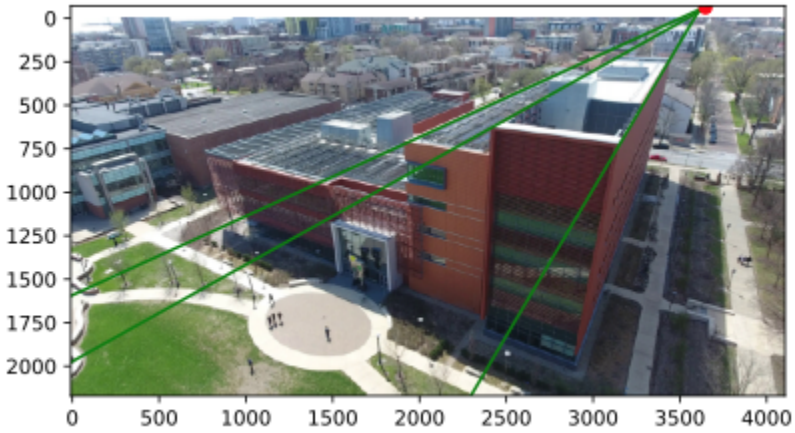
2D reprojection error for the library 1 data: 51.532271221300746

2D reprojection error for the library 2 data: 81.28132134991958

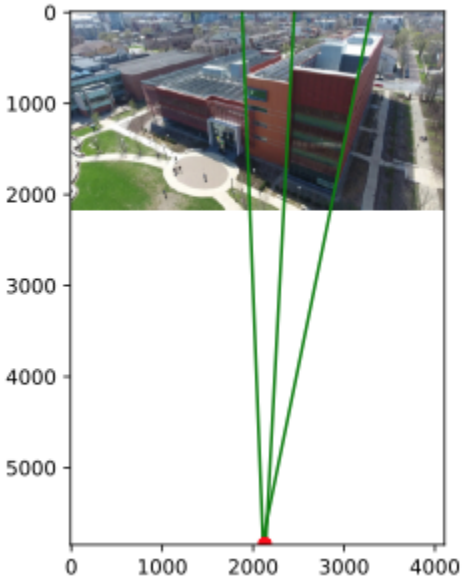


Q3

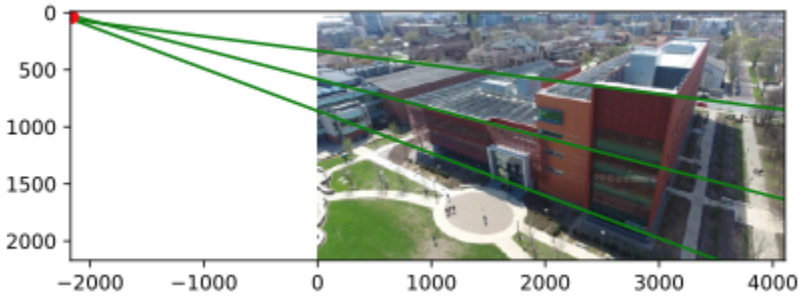
1. **Vanishing Points**



[3.64725304e+03 -6.19815607e+01 1.00000000e+00]



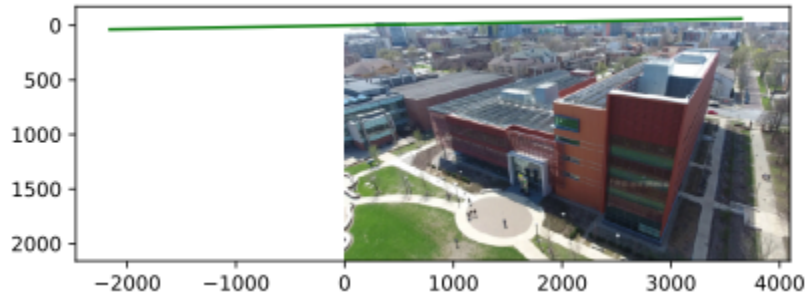
[2.12821831e+03 5.83624632e+03 1.00000000e+00]



[-2.16018606e+03 3.67271545e+01 1.00000000e+00]

2. **Horizon**

[a, b, c] = [-0.01699449 -0.99985558 0.01059219]



3. **Camera Calibration**

To calculate the camera parameters, we use the functions from sympy. First, since we have three vanishing points, we then set three equations:

vpt1.T * K.Inverse.T * K.Inverse *vpt2 = 0

vpt1.T * K.Inverse.T * K.Inverse *vpt3 = 0

vpt2.T * K.Inverse.T * K.Inverse *vpt3 = 0

Based on three equations, we could solve the transformation matrix by sympy.solve().

focal length = -2334.13539207586

principal point = (2048.06544224965, 1120.52389673485)

4. **Rotation Matrix**

[[0.853122   -0.01523129 -0.52148908]

 [0.21971378 -0.89611959 0.3856106]

 [0.47318993 0.44355122 0.76115281]]