16-720 Computer Vision: Homework 2 (Spring 2020)
Augmented Reality with Planar Homographies

PROBLEM 1.1 Homography

Homography → Prove existence of H in two point images (x_1) & (x_2) of two cameras with projection matrices P_1 & P_2 respectively.

$x_1 = H x_2$.

Let O be a point in the homogeneous co-ordinates $[X_i, Y_i, Z_i, 1]$

The image of O in P_1 → $x_1 = P_1 O$
The image of O in P_2 → $x_2 = P_2 O$

$\therefore \quad P_1^{-1} x_1 = P_2^{-1} x_2$

$x_1 = P_1 P_2^{-1} x_2$

$P_1 P_2^{-1} = H$

$\therefore \quad x_1 = H x_2$

$x_1 = H x_2$ ... Equation is correct to a scaling factor.

Hence proved.

PROBLEM 1.2

1.

The ~~total~~ total degrees of freedom in h is $\underline{\underline{8}}$. The value is total number of elements in the matrix from which one is subtracted for scaling factor.

Problem 1.2

2.

Total of eight points which is four point pairs is required to solve for h.

Problem 1.2

3.

Derivation of A_i

We are given the equation,

$$x_1^i \equiv H x_2^i \quad \text{——— (1)}$$

$$x_2 = \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}, \quad x_1 = \begin{bmatrix} a \\ b \\ 1 \end{bmatrix}, \quad H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

To remove $\equiv$ relation scale factor λ used as follows:

$$\begin{bmatrix} \lambda a \\ \lambda b \\ \lambda \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

Putting $\lambda = 1$ & dividing 1^{st} & 2^{nd} row with third row we get

$$-h_{11}p - h_{12}q - h_{13} + (h_{31}p + h_{32}q + h_{33})a = 0 \quad \text{——— (2)}$$

$$-h_{21}p - h_{22}q - h_{23} + (h_{31}p + h_{32}q + h_{33})a = 0 \quad \text{——— (3)}$$

Equations (2) & (3) can be written in matrix form as

$$A_i h = 0$$

$$A = \begin{bmatrix} -p & -q & -1 & 0 & 0 & 0 & ap & aq & a \\ 0 & 0 & 0 & -p & -q & -1 & bp & bq & b \end{bmatrix}$$

$$h = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} & h_{33} \end{bmatrix}^T$$

Problem 1.2

4.

Trival solution for $Ah=0$ would be
$h=[0\,0\,0\,\cdots\,0]^T$ where size of h is 9×1.
Two rows of A cannot be written as a
linear combination of each other therefore A is
full rank. The rank of A plus the nullity
of A is equal to dimensions of square
matrix A. The nullity is the dimension of
the matrix kernel which is all vectors of
the form $Av=0v$. This shows that the eigen
eigen space of eigen value zero is the
kernel of A. The rank will be n minus
dimension of eigenspace corresponding to 0.
If 0 is not an eigen value, then the kernal
is trivial & matrix is full rank n. Rank
depends on eigen values.

Problem 1.3

Given:

Camera 1 : $x_1 = K_1 [I \; 0] X$ I = Identity matrix

Camera 2 : $x_2 = K_2 [R \; 0] X$ R = rotation matrix

X is a point in 3D space = $\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$

On removing zero column and corresponding row from camera 1 equation x_1 we get

$$x_1 = K_1 I X = K_1 X \Rightarrow X = K_1^{-1} x_1$$

Similarly for x_2 on removing zero column we get

$$x_2 = K_2 R X$$

$$X = R^{-1} K_2^{-1} x_2$$

$$x_1 = K_1 R^{-1} K_2^{-1} x_2 \quad \text{———} \quad (1)$$

Therefore by comparing the equation (1) with $x_1 = H x_2$

we get,

$$H = K_1 R^{-1} K_2^{-1}$$

Therefore we can say that there exists a homography H that satisfies

$x_1 \equiv H x_2$

1.4.

Understanding homographies under rotation.

We have found from 1.3 that

$$H = K_1 R^{-1} K_2^{-1}$$

$$H^2 = (K_1 R^{-1} K_2^{-1})(K_1 R^{-1} K_2^{-1})$$

As K_1 & K_2 are constant we can write $H^2 = K R^{-1} R^{-1} K^{-1}$

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R^{-1} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R^{-1} \cdot R^{-1} = \begin{bmatrix} \cos^2\theta - \sin^2\theta & \cos\theta\sin\theta + \sin\theta\cos\theta & 0 \\ -(\sin\theta\cos\theta + \cos\theta\sin\theta) & -\sin^2\theta + \cos^2\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$H^2 = K \begin{bmatrix} \cos 2\theta & \sin 2\theta & 0 \\ -\sin 2\theta & \cos 2\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} K^{-1}$$

Therefore H^2 is the homography corresponding to a rotation of 2θ.

1.5

Planar homography is not completely sufficient to map arbitrary scene to another viewpoint as its repeated pattern handling is not efficient.

1.6

To prove 3D line is preserved in 2D.
Let us consider a 3D line with co-ordinates as $[0,0,1]^T$, $[1,1,1]^T$ & $[4,4,1]^T$.

Perception matrix $P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

On multiplying P with line in 3D space

$$x = \underset{4\times 4}{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}} \underset{4\times 3}{\begin{bmatrix} 0 & 1 & 4 \\ 0 & 1 & 4 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}}$$

$$x = \begin{bmatrix} 0 & 1 & 4 \\ 0 & 1 & 4 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

From the above x value we can see that line is projected to xY plane & line is still preserved. Thus line is preserved.

Problem 2.

2.1.1 Fast Detector

FAST detector and Harris corner detector are used to detect corners in a scene based on the change in intensity value. Fast detector samples a pixel and considers a 16 pixel circle around it, a threshold is defined on which change in intensity based on which pixel intensities out of four pixels on the axis are checked. The decision of the corner is made based on if the intensity of chosen pixel is above or below the threshold. Harris corner detector on the other hand uses a sliding window to mover over a region. FAST detector has a lower computational cost than Harris corner detector as it uses less sampled points for detection.

2.1.2 Brief Descriptor

The filter banks seen in the lectures requires a lot of computations to find binary strings whereas by utilizing less memory, faster matching and higher recognition rate, BRIEF Descriptor is an easy way to get binary descriptors. BRIEF is very fast both to build and to match. It does that by comparing intensities of the selected location pairs from the part of image with smooth patch.

2.1.3 Matching Methods

Binary strings in BRIEF Descriptor that are used to match features can use Hamming distance as a metric for computing the match.

Nearest Neighbor: Make two sets. From the first image, pick N interest points and put them in first set. Then from ground truth data, deduce the corresponding points in the other and put it in 2nd set. After computing the 2N associated descriptors, for each point in first set, use Nearest neighbor to find the second one and call it a match.

Hamming distance as compared to Euclidean distance provides better speed-up in measuring distance because finding hamming distance is just applying XOR and bit count, which are very fast in modern CPUs with SSE instructions.

2.1.4 Feature Matching

Output of feature matching using the default parameters (sigma=0.2, Ratio=0.7) is given as below

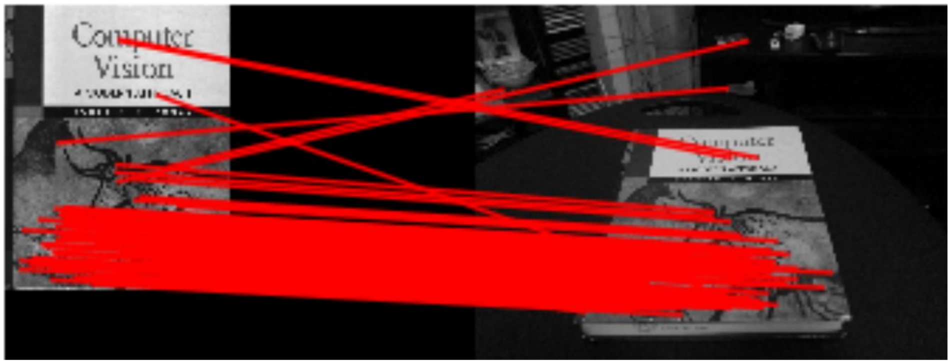
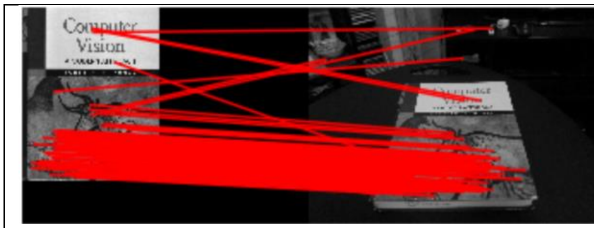
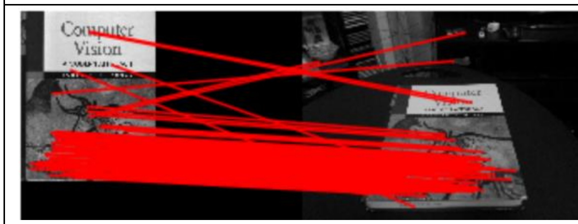
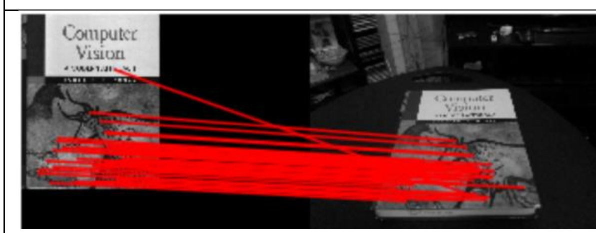
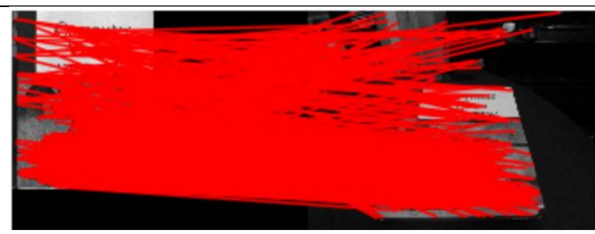


Figure 1

2.1.5 Feature Matching Parameter Tuning

Parameter Turning:

Table 1 Parameter Tuning

 Sigma-0.05 Ratio-0.7	 Sigma-0.12 Ratio-0.7
 Sigma-0.7 Ratio-0.7	 Sigma-0.2 Ratio-0.1
 Sigma-0.2 Ratio-0.6	 Sigma-0.05 Ratio-0.9

Observation:

From the above figures it can be seen that at lower value of sigma there are more matches outside the book. Also, as the value ratio is lowered the number of matches reduces.

2.1.6 Brief and Rotations

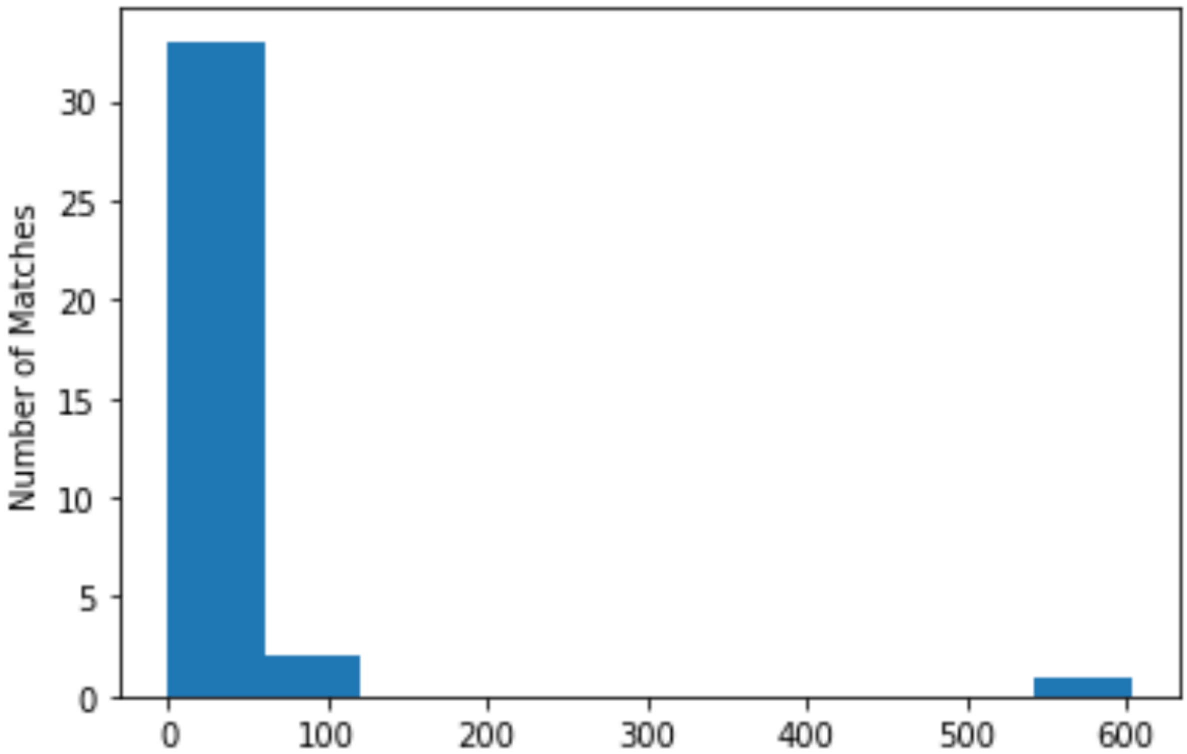


Figure 2 Histogram of number of matches

From the above histogram it can be seen as the image is rotated the number of features mapped dips significantly. From this we can conclude that even if brief descriptor is fast in computation it is unable to detect similar features therefore not good in feature matching.

Figure 3 Maximum match

-When the images are besides each other in the same orientation the maximum matches are observed

2.2.4 Putting it together

The below image is obtained on running harrypotterize.py with default max_iter=500 and inlier_tolerance=2

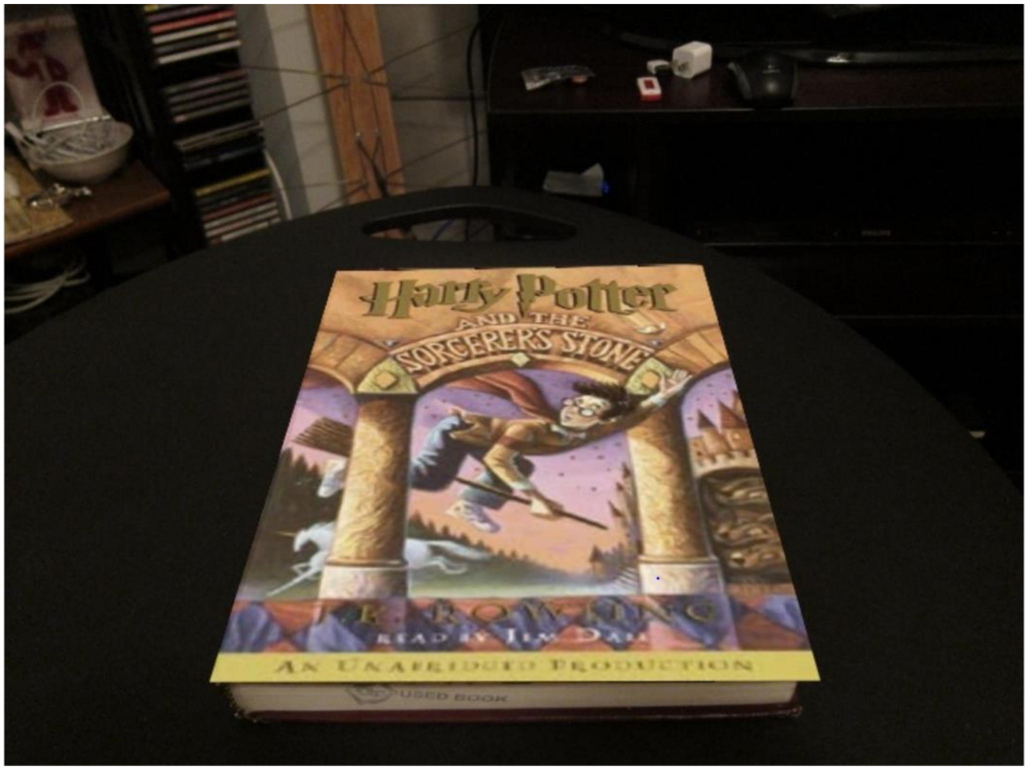


Figure 4 Harrypotterize using default values

Number of inliers=107

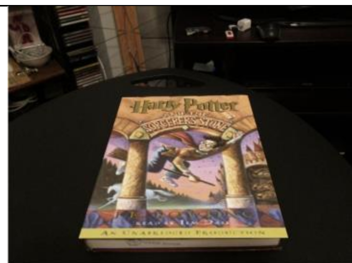
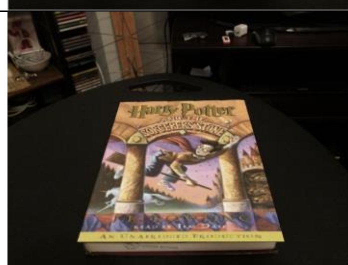
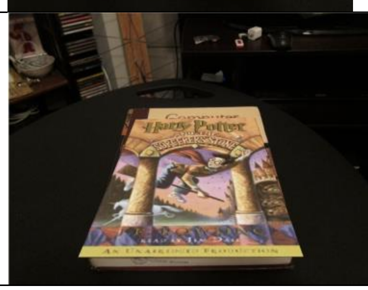
Accuracy=93.85%

2.2.4 Ransac Parameter Tuning

Table 2- Parameter Tuning

Sr No	Maximum Iteration	Inlier Tolerance	Number of inliers
1	5000	5	107
2	500	1	97
4	5000	30	108

Table 3 Parameter Tuning with description

1.		-As the number of iterations are increased the number inliers is not affected upto a certain value
2		-It can be seen that as the number of inlier tolerance is reduced the number of inliers drops
3		-On increasing the inlier tolerance tremendously the number of inliers increases but the image gets distorted