Computer Vision, 16720A - Homework 2

Neeraj Basu
neerajb@andrew.cmu.edu

October 8, 2020

Q 1.1 - Homography

1. Prove that there exists a homography H given two 3x4 camera projection matrices P1 and P2 corresponding to the two cameras and a plane.

Let's start by focusing on Camera C and derive the extrinsic and intrinsic matrices that transform x to x_π. Here f represents the focal point, r the rotation coefficients and t the transformation coefficients. The matrix containing f represents the intrinsic characteristics and the matrix containing r's and t's represents the extrinsic characteristics.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ q \\ 0 \\ 1 \end{bmatrix}$$

We can remove the third column of the r/t matrix since z = 0 in the right most column matrix. The reason z is zero is because x_π is in the planar world. We can also remove the fourth column of the f matrix since this column will have no effect on the matrix multiplication. The new matrix will look like:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ r_{31} & r_{33} & t_z \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ q \\ 1 \end{bmatrix}$$

Simplifying the matrices such that x_1 represents Camera C, P_1 represents the combination of the intrinsic and extrinsic transformations and Q represents the point in x_π, we can form an equation:

$$x_1 = P_1 Q$$

For Camera C', the process will look very similar. The content of the projection matrix will differ and we will denote this with P_2. The simplified equation will be denoted as:

$$x_2 = P_2 Q$$

Isolating Q in the Camera 2 equation and plugging it into the Camera 1 equation we will get:

$$x_1 \equiv P_1 P_2^{-1} x_2$$

We can now see there exists a homography H, denoted by $P_1 P_2^{-1}$ between x_1 and x_2.

Q 1.2 - Correspondences

1. How many degrees of freedom does h have?

Looking at a single point we'll call x_1, we can map the point to the projective plane, using the equation:

$$\begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Solving for u_1 and v_1 we get the two equations:

$$u_1 = \frac{h_{11}x_1 + h_{12}y_1 + h_{13}}{h_{31}x_1 + h_{32}y_1 + h_{33}}$$

$$v_1 = \frac{h_{21}x_1 + h_{22}y_1 + h_{23}}{h_{31}x_1 + h_{32}y_1 + h_{33}}$$

Rearranging the equations to make them compatible with matrix form:

$$h_{11}x_1 + h_{12}y_1 + h_{13} - h_{31}x_1u_1 - h_{32}y_1u_1 - h_{33}u_1 = 0$$

$$h_{21}x_1 + h_{22}y_1 + h_{23} - h_{31}x_1v_1 - h_{32}y_1v_1 - h_{33}v_1 = 0$$

Rearranging this into matrix form:

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1u_1 & -y_1u_1 & -u_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1v_1 & -y_1v_1 & -v_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ \vdots \\ h_{33} \end{bmatrix} = 0$$

We can see that in this form, h will have 9 DoFs. Although one dimension will be dropped since the system is scale invariant. So in reality h will have 8 DoFs.

2. How many point pairs are required to solve h?

Since h has 8 DoFs, we will need 8 equations in total to solve the matrix above. Since each point will generate two rows in the A matrix, we will need four point pairs in order to solve for h.

3. Derive A_i.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1u_1 & -y_1u_1 & -u_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1v_1 & -y_1v_1 & -v_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2u_2 & -y_2u_2 & -u_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2v_2 & -y_2v_2 & -v_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3u_3 & -y_3u_3 & -u_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3v_3 & -y_3v_3 & -v_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4u_4 & -y_4u_4 & -u_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -x_4v_4 & -y_4v_4 & -v_4 \end{bmatrix}$$

4. What will be a trivial solution for h? Is the matrix A full rank? Why/Why not? What impact will it have on the eigen values? What impact will it have on the eigen vectors?

The trivial solution for $Ah = 0$ would be h equal to a zero column vector. If Matrix A was full rank, A would not have a null space and therefore $Ah = 0$ would not have a solution. Since we were able to find a solution, we know A is not full rank. The fact that A has a non-trivial solution, implies that A has an eigenvector which lives in the nullspace of A and is a solution to $Ah = 0$. This also means that A has an eigen value of zero which corresponds to the eigenvector in the nullspace.

Q 1.3 - Homography under rotation

Prove that there exists a homography H that satisfies $x1 \equiv Hx2$, given two cameras separated by a pure rotation.

Starting with the equations where K_1 and K_2 represent the intrinsic matrices and R the rotation matrix :

$$x_1 = K_1[I\ 0]X$$

$$x_2 = K_2[R\ 0]X$$

We can remove the 0 vector in both equations since they will have no impact during the matrix multiplication. In the x_1 equation, we can multiply K_1 by the identity matrix resulting in K_1.

$$x_1 = K_1 X$$

$$x_2 = K_2 R X$$

Solving for X in he first equation and plugging it into the second:

$$x_2 \equiv K_2 R K_1^{-1} x_1$$

We can now see there exists a homography denoted by $K_2 R K_1^{-1}$ between x_1 and x_2.

Q 1.4 - Understanding homographies under rotation

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

Referencing Q1.3 we can see the homography describing the rotation between two cameras is represented by $K_2RK_1^{-1}$. For this problem, we will reuse that homography, only there will be a single K since we are dealing with one camera. We'll use x_1 to indicate our starting point and x_2 to indicate our ending point.

Applying a rotation homography $K_2RK_1^{-1}$ twice we'll get:

$$x_2 \equiv KRK^{-1}KRK^{-1}x_1$$

We can see the $K^{-1}K$ term will simplify to the identity matrix leaving us with:

$$x_2 \equiv KRRK^{-1}x_1$$

Which can be written as:

$$x_2 \equiv KR^2K^{-1}x_1$$

We can then see the homography describing a rotation by θ twice can be representing by the rotation homography squared.

Q 1.5 - Limitations of the planar homography

Why is the planar homography not completely sufficient to map any arbitrary scene image to another viewpoint?

Homography only works under two assumptions:
1. We assume that the world is planar
2. We assume that the camera was rotated about its center

Q 1.6 - Behavior of lines under perspective projections

Verify that the projection P in $x = PX$ preserves lines.

To prove that lines are preserved when projected from 3D to 2D, we first need to pick three points in 3D space. The first two points A & B can be arbitrary, but the third must be a linear combination of A & B to form a line. Let us donate C as a parametric equation involving A & B.

$$C = A + t(B - A), \quad t \in \mathbb{R}$$

C is now defined as every point which can be on a line containing points A & B. Now let's assume we have some projection matrix which takes A, B & C to 2D. We'll call this matrix P.

$$a = PA$$
$$b = PB$$
$$c = PC$$

Substituting the equation for C in 3D space into c for 2D space we get:

$$c = PC = P(A + t(B - A))$$

Simplifying the equation we end on:

$$PC = PA + t(PB - PA))$$

Analyzing this equation we can see we have the parametric equation of a line, but in 2D space after we've applied our projection P. Therefore, our projection P has preserved the lines.

Q 2.1.1 - FAST Detector

How is the FAST detector different from the Harris corner detector that you've seen in the lectures?

Compared to the Harris detector, the FAST detector does pixel to pixel intensity correspondence opposed to kernel convolution to detect edges. FAST picks a point **p**, an intensity **I** to describe **p** and a threshold **t** and assumes a circle around the point p (usually of 16 pixels). FAST then looks for n contiguous pixels whose intensity is either greater or less than **t**. In the case of 16 pixels, FAST would be looking for 12 pixels that meet the intensity criteria. FAST is computationally efficient since the only operations needed to create the binary descriptors are addition and subtraction. This eliminates the need for calculating derivatives in the x and y direction unlike the Harris detector.

Q 2.1.2 - BREIF Descriptor

How is the BRIEF descriptor different from the filterbanks you've seen in the lectures?

Filterbanks and BRIEF descriptors are both methods of consolidating neighboring pixel data into a single point. However, BRIEF is much faster computationally. Filter banks required you to build a multidimensional array to describe your point over multiple filters and scales and then computes the Euclidean distance to match points. Whereas BRIEF is capable of doing this within a single dimension by using pixel to pixel comparison. BRIEF descriptors also contain location data compared to filter banks since you don't need to stretch the data into multiple dimensions.

Q 2.1.3 - Matching Methods

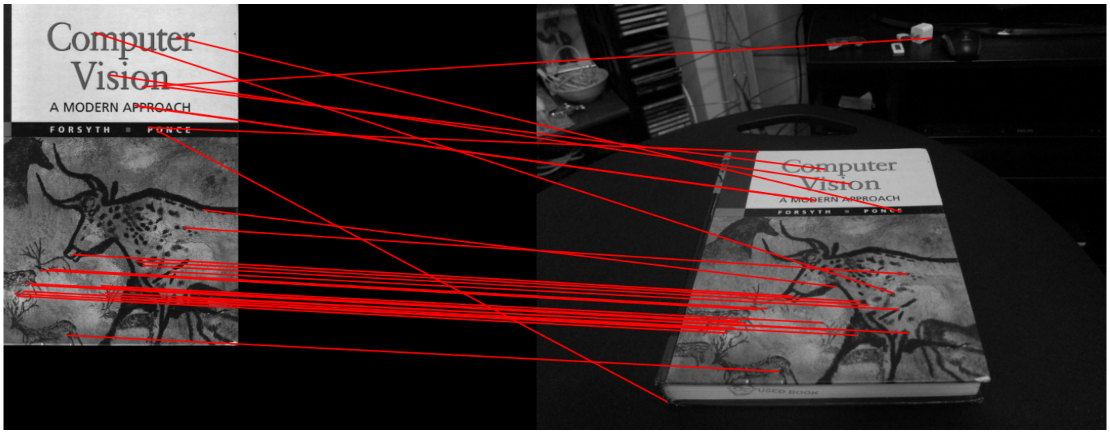
What benefits does the Hamming distance have over a more conventional Euclidean distance measure in our setting?

Hamming distance compares two binary strings of equal length which are created from Feature Descriptors such as FAST, LBP, etc. The Hamming distance is the number of bit positions between the two strings which are different. For example if string1 is [0,0,1] and string2 [1,0,0], the Hamming distance is 3 since we had to flip the bits in string1 3 times to match string2.

The main advantage of Hamming distance compared to Euclidean is speed. Since these strings are managed at a bit level, we can use bitwise functions such as XOR to calculate the Hamming distance which are far more computationally efficient when compared to Euclidean Distance.

Q 2.1.4 - Feature Matching

Use plotMatches() to visualize your matched points and include the result image in your write-up.

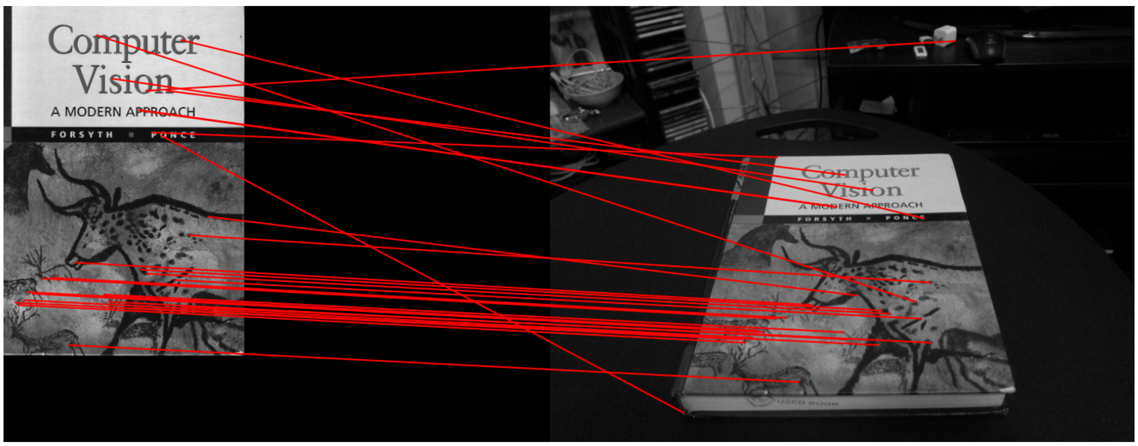


Q 2.1.5 - Feature Matching Parameter Tuning

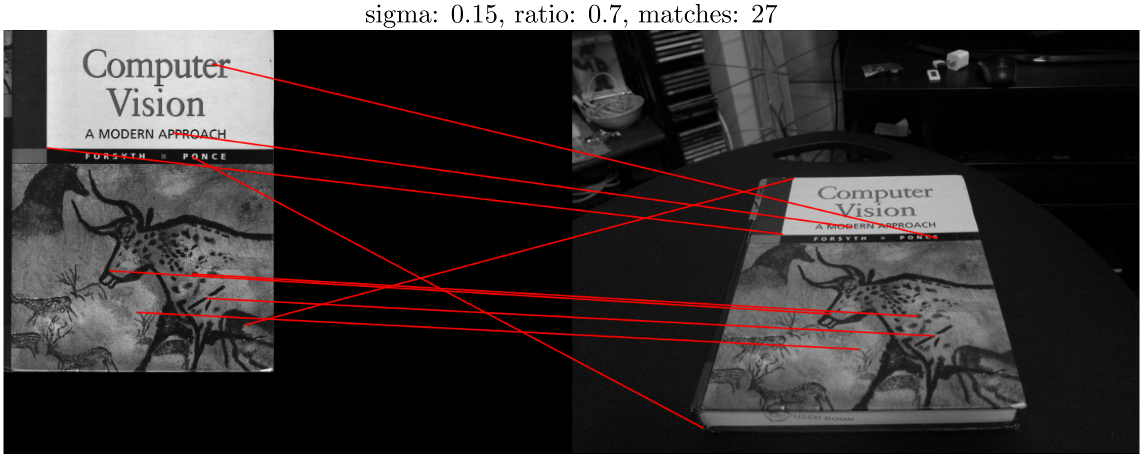
Conduct a small ablation study with various sigma and ratio values.

Ablation Table:

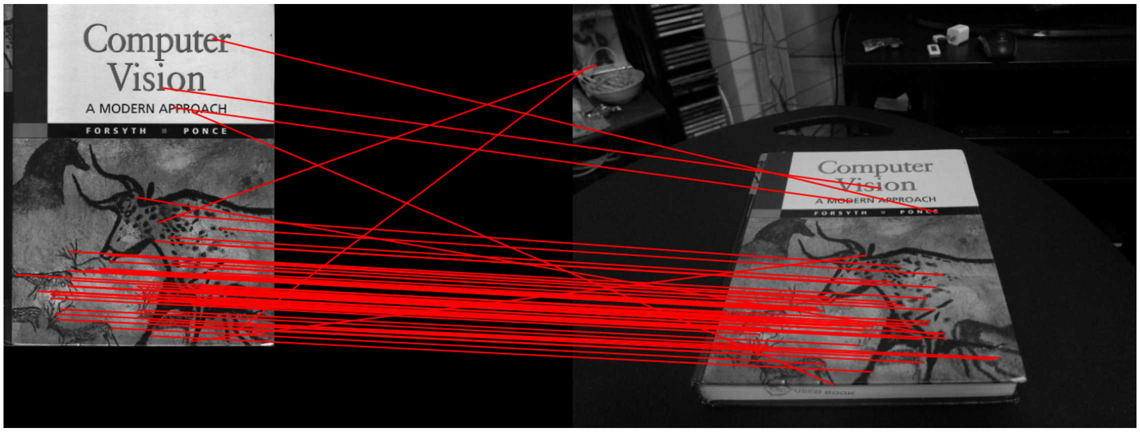
Sigma	Ratio	matches
0.15	0.7	27
0.20	0.7	9
0.10	0.7	60
0.15	**0.8**	62
0.15	**0.6**	7



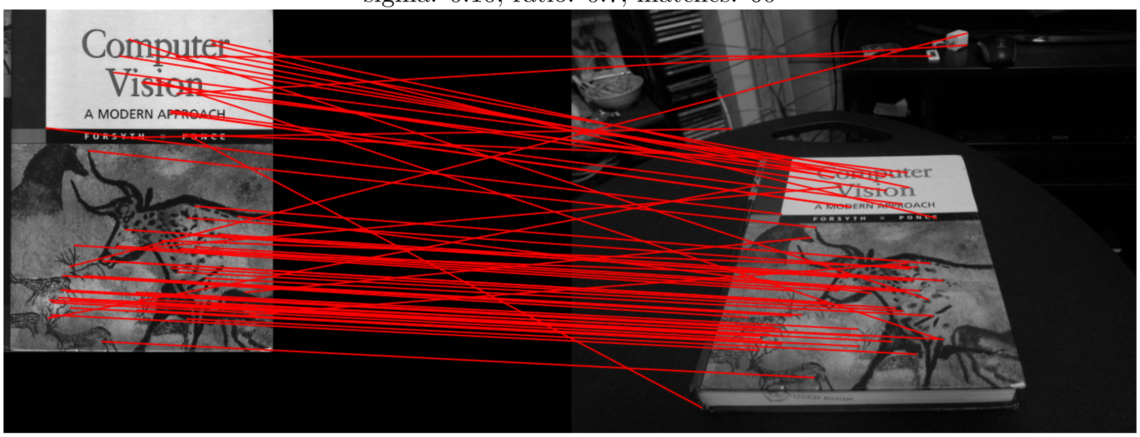
sigma: 0.15, ratio: 0.7, matches: 27



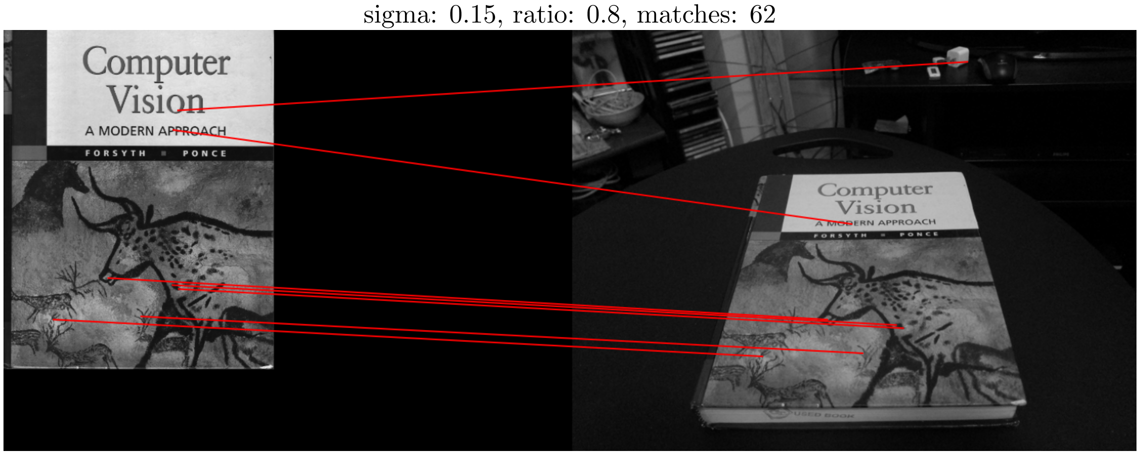
sigma: 0.20, ratio: 0.7, matches: 9



sigma: 0.10, ratio: 0.7, matches: 60



sigma: 0.15, ratio: 0.8, matches: 62

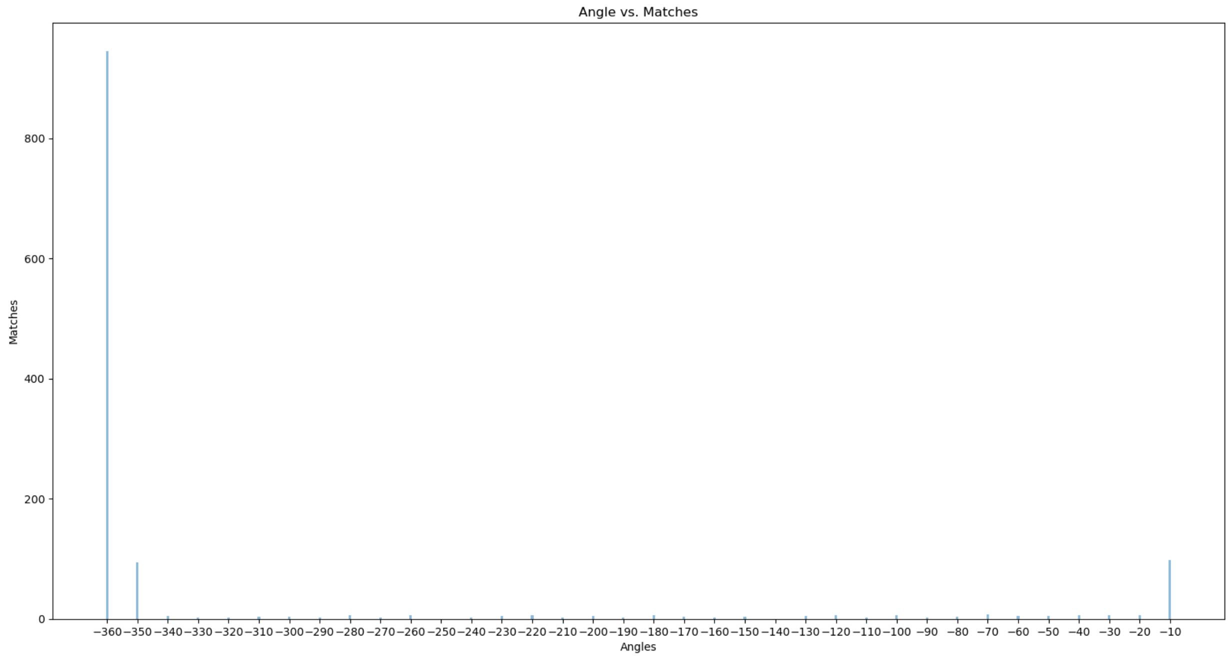


sigma: 0.15, ratio: 0.6, matches: 7

Based on the the ablation table study, decreasing sigma and and increasing ratio leads to an increases in matches. Whereas increasing sigma and decreasing ratio leads to a decrease in matches. We can also see as we increase sigma and decrease ratio, many of the low frequency matches are filtered out. Higher frequency points such as the words in the title are captured and the fidelity of the matches increases.

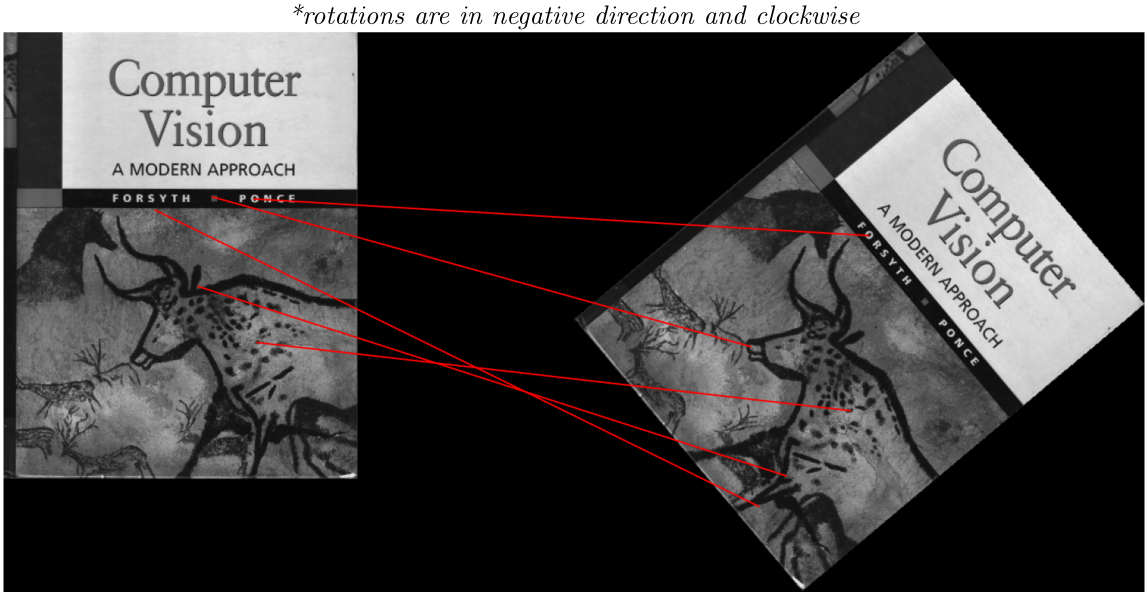
Q 2.1.6 - BRIEF and Rotations

Visualize the histogram and the feature matching result at three different orientations and include them in your write-up. Explain why you think the BRIEF descriptor behaves this way.



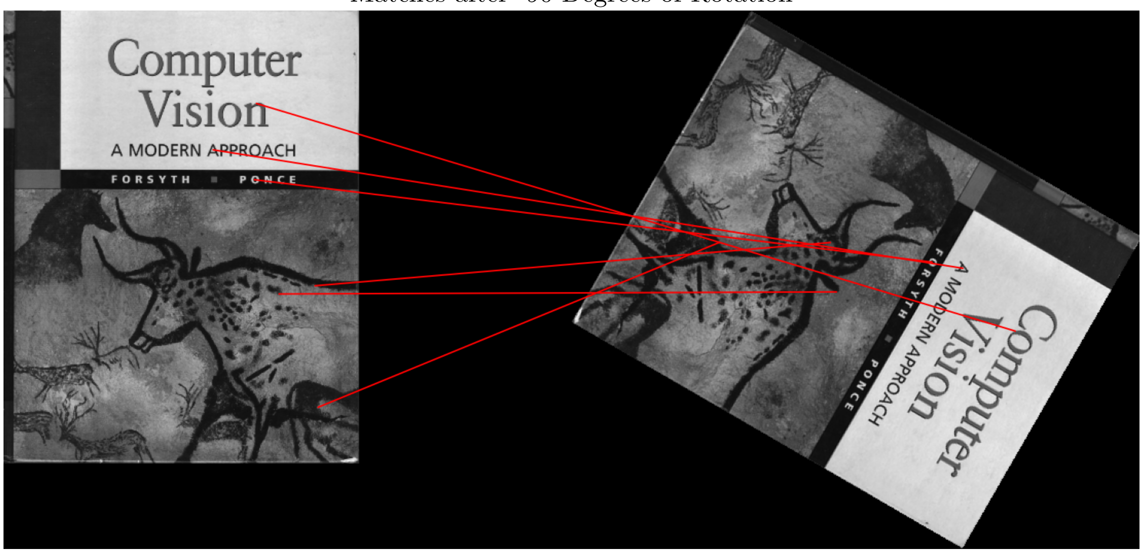
Histogram, Angle vs. Matches

**rotations are in negative direction and clockwise*



Matches after -50 Degrees of Rotation

Matches after -90 Degrees of Rotation



Matches after -120 Degrees of Rotation

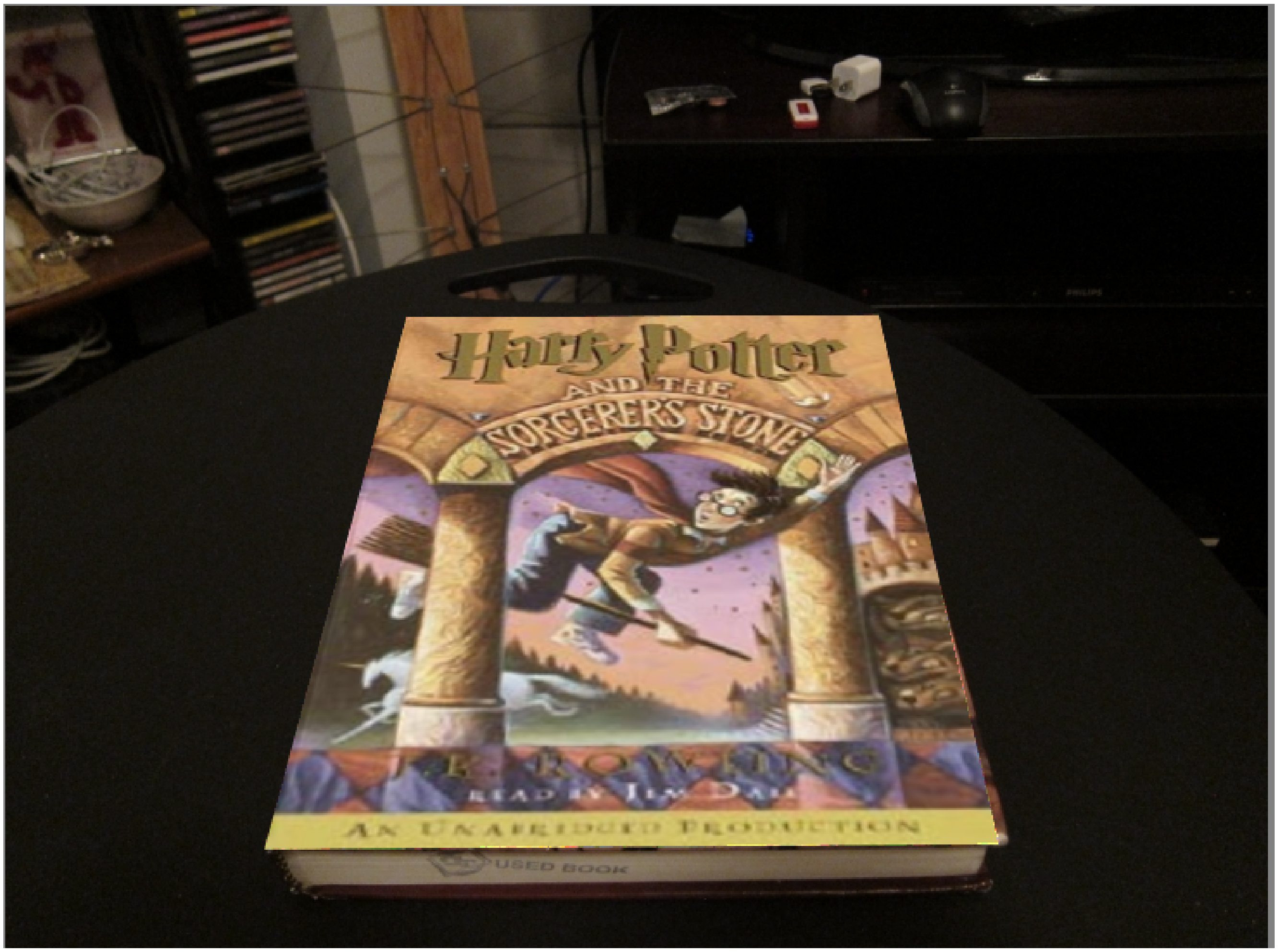
We can see that as the image rotates away from it's original position, the number of matches drops. At 0 and 360 degrees, the descriptor works best. At $\pm$ 10 degrees it finds some matches but any rotation beyond that renders BRIEF futile.

This is expected as the BRIEF Descriptor is not rotation invariant. This is due to the fact that the BRIEF Descriptor does a pixel to pixel correspondence in terms of intensity. If the patch describing a feature point is rotated, the encoded data is not the same. Please see an example of a patch being rotated by 90 degrees.

$$\begin{array}{|c|c|c|}\hline 1 & 2 & 3 \\ \hline 4 & 5 & 6 \\ \hline 7 & 8 & 9 \\ \hline \end{array} \neq \begin{array}{|c|c|c|}\hline 7 & 4 & 1 \\ \hline 8 & 5 & 2 \\ \hline 9 & 6 & 3 \\ \hline \end{array}$$

Q 2.2.4 - HarryPotterize

Compose the warped image with the desk image and include your result in your write-up.



Harry Potter Cover Overlayed on CV Desk Image

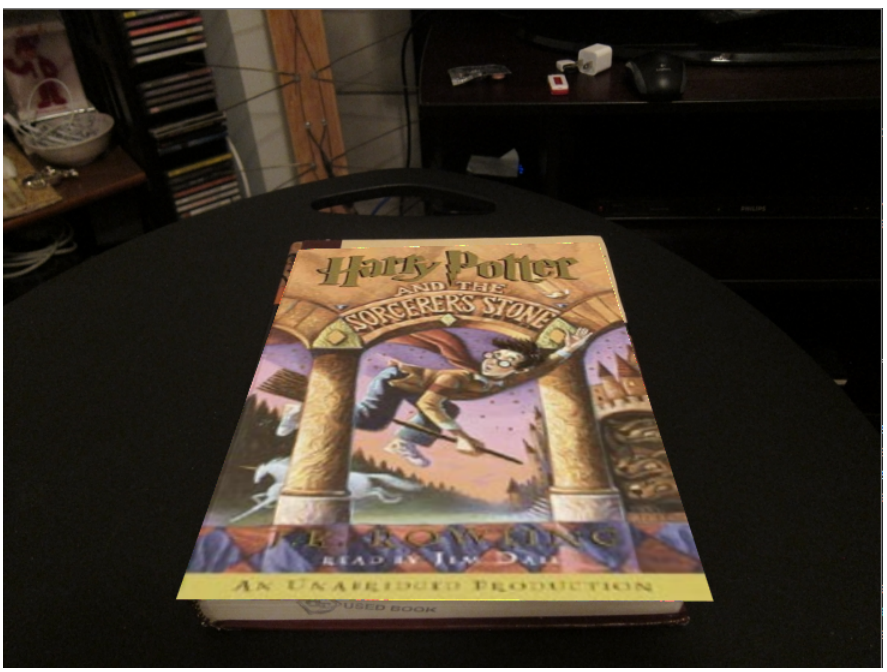
The HP cover did not fill the space on CV Desk since it was smaller than the CV Cover image and applying a homography would not change this scale. I was able to fix the issue of the warped image not filling the CV Desk image by reshaping the warped image to the same size as the CV Cover image.

Q 2.2.5 - RANSAC Parameter Tuning

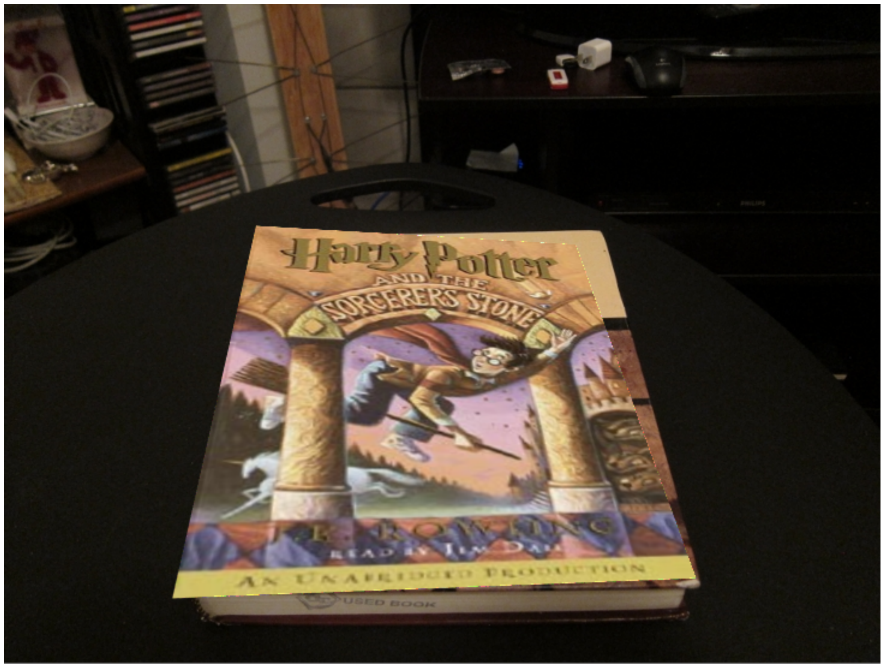
Conduct a small ablation study by running HarryPotterize.py with various max iters and inlier tol values. Include the result images in your write-up, and explain the effect of these two parameters respectively.

max iters	inlier tol	inlier count
500	2	10
250	2	10
100	2	10
10	2	9
1	2	7
750	2	10
1000	2	10
5000	2	10
500	**1.5**	10
500	**1**	9
500	**0.5**	7
500	**2.5**	10
500	**3**	10
500	**13**	12

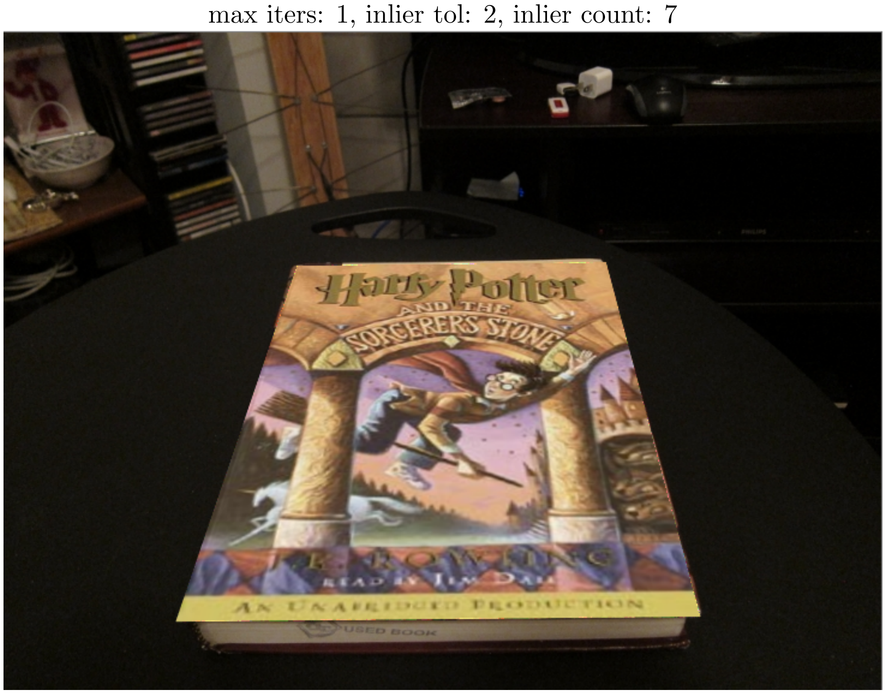
Please note images were omitted that did not show a visual change.



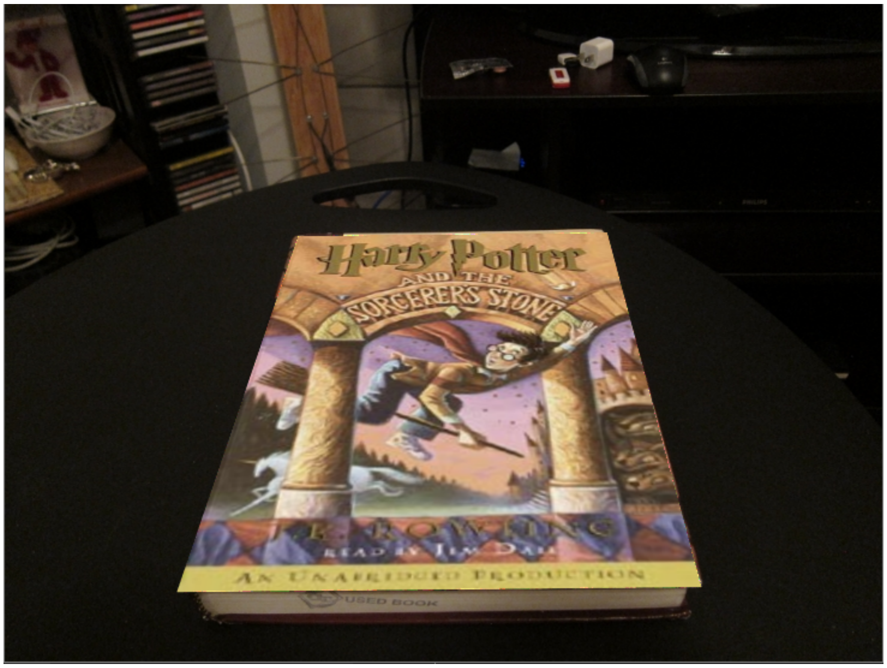
max iters: 100, inlier tol: 2, inlier count: 10



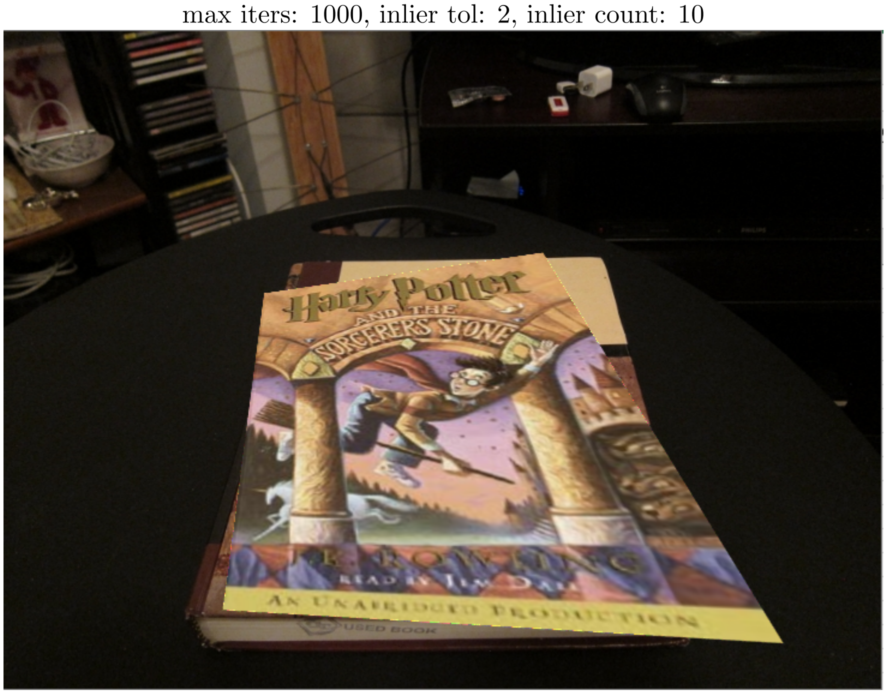
max iters: 1, inlier tol: 2, inlier count: 7



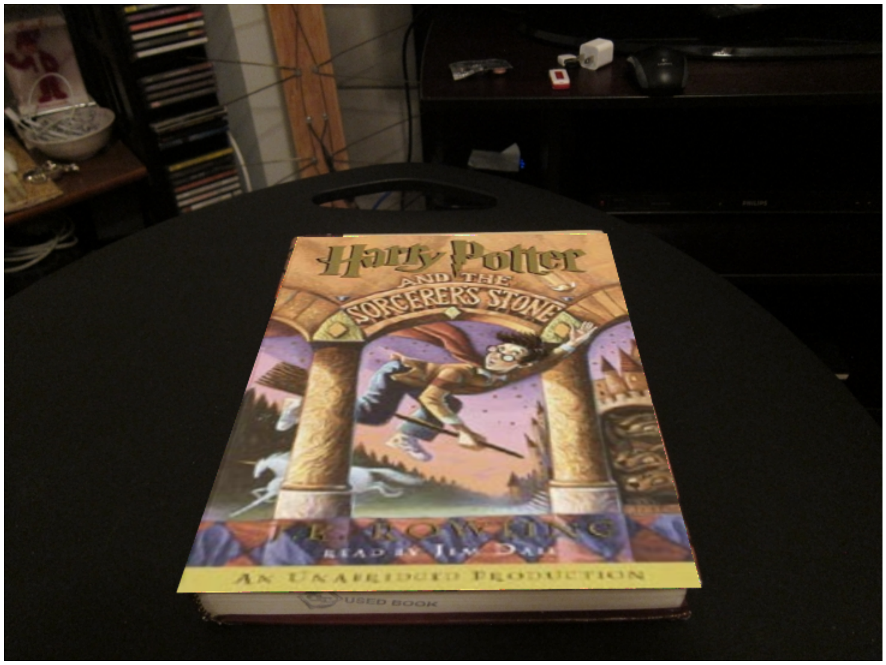
max iters: 750, inlier tol: 2, inlier count: 10



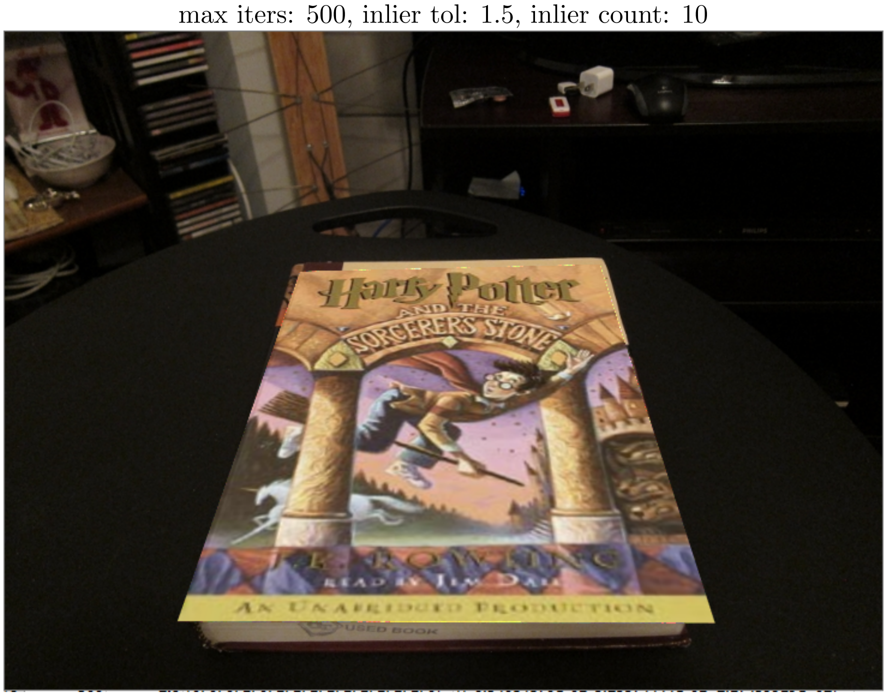
max iters: 1000, inlier tol: 2, inlier count: 10



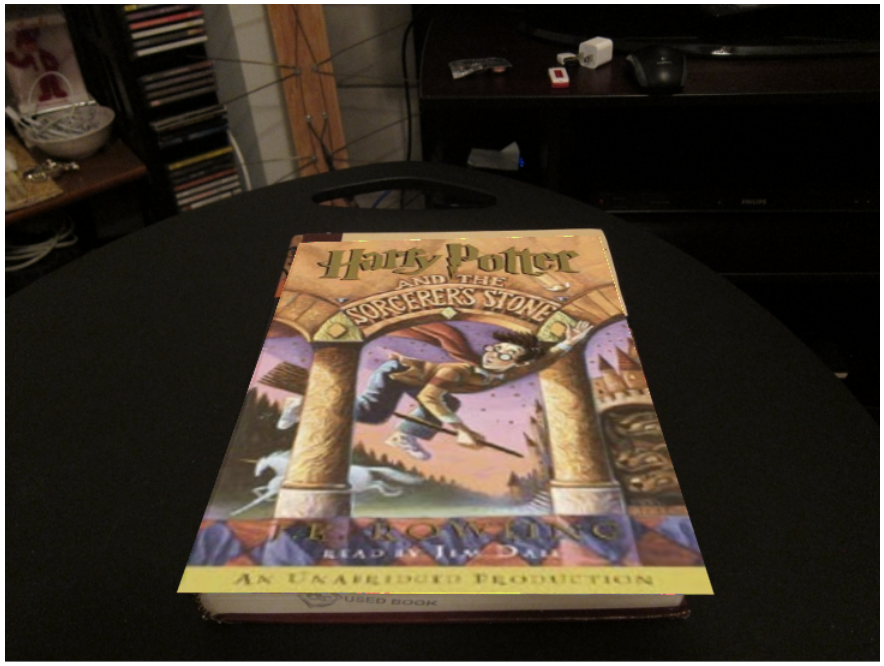
max iters: 5000, inlier tol: 2, inlier count: 12



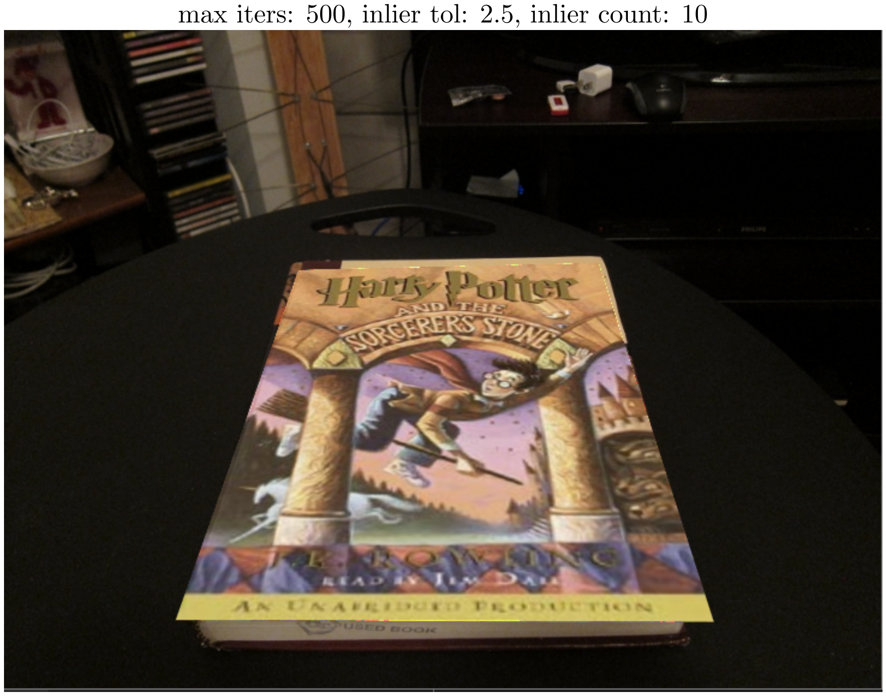
max iters: 500, inlier tol: 1.5, inlier count: 10



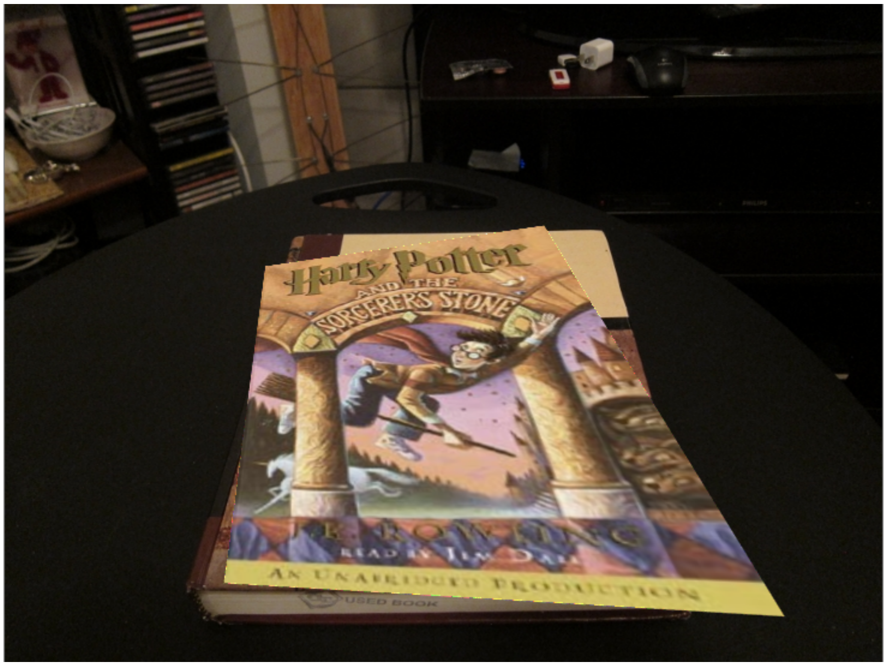
max iters: 500, inlier tol: 0.5, inlier count: 7



max iters: 500, inlier tol: 2.5, inlier count: 10



max iters: 500, inlier tol: 3, inlier count: 10



max iters: 500, inlier tol: 13, inlier count: 12

From the ablation study, it becomes clear that the fidelity of homography decreases, as we get closer to the extremes of turning either knob. If the numbers of iterations is too low, our probability of finding the best fit and picking the best H is low. If the number of iterations is too high, we increase the probability of finding an incorrect fit which still yields the highest number of outliers. If our inlier tolerance is too low, we become less robust in regards to handling image noise and if too high, a higher probability of accepting outliers.

Q 3.1 - Incorporating Video

I needed to deviate from the default feature detection parameters in order for this segment to work. When I reached frame 435 in the Kung-Fu Panda video, my feature detector was unable to collect enough features. The parameters I used were sigma = 0.15, ratio = 0.8.