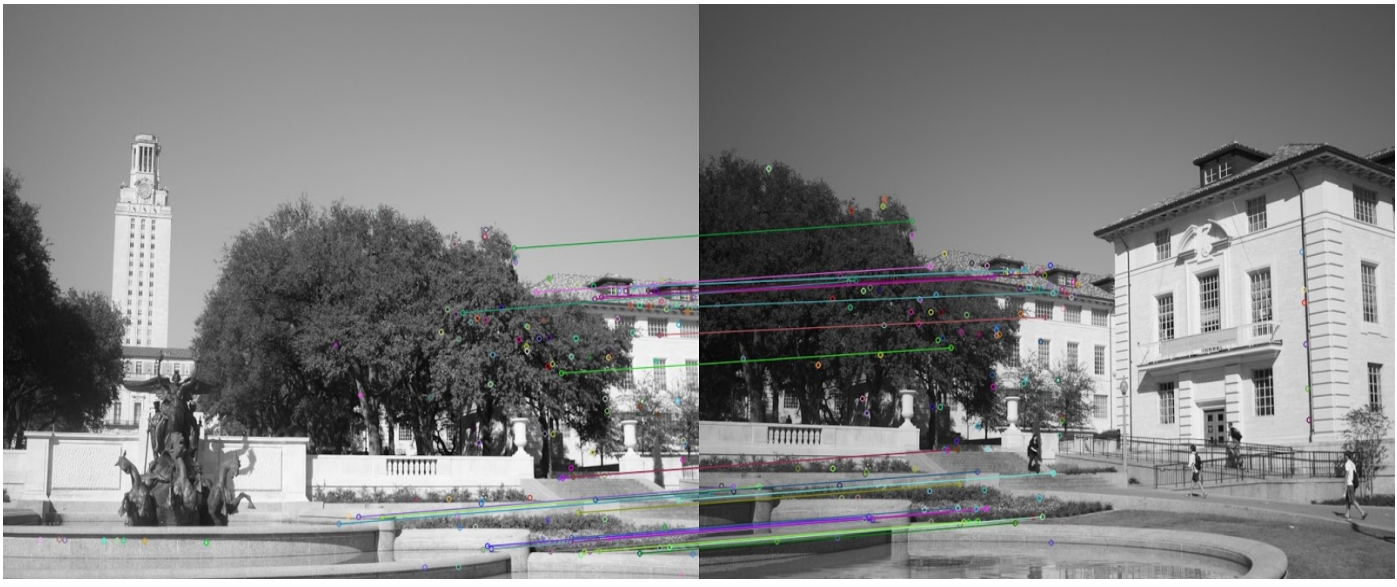
Grading Check List:

PART 1:

(a) Solution description: First, I load the image using cv2.imread, then use cv2.xfeatures2d to extract features in both images. After getting features, I get matching pairs by distance threshold of 5000, which reduces matches to around 150 pairs. As for RANSAC, I use random.sample() in python to randomly sample 4 pairs to build H matrix, and the use this matrix to get projection residuals for these 4 pairs. If average error is less than 3, I transform all pairs by this H matrix and calculate inliers by the standard that "error is less than 1". If the inlier ratio is above 0.5, I accept this H matrix, else, sample H matrix again. Usually this will take 5-10 seconds. **Noted** that by re-fitting H matrix with all inliers will actually make my results worse, because the H matrix in my code is of very high standard so it's just fine to use the H matrix by 4 pairs only. After getting my final H, I use cv2.warpPerspective to project one image and normalize the projected image. After adding them together, I divide pixel values by 2 where there are overlapping.

(b) There are 82 inliers and residual is 0.34.

```
part1|master⚡  ⇒  python3 part1.py
Detecting POI......
Finding Optimal Match......
Total inliers: 82
Average Residual is: 0.3351365737473595
Match Found!
```



(c)

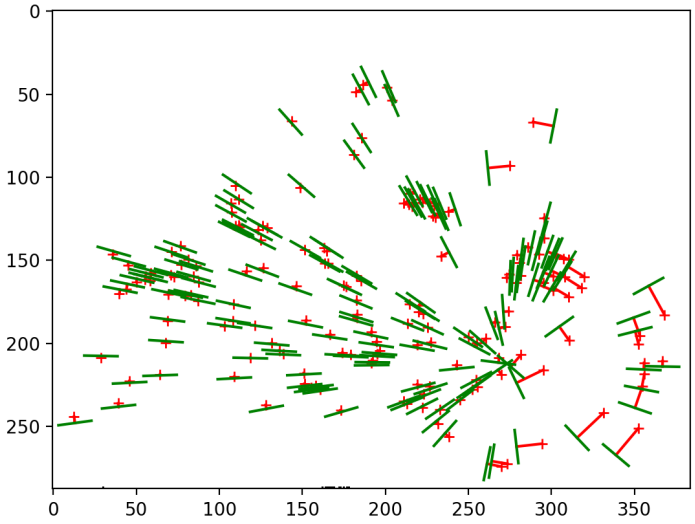
PART 2:

(a)

(1) Unnormalized with ground truth.

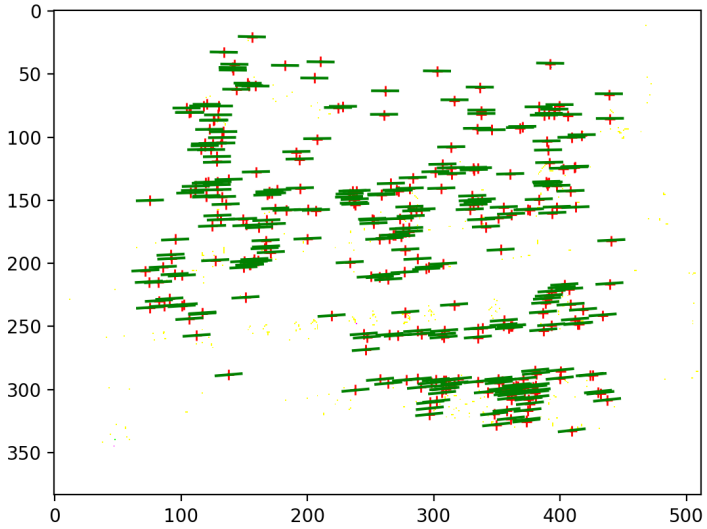
Mean squared error is 26.4 for house.

```
part2|master⚡ ⇒ python3 part2_1.py
26.401303464140256
```



Mean squared error is 0.21 for library.

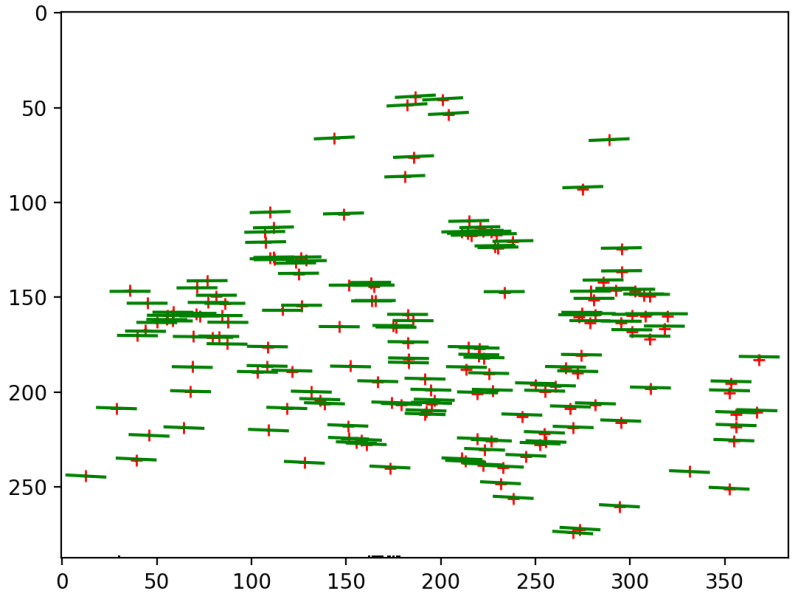
```
part2|master⚡ ⇒ python3 part2_1.py
0.21159591985251422
```



(2) Normalized with ground truth.

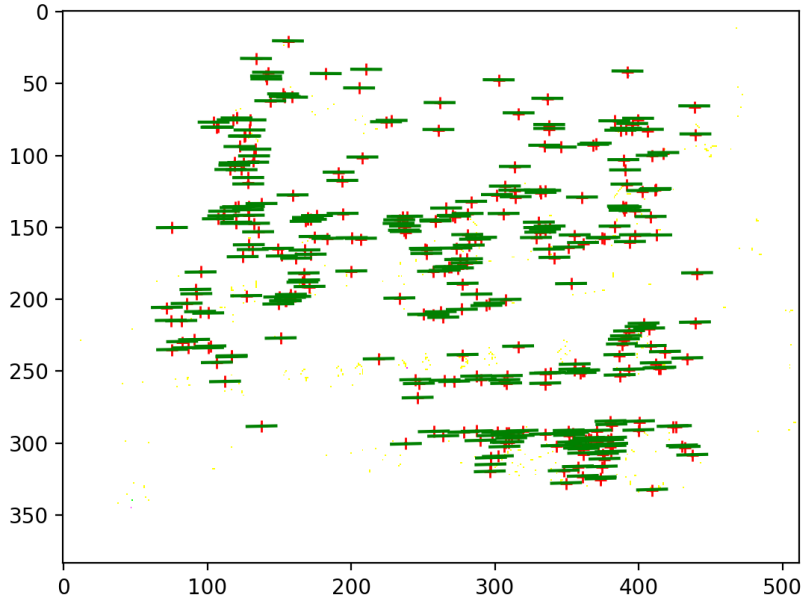
Mean squared error is 0.349 for house.

```
part2|master⚡ ⇒ python3 part2_1.py
0.3494407368695804
```



Mean squared error is 0.179 for library.

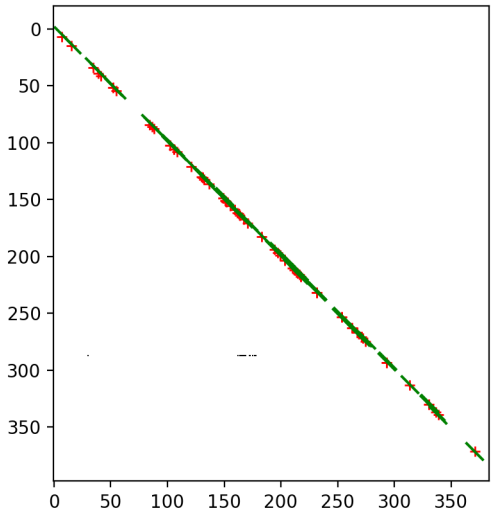
```
part2|master⚡ ⇒ python3 part2_1.py
0.17921336681436997
```



(b) Normalized RANSAC.

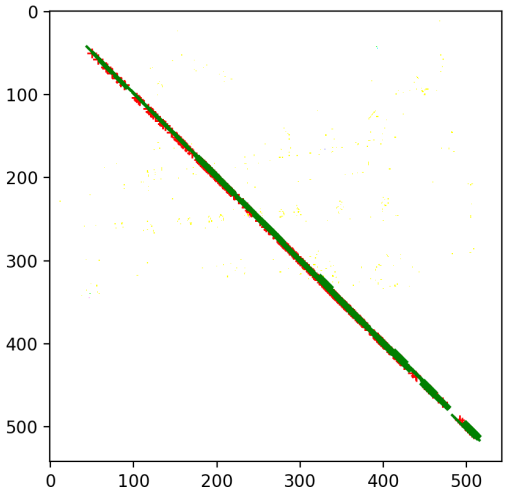
Mean squared error is 1.51 for house, with 68 inliers.

```
part2|master⚡ ⇒ python3 part2_1.py
Detecting POI......
Finding Optimal Match......
68
Match Found!
0.18522347797728456
```

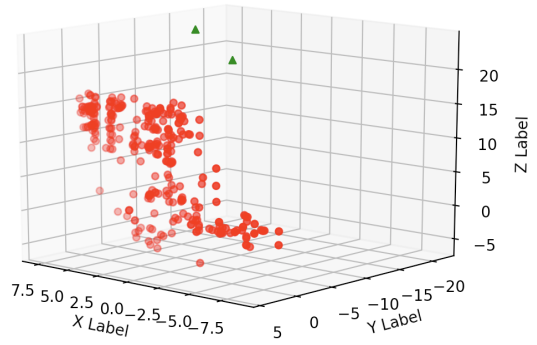


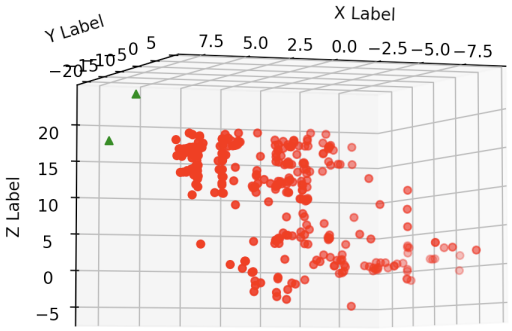
Mean squared error is 1.789 for library, with 579 inliers.

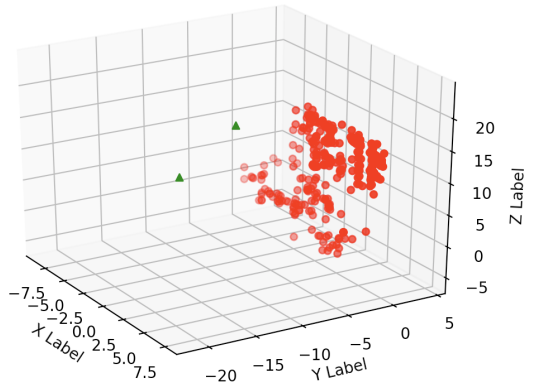
```
part2|master⚡ ⇒ python3 part2_1.py
Detecting POI......
Finding Optimal Match......
579
Match Found!
1.7535991151520105
```

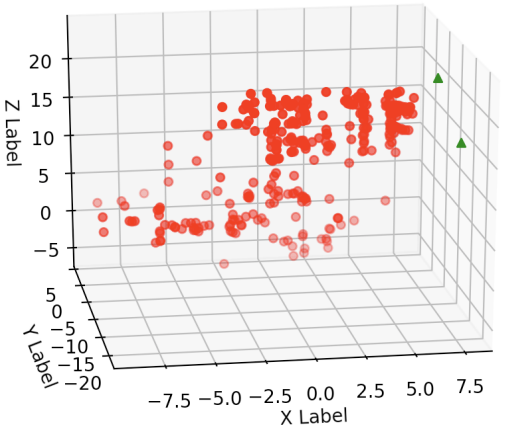


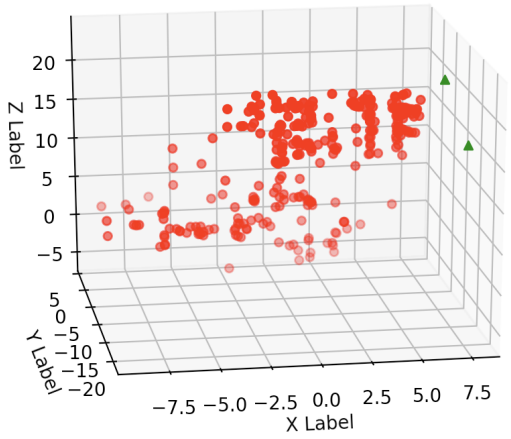
(c) **Visualization of library: (Green triangle is camera locations)**

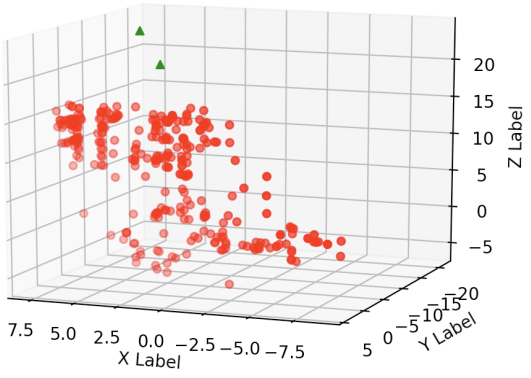




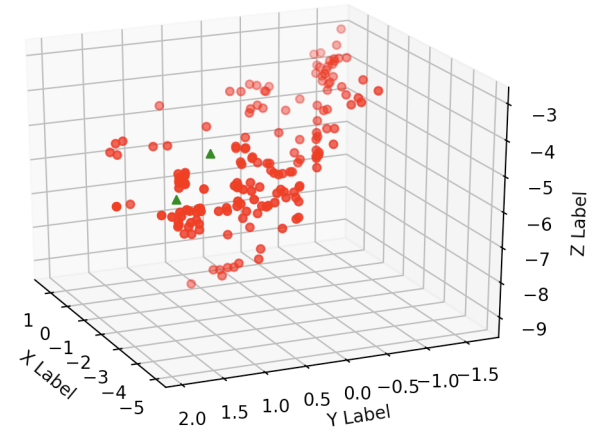


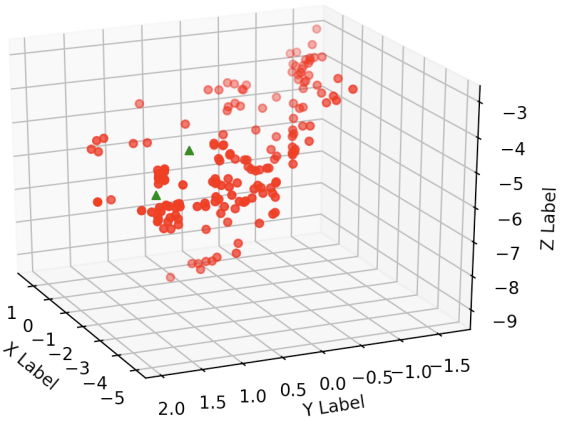


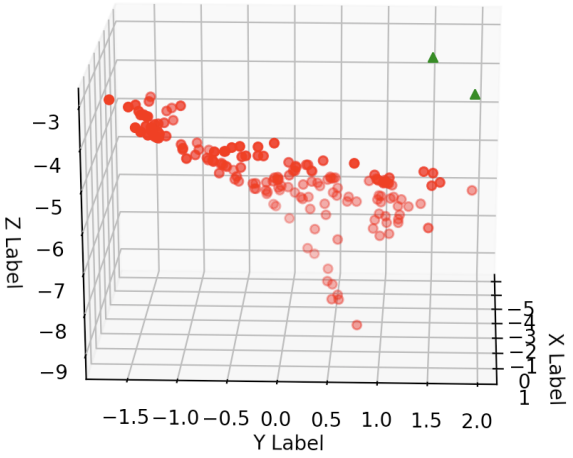


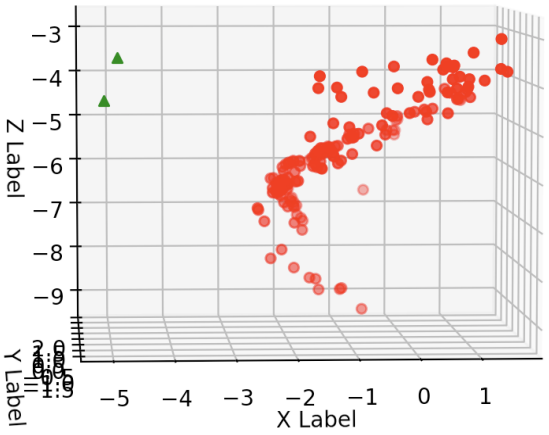


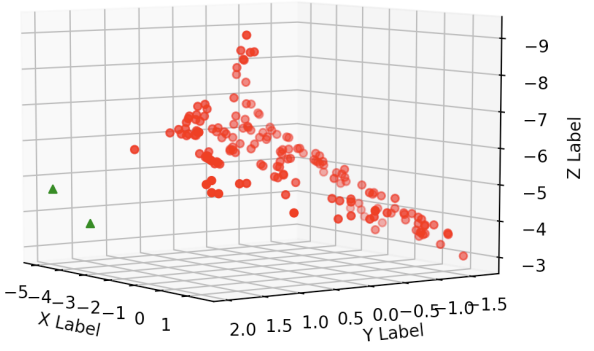
Visualization of house: (Green triangle is camera locations)

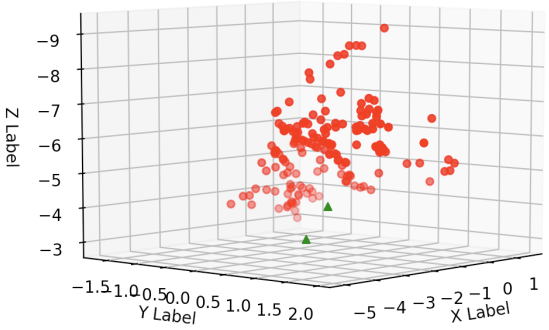












PART 3:

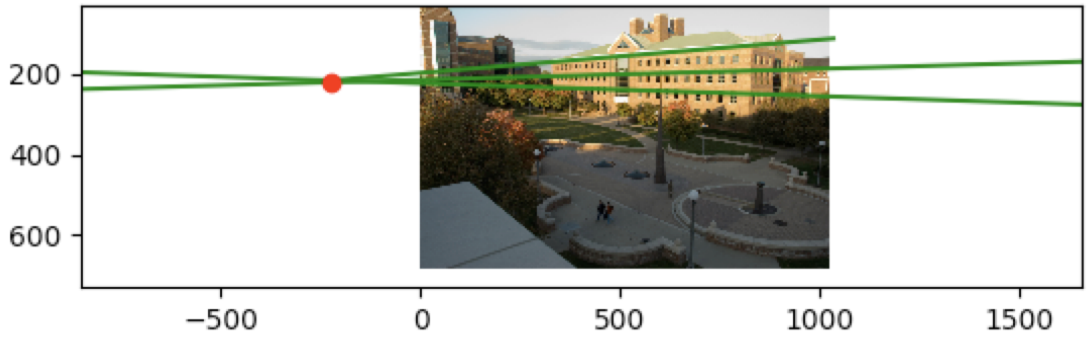
Output overview:

```
vpts:  [[-2.24633188e+02  1.28830140e+03  4.23012805e+02]
 [ 2.18873639e+02  2.32355986e+02  4.94691715e+03]
 [ 1.00000000e+00  1.00000000e+00  1.00000000e+00]]
Horizon line is:  [-8.91103396e-03  9.99960296e-01 -2.20866663e+02]
f, u, v:  743.8905043343768 464.01990434524663 345.26582037334356
R is:  [[ 9.90748275e-01 -8.91103396e-03 -9.83571297e-01]
 [-1.35712404e-01  9.99960296e-01 -1.80520095e-01]
 [ 1.61576709e-06  2.92119153e-07  1.91997834e-06]]
```

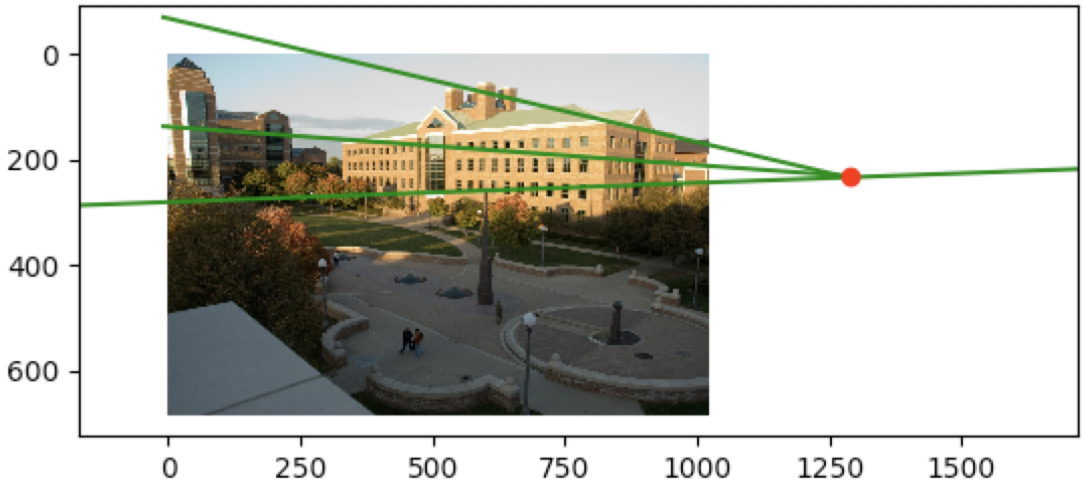
```
Estimating height of CSL building
H/R 0.057676423278618716
CSL building is 1144 inches, which is 95'4
Estimating height of the spike statue
H/R 0.16487728369910698
the spike statue is 400 inches, which is 33'4
Estimating height of the lamp posts
H/R 0.40931794184537645
the lamp posts is 161 inches, which is 13'5
```

1. Noted than the third one is rotated for illustration purpose.

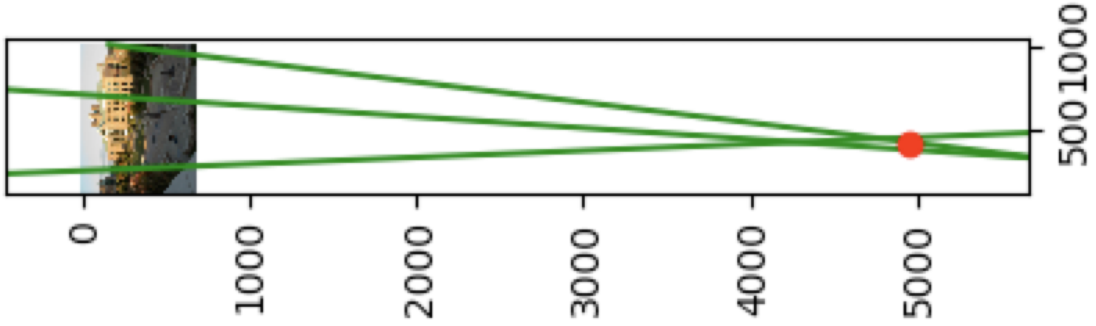
(-224, 218)



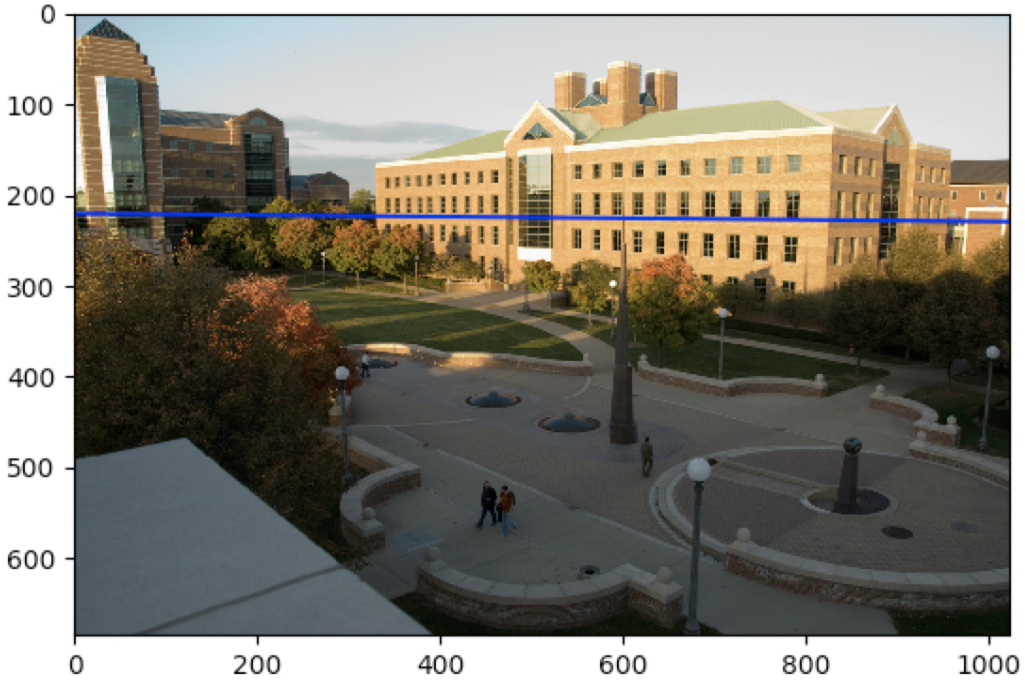
(1288, 232)



(423, 4946)



(a, b, c) = (`-8.91103396e-03  9.99960296e-01 -2.20866663e+02`)



2. `f, u, v:  743.8905043343768 464.01990434524663 345.26582037334356`

```
def get_camera_parameters(vpts):
    [xi, yi, _], [xj, yj, _], [xk, yk, _] = vpts[:,0]/vpts[:,0][-1],
vpts[:,1]/vpts[:,1][-1],vpts[:,2]/ vpts[:,2][-1]
    c11, c12, c13 = xi*xj + yi*yj, xi + xj, yi + yj
    c21, c22, c23 = xi*xk + yi*yk, xi + xk, yi + yk
    c31, c32, c33 = xk*xj + yk*yj, xk + xj, yk + yj

    eq = np.append((np.array([c11, c12, c13]) - np.array([c21, c22,
c23])).reshape(1,3),
            (np.array([c21, c22, c23]) - np.array([c31, c32,
c33])).reshape(1,3), axis=0)

    A, b = eq[:,1:], eq[:,0]
    u, v = nl.solve(A, b)
    f = np.sqrt(c12*u + c13*v - c11 - u*u - v*v)
    return f, u, v
```

Assuming K matrix is [[f,0,u],[0,f,v],[0,0,1]], so for three vanishing points we can get (2C3) = 3 equations satisfying v_i^T*K^-T*K^-1*v_j = 0. Since we have 3 unknowns f, u, v, we can solve three equations to get f, u, v. Be noted that the f here is not actually focal length, but focal length * alpha, where alpha is ratio between real word object size and image object size.

3. R is: [[9.90748275e-01 -8.91103396e-03 -9.83571297e-01]
 [-1.35712404e-01 9.99960296e-01 -1.80520095e-01]
 [1.61576709e-06 2.92119153e-07 1.91997834e-06]]

4.

```
Estimating height of CSL building
H/R 0.057676423278618716
CSL building is 1144 inches, which is 95'4
Estimating height of the spike statue
H/R 0.16487728369910698
the spike statue is 400 inches, which is 33'4
Estimating height of the lamp posts
H/R 0.40931794184537645
the lamp posts is 161 inches, which is 13'5
```

If the person is 6ft tall, which is 72 inches, then:
CSL will be around 72 / 0.0577 = 1248 inches, instead of 1144;
Spike will be around 72 / 0.1649 = 437 inches, instead of 400;
Lamp Post will be around 72 / 0.409 = 176 inches, instead of 161;

PART3 EXTRA CREDIT:

1. The tallest guy is the guy on the most left. 
   He's about 5"11' tall.

2. The smallest window is 142 inches and others are 216 inches.

3. I use the picture taken by me to measure “One Times Square”, NYC. The actualy height is 363 feets + 30 feets (for the billboard above) v.s my approximation 415 feets. The noise is mainly due to people in this picture is too small.