CS 477 HW #12

Questions completed: All undergrad level questions in MATLAB

A. Finding a line using RANSAC.

Below in Figure 1, the homogeneous least squares best fit line found using RANSAC is shown.

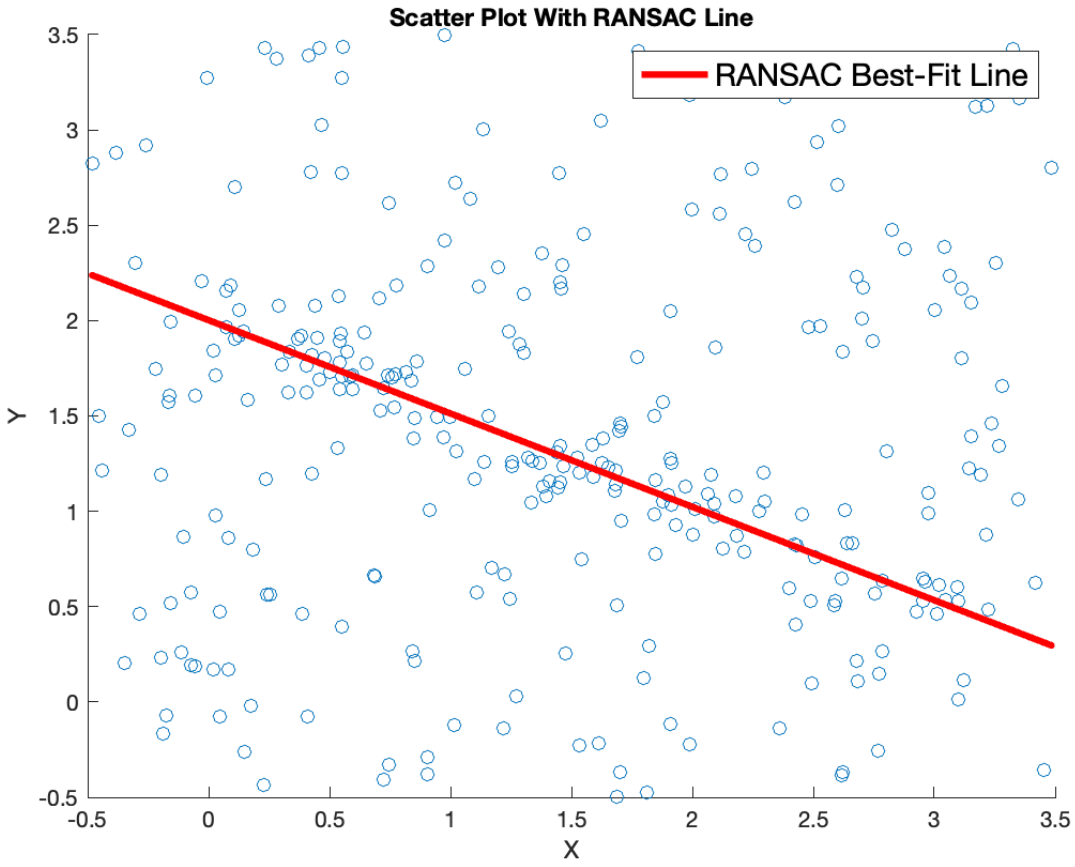


Figure 1: The homogeneous least squares best fit line was found using the RANSAC algorithm. The number of iterations for the RANSAC algorithm to find this line was $K = \log(1 - 0.9999)/\log(1 - 0.25^2) \approx 143$ where 0.9999 is the probability of a valid fit, 0.25 is the probability of getting an inlier (because the problem specified that a quarter of the points lie on a line), and the squared is from the minimum number of points needed for the model (two points to form a line).

In the RANSAC algorithm for each iteration, two random points to form a line were

selected, and the inliers were assigned to the top 25% of points perpendicular to the random line. After the inliers were found, the line was refitted using homogeneous least squares on the inliers, and then the error was computed by taking the sum of the squared perpendicular distances of the inliers to the refitted line. The best error turned out to be 0.2576 which came from the best fit line with a slope of -0.4895 and an intercept of 2.0023.

A plot was then generated as shown above in Figure 1 using MATLAB functions `scatter()` and `plot()`. From visual inspection, it appears that the RANSAC algorithm was effective in identifying the quarter of points that were spread along a good line.

B. Implementing DLT method for computing homography between two planar images.

The RMS error of the transformation for 4, 5, and 6 randomly generated pairs of points inside [0,1]x[0,1] over 10 samples using homography is reported below (from left to right).

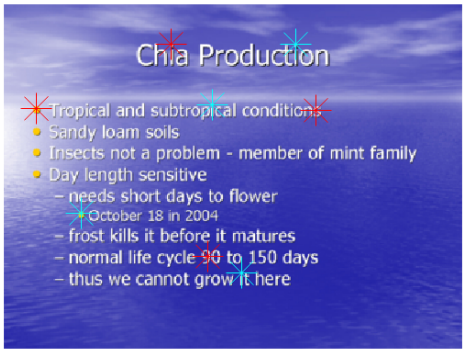
```
RMSE_samples =
0.0000    0.1811    1.1338
```

My results suggest that the code is working because I would expect the RMSE to increase with the number of points that is used in homography computation when we are working purely with randomly generated "pairs" of points.

I also reran the code several times for good measure since we are working with randomly generated pairs, and the results are what is expected most of the time, with the occasional case of 6 random pairs getting a lower RMSE than 5 random pairs (but this is also within expectations of working with random pairs). I also tested that my DLT

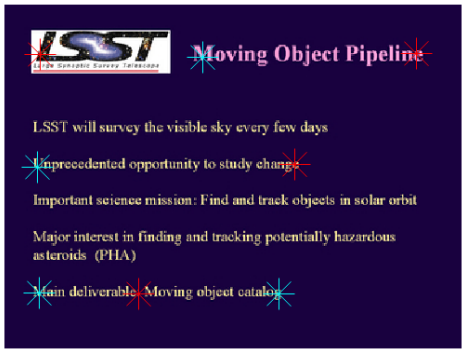
method for computing homography was mapping things correctly before working with randomly generated pairs.

Below in Figure 2, the DLT homography testing for each slide/frame pair is shown.

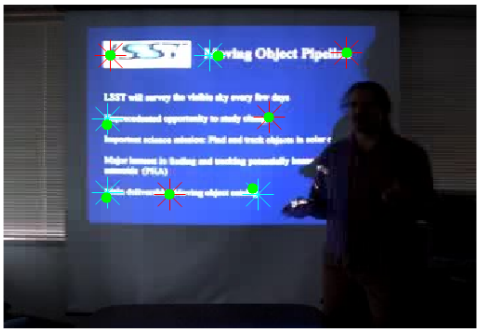


(a) slide1 w/ red/cyan clicked stars.

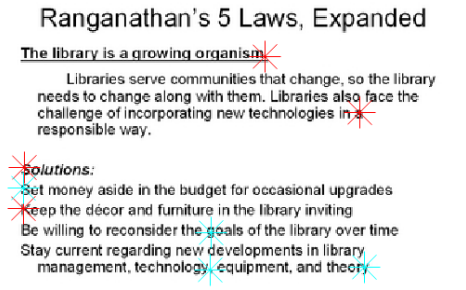
(b) frame1 w/ red/cyan clicked stars and green mapped dots.



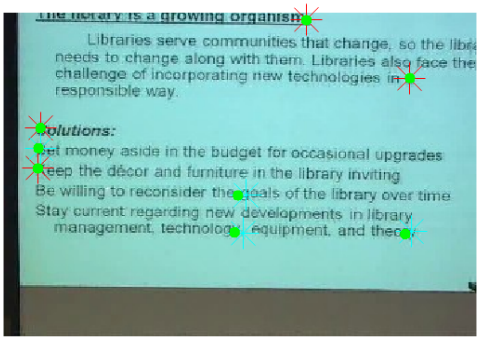
(c) slide2 w/ red/cyan clicked stars.



(d) frame2 w/ red/cyan clicked stars and green mapped dots.



(e) slide3 w/ red/cyan clicked stars.



(f) frame3 w/ red/cyan clicked stars and green mapped dots.

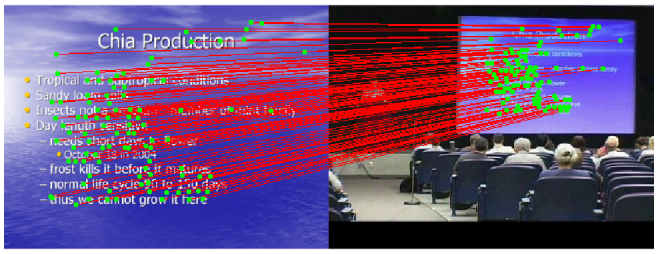
Figure 2: Clicked point pairs used in DLT homography are shown as red stars in the slide/frame pairs (4 red stars per slide/frame), clicked point pairs not used in homography are shown as cyan stars in the slide/frame pairs (4 cyan stars per slide/frame), and the mapped points from DLT computed homography are shown as green dots on the frames (8 green dots per frame).

From Figure 2 above, one can see that all mapped points (green dots) from DLT homography are relatively close to the clicked points (red/cyan stars) in each frame. Specifically, the green dots in each frame are directly on top of their corresponding red stars since these were the clicked point pairs used for homography. The green dots are also relatively close to their corresponding cyan star clicked points which were not used in homography.

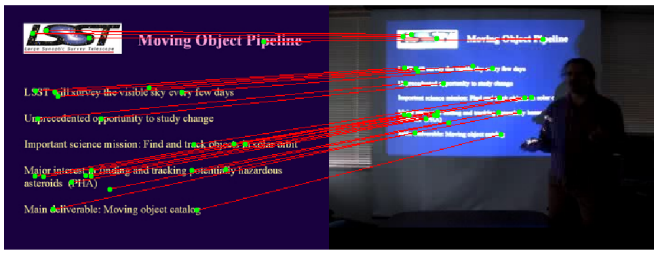
Initially, I had terrible homography for the cyan star point pairs (the green dots were nowhere near their corresponding cyan star points) because I naively used the first four out of the eight clicked point pairs for each slide/image which did not give a good spread of points to work with for homography. My solution to this was to use every other point pair in the list of eight as my four point pairs for homography, and it worked out well and resulted in Figure 2. Next time, I would make sure to select four point pairs in the slide/frame pairs that fit more to the dimensions of the slides.

C. Using RANSAC to improve keypoint matching.

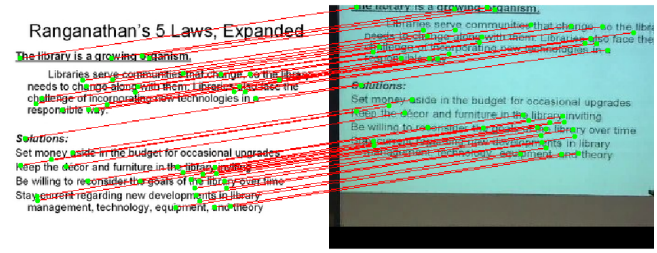
Homography with RANSAC improves keypoint matching based on my findings and by a visually significant margin compared to my results from assignment 9. Below in Figure 3, the results of homography with RANSAC to improve keypoint matching of SIFT vectors is shown.



(a) slide/frame 1 w/ green dot keypoint matches (homography inliers) connected by red lines.



(b) slide/frame 2 w/ green dot keypoint matches (homography inliers) connected by red lines.



(c) slide/frame 3 w/ green dot keypoint matches (homography inliers) connected by red lines.

Figure 3: The results of homography with RANSAC to improve keypoint matching of SIFT vectors from the slide/frame pairs of assignment 9 are shown. The green dots in the slide/frame pairs represent the keypoint matches that were in the homography inliers, and the red lines connecting the green dots signify the matching pairs.

As seen in Figure 3 above, the homography inliers used as the final keypoint matches betweeen slide/frame pairs seem to work well in that there are plenty of keypoint matches produced in each slide/frame pair and in that there are (from visual inspection) no incorrect keypoint matches whatsoever. This is noticeably better than any method used in assignment 9 to narrow down the matches because, from my experiments for that assignment, the vast majority of matches had to be discarded just to get a couple accurate matches for each slide/frame. In contrast, with RANSAC and homography, we see so many more accurate matches now that were missing before.

The parameters for the number of iterations was 99.9999% probability of finding the correct fit, 8% probability of finding a correct keypoint match (percentage that are homography inliers), and a minimum of 4 keypoint matches required for a fit. Using the number of iterations equation, this resulted in 337,286 iterations per slide/frame pair to produce the homography inlier keypoint matches as seen above in Figure 3.

Before using homography with RANSAC, `knnsearch()` was used to produce keypoint matches from the SIFT files from assignment 9 by getting the nearest neighbor for all SIFT points in the slide for all SIFT points in the frame. In each iteration of RANSAC, 4 keypoint matches were randomly chosen in an indefinite while-loop which would not break until all 4 keypoint matches were unique. Then, homography was computed from the four points and then refit to the top 8% of inliers found based on squared error of the homography versus the frame keypoints. The RMSE of the refitted homography was computed, and the homography inliers were saved if the RMSE was the best one seen so far.

Overall, homography with RANSAC produced better results than my methods in assignment 9, and I think the fact that the tuning parameters used to determine the number of iterations is so straight-forward makes this method easier to use. The

downside is that the number of iterations for RANSAC could become quite large depending on the parameters, but it seems that at least for the slide/frame pairs that we are using, the results will be more reliable.