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

Han-Ting (Quentin) Cheng
Andrew id: hantingc

February 23, 2020

1 Homographies

Q1.1 Homography

$\mathbf{x_1^i} \equiv P_1\mathbf{x}_\pi^\mathbf{i}$, $\mathbf{x_2^i} \equiv P_2\mathbf{x}_\pi^\mathbf{i}$, because π is a plane, z coordinate is necessarily 0 and we can reduce the matrix to as shown below.

$$\mathbf{x_1} = P_1\mathbf{x}_\pi = \begin{bmatrix} f_1 & 0 & 0 \\ 0 & f_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_1 & r_4 & p \\ r_2 & r_5 & q \\ r_3 & r_6 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \tag{1}$$

The two matrices can be multiplied together to form H, so the equations become
$\mathbf{x_1} = H_1\mathbf{x}_\pi$ and $\mathbf{x_2} = H_2\mathbf{x}_\pi$.
Invert H_2 and plug $\mathbf{x}_\pi$ in to get $\mathbf{x_1} = H_1H_2^{-1}\mathbf{x_2}$.
$H = H_1H_2^{-1}$ thus exists because we know H is 3x3 and invertible.

Q1.2 Correspondences

1. 8 degrees of freedom because homogeneous coordinates take away 1.
2. Each point pair gives two equations, so 4.
3.
Let $\mathbf{H} = \begin{bmatrix} h_1^T \\ h_2^T \\ h_3^T \end{bmatrix}$, $\mathbf{x_1^i} \equiv H\mathbf{x_2^i}$, where $\mathbf{x_1^i} = \begin{bmatrix} x_1^i \\ y_1^i \\ 1 \end{bmatrix}$ Multiplying it out and writing separate equations for x and y of $\mathbf{x_1}$ gives $x_1^i = \frac{h_1^T\mathbf{x_2^i}}{h_3^T\mathbf{x_2^i}}$ and $y_1^i = \frac{h_2^T\mathbf{x_2^i}}{h_3^T\mathbf{x_2^i}}$. Then we can rearrange to be written in matrix form $\mathbf{A_i h} = 0$

$$\begin{bmatrix} x_2^i & y_2^i & 1 & 0 & 0 & 0 & -x_1^ix_2^i & -x_1^iy_2^i & -x_1^i \\ 0 & 0 & 0 & x_2^i & y_2^i & 1 & -y_1^ix_2^i & -y_1^iy_2^i & -y_1^i \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} = 0 \tag{2}$$

Where A is the left matrix.

4. A trivial solution is $\mathbf{h} = 0$. $\mathbf{A}$ is not full rank, because $\mathbf{h}$ only has 8 degrees of freedom, so $\mathbf{A}$ only has up to rank 8. This means the unique eigenvalue and eigenvector pairs of $\mathbf{A}$ will be n less where n = dimension of null space of $\mathbf{A}$.

1.1 Theory Questions

Q1.3

Since last columns are 0, X will lose a dimension and equations are simplified to $x_1 = K_1\tilde{X}$, $x_2 = K_2R\tilde{X}$ where $\tilde{X}$ is X with last dimension dropped.

$\Rightarrow (K_2R)^{-1}x_2 = \tilde{X}$

$\Rightarrow x_1 = K_1(K_2R)^{-1}x_2$

Thus $\underline{H = K_1(K_2R)^{-1}}$ exists.

Q1.4.

When $R_\theta = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $R_\theta \cdot R_\theta = \begin{pmatrix} \cos2\theta & -\sin2\theta & 0 \\ \sin2\theta & \cos2\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} = R_{2\theta}$

Same can be shown for any Rotational Matrix.

$x_1 = K[R \mid 0]\, x_2$

Similar to previous question.

$x_1 = KR\tilde{x}_2$

where $H = KR$.

$H^2 = KR_\theta R_\theta^T K^T = KR_{2\theta}K^T$

here K^T drops out

$\Rightarrow \underline{H^2 = KR_{2\theta}}$

Q1.5 Limitations of the planar homography

Because in planar homography we assume the scene lies on a 2D plane, thus simplifying a 3D problem. This can not always be assumed for any arbitrary scene.

Q1.6 Behavior of lines under perspective projections

We derived in class that for three points in 3D space that are in a line, $x_3^T(x_1 \times x_2) = 0$. After projection, we get $Px_3^T(Px_1 \times Px_2)$, which after pulling out P we find also equal to 0, thus meaning the three points are still in a line and that lines are preserved after perspective projections.

2 Computing Planar Homographies

2.1 Feature Detection and Matching

Q2.1.1 FAST Detector

The FAST detector looks at a circle of 16 pixels around a point, and checks whether enough pixels have intensity differences with the point of interest larger than sigma to classify corners. Harris corners, in contrast, looks at the gradients of an image to find corners. FAST is faster than Harris corner, hence the name, because it doesn't need to compute gradients and only need simple subtractions.

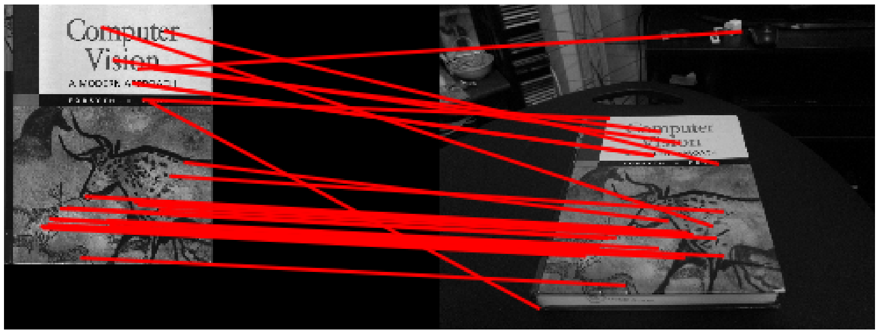
Q2.1.2 BRIEF Descriptor

The BRIEF descriptor describes a point with the patch around it by randomly selecting points around the point of interest, and comparing their pixel intensities to create a binary feature vector of 1's and 0's. The filterbenks method describes a point with a vector of itself after filtered, thus in a way also describing it with points around it. We can use any filerbank as a descriptor, but whether it works well depends on the specific cases.

Q2.1.3 Matching Methods

Hamming distance: While comparing two binary strings, Hamming distance is the number of bit positions in which the two bits are different. This is perfect for BRIEF, as we can easily compute the number of difference between bit digits for the closest match.
Nearest Neighbor: Nearest Neighbor can classify a a point based on its nearest neighbors' classifications. Or in this case, give us the closest feature match using a distance function.

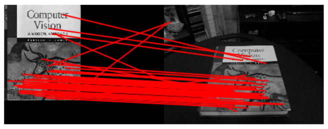
Hamming distance is easier to compute and represents BRIEF difference better compared to Euclidean distance, since BRIEF descriptors do not have a physical distance that makes sense to be taken the norm of. A number of differences between the digits serve the purpose just as well while being faster.



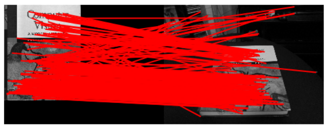
Q2.1.4 Feature Matching

Q2.1.5 Feature Matching Parameter Tuning

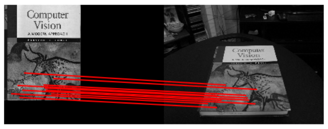
I first lowered sigma from 0.15 to 0.1, and immediately got more matches. Afterwhich, I increased ratio from 0.7 to 0.9, and got even more matches. Finally, I reduced sigma to 0.05 and reduced ratio to 0.5 to drastically decrease the amount of matches.
Sigma is the threshold at which the FAST algorithm determines whether a point is a corner. Decreasing it means points around a corner don't need to be as different in intensity to classify that point as a corner, hence more corners will be found. Ratio is the maximum ratio of distances between the two closest descriptor in the second set of descriptors to the descriptor of interest in the first set of descriptors. Increasing it means the distance differences between those two closest matches don't have to be as drastic, thus increasing amount of matches made.



sigma = 0.1
ratio = 0.7



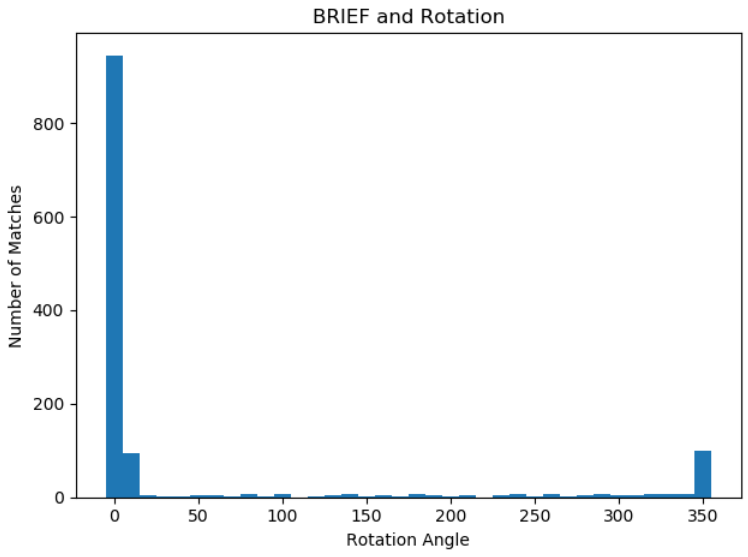
sigma = 0.1
ratio = 0.9

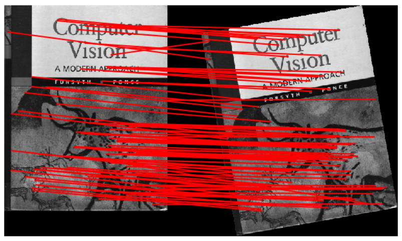


sigma = 0.05
ratio = 0.5

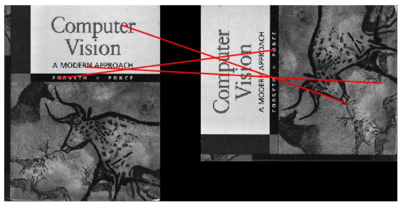
Q2.1.6 BRIEF and Rotations

The histogram and three matching results at different orientations are down below.

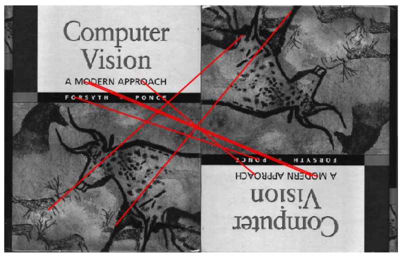




rotation = 10 degree



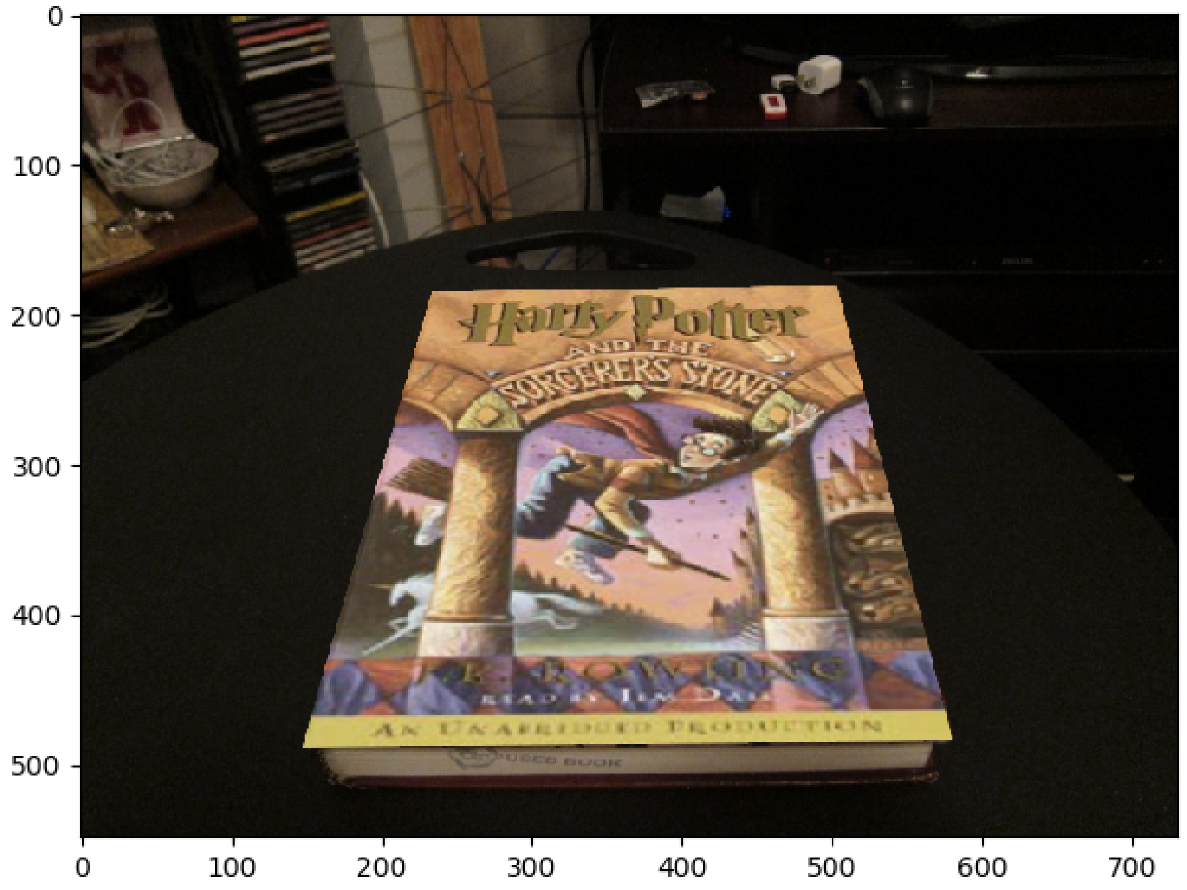
rotation = 90 degree



rotation = 180 degree

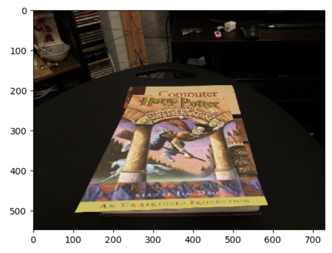
2.2 Homography Computation

Q2.2.4 Putting it together

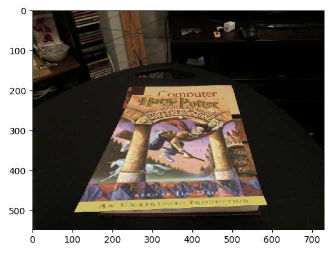


Q2.2.5 RANSAC Parameter Tuning

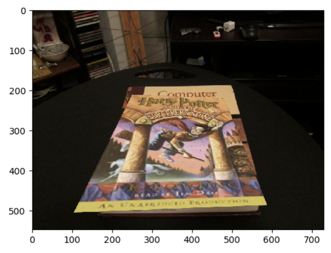
max_iters determines how many iterations to run RANSAC for, and **inlier_tol** determines the threshold for considering a point an inlier after transforming it with a given H. RANSAC is quite robust, and the performance doesn't degrade until **max_iters** reaches below 5. Higher **max_iters** obviously doesn't decrease performance. **inlier_tol** needs to go below 0.0001 or above 50 for performance to noticeably degrade Interestingly, they degradations look very similar for all three cases. The result images are down below.



inlier_tol = 2
max_iters = 4



inlier_tol = 0.0001
max_iters = 500



inlier_tol = 50
max_iters = 500