Assignment 2 Write up

Name: Kai Li

Andrew ID: kail2

1.1

Please refer to createGaussianPyramid.m for detailed implementation.

1.2

Please refer to createDoGPyramid.m for detailed implementation.

1.3

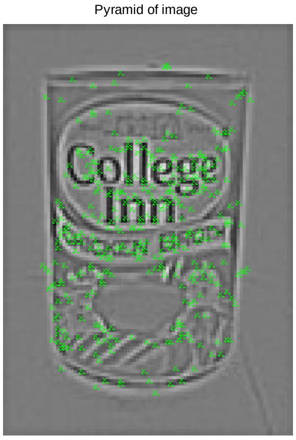
Please refer to computePrincipalCurvature.m for detailed implementation.

1.4

Please refer to getLocalExtrema.m for detailed implementation.

1.5

Please refer to DoGdetector.m for detailed implementation.



2.1

Please refer to makeTestPattern.m for detailed implementation.
‘testPattern.mat’ file is included in my submission.

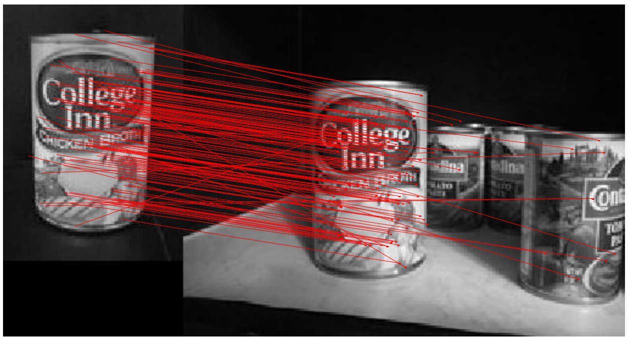
2.2

Please refer to computeBrief.m for detailed implementation.

2.3

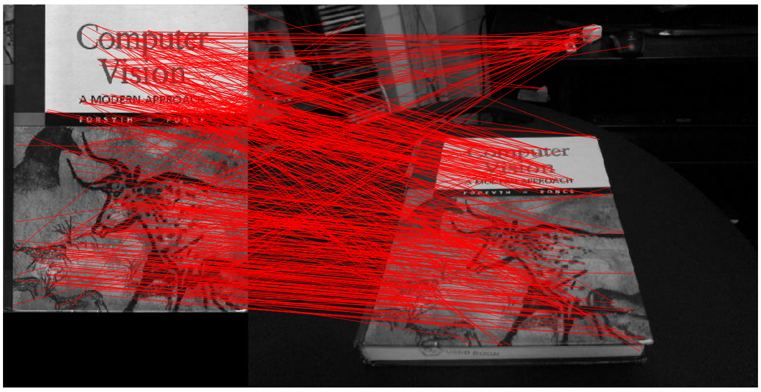
Please refer to briefLite.m for detailed implementation.

2.4

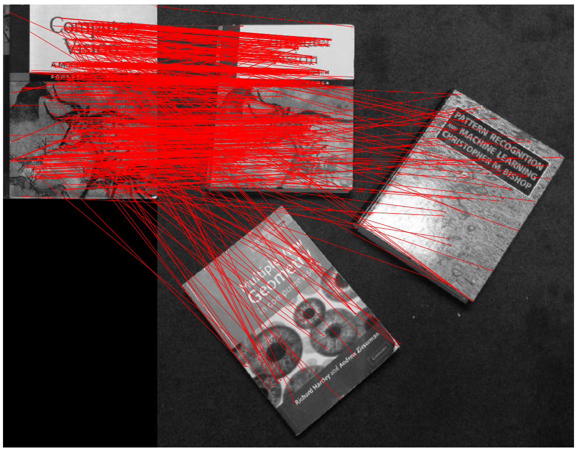


model_chickenbroth.jpg and chickenbroth_01.jpg

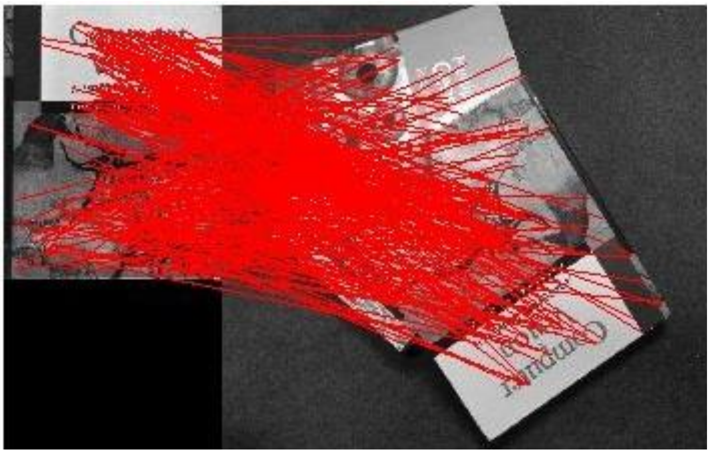
incline_L.png and incline_R.png



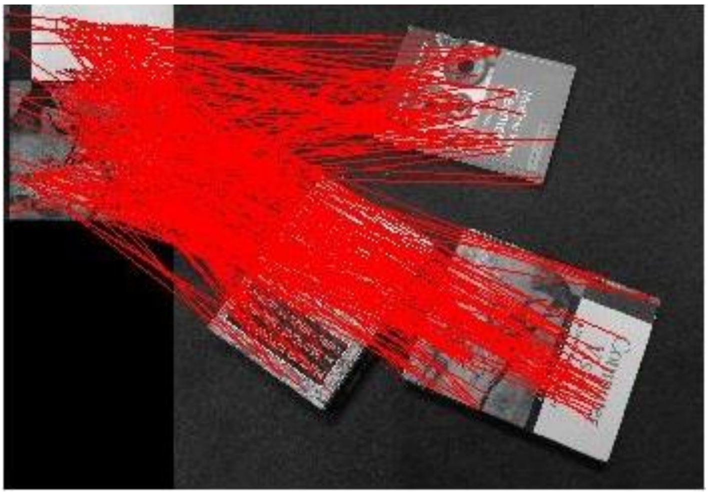
pf_scan_scaled.jpg and pf_desk.jpg


PATTERN RECOGNITION AND MACHINE LEARNING
CHRISTOPHER M. BISHOP
Multiple View Geometry
Richard Hartley and Andrew Zisserman

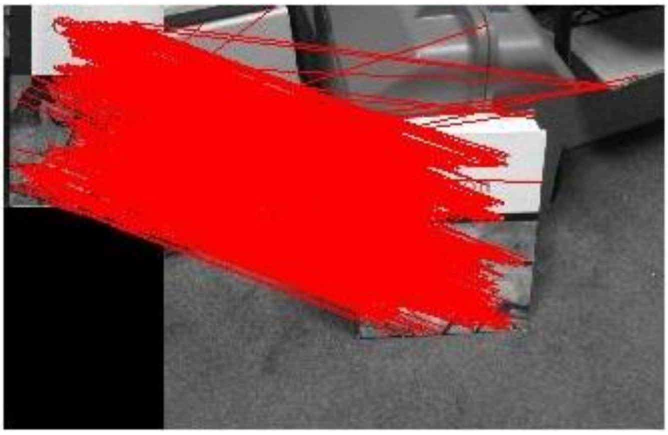
pf_scan_scaled.jpg and pf_floor.jpg



pf_scan_scaled.jpg and pf_pile.jpg



pf_scan_scaled.jpg and pf_floor_rot.jpg



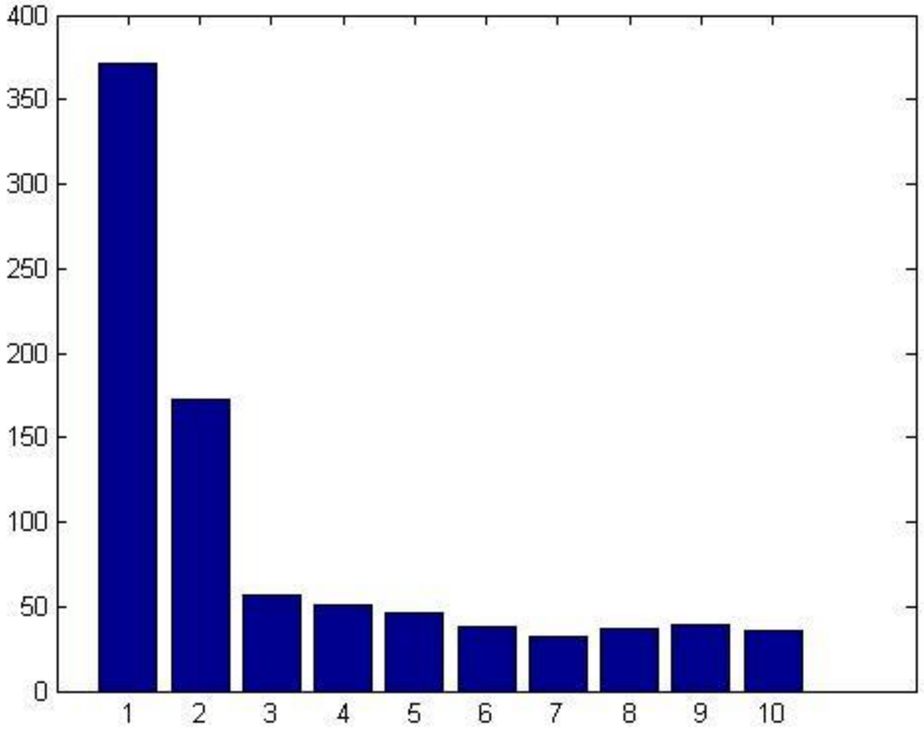
pf_scan_scaled.jpg and pf_stand.jpg

When there is only translation involved, the mapping works pretty well, especially with similar scale (for example the first and second cases shown above); and if we rotate the image, with the rotation angle become larger and larger, the mapping results become worse and worse; also worth notice that the mapping cannot fully handle the scaling of an image, that is to say, if we scale the test image too much, the mapping will have much more errors compared to original size.

2.5

Here because of the time limit, I rotate the image from 0---90 degree.
The following bar is the amount of mappings. We can see with the rotate degree

increases, the amount of mapping decrease. (From 0—90 degree) This is because of the BRIEF descriptor cannot handle well with the rotated image, so that the error will increase if we use the original image mapping with the rotated image.



3

Q 3.1

(a) We know there exists $p^i \equiv Hq^i$ for $i \in \{1, 2 \cdots N\}$.

In homogeneous wordinates:

$$\begin{pmatrix} a x_i \\ a y_i \\ a \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix}$$

$$x_i = \frac{a x_i}{a} = \frac{h_{11} u_i + h_{12} v_i + h_{13}}{h_{31} u_i + h_{32} v_i + h_{33}}$$

$$y_i = \frac{a y_i}{a} = \frac{h_{21} u_i + h_{22} v_i + h_{23}}{h_{31} u_i + h_{32} v_i + h_{33}}$$

Therefore

$$\begin{bmatrix} u_i & v_i & 1 & 0 & 0 & 0 & -x_i u_i & -x_i v_i & -x_i \\ 0 & 0 & 0 & u_i & v_i & 1 & -y_i u_i & -y_i v_i & -y_i \\ & & & & \vdots & & & & \end{bmatrix} \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

(b) There are 9 elements in h.

(c) $\because \|h\|^2 = 1$, and each point pairs offer 2 equations;

Therefore 4 point pairs are required to solve this problem
(at least)

(d) We already have $Ah = 0$

$h = \arg\min_h \|Ah\|$ subject to $\|h\| = 1$, rewrite constraint as $1 - h^T h = 0$.

Then we want to minimize $\|Ah\|$ with constraint $1 - h^T h = 0$.

$$\frac{\partial}{\partial h}\left(h^T A^T A h + \lambda (1 - h^T h)\right) = 0$$

Therefore $2A^T A h - 2\lambda h = 0$

$(A^T A - \lambda I)h = 0$, h is an eigenvector of $A^T A$, and λ is eigenvalue.

We want least-squares error $e = \|Ah\|^2 = h^T A^T A h$
$= h^T \lambda h$
to be minimal.

Therefore we want λ to be the minimal eigenvalue of matrix $(A^T A)$.

Next we conduct SVD for A.

Let $A = USV^T$, because S is diagonal and the elements are sorted descendently.

Therefore the last column of matrix V will be h, which related to the minimal λ.

4

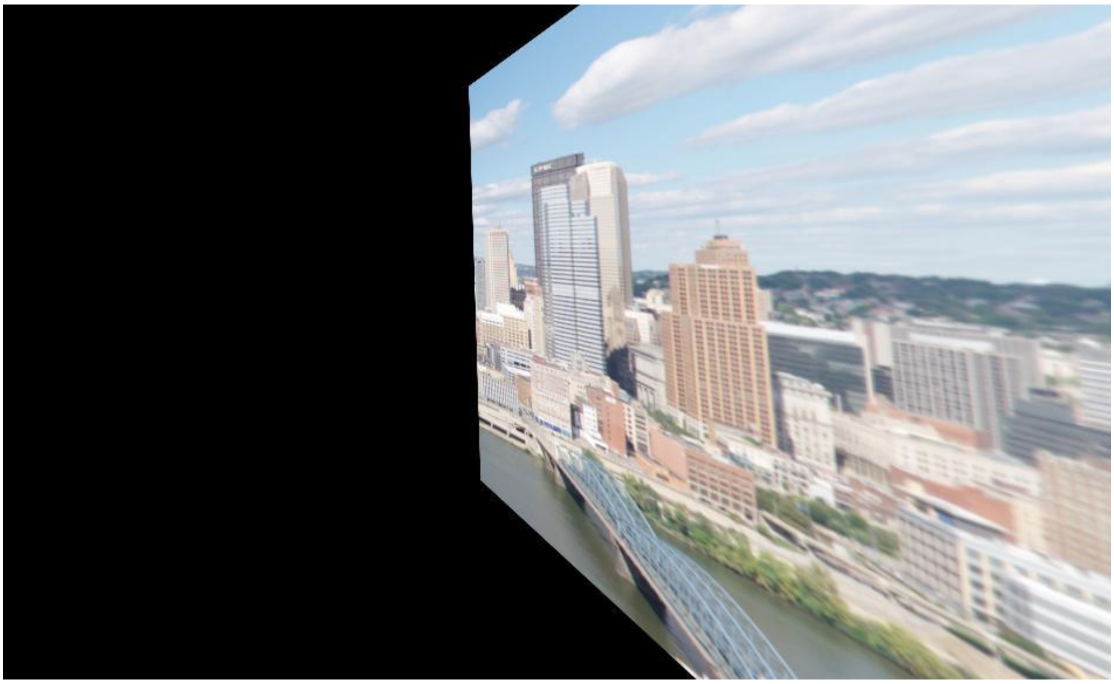
Please refer to computeH.m for detailed implementation.

5.1

Please refer to imageStitching.m for detailed implementation.

Already saved the q5_1.jpg and q5_1.mat in my submission.

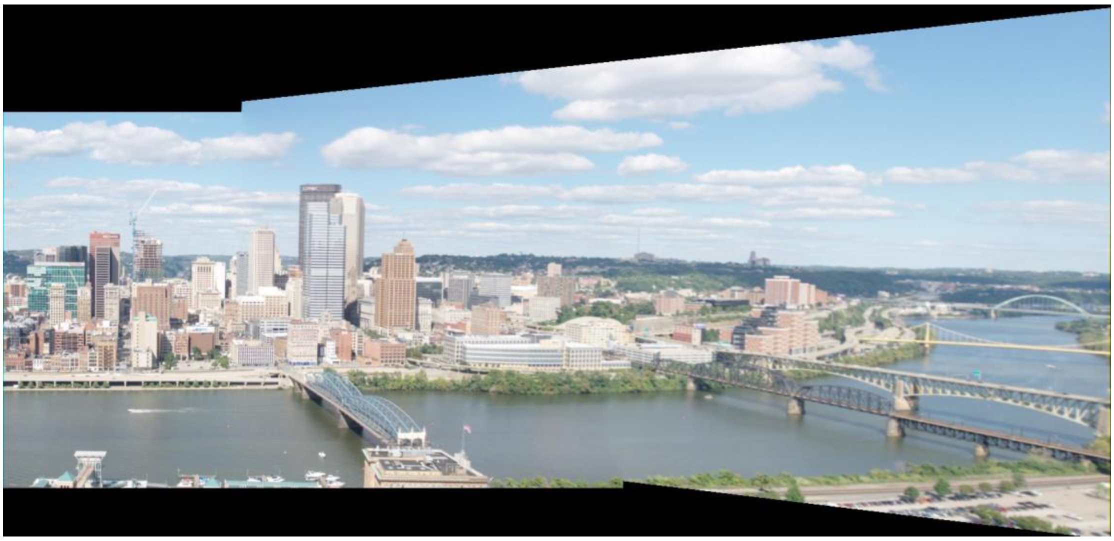
Note: Because without Ransac, the error in computeH will ruin the wrapped image, so what I did here is to manually choose four correct mapping point pairs using ginput(), and formulate my H matrix, therefore I can the following wrapped image:



‘q5_1.jpg’

5.2

Please refer to imageStitching_noClip.m for detailed implementation.
Already saved ‘q5_pan.jpg’ in my submission.



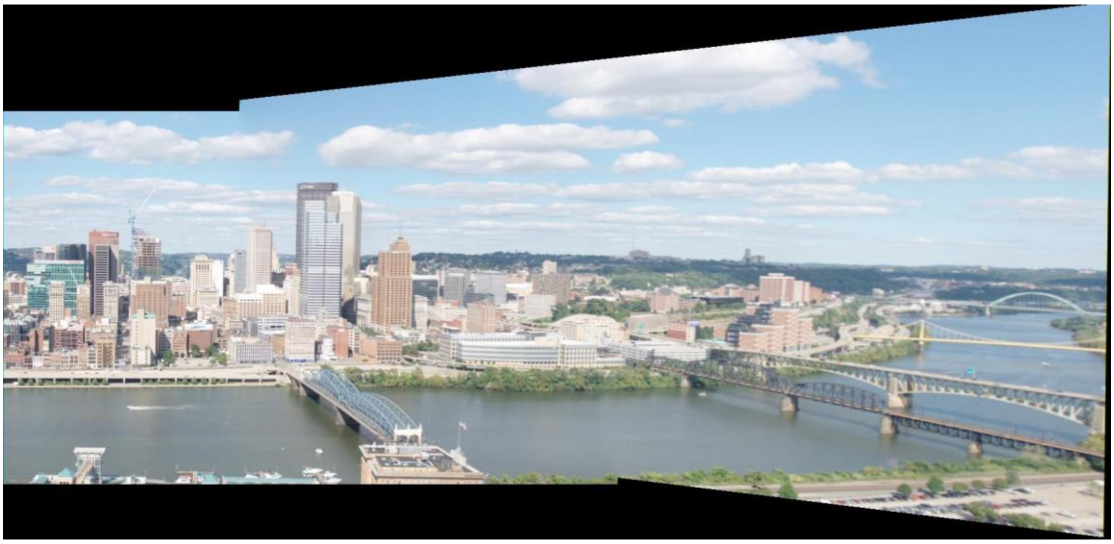
‘q5_pan.jpg’

6.1

Please refer to ransacH.m for detailed implementation.

6.2

Please refer to generatePanorama.m for detailed implementation.



‘q6_2.jpg’