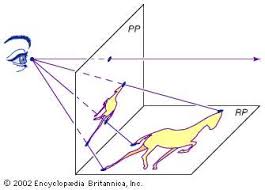
**Image processing using openCV: Perspective transformation**

Perspective projection is used to transform the perspective of the image.



In openCV, we use cv2.getPerspectiveTransform(src, dst) function in which we need four non-collinear points from input image and their corresponding locations in output image to obtain the required 3x3 transformation matrix. The function takes two arguments:

1. src: it specifies 3x2 matrix of three points in the input image.
2. dst: it specifies 3x2 matrix of three points in the output image.

The resultant matrix is then passed to cv2.warpPerspective () function to get the output image.

cv2.warpPerspective():

cv2.warpAffine(image, transformation\_matrix, shape) function takes a 3x3 transformation matrix and convert the image based on the passed transformation matrix. It has three arguments:

1. image: the image which we want to transform.
2. transformation\_matrix: it is a 3x3 matrix which is used to transform the input image.
3. shape: it defines the shape(width, height) of the resultant matrix.

Python code:

import cv2

import numpy as np

img = cv2.imread('geometric.png');

rows,cols,channel = img.shape;

pts1 = np.float32([[56,65],[360,52],[52,380],[389,390]]);

pts2 = np.float32([[0,100],[150,0],[0,300],[300,300]]);

M = cv2.getPerspectiveTransform(pts1,pts2);

result = cv2.warpPerspective(img,M,(cols,rows));

cv2.imshow('original', img);

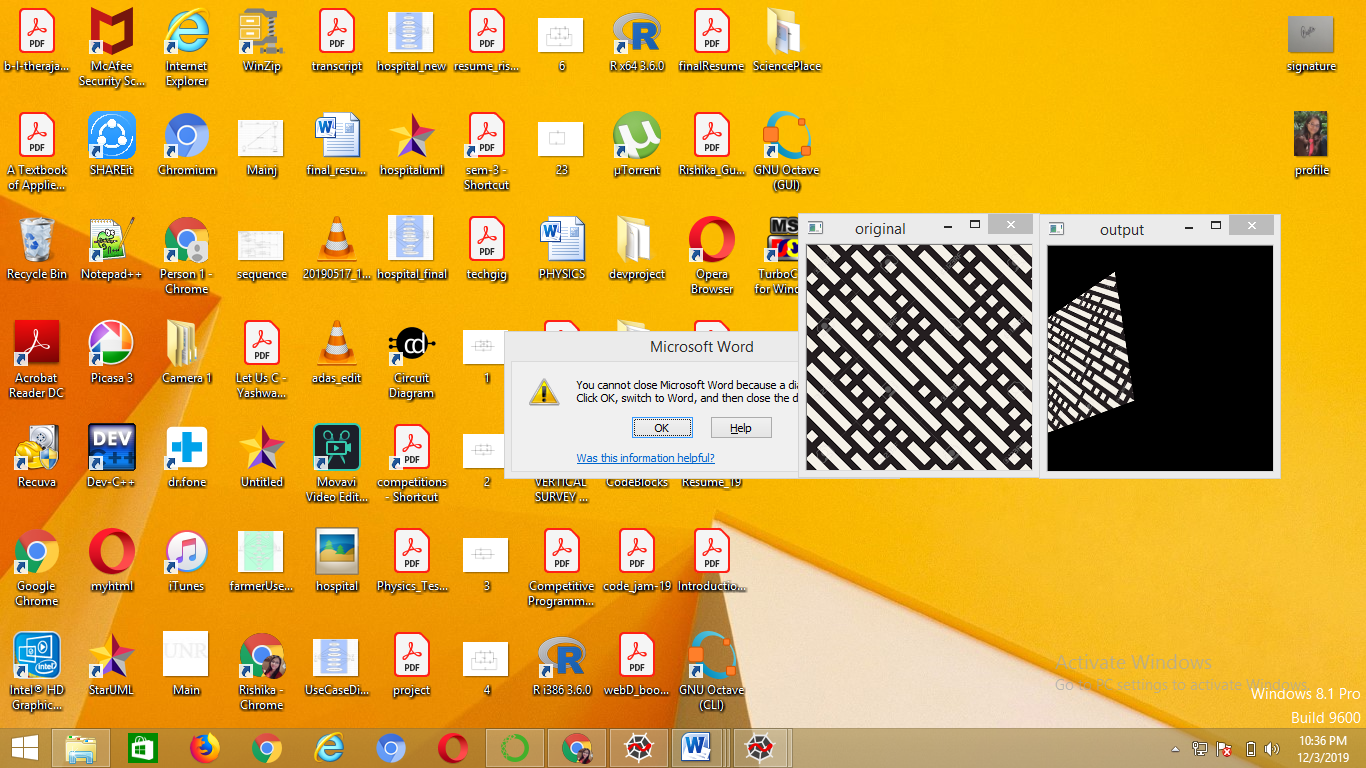
cv2.imshow('output', result);

#wait for 10 seconds

cv2.waitKey(10000);

cv2.destroyAllWindows();

Output:



You can also change the shape of the resultant image by varying the shape attribute in cv2.warpPerspective() function. Let’s try it:

Python code:

import cv2

import numpy as np

img = cv2.imread('tile.jpg');

print(img.shape);

pts1 = np.float32([[0,20],[100,50],[50,335],[150,350]]);

pts2 = np.float32([[0,100],[200,300],[70,400],[175,300]]);

M = cv2.getPerspectiveTransform(pts1,pts2);

result = cv2.warpPerspective(img,M,(300,200));

cv2.imshow('original', img);

cv2.imshow('output', result);

#wait for 10 seconds

cv2.waitKey(10000);

cv2.destroyAllWindows();

Output:

