Andrew: numasank
Nanditha Umasankar

Augmented Reality with Planar Homographies

Q 1.1 Homography

For a point x_π on the plane, using the camera model P_1 the point can be represented as $x_1 = P_1 * x_\pi$ .With the camera model P_2, the same point can be represented as $x_2 = P_2 * x_\pi$.

Since the two equations represent the same point on the plane, The camera projection matrices P1 and P2 being invertible, we can find a relation between x_1 and x_2 .

Hence there exists an ***H*** such that, $x_1 = \boldsymbol{H} x_2$

Q 1.2 Correspondences

1. For a homography H,

$$\boldsymbol{x_1^i} \equiv \boldsymbol{H}\, \boldsymbol{x_2^i} \qquad \text{with } i \in [1, N]$$

For each point pair with relation A_i h = 0, where h is a reshaped column vector of H with dimensions 3x3, **h** has <u>8 degrees of freedom</u> since the 9th element is used to normalize h.
2. 4-point pairs – that have 2 relations A_i h = 0 each, giving rise to 8 equations are needed to solve **h**
3. **A**$_i$
Estimating A_i by considering the Homography H between 2 points, $x_2 = H\, x_1$

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$$

With the inhomogeneous coordinates $x_2' = x_2/z_2$ and $y_2' = y_2/z_2$

$$x_2' = \frac{H_{11}x_1 + H_{12}y_1 + H_{13}z_1}{H_{31}x_1 + H_{32}y_1 + H_{33}z_1}$$

$$y_2' = \frac{H_{21}x_1 + H_{22}y_1 + H_{23}z_1}{H_{31}x_1 + H_{32}y_1 + H_{33}z_1}$$

Setting z_1 = 0,

$$x_2'(H_{31}x_1 + H_{32}y_1 + H_{33}) = H_{11}x_1 + H_{12}y_1 + H_{13}$$

$$y_2'(H_{31}x_1 + H_{32}y_1 + H_{33}) = H_{21}x_1 + H_{22}y_1 + H_{23}$$

We can rearrange the above equations to the form,

$$a_x^T h = 0$$
$$a_y^T h = 0$$

where,

$$h = \left(H_{\{11\}}, H_{\{12\}}, H_{\{13\}}, H_{\{21\}}, H_{\{22\}}, H_{\{23\}}, H_{\{31\}}, H_{\{32\}}, H_{\{33\}}\right)^T$$

$$\mathrm{a_x} = (-\mathrm{x}_1, -\mathrm{y}_1, -1, 0, 0, 0, \mathrm{x}_2' x_1, \mathrm{x}'_2\, y_1, \mathrm{x}_2')^\mathrm{T}$$
$$\mathrm{a}_y = (0,0,0, -\mathrm{x}_1, -\mathrm{y}_1, -1, \mathrm{y}_2' x_1, \mathrm{y}'_2\, y_1, \mathrm{y}_2')^\mathrm{T}$$

Hence, we have A_i

Thus, we have Ah = 0 where

4. The trivial solution for Ah = 0 is h = 0.
If only trivial solutions exist then A is full rank. Since there is some non-trivial solution, A is not full rank. The equation Ah = 0 can be solved as a Linear Least Squares problem using Singular Value Decomposition. The non-trivial solution is the eigen vector corresponding to the minimum eigen value of $A^T A$.

Q 1.3 Homography under rotation

For two cameras,

$$x_1 = K_1[I \quad 0]\begin{bmatrix} X \\ 1 \end{bmatrix}$$

$$x_2 = K_2[R \quad 0]\begin{bmatrix} X \\ 1 \end{bmatrix}$$

Since the cameras are separated by pure rotation, we can obtain the Homography H

$$X = K_1^{-1} x_1$$

$$x_2 = K_2 R \,(K_1^{-1}\, x_1)$$

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

Therefore, there exists a homography $H = K_1 R K_1^{-1}$ for the two cameras separated by a pure rotation.

Q 1.4 Understanding Homographies under Rotation

For a point X in the 3D world the homography between the two views at orientation θ of the camera is given by,

$$x_1 = H\,x_2$$

Now for the third view, with the camera at orientation θ from the second position, the homography between the points in second and third views is given by the same homography since there is only a rotation between the two views,

$$x_2 = H\,x_3$$

Homography between the first and third view is, H

$$x_1 = H\,(Hx_3) = H^2 x_3$$

Therefore, the homography for camera with rotation 2θ is H^2

Q 1.5 Limitations of Planar Homography

Planar homography assumes that all points in the 3D world lie on a plane and the camera is purely rotated. While computing the homography, the distance between the camera and the point is not taken. Without the depth information, planar homography is insufficient to map any scene to another viewpoint unless the point is on the same plane or the camera is purely rotated.

Q 1.6 Behaviour of lines under perspective projection

A point in 3D is represented as, $[x \quad y \quad z]^T$. When projected to 2D the points are $[x' \quad y']^T$ where $x' = x/z$ and $y' = y/z$

A line in 3D has the following equation,

$$[a \quad b \quad c]\begin{bmatrix} x \\ y \\ z \end{bmatrix} = 0$$

Which can be represented as

$$[a \quad b \quad c]\begin{bmatrix} x/z \\ y/z \\ 1 \end{bmatrix} = 0 \;=>\; [a \quad b \quad c]\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = 0$$

Which is equivalent to the 2D line representation

$$[l \quad m]\begin{bmatrix} x \\ y \end{bmatrix} = 0$$

Thus, Perspective projection preserves lines.

Q 2.1.1 FAST Detector

FAST or Feature Accelerated Segment Test Detector uses a circular area of pixels around the candidate pixel p with intensity I_p, to identify corners. Using a threshold t, pixel p is classified as a corner if the 16 contiguous pixels around p that form the circular area have an intensity brighter or darker than I_p +/- t. Harris corner detector on the other hand uses a sliding window and computes the Sum of Squared Differences (SSD) between each step which is computationally expensive.

FAST, using a circular area, allows high-speed test to reject non-corner points by simply looking at 4 out of the 16 points considered in the circle. If the point is not rejected based on the first four points, all 16 points are considered. Thus, making FAST one of the more computationally efficient corner detectors.

Q 2.1.2 BRIEF Descriptor

BRIEF or Binary Robust Independent Elementary Features computes a binary string in image patches. This binary string is computed after smoothing the patch with Gaussian filters of varying kernels. Unlike other algorithms that first computing full descriptors and then reducing the dimensionality, BRIEF directly computes binary strings.

The filter bank can be used as feature descriptors. They use convolution to extract features from images. The filter bank increases the dimensionality of the image as it applies filter with varying scales.

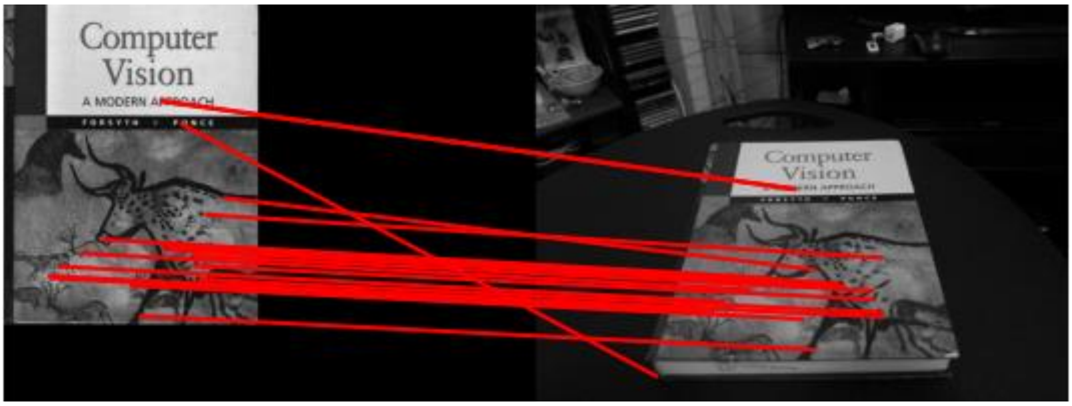
Q 2.1.3 Matching Methods

Hamming distance computes the distance between the binary strings from the BRIEF descriptor. The distance is computed by comparing the value of the two binary strings at each position, i.e. Bitwise XOR operation. The Euclidian distance is a point operation.

Hamming distance is computationally faster and is better suited for our setting.

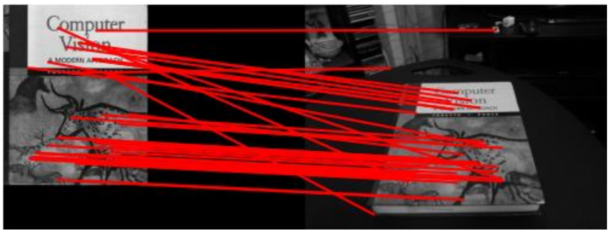
Q 2.1.4 Feature Matching

Result for the matchPics.py using default values: sigma = 0.15 and ratio = 0.7,

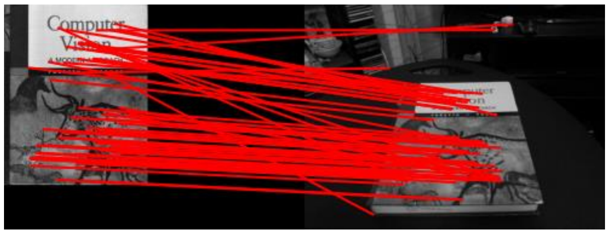


Q 2.1.5 Feature Matching Parameter Tuning

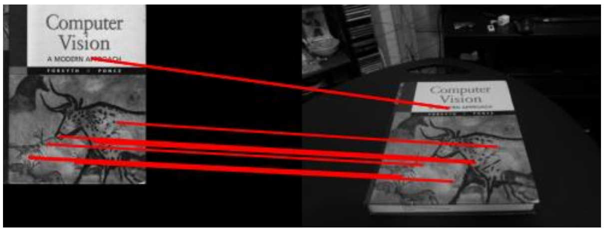
With sigma = 0.15 and tuning ratio



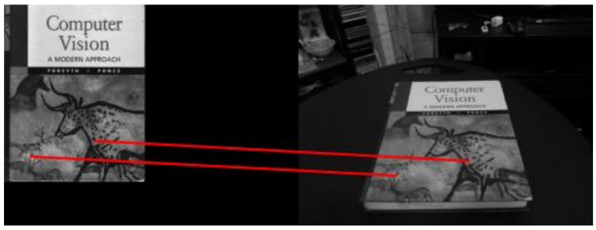
Ratio = 0.75



Ratio = 0.8



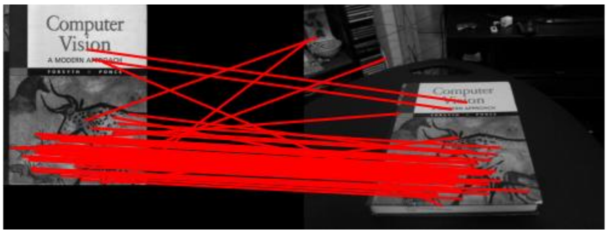
Ratio = 0.65



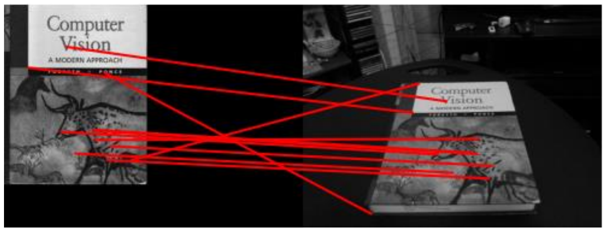
Ratio = 0.5

As the ratio is increased, briefMatch() computes multiple matches for each computed point. There is a drastic drop with decrease in ratio.

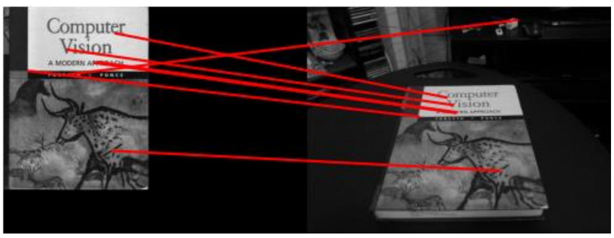
With ratio = 0.7 and tuning sigma



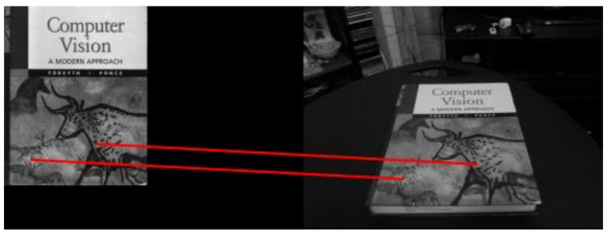
Sigma = 0.1



Sigma = 0.2



Sigma = 0.25

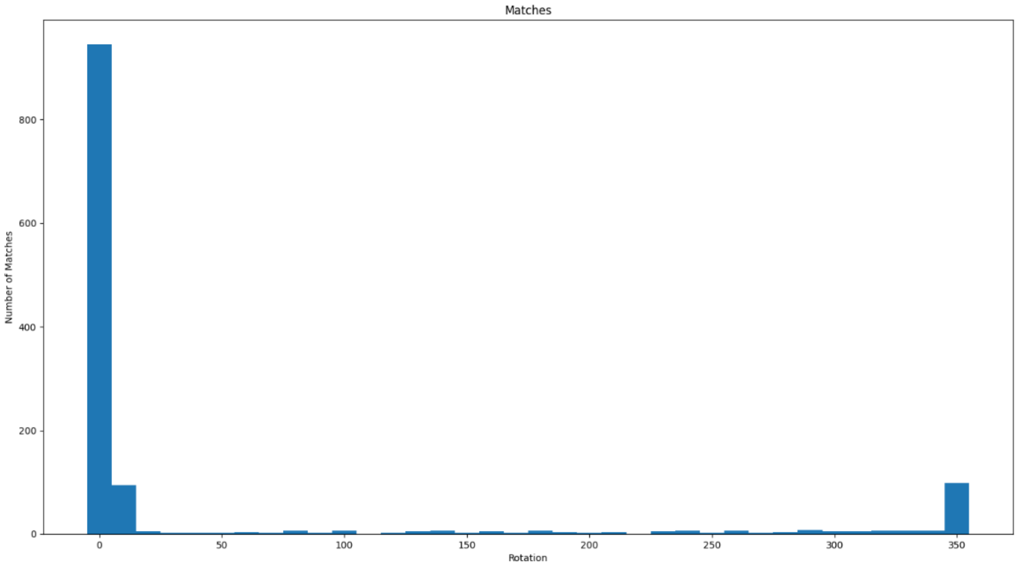


Sigma = 0.5

With a lower threshold sigma, more corners are detected. This results in a higher number of matches as visible in the sigma=1 image. As sigma is increased, there is a higher restriction in computing corners and hence fewer matches are found between the images.

Q 2.1.6 BRIEF and Rotations

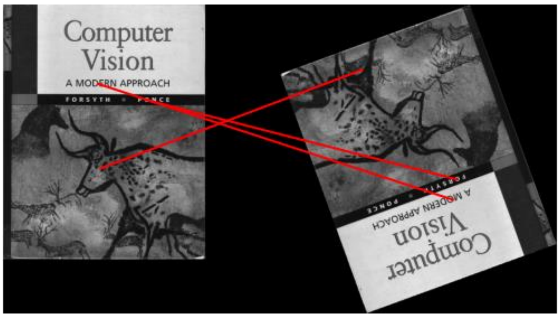
Histogram indicating the matches obtained for 10-degree rotation increments from 0 to 350,



Examples of matches for different rotation angles

Rotation = 10

Rotation = 90

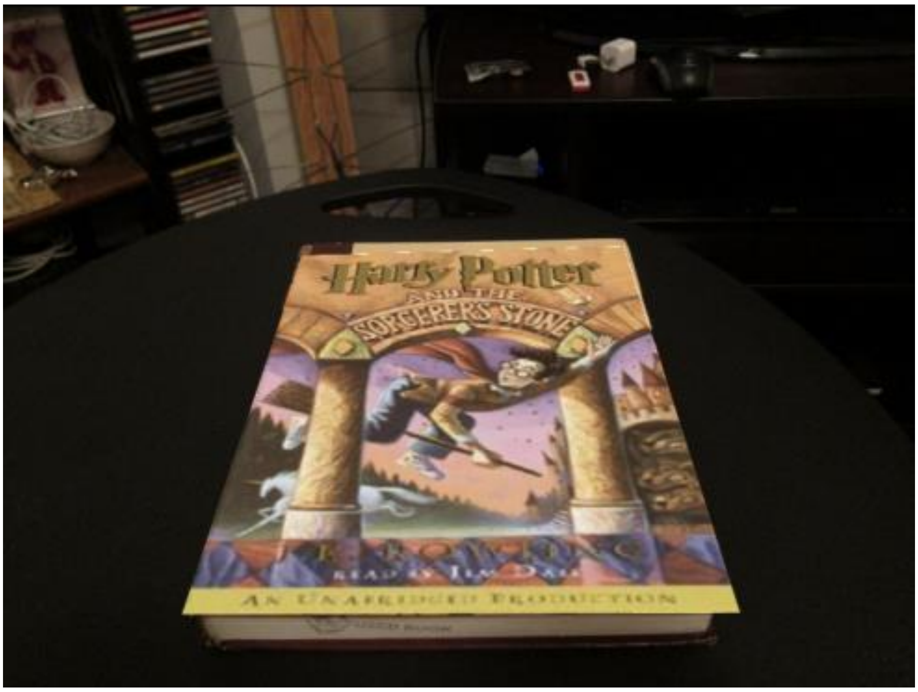


The BRIEF descriptor is very sensitive to angle rotation. From the histogram it is clear that for rotations greater than +/- 10 degrees, the matches drastically reduce.
As the neighbouring patches vary with rotation of the image, the BRIEF descriptor isn't very good at finding good matches.

Rotation = 200

Q 2.2.4 Putting it Together

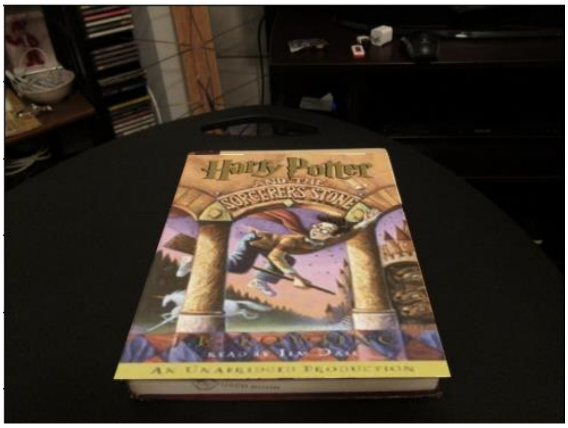
Composite image of Harry Potter cover warped to fit the CV book placed on a desk



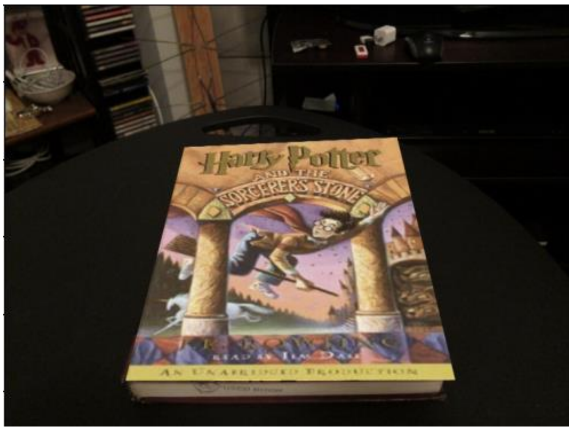
This is the warping obtained from the default parameters,
sigma = 0.15
ratio = 0.7
max_iters = 500
inlier_tol = 2

Q 2.2.5 RANSAC Parameter Tuning

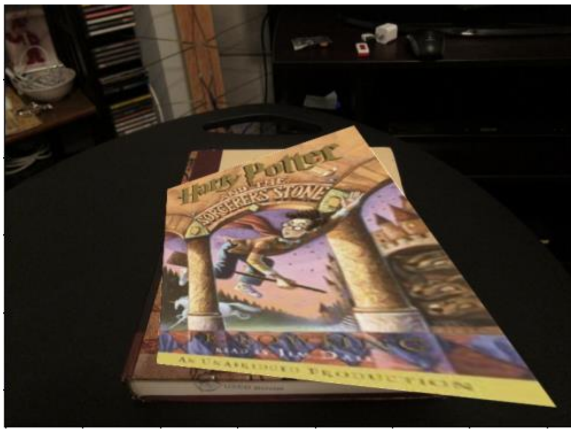
With max_iters = 500, tuning inlier_tol



inlier_tol = 1

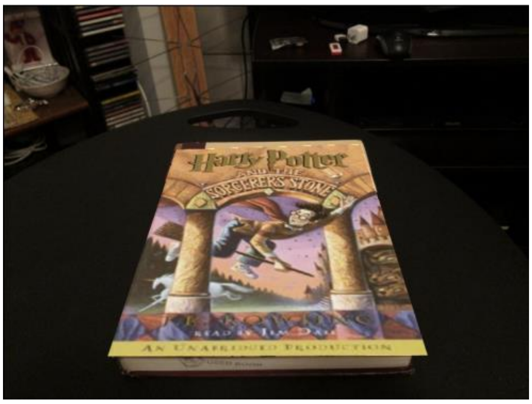


inlier_tol = 5

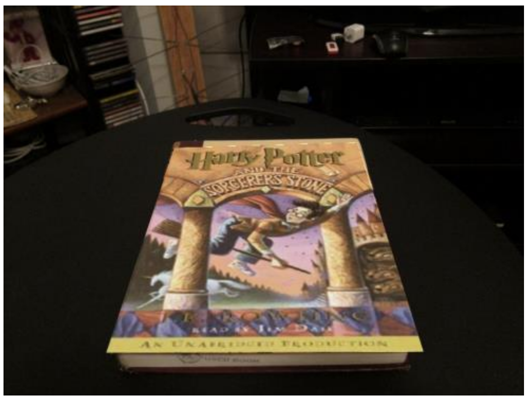


inlier_tol = 20

With inlier_tol = 2.0, tuning max_iters



max_iters = 100



max_iters = 1000

Increasing inlier_tol improves the wrap but for a very high value i.e. 20, the homography is overfitted and hence the warp isn't good.

Increasing max_iters helps find a better homography. The two values must be kept at an optimum to find the best warp.

Q 3.1 Incorporating Video

Screenshots of the result video – ‘ar.avi’

Video Link: [ar.avi](ar.avi)

Q 4.1 Real-time AR

To process real-time AR, the following steps were taken

1. ORB features were detected
2. Matches were found using Brute Force match
3. Computation speed was improved by using in built homography

Issues are observed with certain frames with respect to the warping
The FPS obtained was on an average ~24fps.

Q 2.4 Panorama

Image – left

Image – right

Panorama Image