16-720: Assignment 2

Abhinav Maurya
amaurya@andrew.cmu.edu

1.1: createGaussianPyramid

Already provided to us in `code/createGaussianPyramid.m` file.

1.2: createDoGPyramid

Included in `code/createDoGPyramid.m` file.

1.3: computePrincipalCurvature

Included in `code/computePrincipalCurvature.m` file.

1.4: getLocalExtrema

Included in `code/getLocalExtrema.m` file.

1.5: DoGdetector

Included in `code/DoGdetector.m` file. Results in figure (1).

2.1: makeTestPattern

Included in `code/makeTestPattern.m` file. Created pattern saved in file `code/testPattern.mat`.

2.2: computeBrief

Included in `code/computeBrief.m` file.

2.3: briefLite

Included in `code/briefLite.m` file.

2.4: plotMatches

Included in `code/plotMatches.m` file. Results in figure (2). We observe that rotation of the image greatly reduces the number of matches, which is observed in the matches of the original grayscale image with both the image showing all 3 textbooks in a rotated fashion as well as the three textbooks piled up and the PF textbook rotated and on top. We see a significant number of matches on the incline images where there is perspective rotation but no rotation in the plane of the image. Placing the book on the desk or against the stand leads to some reduction in the number of matches with the original grayscale textbook image. However, it is much lesser than the reduction in matches caused due to image rotation.

2.5: briefRotTest

Included in `code/briefRotTest.m` file. Results in figure (3).

3.1.a: Equation for computing homography H

Consider a single 3D point represented in two image capturing planes by two points $p = (x_1, y_1, 1)$ and $q = (x_2, y_2, 1)$ respectively. Our planar homography H maps p to q through an affine operation.

$$\therefore q \equiv Hp$$

$$\therefore \lambda q = Hp$$

If $H = [a, b, c; d, e, f; g, h, i]$, then substituting $p = (x_1, y_1, 1)$ and $q = (x_2, y_2, 1)$ in $\lambda q = Hp$ gives:

$$\lambda = gx_1 + hy_1 + i$$
$$\lambda x_2 = ax_1 + by_1 + c$$
$$\lambda y_2 = dx_1 + ey_1 + f$$

Substituting λ gives us the following two equations in the 9 unknows of matrix H:

$$(gx_1 + hy_1 + i) * x_2 = ax_1 + by_1 + c$$
$$(gx_1 + hy_1 + i) * y_2 = dx_1 + ey_1 + f$$

If we vectorize H to $h = [a, b, c, d, e, f, g, h, i]'$, then

$$A = [[x_1, y_1, 1, 0, 0, 0, -x_1x_2, -x_2y_1, -x_2]; [0, 0, 0, x_1, y_1, 1, -x_1y_2, -y_1y_2 - y_2]]$$

The linear equations from further point correspondences can be stacked down as further rows in A to obtain the final linear system $Ah = 0$ to solve.

3.1.b: Elements in H

We have seen that there are 9 unknown elements in $H = [a, b, c; d, e, f; g, h, i]$.

3.1.c: Required number of point correspondences to solve for H

Even though there are 9 unknown elements in $H = [a, b, c; d, e, f; g, h, i]$, there are only 8 degrees of freedom because we need to know H only up to a scaling factor λ. This is why we were able to resolve the three equations of $\lambda q = Hp$ to two by eliminating λ. Since each point correspondence gave us two equations and we have eight effective unknowns, we need *four* point correspondences to determine H.

3.1.d: Least squares solution for H

Instead of solving for $Ah = 0$ which may not have a solution due to the large number of constraint rows and noise in A, we instead solve for h to minimize the l_2 nor of Ah.

$$\hat{h} = argmin_{||h||_2=1} \ (||Ah||_2)^2 = argmin_{||h||_2=1} \ ||h'A'Ah||$$

Rayleigh Quotient for matrix M and vector x is given as

$$R(M, x) = \frac{x'Mx}{x'x}$$

Also, $R(M, cx) = R(M, x)$ i.e. the vector x that provides a particular value of Rayleigh Quotient for a given matrix M is defined up to a scaling factor.

Rayleigh Quotient Theorem establishes that the vector x that maximizes $R(M, x)$ is the eigenvector corresponding to the largest eigenvalue of M. Correspondingly, it also states that the vector x that minimizes $R(M, x)$ is the eigenvector corresponding to the smallest eigenvalue of M. If $M = A'A$, then the vector x that minimizes $R(A'A, x)$ is the smallest eigenvector of $A'A$ which we know is the smallest right singular vector of A. Hence, by calculating the SVD of A, we can calculate h that minimizes $||h'A'Ah||$. We do not have to worry about satisfying the norm condition $||h||_2 = 1$, because out of all minimizers of $R(A'A, x)$, SVD returns the smallest right singular vector which is normalized to already have a length of 1. Thus, we conveniently get a unique minimizer h with a fixed scaling factor to 1.

4.1: computeH

Included in `code/computeH.m` file.

5.1: imageStitching

Included in `code/imageStitching.m` file. Result in figure (4.a). Please zoom if result image is not clear.

5.2: imageStitching_noClip

Included in `code/imageStitching_noClip.m` file. Result in figure (4.b). Please zoom if result image is not clear.

6.1: ransacH

Included in `code/ransacH.m` file.

6.2: generatePanorama

Included in `code/generatePanorama.m` file. Results in figure (5). Please zoom if result image is not clear.

7.1: ec_1

Included in `code/ec_1.m` file. Results in figure (6). We decided to enhance the `computeBrief` function so as to include BRIEF descriptors on the rotated versions of the input image. We chose all rotations of the image spaced 30 degrees apart i.e. $30, 60, .., 360$. Figure (6.a) shows the number of matches without BRIEF enhancement while figure (6.b) shows the number of matches with the suggested BRIEF enhancement. We observe that the number of matches has significantly improved between the original image and its rotated version where rotation is on the X-axis of the figure, while keeping the number of matches between the original image and its unrotated version (at 360 degrees in the figure) fairly high.

7.2: ec_2

Included in `code/ec_2.m` file. Results in figure (7). We decided to enhance the `computeBrief` function so as to include BRIEF descriptors on the scaled versions of the input image. We chose the following scalings of the original image for enhancement $0.4, 0.7, 1.0$. Figure (7.a) shows the number of matches without BRIEF enhancement while figure (7.b) shows the number of matches with the suggested BRIEF enhancement. We observe that the number of matches has significantly improved between the original image and its scaled version where the applied scaling for the matched image is on the X-axis of the figure, while keeping the number of matches between the original image and its unscaled version (at 1.0 scaling in the figure) fairly high. It is odd that the absolute number of matches between the original image and its unscaled version is lower in the case of the BRIEF descriptor enhancement than in the case without the enhancement. This is not the case in the case of BRIEF enhancement to handle image rotations.

Chicken Broth Image	Duquesne Incline Image
	

Figure 1: Output of `DoGDetector`

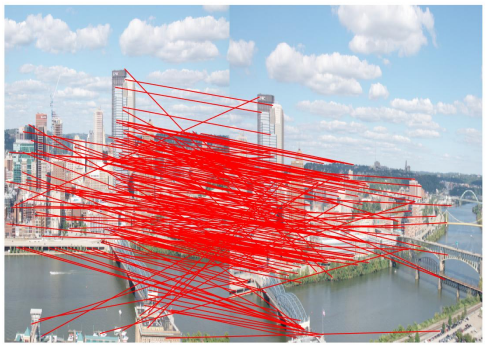
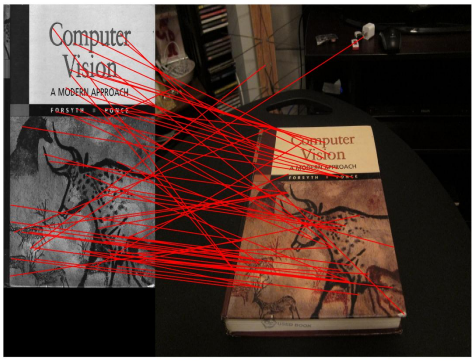
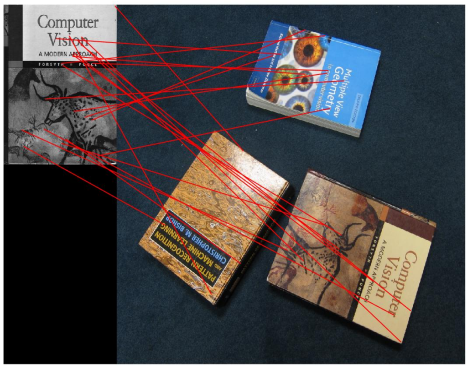
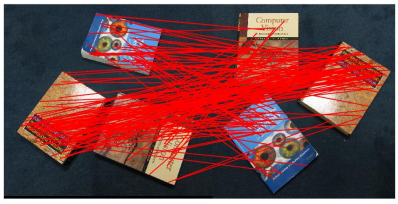
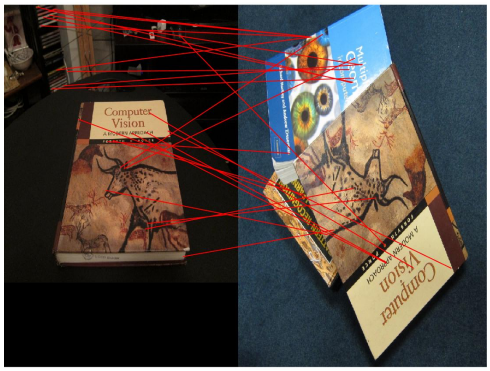
Incline Image Matches	PF Scaled and Desk Matches
	
PF Scaled and Floor Matches	**PF Scaled and Floor Rotated Matches**
	
PF Scaled and Pile Matches	**PF Scaled and Stand Matches**
	
PF Floor and Floor Rotated Matches	**PF Desk and Pile Matches**
	

Figure 2: Output of `testMatches`

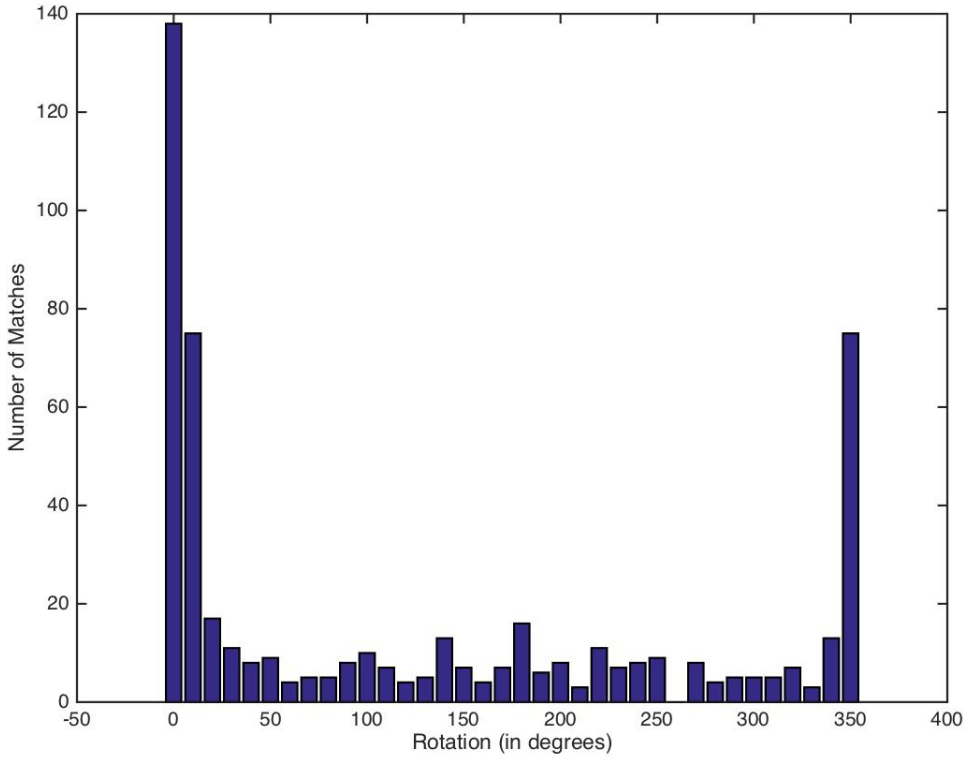
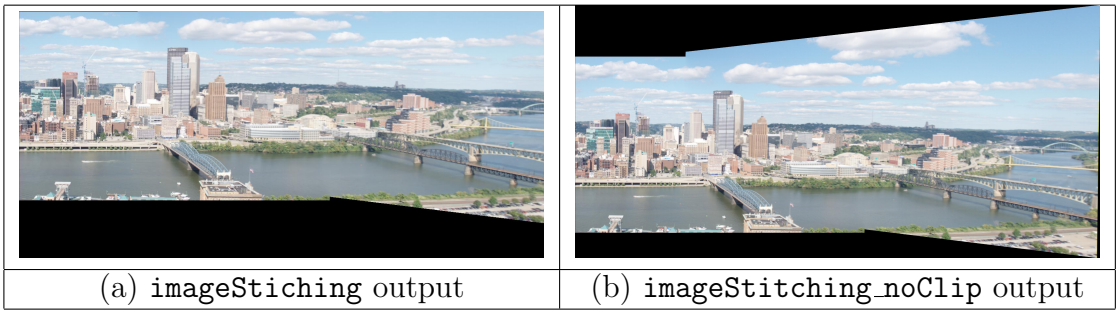
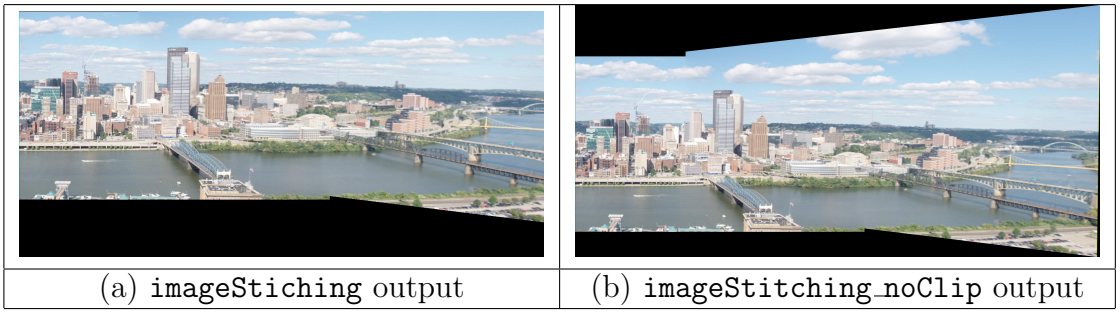


Figure 3: Output of `briefRotTest`



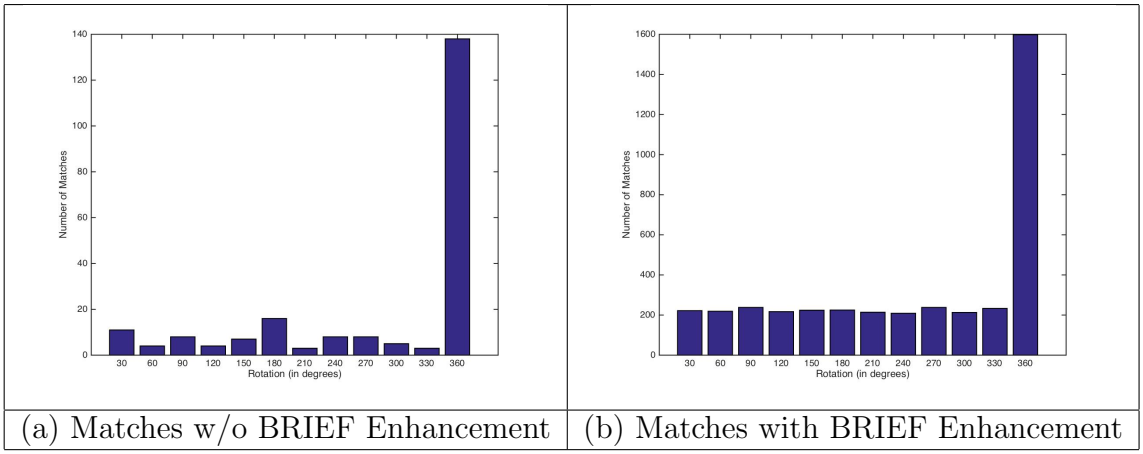
(a) `imageStiching` output (b) `imageStitching_noClip` output

Figure 4: Output of Image Stiching



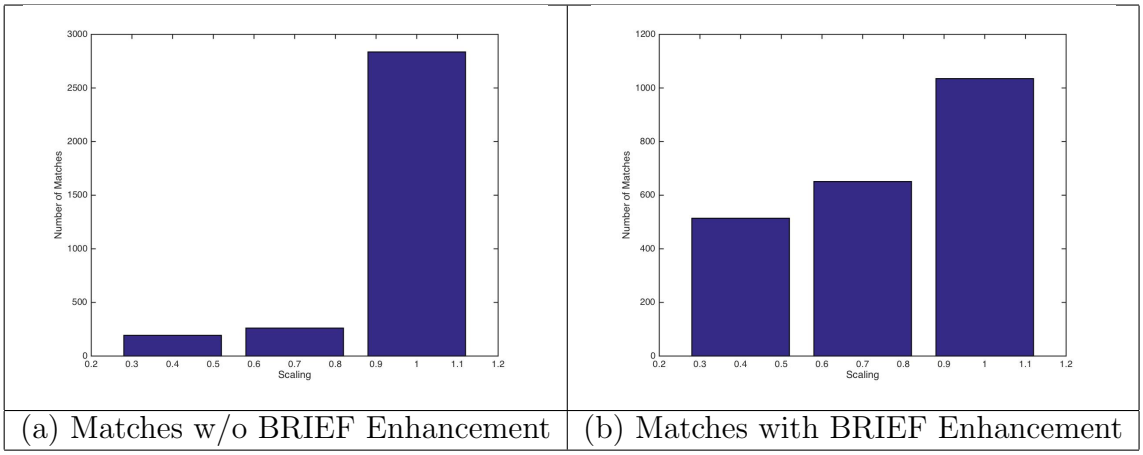
(a) `imageStiching` output (b) `imageStitching_noClip` output

Figure 5: Output of RANSAC Image Stiching



(a) Matches w/o BRIEF Enhancement (b) Matches with BRIEF Enhancement

Figure 6: Output of BRIEF Descriptor Enhancement for Rotation



(a) Matches w/o BRIEF Enhancement (b) Matches with BRIEF Enhancement

Figure 7: Output of BRIEF Descriptor Enhancement for Scaling