Project 5 Writeup

Instructions

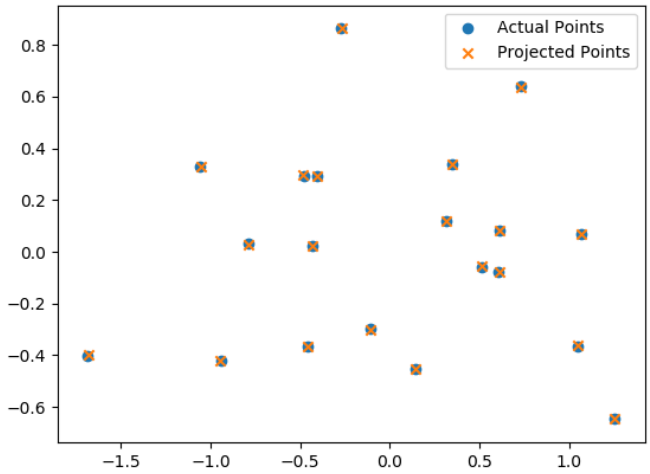
- Provide an overview about how your project functions.
- Describe any interesting decisions you made to write your algorithm.
- Show and discuss the results of your algorithm.
- Feel free to include code snippets, images, and equations.
- List any extra credit implementation and result (optional).
- Use as many pages as you need, but err on the short side.
- **Please make this document anonymous.**

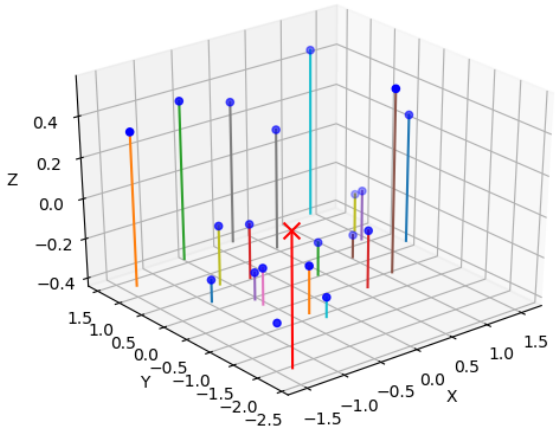
Project Overview

The goal of this project is to introduce you to camera and scene geometry. Specifically we will estimate the camera projection matrix, which maps 3D world coordinates to image coordinates, as well as the fundamental matrix, which relates points in one scene to epipolar lines in another. The camera projection matrix and the fundamental matrix can each be estimated using point correspondences. To estimate the projection matrix—intrinsic and extrinsic camera calibration—the input is corresponding 3d and 2d points. To estimate the fundamental matrix the input is corresponding 2d points across two images. You will start out by estimating the projection matrix and the fundamental matrix for a scene with ground truth correspondences. Then you'll move on to estimating the fundamental matrix using point correspondences from SIFT and RANSAC.

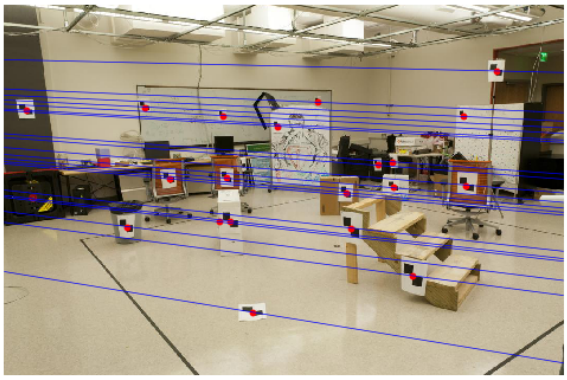
Result

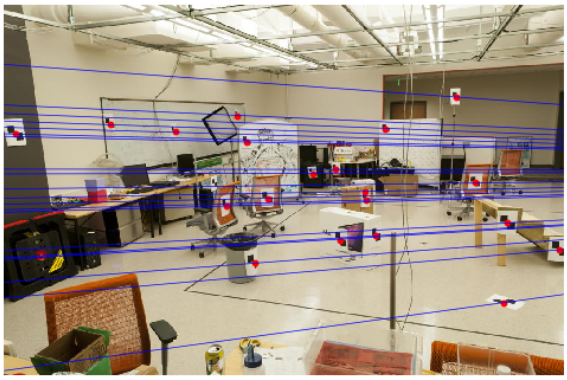
Camera projection are close to the actual points, which is good...



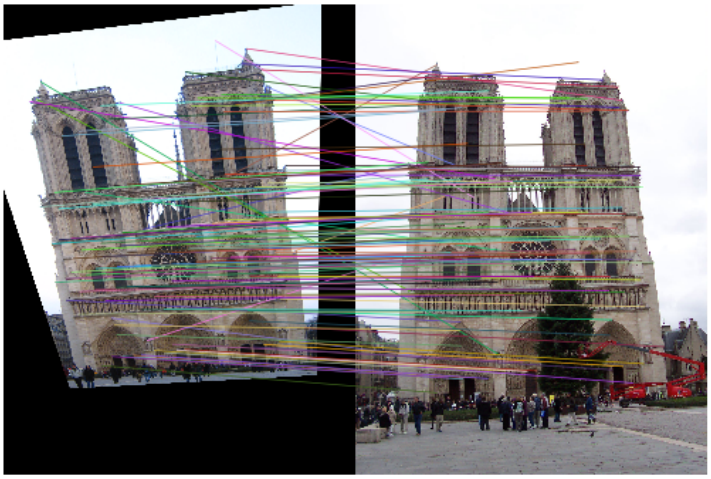


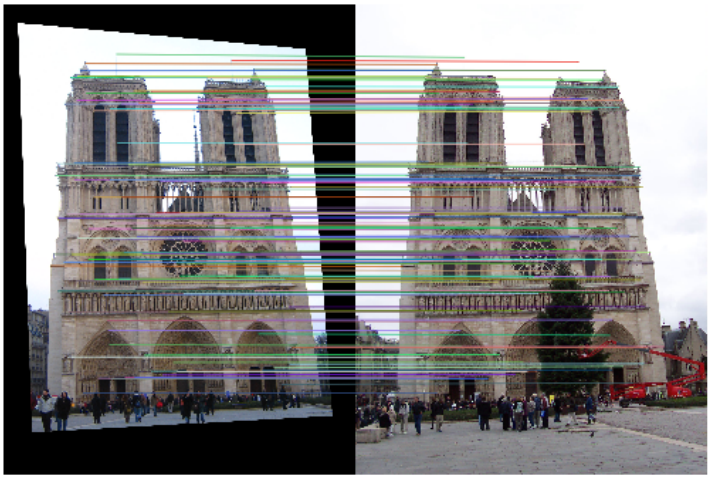
All epipolar lines cross the red center...





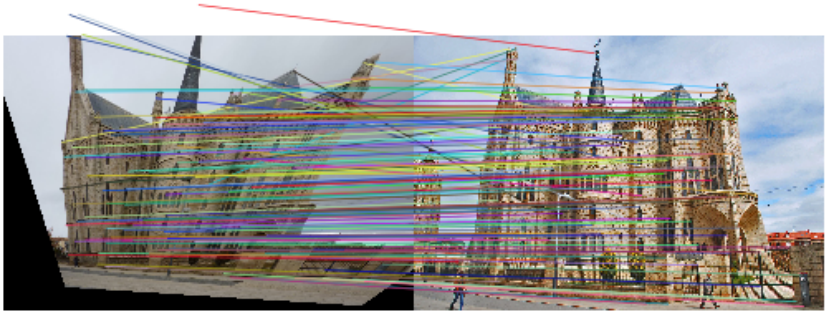
This first image of this page is notre dame after applying noise (high noise like 0.3) and ransac, it matches well. Even though some lines are mismatched and images are little bit distorted. I have also applied normalization, so the performance is better. After normalizing my resiude is low so i can pass the ransac gradescope test...
The second image is ground truth points which matches perfectly...And the same result image is when i also applying low noise and ransac...

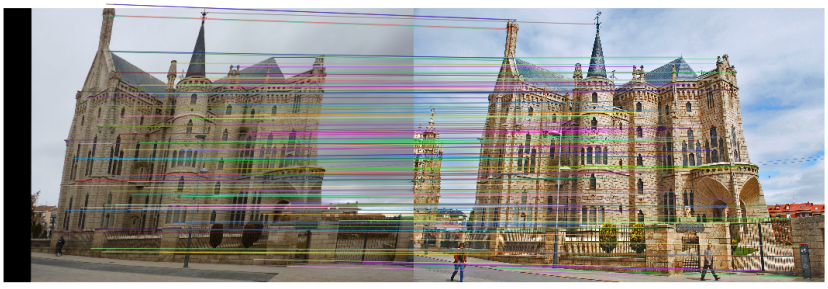




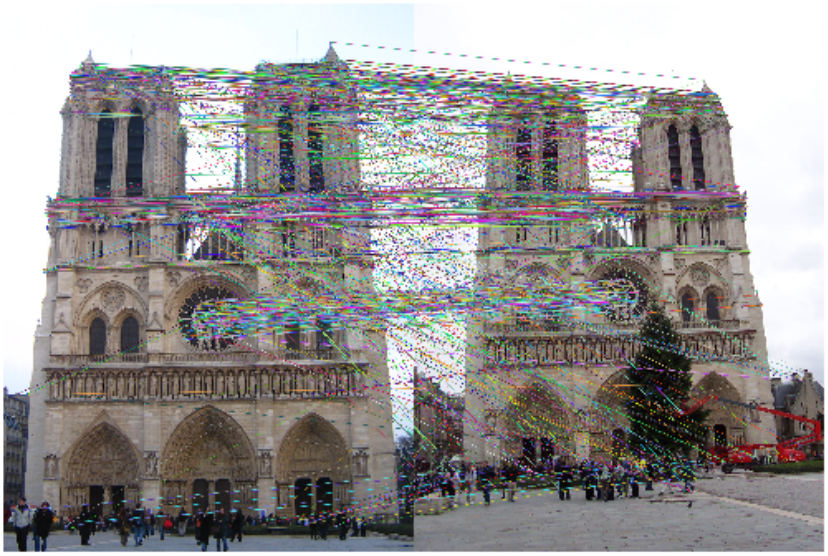
This image is mt rushmore after applying noise and ransac, it matches well. I have also applied normalization, so the performance is better... Ground truth points are perfectly matched...It seems like noise function give us a little trouble but ransac can help! The threshold is 0.000001 but I don't think it needs to be very small. 0.1 should also works. The second image of mt rushmore is ground truth points which matches perfectly...And the same result image is when i also applying low noise and ransac...

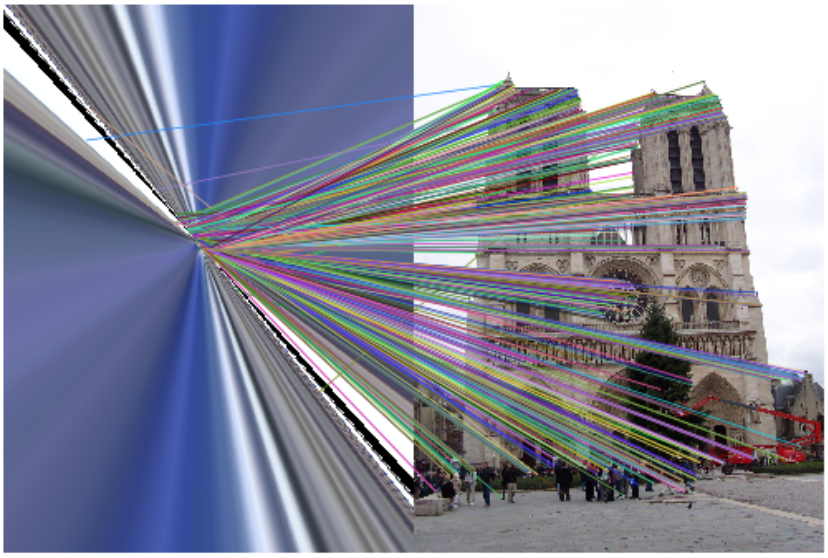
This image is gaudi after applying noise and ransac, it matches well. I have also applied normalization, so the performance is better... I think the number of iteration is better to be around 2000. If it is lower than 2000, the images are distorted badly. If it over 2000 too much, it also won't improve the performance of matching..So 2000 is my selection of iterations. The second image of gaudi is ground truth points which matches perfectly...And the same result image is when i also applying low noise and ransac...





I am also attching orb images, which looks wired





1. Result 1 was a total failure, because...
2. Result 2 (Figure 1, left) was surprising, because...
3. Result 3 (Figure 1, right) blew my socks off, because...

`placeholder.jpg`	`placeholder.jpg`

Figure 1: *Left:* My result was spectacular. *Right:* Curious.

My results are summarized in Table 1.

Condition	Time (seconds)
Test 1	1
Test 2	1000

Table 1: Stunning revelation about the efficiency of my code.

Extra Credit (Optional)

1. Implementation A, code snippets, and results

```
one = 1;
two = one + one;
if two == 2
    disp( 'This computer is not broken.' );
end
```

2. Implementation B, code snippets, and results

```
one = 1;
two = one + one;
if two == 2
    disp( 'This computer is not broken.' );
end
```