ENPM 673: Perception for Autonomous Robots

Midterm Report

3/15/2022

Question 1 :

Coins extraction :

- Reading the image
- a kernel is created using *cv.getStructuringElement is used as it returns a structuring element of the specified size and shape for morphological operations here we need circle*
- This kernel for performing morphological operations such as erosion and dilation to separate the coins from each other using inbulit functions cv.erode and cv.dilate.
-

- The eroded image is then dilated using inbuilt OpenCv function . The dilated image is shown in figure 2

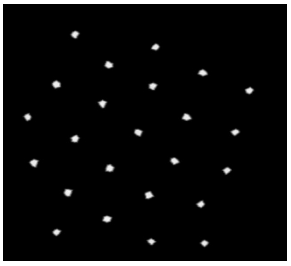


Fig1: Eroded image

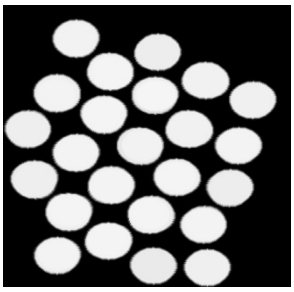


Figure 2: Dilated Image

Counting Coins:

- After this the image is converted into a gray scale,blurred and canny is performed for this image.
- Using Find Contours function, we can find the number of circles in the dilated image.
- Then using inbulit draw contours functions to highlight the coins.

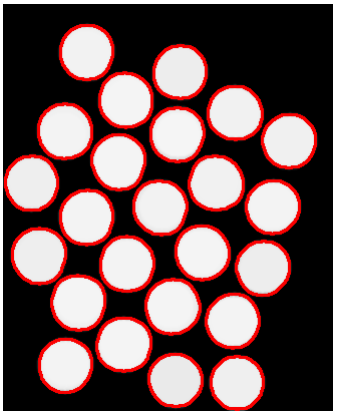


Fig3: highlighted coins

```
/usr/bin/python3 "/home/chaosmachete/Documents/project1 667/que1.py"
chaosmachete@chaosmachete:~/Documents/project1 667$ /usr/bin/python3 "/home/chaosmachete/Documents/project1 667/que1.py"
coins in the image :  24
```

Figure 4 : No of coins

QUESTION 2 :

- To stitch two or more images as one, we first need to detect the matching features between the two. Two images are read to extract the features using inbuilt functions for image stitching
- For this I am using ORB (Oriented FAST and Rotated BRIEF) function to extract the key-points and descriptions which are required for matching.
- For matching the features,I'm using inbuilt bfmatcher and knnmatch functions and matched lines can be seen in the fig below.
-

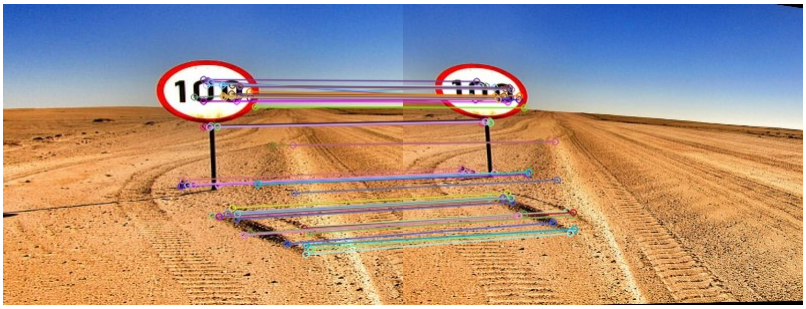


Fig 1: matching lines

- Homography is done on these matching points and we warp the destination image (the second image) onto the source image (the first one) to get the stitched image. But unwanted line is formed which can be seen below figure.

Fig 2: stitched image with noise line

- This line is removed using median blur in built function in openCV module. to remove the line-like disturbance that was visible when I warped the images. The output is as shown in Figure below.

QUESTION 3:

1A) The minimum number of points required are six.

2A)

Calibration procedure:

Estimate the P matrix using scene points and their images

Estimate the intrinsic camera parameters and the extrinsic camera parameters

We find homogeneous coordinates of C in the scene, C is the null vector of P, this gives us the camera translation. We use SVD of P and take out the last column to get C.

The Camera orientation and internal parameters M are in the left 3×3 submatrix of P.

M = KR, K is the camera calibration matrix (upper triangular matrix) and R is an orthogonal rotation matrix. M is a nonsingular submatrix.

3)

$$x = K\left[I|0\right]\begin{bmatrix} R & -RC \\ 0 & 1 \end{bmatrix} X$$

$$x = K\left[R \quad -RC\right] \rightarrow x = KR\left(I|-C\right)X$$

$$P = KR(I|-C) = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix} = [M|C]$$

$$M = KR = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix} \quad C = \begin{bmatrix} P_{14} \\ P_{24} \\ P_{34} \end{bmatrix}$$

$$x = PX \rightarrow \begin{bmatrix} x \\ y \\ z \end{bmatrix} = P \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}$$

$$K = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

these are the unknown camera intrinsic parameters

4)

Steps:

1)we have the image coordinates and the world coordinates.

2)We can use the procedure of improved computation of projection matrix here.

From the relationship between image coordinates, projection matrix and world coordinates, we create a linear system of equations by performing matrix manipulations.

x = PX - AP = 0

Now can solve using SVD

Singular Value decomposition is a generalized version of eigen decomposition where we factorize an (m x n) matrix A of rank r as $A = UV^T > A = (AA^T)$ (sumation of $(A^T A)^T$

3)For a homogeneous system of equations, there is a unique trivial solution P = 0. We are trying to avoid getting a trivial solution; therefore, we need to impose a constraint /IP|| = 1 to avoid it. Now we must find P such that $||AP||^2$ is minimized.

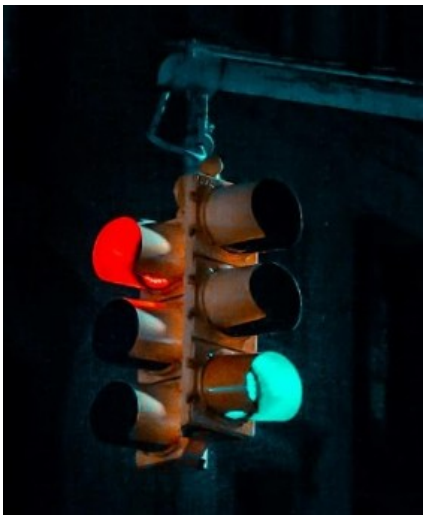
$||AP||^2 = PAA^T p$

4)Once we solve the SVD for A, to find the elements of the projection matrix, all we must do is single out the last column of VT which corresponds to the least eigen value and is hence the least eigen vector. We have 12 values from the last column, so we can reshape it into a 3x4 Projection matrix. we get K, which is the camera calibration matrix. (Normalize if needed). The below figure shows camera calibration matrix

```
chaosmachete@chaosmachete:~/Documents/project1 667$ /usr/bin/python3 "/home/chaosmachete/Documents/project1 667/Question3.py"
[[-0.04423173  0.04947733  0.05626339]
 [ 0.         -0.06674675  0.07361852]
 [ 0.          0.         -0.00013068]]
```

QUESTION 4:

Input image:



1. To create the pipeline, I read and converted the image from BGR form to RGB form. Then we convert the 3d image into 2d format by flattening the width and height.

2. after that I randomly initialized the centroids or centers for each clusters and compute the euclidean distance from each point to the centroid.

3. Now adjust the centroid of each cluster by taking the mean of all the data points in the cluster and repeat the above step until there is no considerable change in the centroid of the cluster Setting No. Of Clusters as K=4 {given in question.

4. Using these centroid points, I plotted the segmented image, for which I obtained the output as shown in the Figure 4.