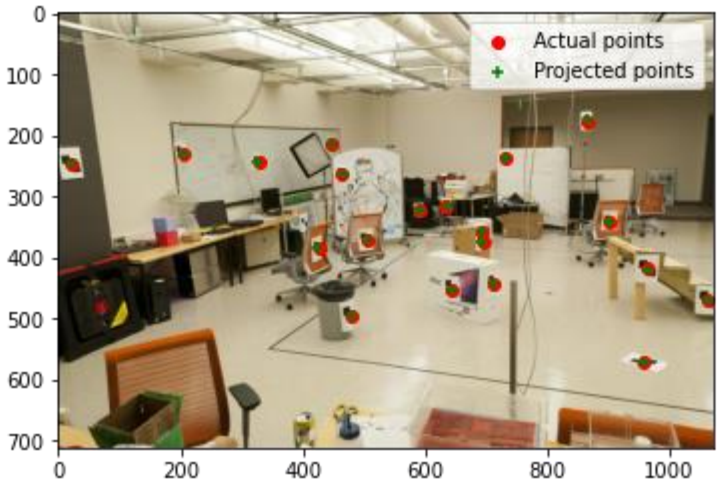
CS 4476 Project 3

Younes Djemmal
ydjemmal3@gatech.edu
903794219

Part 1.5: Projection Matrix for provided image

<insert visualization of projected 3D points and actual 2D points for image provided by us here [1]>



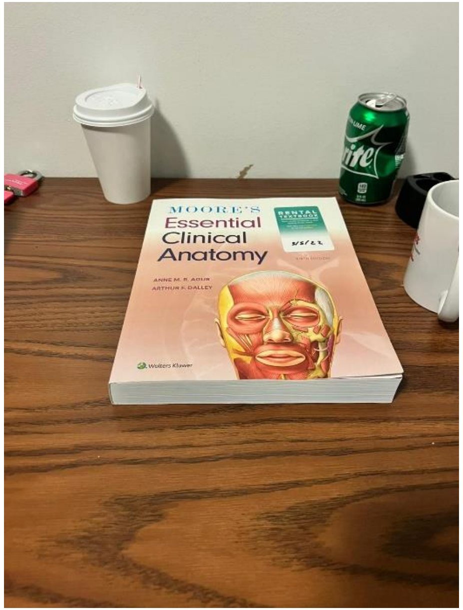
<What is the minimum number of 3D-2D point correspondences needed to estimate the projection matrix? Why? [2]>

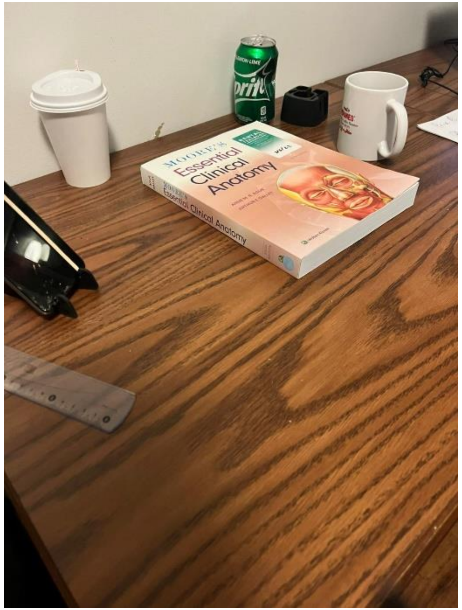
Our matrix P is unique up to a scale only, that's why we fixed the element $\mathbf{P}_{34}$=1. Which leads us to only have degrees of freedom.

With 11 degrees of freedom, we need a minimum of 6 2D/3D correspondences to estimate our matrix. Each correspondence takes off 2 degrees of freedom as each corresponding point will give two equations, one for x and one for y.

Part 2.1: Projection Matrix for custom images

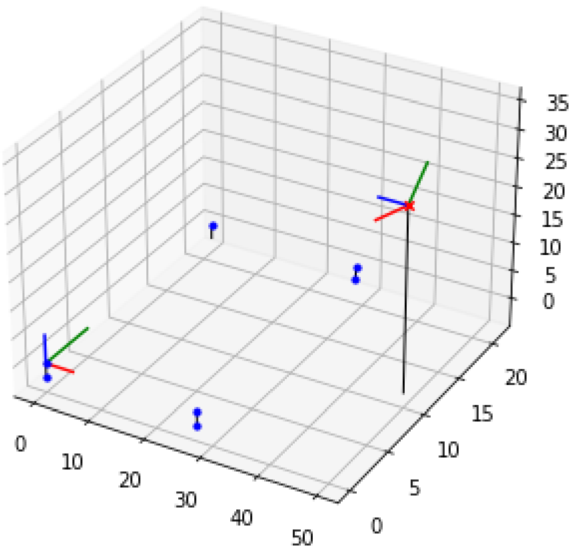
<Copy two images of your fiducial object here [2]>



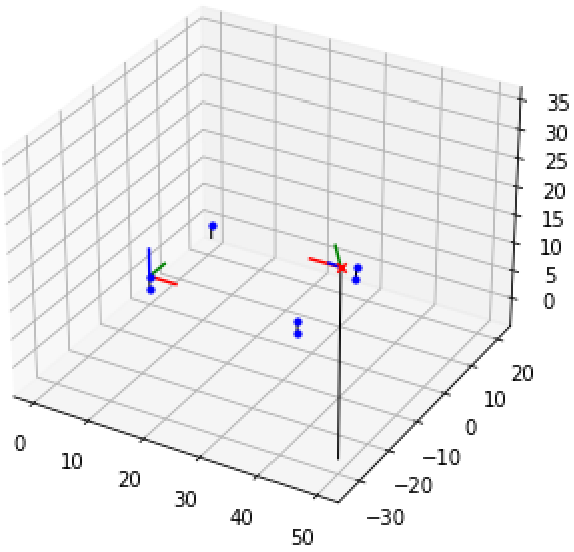


Part 2.2: Pose init for custom images

<Insert visualization for the initialized camera pose for 1st image> [1]

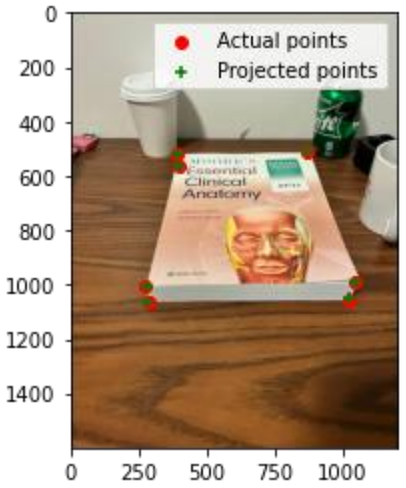


<Insert visualization for the initialized camera pose for 2nd image> [1]

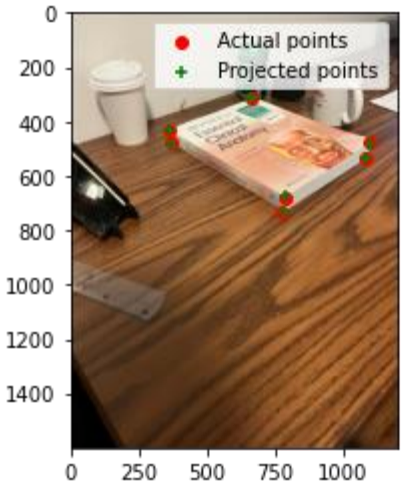


Part 2.2: Optimized results for custom images

<Insert visualization for projected 3D points and actual 2D points for 1st image> [1.5]

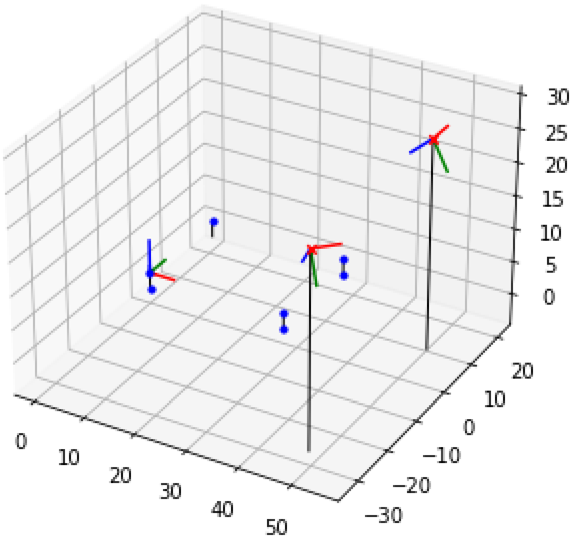


<Insert visualization for projected 3D points and actual 2D points for 2nd image> [1.5]



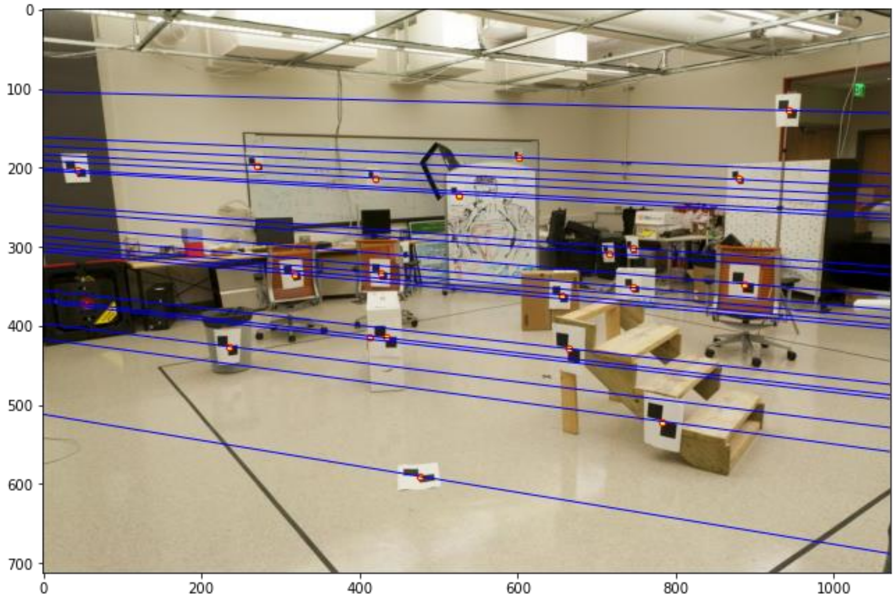
Part 2.3: Optimized Camera Poses

<Insert pose with world and optimized camera’s coordinate systems [1]>

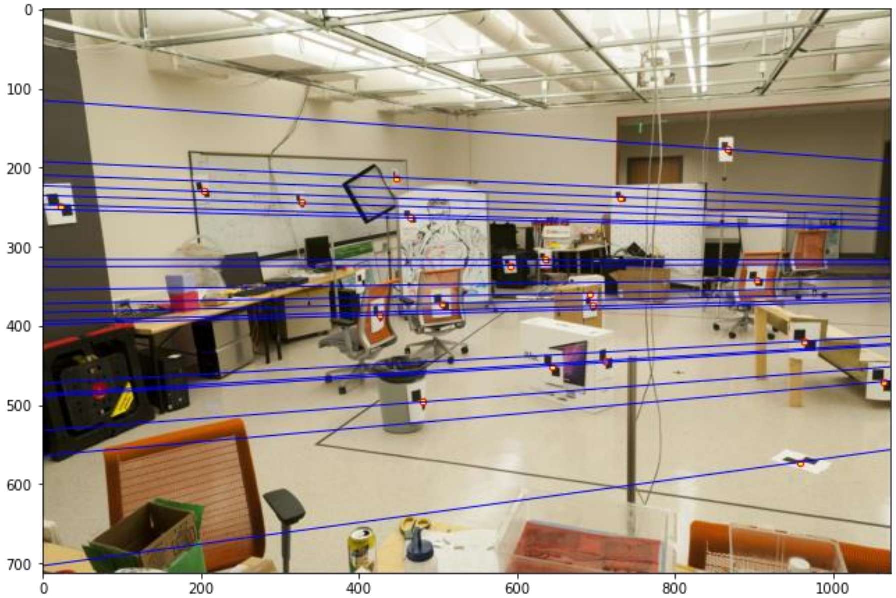


Part 3.2: Optimized Epipolar Lines (given images)

<Insert left image with epipolar lines> [1]



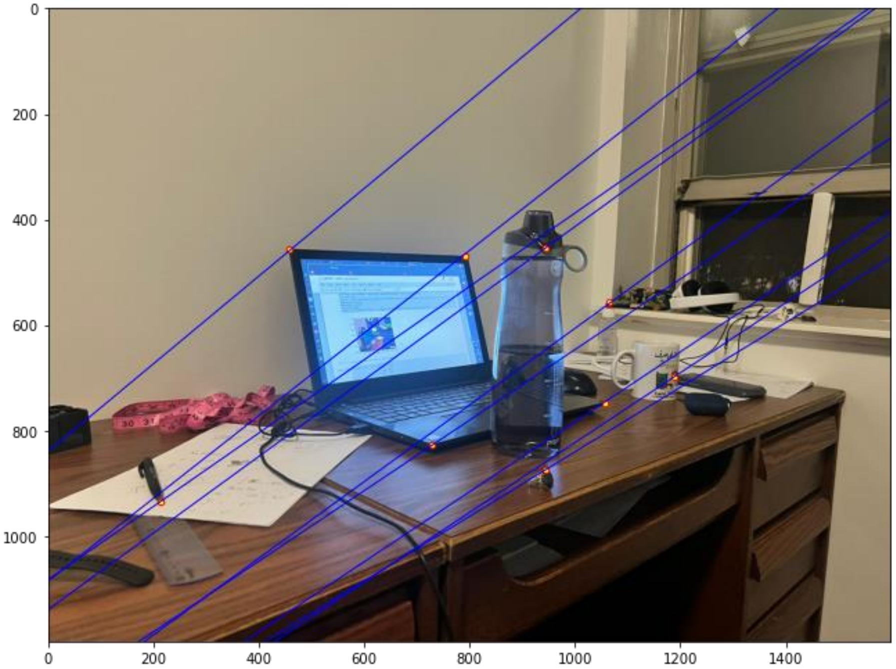
<Insert right image with epipolar lines> [1]

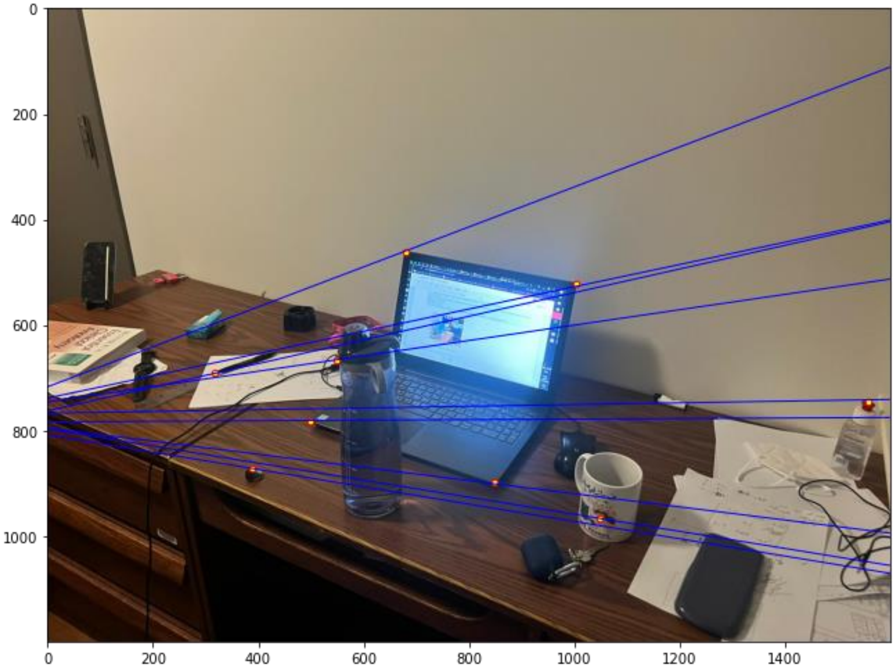


Part 3.3: Optimized Epipolar Lines (custom images)

<Insert left image with epipolar lines> [1.5]

<Insert right image with epipolar lines> [1.5]





Part 3.4: Reflection Questions [1x3]

1. The whole point of stereo vision is uncovering the depth of the image that we lose when converting from 3D 2D. Using only one camera would result in not being able to find the 3D point corresponding to a point x in our image because every point on the line OX (O being camera center) will project to the same point on our image plane. But using another camera center, different points on the line OX would project to different points in the second image plane, using this we can calculate the 3D component of our image.

2. Having a point x in image 1, x's projection into image plane two is the projection of the line OX. Each point on line OX will give a different x' on image plane two (by projecting it using O'). The line connecting all these projections is called epipolar line. This line goes through the epipole e' which is the projection of the first camera center O into the second image plane. And vice versa.

3. What happens is the epipole will be the focus of expansion and would usually lie somewhere within image place. If the translation is pure, the epipoles would have the same coordinates in both images and same epipolar lines would be overlaid on top of each other in both cases.

Part 3.4: Reflection Questions [1x3]

4. This case happens when the two image planes are parallel to each other. This would result in having epipoles located at infinity because the baseline is also horizontal and parallel to the image planes.

5. F is defined up to a scale means it’s not a unique matrix. There could be several matrices that satisfy $\boldsymbol{p}'^T\boldsymbol{F}\boldsymbol{p} = 0$.

6. Because: The fundamental matrix F may be written as the dot product of two matrices, one of rank 2 and the other rank 3. Therefore F is rank 2.

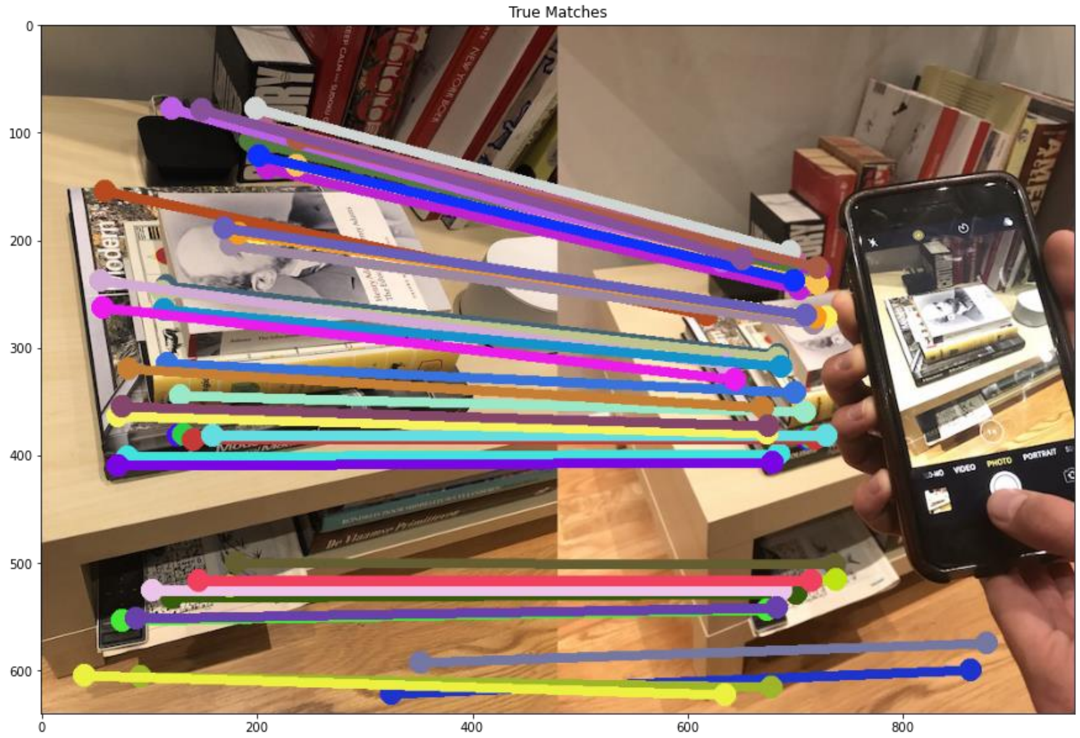
Part 4.2: RANSAC Iterations Questions [1x3]

Type your answers to the three RANSAC Iterations questions from the jupyter notebook below:

1. S = 14 is the minimum number of iterations.
2. S = 42 in the case of 18 points.
3. S = 167 iterations.

Part 4.4: RANSAC Inlier Matches

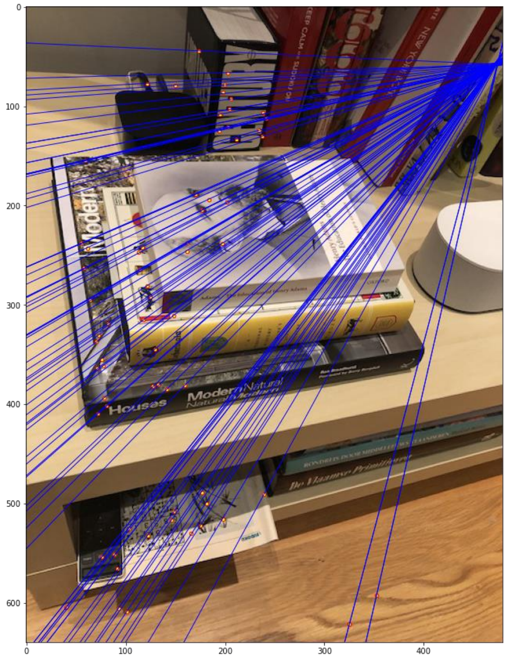
<Paste the inlier matches found by Ransac [2]>

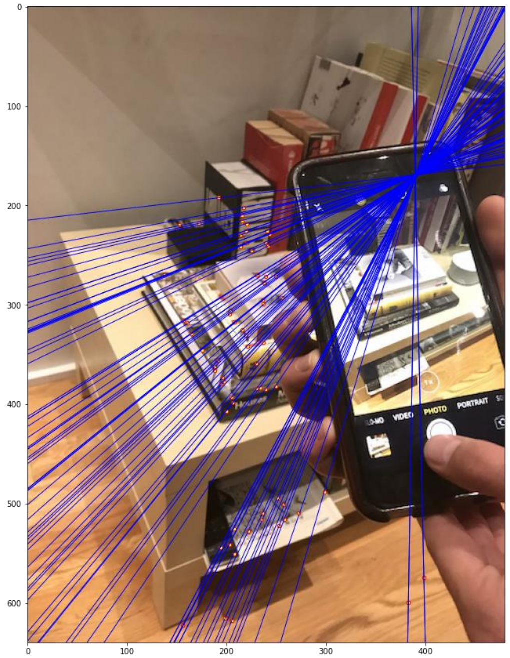


Part 4.4: RANSAC Epipolar Lines

<Paste the left image with epipolar lines> [1]

<Paste the right image with epipolar lines> [1]





Local Unit tests results

<Paste the screenshot when you run all provided unit tests using `pytest`> [1]

```
(cs4476_proj3) C:\proj3_release-2\proj3_unit_tests>pytest ./.
================================================= test session starts =================================================
platform win32 -- Python 3.6.13, pytest-6.2.4, py-1.11.0, pluggy-0.13.1
rootdir: C:\proj3_release-2
collected 20 items

part1_unit_test.py .....                                                                                         [ 25%]
test_essential_matrix_decomposition.py ..                                                                        [ 35%]
test_fundamental_matrix.py .........                                                                             [ 80%]
test_ransac.py ....                                                                                              [100%]

================================================== warnings summary ===================================================
proj3_unit_tests/test_ransac.py::test_ransac_find_inliers
  C:\proj3_release-2\proj3_unit_tests\test_ransac.py:46: DeprecationWarning: elementwise comparison failed; this will raise an error in the future.
    assert outliers not in inliers

-- Docs: https://docs.pytest.org/en/stable/warnings.html
=========================================== 20 passed, 1 warning in 24.03s ============================================
```

Conclusions

<Describe what you have learned in this project. Feel free to include any challenges you ran into.> [2]

I now know how to estimate the projection for a certain image and uncover the depth of the image. I also know how to do it without knowing anything about the cameras that took the shots and only using a set of given correspondences using the Fundamental matrix.
I am more comfortable with optimization problems.
I also know how to do the previous process without having a set of correspondences but instead using RANSAC to get my interest points matches. I learnt a way to get the optimal number of RANSAC iterations for my algorithm.’
On top of all that, I also know how to recover the details of the shot on an image given the camera intrinsics. I get the rotation and translation matrices.

I enjoyed this project much more than the previous one, I felt like the notebook was very helpful and the explanations provided were more than enough for a smooth delivery.