CS 512 : Assignment: 5 Report

By:Charu Saxena
Department of Computer Science :IIT
29th November,2017

Abstract

In this report I will discuss about various implementation and usage of opencv functions using python 3.6. Some of the implementations and usage discussed are basic computer vision and image processing problems including capturing images and live images, modifications on image.

1.Problem Statement

To write a program to input a pair of stereo image and should be displayed next to each other and perform the following:

a) Allow user to input corresponding points using mouse
b) Using these points estimate a fundamental matrix related to two images and print it,but normalise these points before calculating the matrix.
c) Make the matrix of rank 2
d) Using this matrix if person marka point in one image should display the epipolar line in the other image.and vice versa for both the images.
e) Next will compute and display the epipoles on both the images using fundamental matrix.

2.Proposed Solutions

In this report, we will discuss about the stereo pairs,and how define fundamental matrix and calculate corresponding epipolar line and epipole.

a) After we input stereo, we will input corresponding points as clicked by user on both the images one by one using: cv2.EVENT_LBUTTONDOWN function. Input should be at **least 8 points** for both the images.
b) We will normalise the points using matrices of 1/sigma and other of -mu defined in function as:

$$\begin{bmatrix} \hat{x}_i \\ \hat{y}_i \\ 1 \end{bmatrix} = \begin{bmatrix} (x_i - \bar{x})/\bar{d} \\ (y_i - \bar{y})/\bar{d} \\ 1 \end{bmatrix} = \begin{bmatrix} 1/\bar{d} & 0 & -\bar{x}/\bar{d} \\ 0 & 1/\bar{d} & -\bar{y}/\bar{d} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

```
def normalise(pts1):
    a,b,z =std(pts1, axis=0)
    sig=np.zeros((3,3))
    for i in range(0,3):
        for j in range(0,3):
            if i==j==0:
                sig[i,j] = 1/a
            elif i==j==1:
                sig[i,j] = 1/b
            elif i==j==2:
                sig[i,j]=1
    c,d,q=mean(pts1, axis=0)
    mu = np.zeros((3,3))
    for i in range(0,3):
        for j in range(0,3):
            if i==j==0:
                mu[i,j] = 1
            elif i==j==1:
                mu[i,j] = 1
            elif i==j==2:
                mu[i,j]=1
    mu[0,2]=-c
    mu[1,2]=-d
    M =np.matmul(sig,mu)

    return M
```

c) Then we will save the points entered from both the images in a list of corresponding points for both the images separately. Here u and v are the coordinates for right image and u' and v' are for right image.

$$\begin{bmatrix} u & v & 1 \end{bmatrix} \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix} \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = 0$$

$$\begin{bmatrix} uu' & uv' & u & u'v & vv' & v & u' & v' & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

Now to calculate the fundamental matrix we will find the svd of the matrix form by A matrix and rearrange the last column of V to get the fundamental matrix

d) These points are then send as input to make fundamental matrix of the normalised points function:

```
def make_Fprime_matrix(left,right):
    total_matrix = []
    for i,j in zip(left,right):
        lx,ly,d = i
        rx,ry,e= j
        a= lx*rx,ly*rx,rx,lx*ry,ly*ry,ry,lx,ly,1
        matrix= []
        matrix.extend(a)
        total_matrix.append(matrix)
    U, s, V = np.linalg.svd(total_matrix)
    m = V.T[:,-1].reshape(3,3)
    U, s, V = np.linalg.svd(m)
#making it rank 2
    S = np.diag(s)
    row=S.shape[0]-1
    col=S.shape[1]-1
    S[row,col]=0
    print("\nprint S",S)
#returning the f prime.
    return (np.dot(U,np.dot(S,V)))
```

- As shown in the matrix above we will define each row A matrix and define it.
- To make sure it is of rank 2 we will take SVD of F matrix and make last row of S matrix 0 and then again combine this to F matrix.
- Now we will find epipole using this F matrix, by take left epipole as last column of V and right epipole as last column of U.

```
def epipolar(f_matrix):
   # f_matrix=f_matrix.T
    U, s, V = np.linalg.svd(f_matrix)
    #print(" v \n",V)
    left_epipolar = V.T[:,-1]
    right_epipolar = U[:,-1]
    return left_epipolar,right_epipolar
```

e) Now we will draw a epipolar line using:
We will first find that if the prospective line we are going to draw is going to be horizontal or vertical by checking the x and y coordinate of the point for who we have to get respective epipolar line.
Now if x coordinate is more, that is line is going to be horizontal then for each x in in other image we will determine y=(-c-ax)/b and plot the line. Similarly we will check if the vertical coordinate is more so for each y in 0ther image we will determine x=(-c-by)/a and plot the line

3.Implementation Details

This program is developed using spyder 3.2.3 and runs on Python 2.7

Design issues and their solution

- Only after all points in left is marked to get right epipolar lines only after that only program will allow to enter points for right image to get left epipolar lines(after pressing ‘esc’)
- If corresponding points are taken bad then epipoles might be plot outside the image and error is shown.

To run the program

- Enter the two stereo pair file names in the command line:

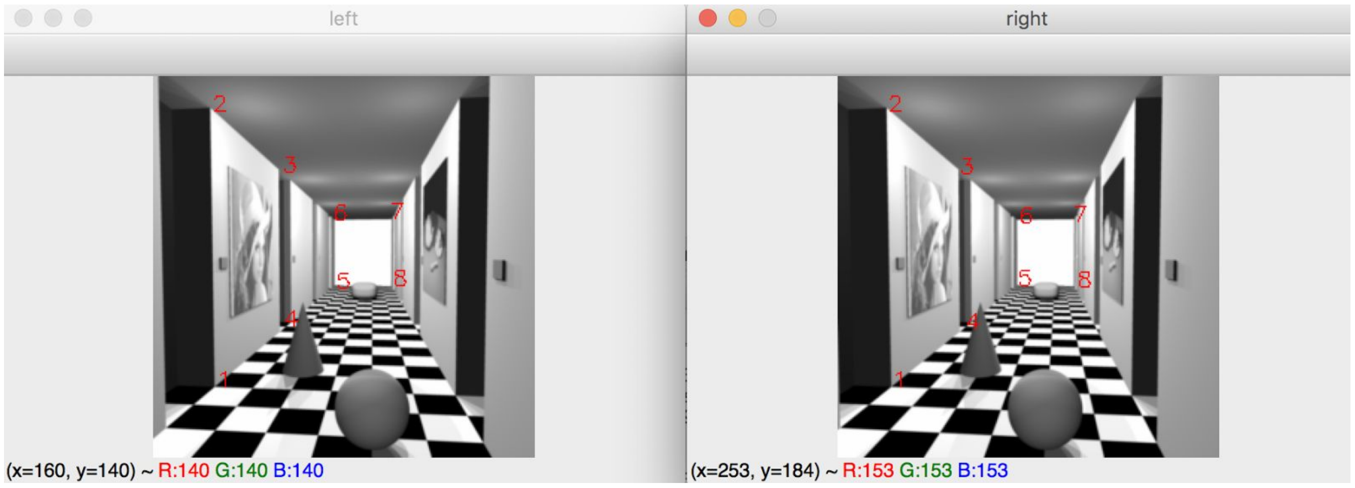
```
python csaxena5.py <left_image> <right_image>
```

- Image one will pop up- press left key to select a point and enter ‘a ’ form keyboard to select it into the list of corresponding points. Do these two step for each point for one image. Press esc to now enter for second image.
- For second image press left key of mouse and area to be selected as corresponding point and press ‘b’ from keyboard to enter that point in the list, do this for each point. After selecting point press esc to do next steps.
- Fundamental matrix and epipoles will be displayed.
- Now to mark epipolar lines:
    - Select a point in left image by using left mouse button and press ‘a’ on the keyboard. This will plot the corresponding epipolar line on right image.
    - Once you are done plotting line of the left image press ‘’esc to go on right image to plot line on left image.
    - Similarly now in right image press a point using left mouse button then press ‘a’ an this will plot the line on left image

4.Results

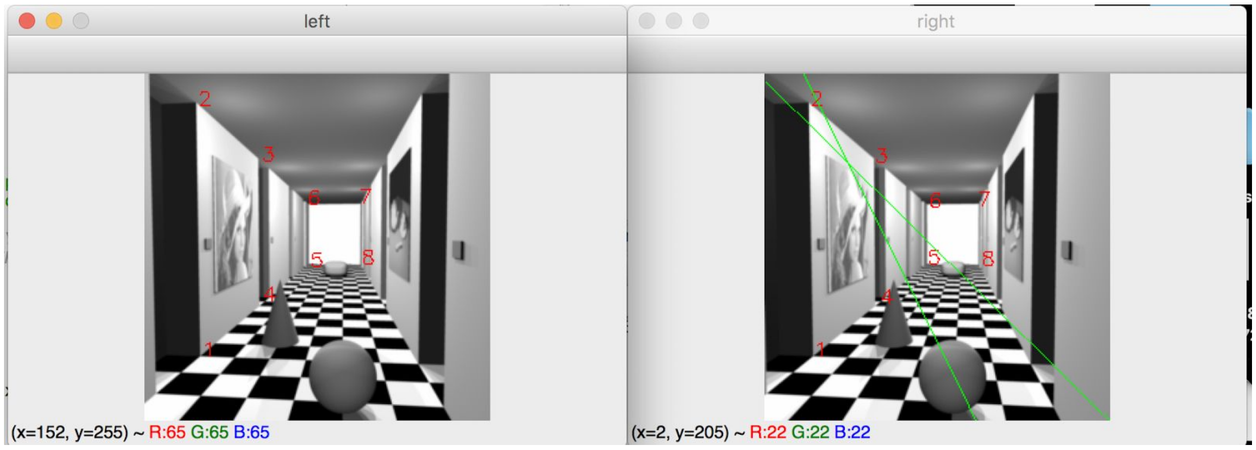
The first here we compare id the corridor image from (http://www.ijrr.org/historic/contents/20_07/abstract/banks/1196-1.html) and we loaded the left and right image and following were the results:

1) Result1 this shows the entering of the corresponding points:


left
right
1
2
3
4
5
6
7
8
(x=160, y=140) ~ R:140 G:140 B:140
(x=253, y=184) ~ R:153 G:153 B:153

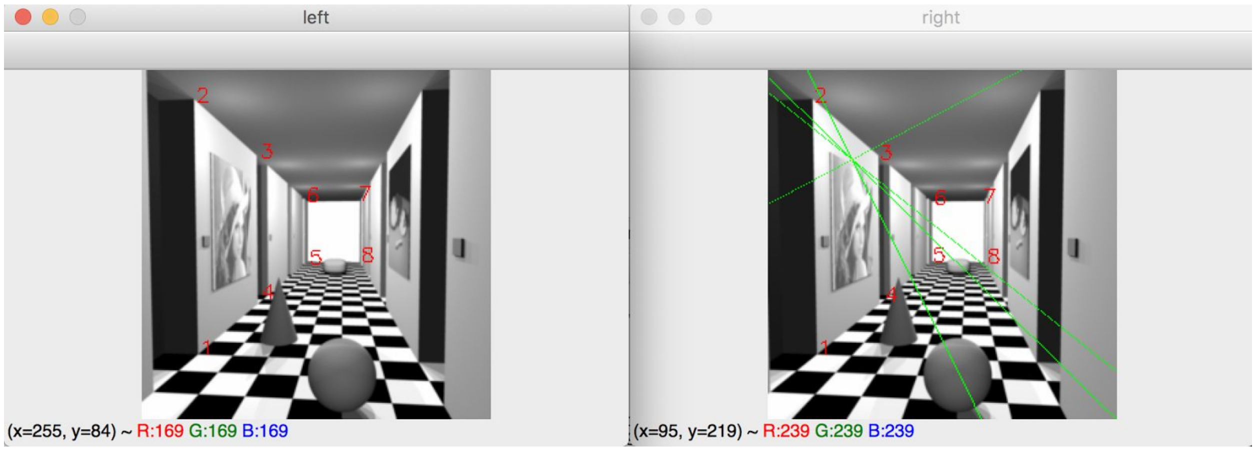
Here first points are entered to left image then to write image and the corners are marked with numbers on both left and right image.

2) Result2 shows plotting of points left image and the corresponding epipolar lines on right image:


left
right
1
2
3
4
5
6
7
8
(x=152, y=255) ~ R:65 G:65 B:65
(x=2, y=205) ~ R:22 G:22 B:22

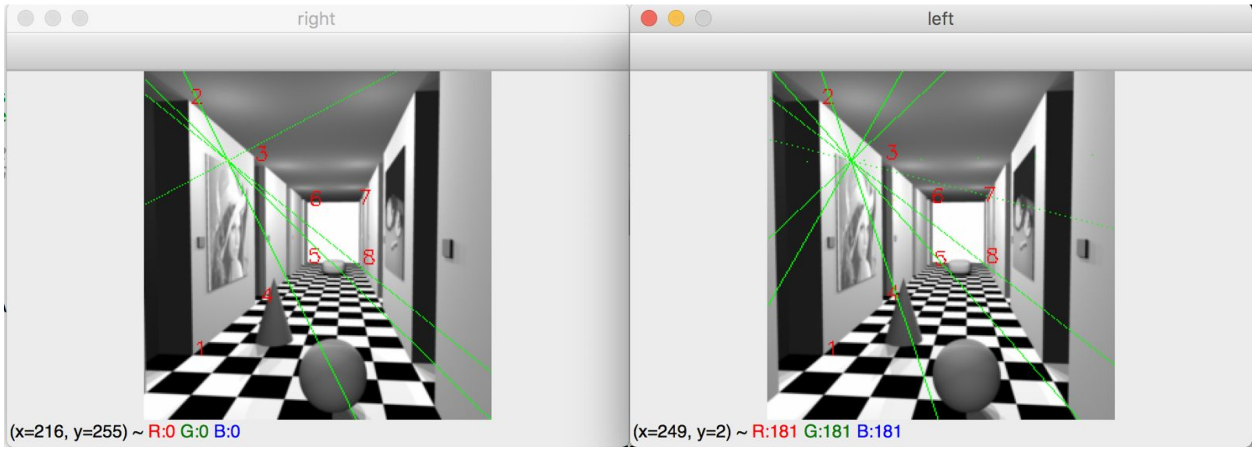
We see that lines on left image were successfully drawn on right image

3) Result 3 shows continue plotting of epipolar lines on right image with points on left image, this is done to show that all lines pass through the epipole and we got following results:


left
right
(x=255, y=84) ~ R:169 G:169 B:169
(x=95, y=219) ~ R:239 G:239 B:239

Thus we see that in right image we got a right epipole successfully.

4) Result4 shows that after pressing 'esc ' now program switches to right image to plot points to show lines on left image and following is the result we observed:


right
left
(x=216, y=255) ~ R:0 G:0 B:0
(x=249, y=2) ~ R:181 G:181 B:181

Thus we see that we able to get points on both left and right image and we also get the epipoles on both the images. The point where all lines are passing through is epipole for here for both left and right image.

CODE:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sat Nov 25 14:38:35 2017

@author: charusaxena
"""
```

```
import cv2
import numpy as np
from numpy import mean,std


left = cv2.imread("/Users/charusaxena/Desktop/corridor-l.tiff")
right = cv2.imread("/Users/charusaxena/Desktop/corridor-r.tiff")

#left = cv2.cvtColor(left, cv2.COLOR_BGR2GRAY)
#right = cv2.cvtColor(right, cv2.COLOR_BGR2GRAY)

rows_r,cols_r,_ = right.shape
rows_l,cols_l,_ = left.shape



#normlaise the list of points we enter and return the M matrix
def normalise(pts1):

    a,b,z =std(pts1, axis=0)
    sig=np.zeros((3,3))
    for i in range(0,3):
        for j in range(0,3):
            if i==j==0:
                sig[i,j] = 1/a
            elif i==j==1:
                sig[i,j] = 1/b
            elif i==j==2:
                sig[i,j]=1
    c,d,q=mean(pts1, axis=0)
    mu = np.zeros((3,3))
    for i in range(0,3):
        for j in range(0,3):
            if i==j==0:
                mu[i,j] = 1
            elif i==j==1:
                mu[i,j] = 1
            elif i==j==2:
                mu[i,j]=1
    mu[0,2]=-c
    mu[1,2]=-d
    M =np.matmul(sig,mu)

    return M
```

```
# make a f matrix , but here used for making fundamental matrix for normalised
#points and make sure that the rank is 2.

def make_Fprime_matrix(left,right):

    total_matrix = []
    for i,j in zip(left,right):
        lx,ly,d = i
        rx,ry,e= j
        a= lx*rx,ly*rx,rx,lx*ry,ly*ry,ry,lx,ly,1
        matrix= []
        matrix.extend(a)
        total_matrix.append(matrix)

    U, s, V = np.linalg.svd(total_matrix)

    m = V.T[:,-1].reshape(3,3)
    U, s, V = np.linalg.svd(m)
#making it rank 2
    S = np.diag(s)

    row=S.shape[0]-1
    col=S.shape[1]-1
    S[row,col]=0
    #print("\nprint S",S)
#returnin the f prime.
    return (np.dot(U,np.dot(S,V)))



## this function will allow to enter points using mouse left key and get x,y coordinates
def on_mouse(event, x, y, flags, params):
   global ix,iy,drawing

   if event==cv2.EVENT_LBUTTONDOWN:

       ix,iy=x,y

       #print("points getting")

a='a'

##this function is used to allow points for making epipolar points
def press_mouse(event, x, y, flags, params):
   global ix,iy,drawing
```

```
    if event==cv2.EVENT_LBUTTONDOWN:
        cv2.putText(left,str(a),(ix,iy), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,255))
        ix,iy=x,y

count=0
xy_left=[]
total_left=[]
total_right=[]

while(1):
    cv2.imshow('left',left)
    cv2.setMouseCallback('left',on_mouse, 0)


    k = cv2.waitKey(20) & 0xFF
    if k == 27:
        break
    elif k == ord('a'):
        count = count +1
        cv2.putText(left,str(count),(ix,iy), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,255))
        print (ix,iy)
        xy_left = []

        xy_left.append(ix)
        xy_left.append(iy)
        xy_left.append(1)
        total_left.append(xy_left)

        #print("\nadded to list",total_left)


#print("\nfinal list ",total_left)



counter = 0
while(1):
    cv2.imshow('right',right)
    cv2.setMouseCallback('right',on_mouse, 0)


    k = cv2.waitKey(20) & 0xFF
    if k == 27:
        break
    elif k == ord('b'):
        counter = counter +1
        cv2.putText(right,str(counter),(ix,iy), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,255))
        print (ix,iy)
        xy_right = []

        xy_right.append(ix)
        xy_right.append(iy)
        xy_right.append(1)
        total_right.append(xy_right)
        #print("\nadded to list right",total_right)




#print("\nnormal F for test not final ",make_Fprime_matrix(total_left,total_left))
norm_left=[]
norm_right=[]

M_left = normalise(total_left)
for i in total_left:
    new= np.matmul(M_left,i)
    norm_left.append(new)


M_right = normalise(total_right)
for i in total_right:
    new2 =np.matmul(M_right,i)
    norm_right.append(new2)


#print("\n\nfinallyy normlaised left ",norm_left)
#print("\nfinallyy  normalised right ",norm_right)

f_prime =make_Fprime_matrix(norm_left,norm_right)
#print("\n prinintg f prime: ",f_prime)


f=np.dot(M_left.T,np.dot(f_prime,M_right))
print("\nfinal fundamental matrix :",f)

def epipolar(f_matrix):
   # f_matrix=f_matrix.T
    U, s, V = np.linalg.svd(f_matrix)
    #print(" v \n",V)
    left_epipolar = V.T[:,-1]
    right_epipolar = U[:,-1]
    return left_epipolar,right_epipolar

el,er=epipolar(f)
t,y,u =el
left_epipole = [t/u,y/u]
i,o,p = er
right_epipole = [i/p,o/p]

pts1 = np.int32(total_left)
pts2 = np.int32( total_right)

print("left epipole:",left_epipole)
print("right epipole:",right_epipole)



while(1):
            cv2.imshow('left',left)
            cv2.setMouseCallback('left',press_mouse, 0)
            k = cv2.waitKey(20) & 0xFF
            if k == 27:
                cv2.destroyAllWindows()
                break
            elif k == ord('a'):
                right[int(right_epipole[1]),int(right_epipole[0])] = [0,255,255]
                line = np.array([ix,iy,1])
                d = np.matmul(line,f.T)
                a = d[0]
                b = d[1]
                c = d[2]
                if a >= b:

                    for x in range(0,rows_r):
                        y = int((-c -a*x)/b)
                        if y < right.shape[1] and y > 0:
                            right[y,x] = [0,0,255]

                else:
                    #print("else")
                    for y in range(0,cols_r):
                        x = int((-c -b*y)/a)

                        if x < right.shape[0] and x > 0:
```

```
                right[y,x] = [0,255,0]
        cv2.imshow('right',right)

while(1):
        cv2.imshow('right',right)
        cv2.setMouseCallback('right',on_mouse, 0)
        k = cv2.waitKey(20) & 0xFF
        if k == 27:
            cv2.destroyAllWindows()
            break
        elif k == ord('a'):
            #print (ix,iy)
            left[int(left_epipole[1]),int(left_epipole[0])] = [0,255,255]
            line = np.array([ix,iy,1])
            d = np.matmul(line,f.T)
            a = d[0]
            b = d[1]
            c = d[2]
            if a >= b:

                for x in range(0,rows_r):
                    y = int((-c -a*x)/b)
                    if y < left.shape[1] and y > 0:
                        left[y,x] = [0,0,255]

            else:
                #print("else")
                for y in range(0,cols_r):
                    x = int((-c -b*y)/a)
                    if x < left.shape[0] and x > 0:
                        left[y,x] = [0,255,0]
            cv2.imshow('left',left)

cv2.destroyAllWindows()
```