ECE549 / CS543 Computer Vision

Assignment 3 Report

Zheng Liu (zliu93)

Part 1 Homography Estimation

a. **Describe your solution, including any interesting parameters or implementation choices for feature extraction, putative matching, RANSAC, etc.**
I followed instruction on the website to load convert the images. I used the given SIFT detector to get the key points with descriptors, because this is easier and more robust. And I used scipy.spatial.distance.cdist(left_des, right_des, 'sqeuclidean') to get the pairwise distance between descriptors. I chose the first 200 smallest pairwise descriptor distances to get matched key points. I also tried first 150 smallest pairwise distance. And I run the RANSAC loop for 4000 times in order to try more random seed selection to get more Homography fitting. And the inlier threshold is 5, which is tight enough to get good inliers. The algorithm for RANSAC is that, in each iteration, 4 points are randomly selected from those 200 closest matches. And I built the matrix A, based on the formula from lecture, with these 4 points. Then solve the matrix A and select the singular vector corresponding to the smallest singular value, as described on the website. To find the inliers, I used the candidate Homography matrix, H, to transform the matched key points to new coordinate and compare this with the original matched points, if the distance is small enough, then this matched pair is an inlier. Then I chose the homography matrix with most inliers to be the optimal H.

b. **For the image pair provided, report the number of homography inliers and the average residual for the inliers (squared distance between the point coordinates in one image and the transformed coordinates of the matching point in the other image). Also, display the locations of inlier matches in both images.**
Number of homography inliers: 68.
Average residual: 0.937141
Matched inliers image:

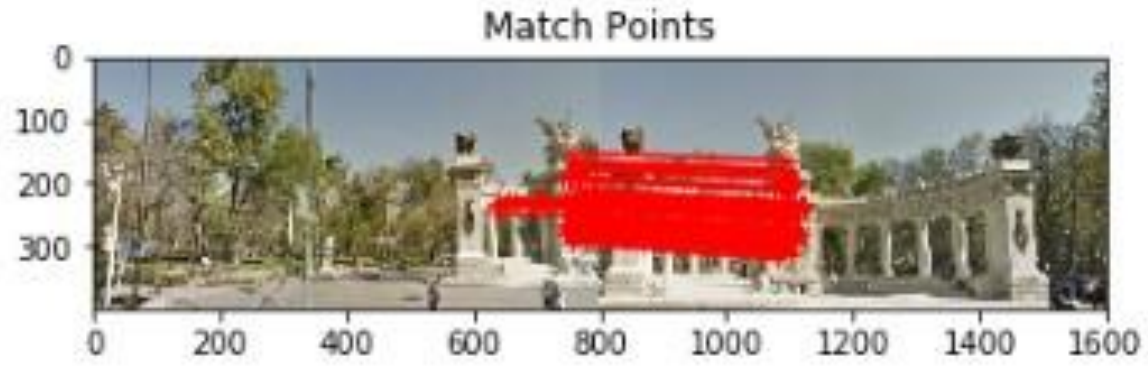


Figure 1: Matched inliers

c. **Display the final result of your stitching.**

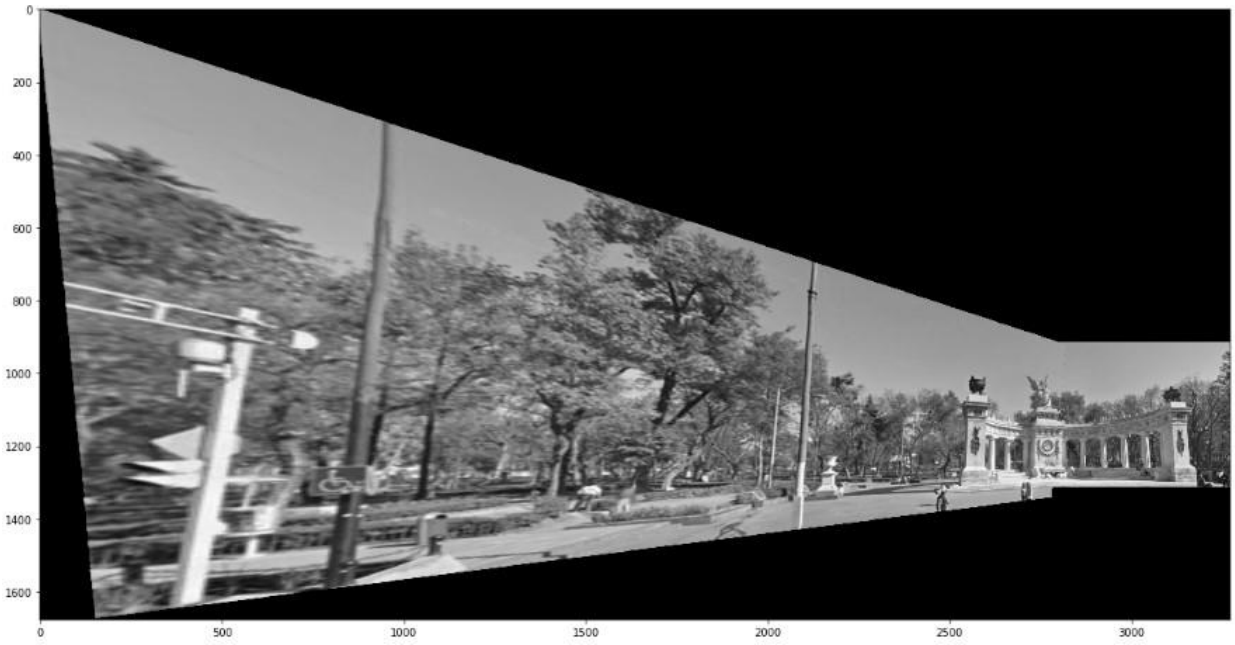


Figure 2: Stitched image in gray

Figure 3: Stitched image in color

Part 2 Fundamental Matrix Estimation, Calibration, Triangulation

a. For both image pairs, for both unnormalized and normalized fundamental matrix estimation, display your result (point and epipolar lines) and report your residual.

- **Library:**
  - **Normalized:**
    - Residual: 0.1835966
    - Image:

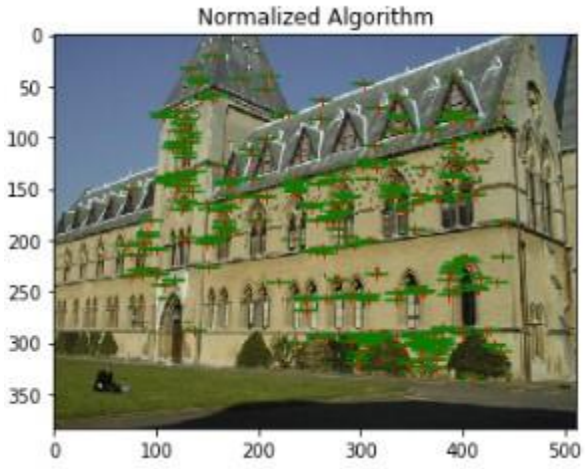


Figure 4: Library Epipolar, normalized

  - **Unnormalized:**
    - Residual: 0.338495
    - Image:

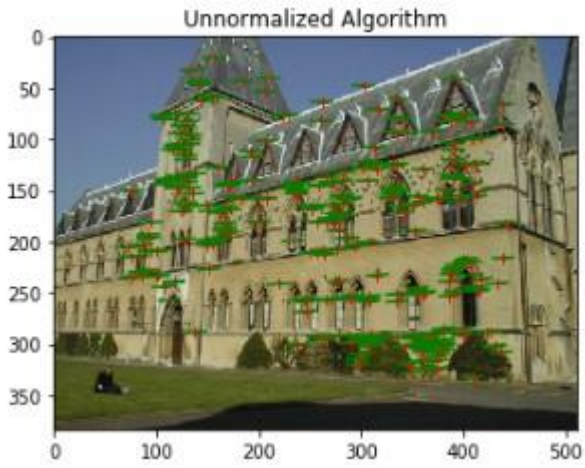


Figure 5: Library Epipolar, Unnormalized

- **Lab:**
  - **Normalized:**
    - Residual: 0.617251755
    - Image:

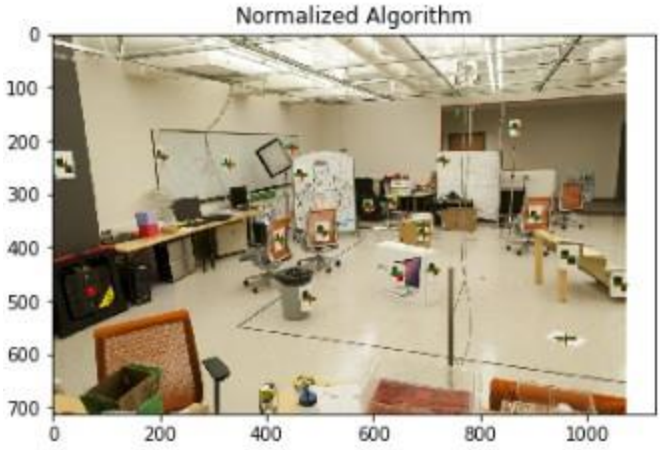


Figure 6: Lab Epipolar, normalized

  - **Unnormalized:**
    - Residual: 2.237977425
    - Image:

Figure 7: Lab Epipolar, normalized

b. For the lab image pair, show your estimated 3x4 camera projection matrices. Report the residual between the projected and observed 2D images.

Estimated camera projection matrix for lab1 is:
[[3.09971524e-03, 1.46250174e-04, -4.48354919e-04, -9.78974905e-01]
[3.06744636e-04, 6.36810842e-04, -2.77389022e-03, -2.03932211e-01]
[1.67995219e-06, 2.74565792e-06, -6.83395792e-07, -1.32842138e-03]]

Estimated camera projection matrix for lab2 is:
[[-6.88970692e-03, 3.96429852e-03, 1.39263702e-03, 8.28289829e-01]
[-1.53909600e-03, -1.02084411e-03, 7.22962251e-03, 5.60181867e-01]
[-7.58603647e-06, -3.72293087e-06, 2.03836990e-06, 3.38133189e-03]]

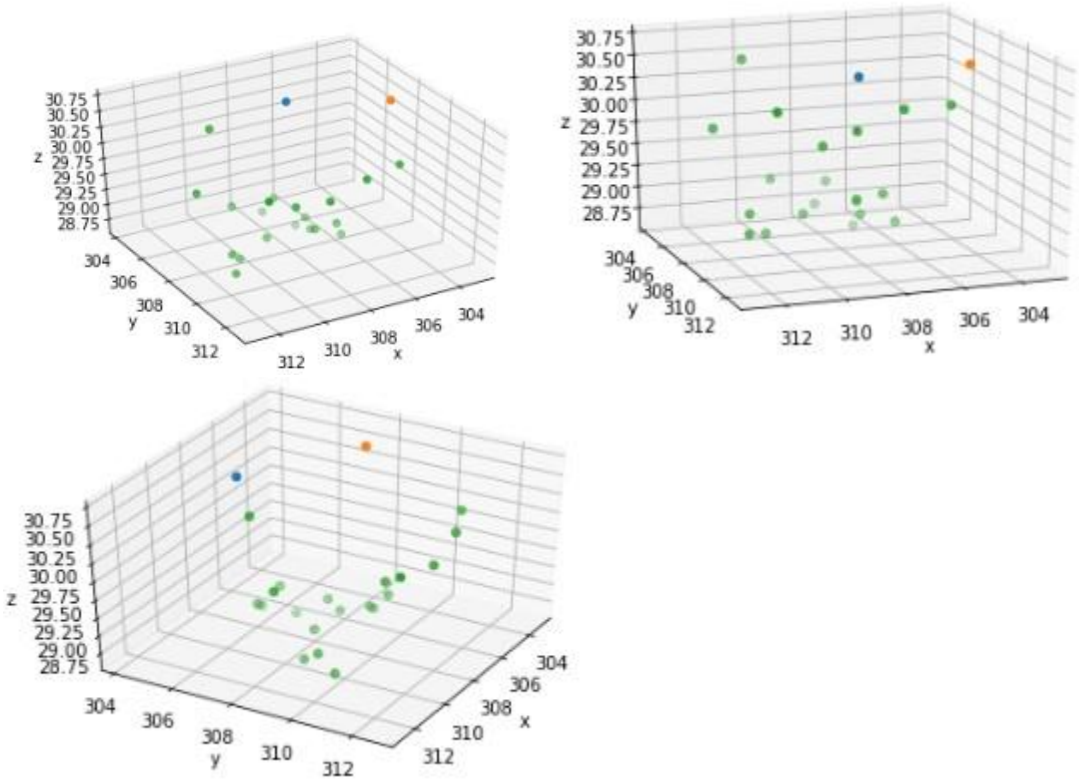
The residual between the projected and observed 2D points for lab1 is: 13.765505109334178
The residual between the projected and observed 2D points for lab2 is: 17.781125905791964

c. For both image pairs, visualize 3D camera centers and triangulated 3D points.

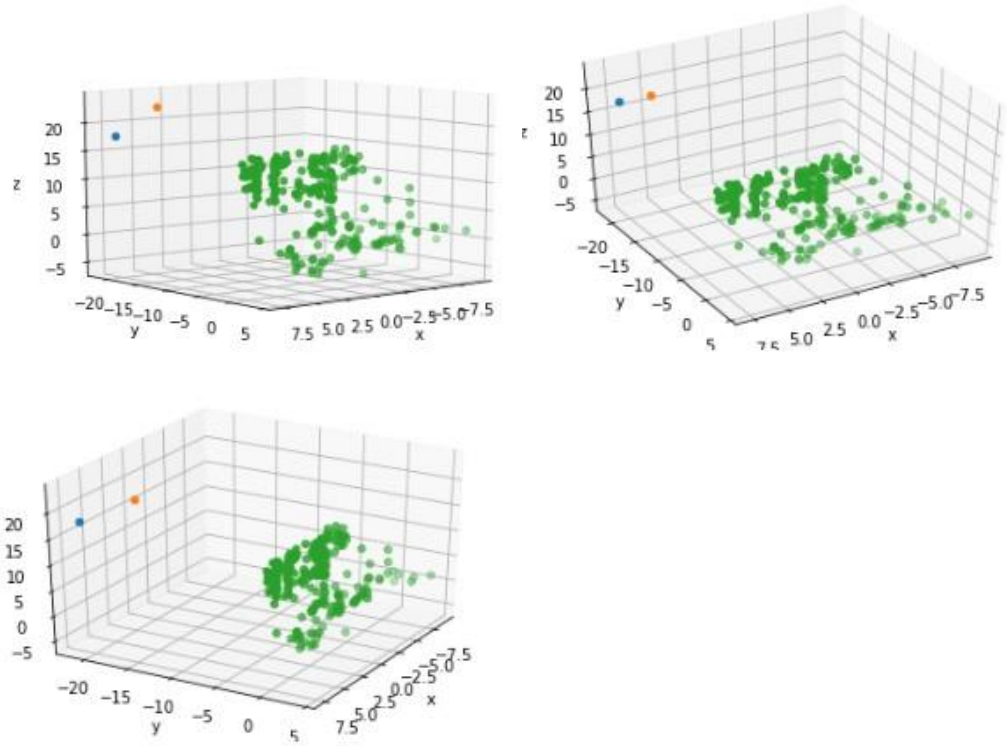
lab1 center: [305.83387882 304.20073002 30.13782356 1.]
lab2 center: [303.14001018 307.21161306 30.4296492 1.]
library1 center: [7.28863053 -21.52118112 17.73503585 1.]
library2 center: [6.89405488 -15.39232716 23.41498687 1.]

The blue and orange points are camera positions

Lab:



Library:



To show the triangulation is correct, there are the points after triangulation and triangulated 3D points for the lab pair match very closely the originally provided 3D points in lab_3d.txt

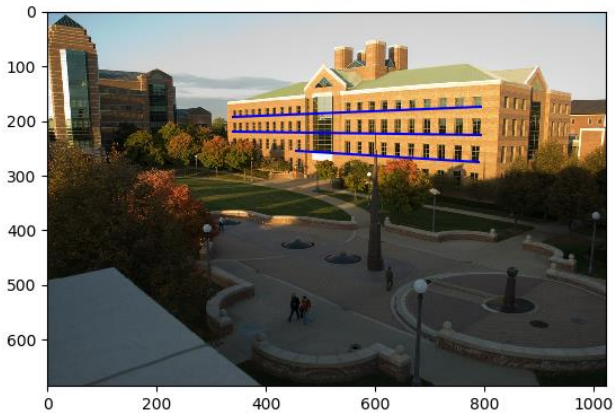
```
[[312.78291956   309.14793677  30.08825937   1.      ]
 [305.79583985   311.647791    30.35938704   1.      ]
 [307.7005947    312.37217915  30.41674895   1.      ]
 [310.13440522   307.17943144  29.30057835   1.      ]
 [311.95788273   310.12008416  29.21275648   1.      ]
 [311.20691609   307.57555977  30.67950481   1.      ]
 [307.10774467   306.88091469  28.65793142   1.      ]
 [309.28133295   312.4486935   30.23054336   1.      ]
 [307.43935436   310.15100191  29.31303864   1.      ]
 [308.24268375   306.29876816  28.88661985   1.      ]
 [306.64235566   309.2992755   28.90589268   1.      ]
 [308.0651248    306.84106044  29.19401194   1.      ]
 [309.64168792   308.81128988  29.03262552   1.      ]
 [308.27460678   309.9739103   29.25778192   1.      ]
 [307.58012218   308.62549965  28.95332987   1.      ]
 [311.08075193   309.20605436  28.88932969   1.      ]
 [307.52743213   308.18738654  29.06369628   1.      ]
 [309.93933906   311.25779992  29.99069135   1.      ]
 [312.24541544   310.81511246  29.05823813   1.      ]
 [311.98515006   312.70461737  30.51474434   1.      ]]
```

Residuals between the observed 2D points and the projected 3D points for lab1
1.4784850377065275
Residuals between the observed 2D points and the projected 3D points for lab2
0.043630386365192765

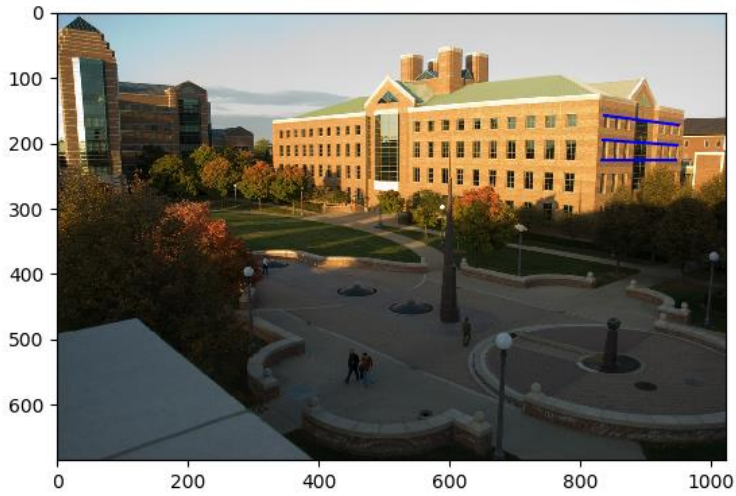
Part 3 Single-View Geometry

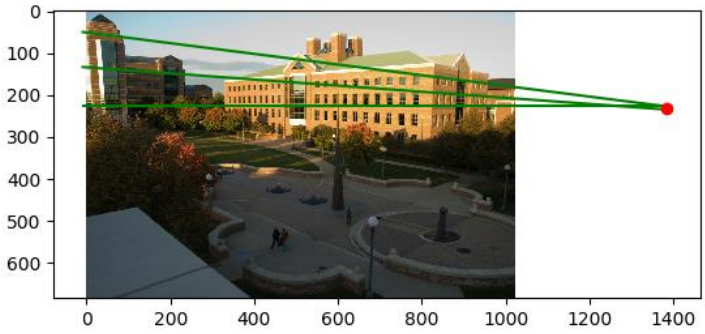
1.

a. Plot the VPs and the lines used to estimate them on the image plane using the provided code.
Left most vanishing point:

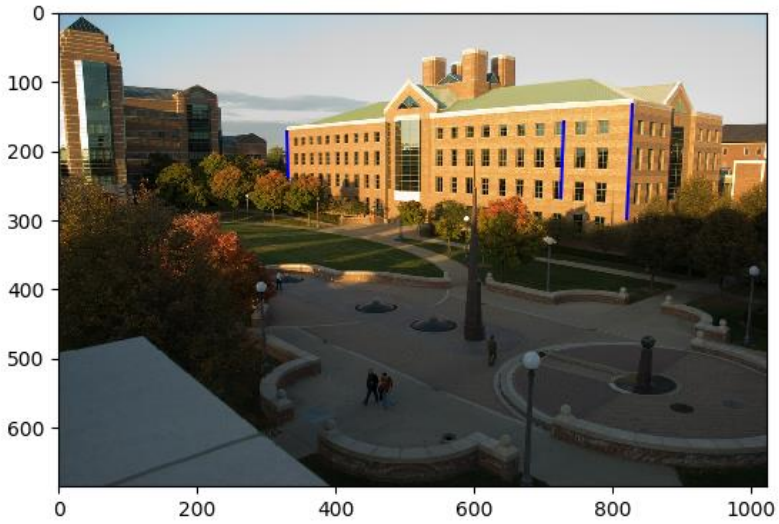


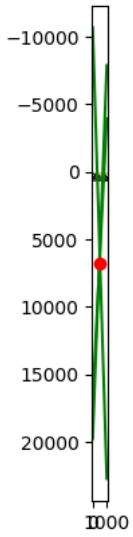
Right most vanishing point:





Vertical Vanishing point:

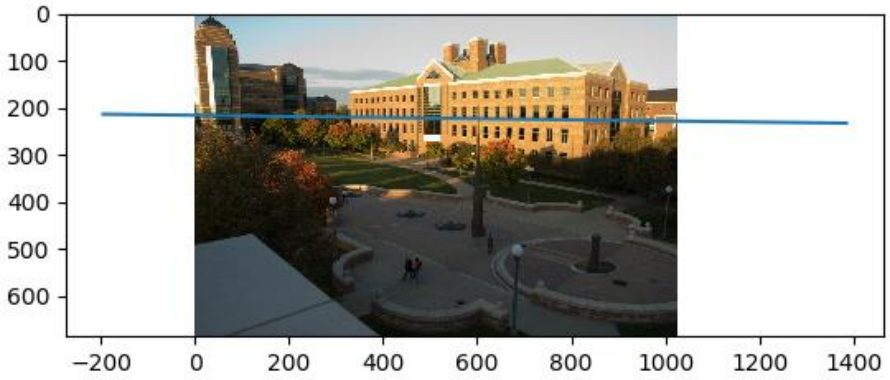




b. Specify the VP pixel coordinates.

Coordinate of left most vanishing point: [-195.63453054 212.82282092 1.]
Coordinate of right vanishing point: [1.38538838e+03 2.31871619e+02 1.00000000e+00]
Coordinate of vertical vanishing point: [4.87156649e+02 6.80106869e+03 1.00000000e+00]

c. **Plot the ground horizon line and specify its parameters in the form a * x + b * y + c = 0. Normalize the parameters so that: a^2 + b^2 = 1.**
Parameter for horizontal line: [-1.20475269e-02 9.99927426e-01 -2.15164288e+02]



2. **Using the fact that the vanishing directions are orthogonal, solve for the focal length and optical center (principle point) of the camera. Show all your work.**
f: -784.340104827719
u: 565.280928705448
v: 316.865735197555

```python
def get_camera_parameters(vpts):
    """
    Computes the camera parameters. Hint: The SymPy package is suitable for this.
    """
    # <YOUR IMPLEMENTATION>
    f = sp.Symbol('f')
    u = sp.Symbol('u')
    v = sp.Symbol('v')
    v1 = sp.Matrix(vpts[:, 0])
    v2 = sp.Matrix(vpts[:, 1])
    v3 = sp.Matrix(vpts[:, 2])

    inverse_K = sp.Matrix(((f, 0, u), (0, f, v), (0, 0, 1))).inv()

    e12 = v1.T * inverse_K.T * inverse_K * v2
    e13 = v1.T * inverse_K.T * inverse_K * v3
    e23 = v2.T * inverse_K.T * inverse_K * v3
    sol = sp.solve([e12, e13, e23], [f, u, v])
    f = sol[0][0]
    u = sol[0][1]
    v = sol[0][2]
    K = sp.Matrix(((f, 0, u), (0, f, v), (0, 0, 1)))
    return f, u, v, K
```

The formula comes from lecture 15, page 12:

Calibration from vanishing points

- Let us align the world coordinate system with three orthogonal vanishing directions in the scene:

$$\mathbf{e}_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}, \quad \mathbf{e}_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}, \quad \mathbf{e}_3 = \begin{bmatrix}0\\0\\1\end{bmatrix}$$

$$\lambda_i \mathbf{v}_i = \mathbf{K}\mathbf{R}\mathbf{e}_i$$

$$\mathbf{e}_i = \lambda_i \mathbf{R}^T \mathbf{K}^{-1} \mathbf{v}_i$$

- Orthogonality constraint: $\mathbf{e}_i^T \mathbf{e}_j = 0$

$$\mathbf{v}_i^T \mathbf{K}^{-T} \mathbf{K}^{-1} \mathbf{v}_j = 0$$

First, set up the camera matrix K, then set up the equations for solving the K.

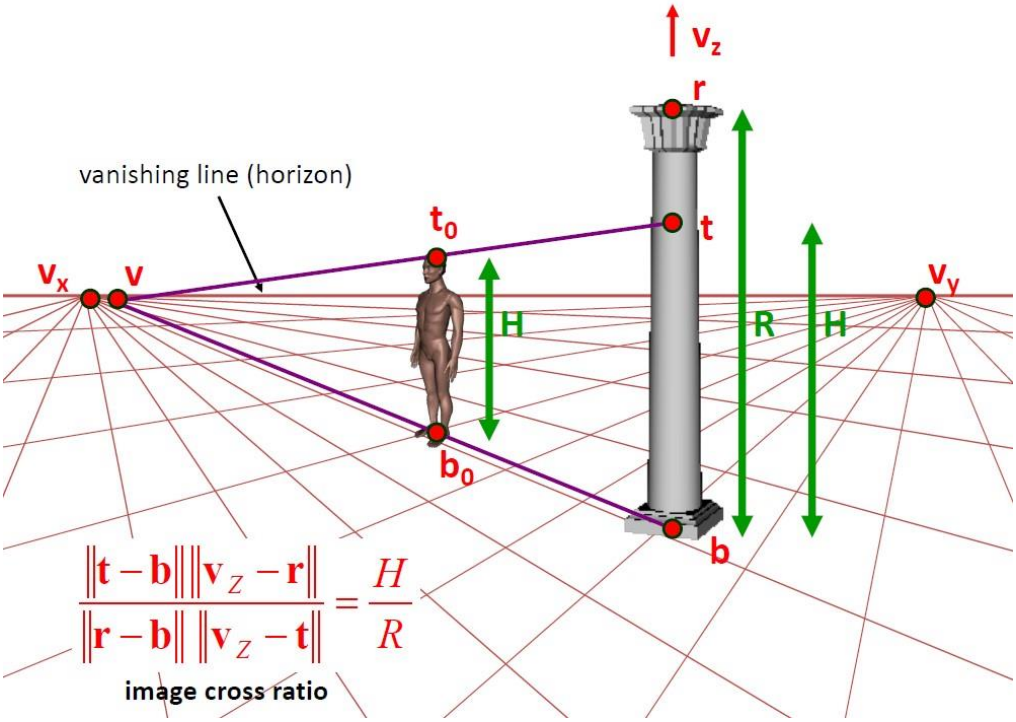
3. **Compute the rotation matrix of the camera, setting the vertical vanishing points as the Y-direction, the right-most vanishing point as the X-direction, and the left-most vanishing point as the Z-direction.**

This is the rotation matrix of the camera.

```
[[-0.72067194    0.01196036    0.69317307]
 [ 0.07468884   -0.99269244    0.09478024]
 [ 0.68924128    0.12007775    0.71451228]]
```

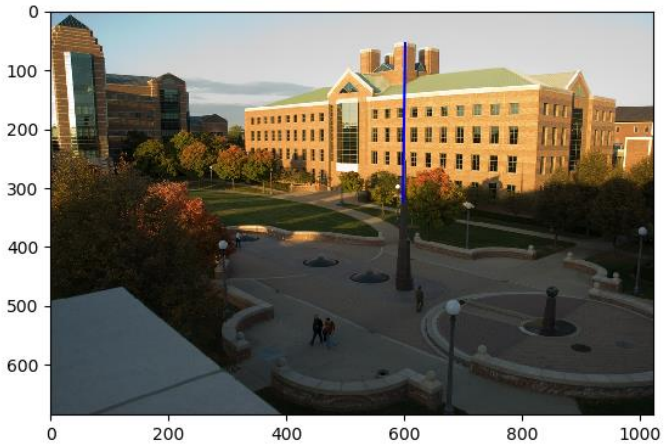
```python
def get_rotation_matrix(vpts, K):
    Y_dir = vpts[:, 2]
    X_dir = vpts[:, 1]
    Z_dir = vpts[:, 0]

    inverse_K = np.array(K.inv()).astype(np.float)

    r1 = np.matmul(inverse_K, X_dir)
    r2 = np.matmul(inverse_K, Y_dir)
    r3 = np.matmul(inverse_K, Z_dir)

    r1 = r1 / np.linalg.norm(r1)
    r2 = r2 / np.linalg.norm(r2)
    r3 = r3 / np.linalg.norm(r3)

    R = np.zeros((3, 3))
    R[:, 0] = r1.T
    R[:, 1] = r2.T
    R[:, 2] = r3.T
    return R
```

4. **Estimate the heights of (a) the CSL building, (b) the spike statue, and (c) the lamp posts assuming that the person nearest to the spike is 5ft 6in tall. In the report, show all the lines and measurements used to perform the calculation. How does the answer change if you assume the person is 6ft tall?**

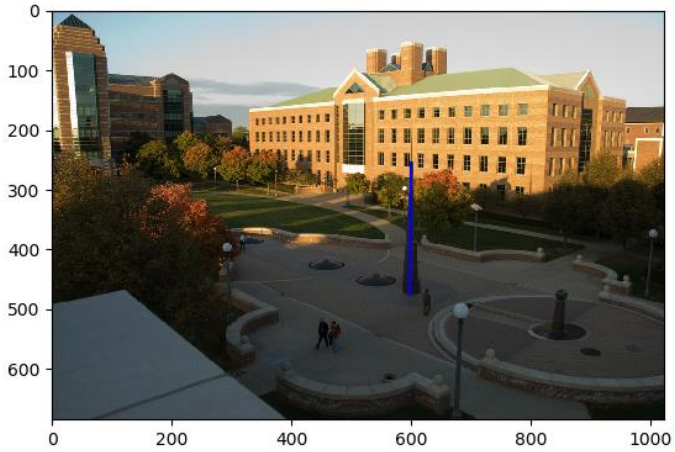



```python
def estimate_height(vpts, reference_height, reference_coord, object_coord):
    """
    Estimates height for a specific object using the recorded coordinates. You might need to plot additional images here for
    your report.
    """
    # <YOUR IMPLEMENTATION>
    vx = vpts[:, 0]
    vy = vpts[:, 1]
    vz = vpts[:, 2]
    r = object_coord[:, 0]
    t_0 = reference_coord[:, 0]
    b = object_coord[:, 1]
    b_0 = reference_coord[:, 1]
    v = np.cross(np.cross(b, b_0), np.cross(vx, vy))
    v = v / v[-1]
    t = np.cross(np.cross(v, t_0), np.cross(r, b))
    t = t / t[-1]
    height = reference_height \
    / ((np.linalg.norm(t - b) * np.linalg.norm(vz - r)) / (np.linalg.norm(r - b) * np.linalg.norm(vz - t)))
    return height
```

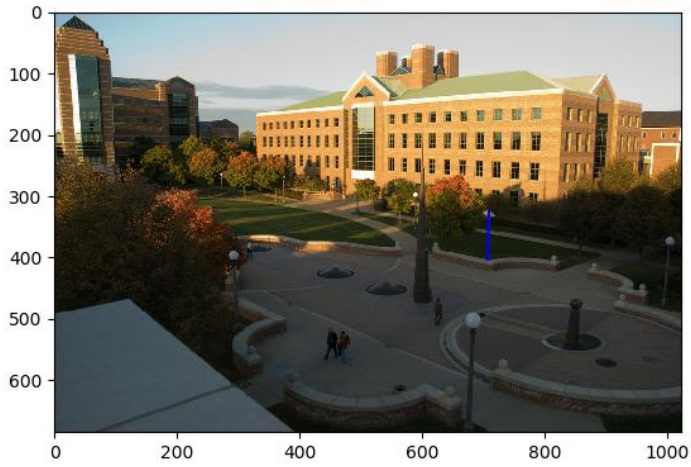
If the reference person is 5ft 6in tall:



Height of CSL building is: 1276.342646 inches



Height of the spike statue is: 391.020097 inches



Height of the lamp posts is: 192.257273 inches

If the reference person is 6ft tall:

Height of CSL building is: 1392.373796 inches

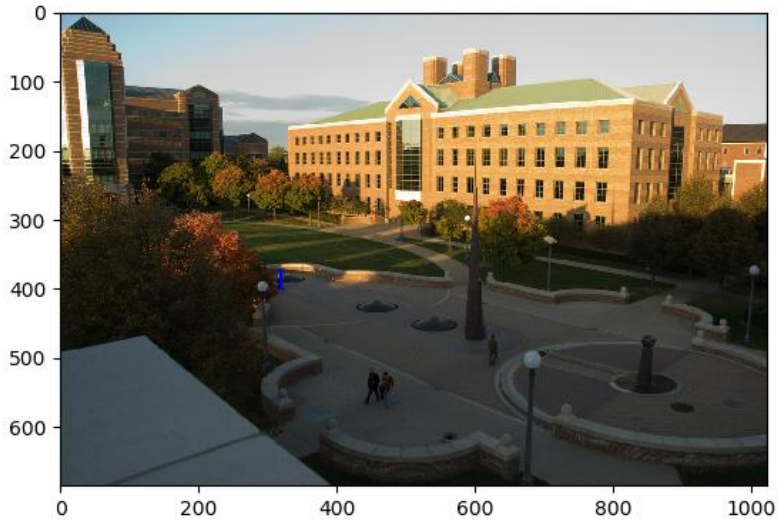
Height of the spike statue is: 426.567379 inches

Height of the lamp posts is: 209.735207 inches

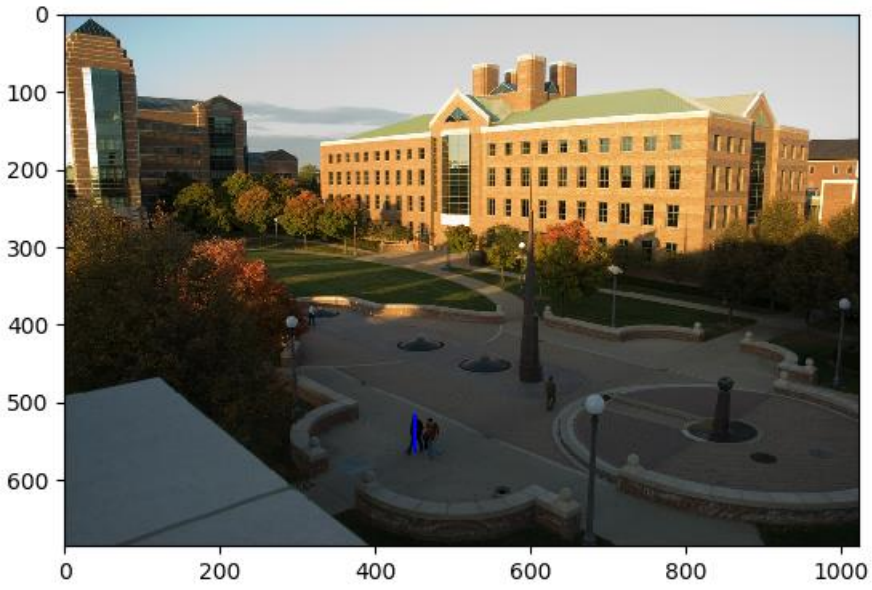
By comparing the result, we can see that if the reference person is taller, then other objects also get a taller estimation.

Part 3 Extra credit:

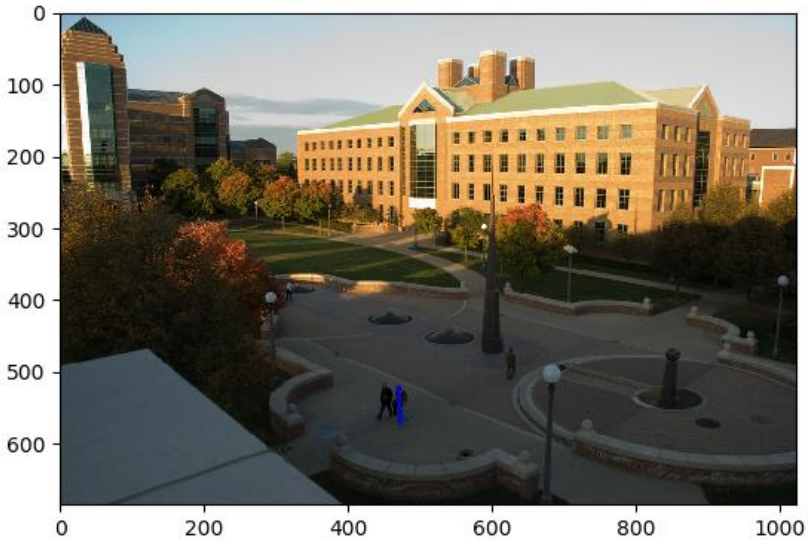
If the reference person is 5ft 6in tall:



Height of person on the left with white shirt is: 68.093012 inches

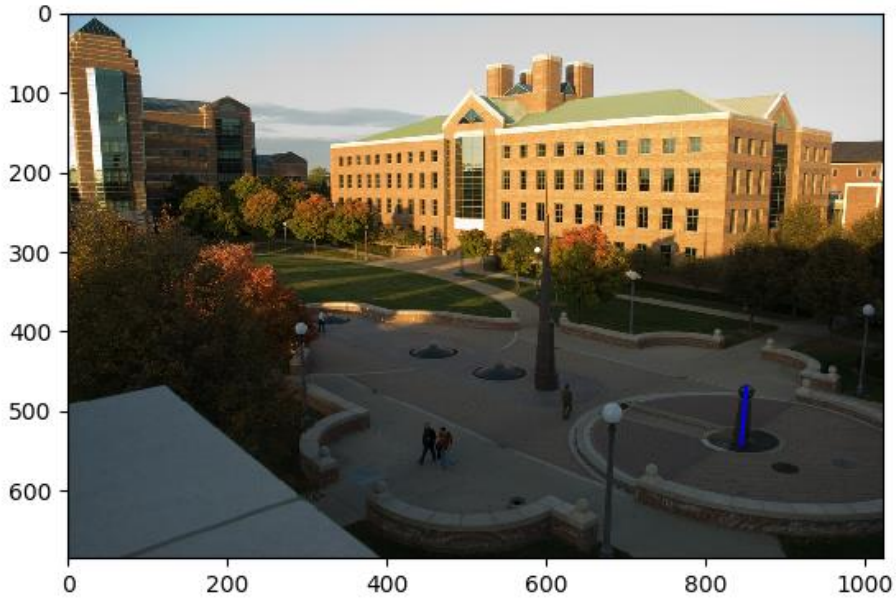


Height of person in the middle with black shirt is: 60.775661 inches



Height of person on the right with red shirt is: 66.602252 inches

So the person on the left with white shirt is tallest among all the people in the image.



Height of fountain is: 105.111099 inches