# COL - 780 Computer Vision - Assignment 2
Dharmisha Sharma(2022MCS2062)

**Results:CV_assign2_panorama**

## Corner Detection Algorithm (for Y=0 the Harris Corner Detector):

1. Take the input image in gray scale and smoothen it.
2. Compute the image gradients in the x and y directions. (Ix and Iy)

   For each pixel compute gradients with the following kernel

   **For Ix :**

   [-0.5  0  0.5]

   **For Iy:**

   [-0.5]

   [ 0 ]

   [ 0.5]

3. Compute Ixx = $Ix^2$, Iyy = $Iy^2$ and Ixy = Ix*Iy
4. Set a window size for corner detection.
5. Calculate the R scores of each pixel using the following method:

   i. a = $\Sigma$Ixx, for all pixels falling in the window of the target pixel

      b = 2*$\Sigma$Ixy, for all pixels falling in the window of the target pixel

      c = $\Sigma$Iyy, for all pixels falling in the window of the target pixel

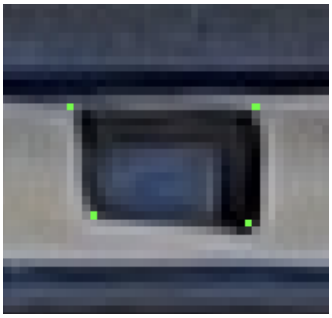
   ii. Compute the Eigenvalues as follows:

      **I1 =( (a+c+$\sqrt{(b^2 + (a-c)^2)}$) / 2**

      **I2 =( (a+c-$\sqrt{(b^2 + (a-c)^2)}$) / 2**

   iii. Compute R value as : **R = (I1*I2)-0.06*(I1+I2)$^2$**

6. After finding the corners perform Non-maximal suppression to remove unwanted corners which might be detected due to noise.
7. Take a window size for non-maximal suppression.
8. Iterate over a window for each pixel and check if the R score for a pixel in the center of the window is maximum. If it is maximum then add it to the list of corners, else change it's R score to 0.
9. For better accuracy I have given a threshold for R score. If the R score of a pixel is max in its proximity and greater than the threshold only then it will be considered as a corner.
10. Return the corners' list after performing non-maximal suppression.



The green pixels in the image denote corners.

## Matching using Sum of Squared Differences Approach:

1. Read the frames which we want to match.
2. Detect corners of these frames using the Corner Detection algorithm.
3. Select a window size for matching.
4. For each corner detected in frame1, check the corners in its proximity in frame2.
5. For each corner in frame2 which is in proximity of the corner in frame1, check the sum of squared differences(ssd) of intensities between patches (around the corner of frame1 and frame2) of size equal to the window size.
6. Select the corner of frame2 for which the ssd value is minimum, i.e. the patch around that corner best matches with the corner of frame1.
7. For better accuracy I have given a threshold ‘t’ for ssd, i.e. if the ssd < t only then I consider it as a match.(The threshold could be subjective to the dataset. A dataset with higher intensity variations or movements could have a higher threshold due to noise.)


Each end point of the green line here depicts a corner matching from one frame to the other.

## Affine Matrix Calculation:

1. Take 3 best matches for affine matrix calculation.
2. Calculate the affine matrix as follows:

$$\begin{bmatrix} a11 & a12 & a13 \\ a21 & a22 & a23 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x1 \\ y1 \\ 1 \end{bmatrix} = \begin{bmatrix} x2 \\ y2 \\ 1 \end{bmatrix}$$

To calculate using 3 matches:

$$\begin{bmatrix} x11 & y11 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x11 & y11 & 1 \\ x21 & y21 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x21 & y21 & 1 \\ x31 & y31 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x31 & y31 & 1 \end{bmatrix} \times \begin{bmatrix} a11 \\ a12 \\ a13 \\ a21 \\ a22 \\ a23 \end{bmatrix} = \begin{bmatrix} x12 \\ y12 \\ x22 \\ y22 \\ x32 \\ y32 \end{bmatrix}$$

xi1,yi1 : coordinates of ith match of Frame 1

xi2,yi2 : coordinates of ith match of Frame 2

**FInal Affine matrix:**

$$\begin{bmatrix} a11 & a12 & a13 \\ a21 & a22 & a23 \\ 0 & 0 & 1 \end{bmatrix}$$

## Stitching for Panorama:

1. Calculate corners for each frame. Find matching between adjacent frames. Based on these matchings, find affine matrices for all adjacent frames. $A_{ij}$ corresponds to the affine matrix between frame i and j.
2. Read the 2 frames to stitch.
3. Take the coordinates of the 4 corners of frame2 and apply affine transformation(using $A_{12}$) to find the coordinates of these corners with respect to frame1.
4. Create a new image of height: max(height of frame 1 + translation with respect to height, height of frame 2 + translation with respect to height).
5. Create a new image of width: max(width of frame 1 + translation with respect to width, width of frame 2 + translation with respect to width).
6. Place the Frame 1 on the new image starting from (0,0) .
7. For placing Frame 2, run a loop from (0, (width of frame 1 +1 )), get the corresponding pixel coordinates of Frame 2 by applying affine inverse (inverse of $A_{12}$).
8. Save this new image (stitched image for frame 1 and frame 2).
9. Now for stitching further frames calculate $A_{ik} = A_{ij} A_{jk.}$ For example, for stitching Frame 3 with the already stitched Frame 1 and Frame 2. We will calculate $A_{13} = A_{12} A_{23}$ . Repeat steps 3 to 8 for stitching Frame 3.
10. Repeat steps 3 to 10 for stitching all frames.
11. Return the stitched image.

### Results of stitching 2 images

Stitched image (Panorama):