Assignment 5: Epipolar lines estimation

Ana Maria Martinez Sidera
Department of Computer Science
Illinois Institute of Technology

November 26, 2017

1. Problem statement

When we take an image using pin-hole camera, we loose an important information, ie depth of the image. Or how far is each point in the image from the camera because it is a 3D-to-2D conversion. So it is an important question whether we can find the depth information using these cameras. Our eyes works in similar way where we use two cameras (two eyes). We will try to solve this case by calculating the fundamental matrix, the epipole and the epipolar lines.

1. First, we need two images that have the same objects but are focused in different ways.
2. We will draw the characteristic points of both photos so that the user can click on them and choose which points to use to estimate the fundamental matrix.
3. Once the user has clicked on at least 8 points we can calculate the fundamental matrix.
4. Calculated the fundamental matrix, we can calculate both epipole both the right and the left.
5. To draw the epipolar lines we will let the user enter a point in the opposite image and draw the epipolar line in the other image.

2. Proposed solution

First, we read the two images and draw all the SIFT features of each of them. Then we draw the features on the image so that the user can select which one he wants to enter in the algorithm to calculate the fundamental matrix. Finally we combine both photos to appear next to each other.

```
    img1 = cv2.imread('corridor-r_1.jpg')
    img2 = cv2.imread('corridor-l_1.jpg')
    orb = cv2.ORB_create()
    bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
    kp1, des1 = orb.detectAndCompute(img1,None)
    kp2, des2 = orb.detectAndCompute(img2,None)
    bf = cv2.BFMatcher()
    matches = bf.knnMatch(des1,des2, k=2)
    for i,(m,n) in enumerate(matches):
        pts2.append(kp2[m.trainIdx].pt)
        pts1.append(kp1[m.queryIdx].pt)
    img1, img2 = drawlines(img1,img2,pts1,pts2)
```

```python
    imagen_final = combinar(img1, img2)
    cv2.imshow('CS_512_imagen6', imagen_final)
    cv2.setMouseCallback('CS_512_imagen6',draw_circle)
```

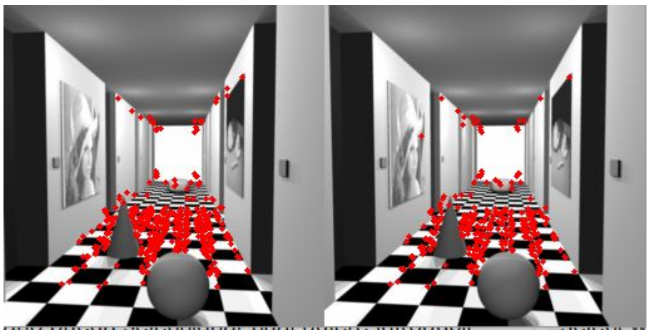

Figure 1: Image with features draw.

Process that we will continue listening to the keys that the user presses at all times.

- The user has to press at least 8 points in the previous images.
- The user must press the letter F to calculate the fundamental matrix.
- The user can press the e key to calculate the right epipole or press the t for the epipole left. In any case, you will get an error if you have not pressed the letters e or t before and want to calculate the epipolar lines.
- The user must press or r or l to let the program begin to listen to what point you want to calculate an epipolar line.
- To show the epipolar line previously calculated you must press or m or n.
- The letter h makes a brief description to the user of how to use the program.

```python
while True:
        key=cv2.waitKey()
        if key == ord('f'):
            if len(lista1) > 8 and len(lista2) > 8 and len(lista1) ==len(lista2):
                lista1 = np.asarray(lista1)
                lista2 = np.asarray(lista2)
                F = compute_fundamental(lista1,lista2)
                print("The fundamental matrix is:")
                print(F)
        if key == ord('e'):
            print("The epipole right is: ")
            er = compute_epipole_right(F)
            print(er)
        if key == ord('t'):
            print("The epipole left is: ")
            el = compute_epipole_left(F)
            print(el)
        if key == ord('r'):
            print("Listening... give me a point in the right image: ")
            cv2.setMouseCallback('CS_512_imagen6',point2_draw)
        if key==ord('m'):
            img1 = plot_epipolar_line(img1,F,point2,er,1)
            imagen_final = combinar(img1, img2)
            cv2.imshow('CS_512_imagen6', imagen_final)
        if key == ord('l'):
            print("Listening... give me a point in the left image: ")
            cv2.setMouseCallback('CS_512_imagen6',point1_draw)
        if key==ord('n'):
            img2 = plot_epipolar_line(img2,F,point1,el,2)
            imagen_final = combinar(img1, img2)
            cv2.imshow('CS_512_imagen6', imagen_final)
        if key == ord('h'):
            print('text help')
        elif key==27:
            cv2.destroyAllWindows()
            break
```

The user has to select the least 8 points in the images to be able to calculate the fundamental matrix. In this case we make sure that the selected points belong to the highlights of the image, taking only those that we have in the SIFT features. There is a 5 error margin so that the user does not have to be more precise than a machine.

```python
def draw_circle(event,x,y,flags,param):
    global lista1, lista2, pts1, pts2, img1
    flag = 0
    if event == cv2.EVENT_LBUTTONDOWN:
        if x < img1.shape[0]:
            k = 0
            for i,j in pts1:
                k += 1
                if abs(i-x)<5 and abs(j-y)<5 and flag == 0:
                    flag = 1
                    lista1.append((i,j,1))
                    lista2.append((pts2[k][0],pts2[k][1], 1))
                    k = 0
            flag = 0

        elif x > img1.shape[0]:
            k = 0
            for i,j in pts2:
                k += 1
                if abs(x-img1.shape[0]-i)<5 and abs(j-y)<5 and flag == 0:
                    flag = 1
                    lista2.append((i,j,1))
                    lista1.append((pts1[k][0],pts1[k][1], 1))
                    k = 0
            flag = 0
        print("Number of points selected:")
        print(len(lista1))
        print("At least 8.")
```

Next, we calculate the fundamental matrix using the 8-point algorithm.

```python
def compute_fundamental(x1,x2):

    n = x1.shape[1]
    if x2.shape[1] != n:
        raise ValueError("Number of points don't match.")
    A = np.zeros((n,9))
    for i in range(n):
        A[i] = [x1[0,i]*x2[0,i], x1[0,i]*x2[1,i], x1[0,i]*x2[2,i],
                x1[1,i]*x2[0,i], x1[1,i]*x2[1,i], x1[1,i]*x2[2,i],
                x1[2,i]*x2[0,i], x1[2,i]*x2[1,i], x1[2,i]*x2[2,i] ]
    U,S,V = np.linalg.svd(A)
    F = V[-1].reshape(3,3)
    U,S,V = np.linalg.svd(F)
    S[2] = 0
    F = np.dot(U,np.dot(np.diag(S),V))

    return F/F[2,2]
```

We are going to calculate the right and left epipole using the following formulas with the fundamental matrix that we have calculated before with the algorithm of the 8 points.

```python
def compute_epipole_right(F):
    U,S,V = np.linalg.svd(F)
    e = V[-1]
    return e/e[2]
```

In the same way as before, but doing the transpose of the fundamental matrix, we can calculate the other epipole on the left side.

```python
def compute_epipole_left(F):
    U,S,V = np.linalg.svd(F.T)
    e = V[-1]
    return e/e[2]
```

The right epipolar line is represented by u_r= F $\overline{p_l}$ $\overline{p_r}$ lies on u_r, that is, $\overline{p_r^T} u_r$=0 or $\overline{p_r^T} F \overline{p_l} = 0$
$Theleftepipolarlineisrepresentedbyu_l$= F^T $\overline{p_r}$ $\overline{p_l}$ lies on u_l, that is, $\overline{p_l^T} u_l$=0 or $\overline{p_l^T}$F $\overline{p_r} =$ 0

```python
def plot_epipolar_line(im,F,x,epipole,number):
    global point1, point2
    m,n = im.shape[:2]
    line = np.dot(F,x)
    t = np.linspace(0,n,100)
    lt = np.array([(line[2]+line[0]*tt)/(-line[1]) for tt in t])
    ndx = (lt>=0) & (lt<m)
    if number == 1:
        cv2.line(im,(int(point1[0]),int(point1[1])),(m-int(t[ndx][-1])
            ,int(lt[ndx][-1])),(0,255,0),2)
        return im
    if number == 2:
        cv2.line(im,(int(point2[0]),int(point2[1])),(m-int(t[ndx][-1])
            ,int(lt[ndx][-1])),(0,255,0),2)
        return im
```

To draw the epipolar line on the left side we need a point on the right side.

```
def point2_draw(event,x,y,flags,param):
    global img1,pts2,pts1, point2, point1
    flag = 0
    if event == cv2.EVENT_LBUTTONDOWN:
        if x > img1.shape[0]:
            k = 0
            for i,j in pts2:
                k+=1
                if abs(x-img1.shape[0]-i)<5 and abs(j-y)<5 and flag == 0:
                    flag = 1
                    point2 = (i,j,1)
                    point2 = np.asarray(point2)
                    point1 = (pts1[k][0],pts1[k][1], 1)
                    point1 = np.asarray(point1)
                    k = 0
        print("Point selected:")
        print(point2)
        flag = 0
```

3. Implementation details

- **Problems found during the implementation of the program:**

Problem 1 : The epipolar lines represented in the images do not make much sense or are badly drawn. They should have the end point of the hall as an epipole and often point to different places so we have miscalculated the lines or the representation is poorly made.

Problem 2 : Due to the first problem maybe the problem comes from before and we are doing a bad computation of the fundamental matrix.

- **Instructions for using the program:**

Step 1 : Select at least 8 points in the two images represented.

Step 2 : Press the letter F to obtain the fundamental matrix of the previous points.

Step 3 : Press e to obtain one of the two epipole.

Step 4 : Press t to get the other epipole.

Step 5 : Press the letter r and then press a point of the red ones in the image on the right.

Step 6 : Press m to represent the epipole line of the point previously captured.

Step 7 : Press the letter l and then press a dot of the red ones in the image on the left.

Step 8 : Press the letter n to represent the epipole line of the point previously captured.

4. Results and discussion

Program 1 : We have obtained all the expected results except for the bad representation of the epipolar line.