16-720B Computer Vision: Homework 2

Vigneshram Krishnamoorthy - vigneshk@andrew.cmu.edu

11th October 2017

1 Solution to Question 1

1.1 Q1.2 - Gaussian and DoG Pyramid Output

The createDoGPyramid function was implemented on the given Gaussian Pyramid image (As shown in Figure 1). The resultant DoGPyramid output from this function is given by Figure 2.

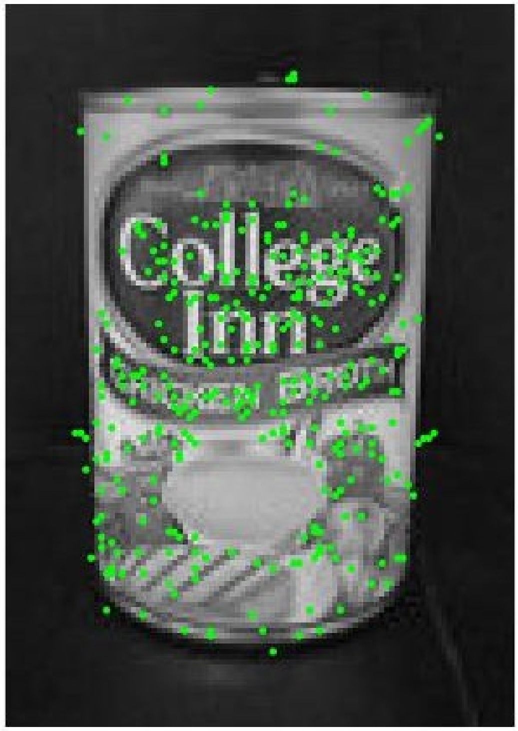
Figure 1: The Gaussian Pyramid Output for model_chickenbroth.jpg image

Figure 2: The Difference of Gaussian Pyramid Output for model_chickenbroth.jpg image

1.2 Q1.5 - DoG Keypoint Detector Output

The results of the DoG Keypoint Detector algorithm is shown in Figure 3.

Figure 3: The Difference of Gaussian Pyramid Output for model_chickenbroth.jpg image with edge supression

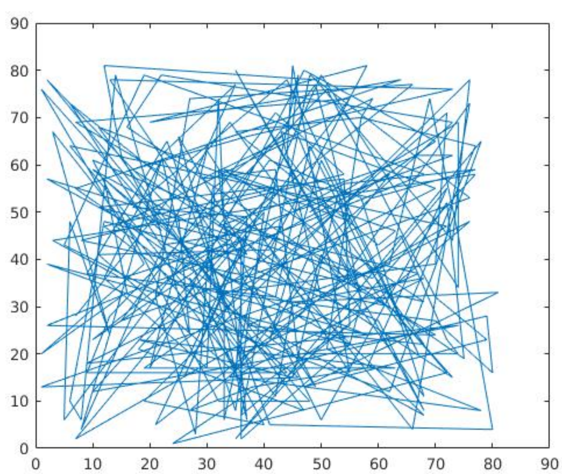


2 Solution to Question 2

2.1 Q2.1 Creating a set of BRIEF tests

According to the section 3.2 of the paper by "BRIEF : Binary Robust Independent Elementary Features by Calonder, Lepetit, Strecha and Fua", the uniform and gaussian distributions of the generated test set gives the best results. Here, I implemented the Type I - uniform distribution for X and Y. The testPattern is illustrated in Figure 4, and saved in testPattern.mat

Figure 4: A random set of BRIEF tests generated using uniform distribution



2.2 Q2.4 - BRIEF Descriptor Matching

Figure 5: The result of BRIEF feature matching on model_chickenbroth.jpg and chickenbroth_01.jpg

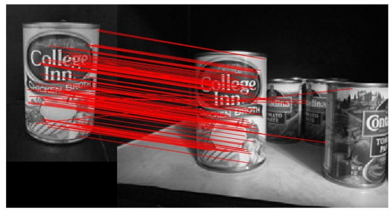
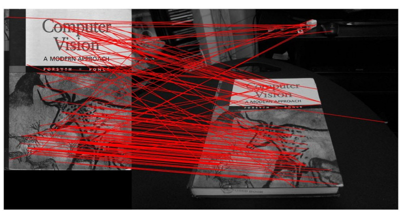
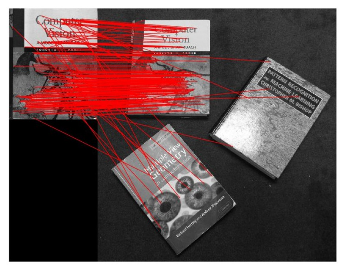


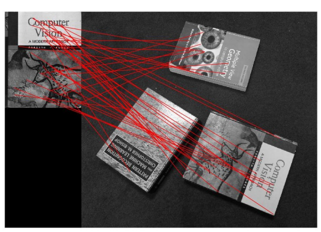
Figure 6: The result of matching BRIEF features on 'incline_L.png' and 'incline_R.png'.



(a) BRIEF matching with pf_desk.jpg

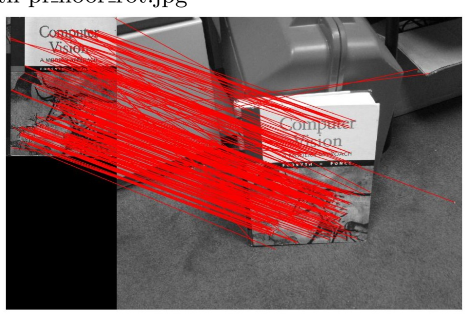


(b) BRIEF matching with pf_floor.jpg



(c) BRIEF matching with pf_floor_rot.jpg

(d) BRIEF matching with pf_pile.jpg



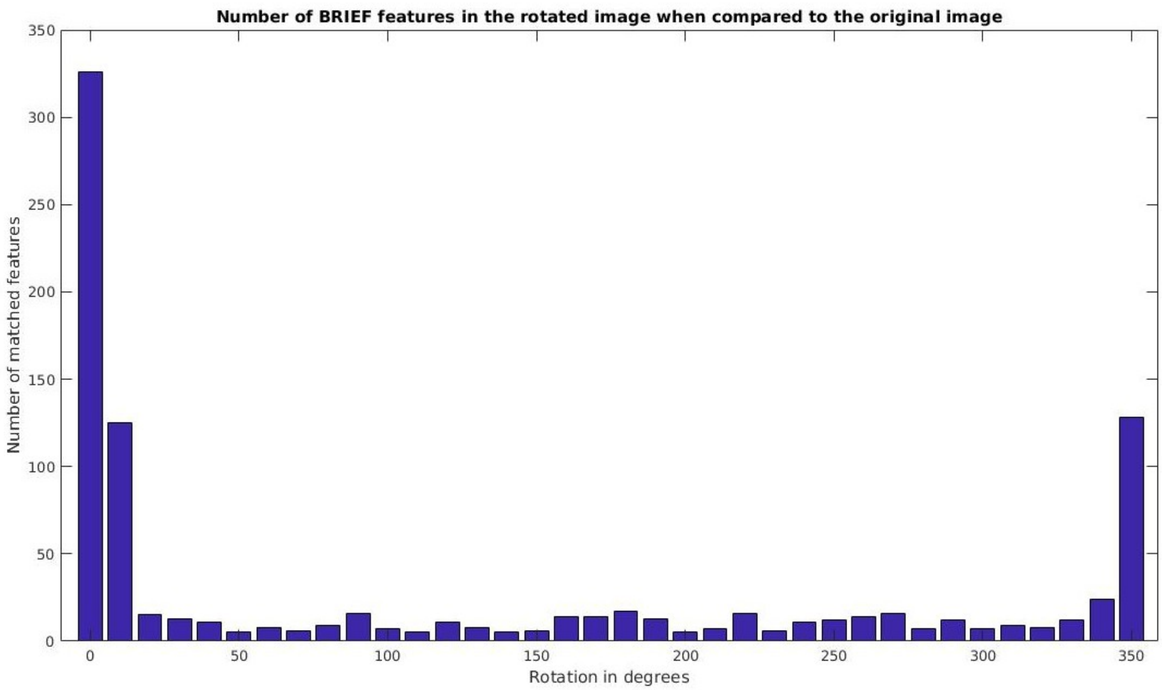
(e) BRIEF matching with pf_stand.jpg

Figure 7: The result of matching BRIEF features on pf_scan_scaled.jpg against the other 5 given images, as referred in the corresponding captions

We observe that in the cases where the two images of the book are negligibly rotated, the algorithm performs well. In case of rotation of the book (For example in pf_floor_rot.jpg), we see that the BRIEF matches are very less, and the algorithm doesn't perform well, the reason for which is discussed in Q2.5.

2.3 Q2.5 BRIEF and Rotations

Figure 8: Rotation in degrees vs Number of correct matches for 'model_chickenbroth.jpg'



We observe from the above plot that the number of correct matches has a huge drop when the image rotates. This can be attributed to the BRIEF implementation procedure which uses the pixel intensities of a patch (fixed points) around a keypoint. These pixel intensities change when the image is rotated. Hence, the keypoints in the rotated image can't be matched with the original image.

3 Solution to Question 3

3.1 Q3.1 (a)

$$\lambda_n \tilde{\boldsymbol{x}}_n = \boldsymbol{H} \tilde{\boldsymbol{u}}_n$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \times \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

$$x = \frac{h_{11}u + h_{12}v + h_{13}}{h_{31}u + h_{32}v + h_{33}}$$

$$y = \frac{h_{21}u + h_{22}v + h_{23}}{h_{31}u + h_{32}v + h_{33}}$$

Multiplying both sides with the denominator, we get

$$(h_{31}u + h_{32}v + h_{33})x = h_{11}u + h_{12}v + h_{13}$$

$$(h_{31}u + h_{32}v + h_{33})y = h_{21}u + h_{22}v + h_{23}$$

Rearranging the above, we obtain,

$$h_{31}ux + h_{32}vx + h_{33}x - h_{11}u - h_{12}v - h_{13} = 0$$

$$h_{31}uy + h_{32}vy + h_{33}y - h_{21}u - h_{22}v - h_{23} = 0$$

Rearranging the above, we get

$$\begin{bmatrix} -u & -v & -1 & 0 & 0 & 0 & ux & vx & x \\ 0 & 0 & 0 & -u & -v & -1 & uy & vy & y \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \mathbf{0}$$

Thus for all $i = 1 : N$, we have $\boldsymbol{a}^i \boldsymbol{h} = \mathbf{0}$, where

$$\mathbf{a}^i = \begin{bmatrix} -u & -v & -1 & 0 & 0 & 0 & ux & vx & x \\ 0 & 0 & 0 & -u & -v & -1 & uy & vy & y \end{bmatrix}$$

$$, \mathbf{h} = \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix}$$

For each point we get 2 independent equations as above. Therefore, for N points, we will have $2N$ independent equations and can form the matrix A as:

$$\mathbf{A} = \begin{bmatrix} \boldsymbol{a}^1 \\ \boldsymbol{a}^2 \\ . \\ . \\ . \\ \boldsymbol{a}^N \end{bmatrix}$$

with $\boldsymbol{Ah} = \mathbf{0}$, where $\mathbf{A}$ is of size $2N \times 9$, $\mathbf{h}$ is of size 9×1 and $\mathbf{0}$ is of size $2N \times 1$

3.2 Q3.1 (b)

From the above derivation, we have that the number of elements in **h** is 9. However, we have an additional constraint that $\|\boldsymbol{h}\| = 1$. Therefore, the degrees of freedom of H is only 8 (due to the constraint). We can force one of the elements to be a given value (scale factor), and hence reducing the number of elements in **h** to 8.

3.3 Q3.1 (c)

As mentioned in (b), we have a constraint that $\|\boldsymbol{h}\| = 1$, making **h** to have only 8 degrees of freedom. Each new point pair correspondence gives us 2 independent equations. Thus, we will require a minimum of 4 point pair (correspondences) to find all the elements of **h**.

3.4 Q3.1 (d)

To find a solution to minimize this homogeneous linear least squares system, we consider the sum squared error for $\boldsymbol{Ah} = \boldsymbol{0}$, which is $S = \|\boldsymbol{Ah}\|^2$

$$S = (\boldsymbol{Ah})^T(\boldsymbol{Ah}) = \boldsymbol{h}^T\boldsymbol{A}^T\boldsymbol{Ah}$$

To minimize S w.r.t $\boldsymbol{h}$:

$$S^{'} = 0 = (\boldsymbol{h}^T\boldsymbol{A}^T\boldsymbol{Ah})^{'}$$

Therefore, we get $\boldsymbol{A}^T\boldsymbol{Ah} = 0$. $\boldsymbol{h}$ cannot be $\boldsymbol{0}$ as it won't satisfy the constraint that $\|\boldsymbol{h}\| = 1$

We observe that the SVD decomposition of matrix **A** gives the eigenvectors of $\boldsymbol{A}^T\boldsymbol{A}$. Hence, we can use this relation to directly compute the eigenvector corresponding to the smallest eigenvalue of the matrix $\boldsymbol{A}^T\boldsymbol{A}$.

Steps to compute **h** using SVD:

- Compute the matrix **A** from the given set of N point pairs, by finding the individual $\boldsymbol{a}^i$ matrices as explained in Q3.1 (a). This matrix will be of size $2N \times 9$
- Perform SVD on **A** to get it of the form USV^T
- The solution for **h** corresponds to the last column of V, if the matrix S has eigenvalues arranged in the decreasing order.

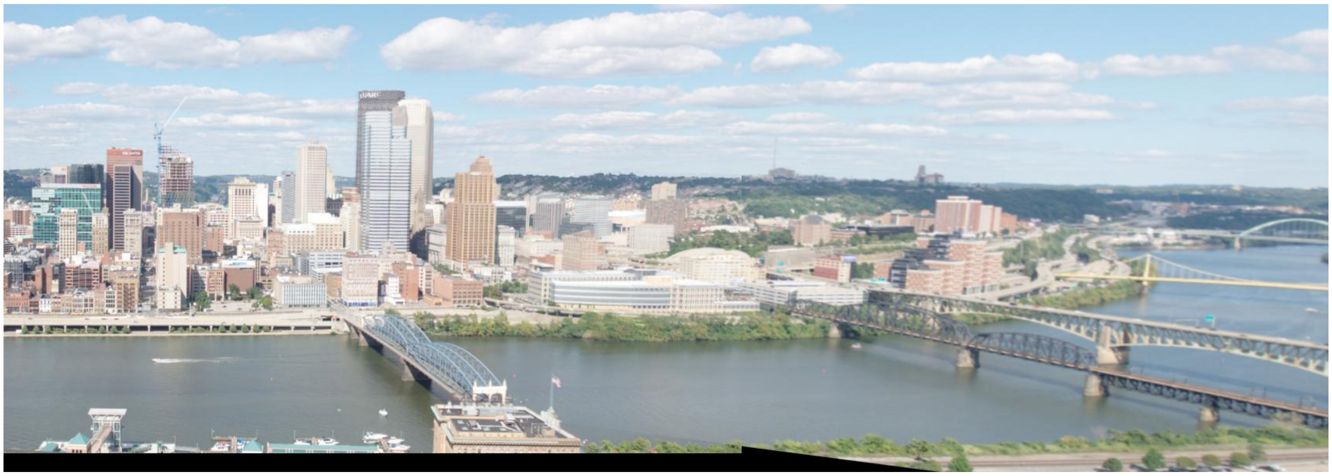
The function computeH.m was implemented based on the above steps.

4 Solution to Question 6

In order to blend two images given the images, I wrote a function blendingfn_max.m. This function blends the images by substituting the maximum intensity of a pixel from the given two images in each location of the overlapping region.

4.1 Q6.1

Figure 9: Result of stitching 'incline_L.jpg' and 'incline_R.jpg' with clipping and with estimating homography matrix H using RANSAC.



4.2 Q6.2

Figure 10: Result of stitching 'incline_L.jpg' and 'incline_R.jpg' without clipping and estimating homography matrix H without using RANSAC.

4.3 Q6.3

Figure 11: Result of stitching 'incline_L.jpg' and 'incline_R.jpg' without clipping and estimating homography matrix H using RANSAC.

5 Solution to Question 7

5.1 Q7.1

Using section 15.4.1 of Prince's textbook, the function compute_extrinsics was implemented using the given K and by finding $\mathbf{H}$ using $\mathbf{X}$ and $\mathbf{W}$. The code main_qn7.m computes H given $\mathbf{X}$ and $\mathbf{W}$, and then calls the function compute_extrinsics to calculate $\mathbf{R}$ and $\mathbf{t}$. The value of $\mathbf{R}$ and $\mathbf{t}$ are as follows:

$$\mathbf{R} = \begin{bmatrix} 0.99939 & 0.034044 & 0.0083701 \\ -0.029252 & 0.67818 & 0.73432 \\ 0.019323 & -0.73411 & 0.67875 \end{bmatrix}, \ \mathbf{t} = \begin{bmatrix} -10.6287 \\ -11.8362 \\ 45.3727 \end{bmatrix}$$

5.2 Q7.2

Figure 12: Projection of the sphere on the given image, with the bottom of the ball in the middle of 'o' in 'Computer'.