tal-image-processing-laboratory-1

June 15, 2023

1 1)Write a Program to read a digital image. Split and display image into 4 quadrants, up, down, right and left

[ ]:
```
import cv2
img=cv2.imread('fruit4.jpg')
height, width = img.shape[:2]
half_width = width//2
half_height = height//2

top_left = img[0:half_height, 0:half_width]
top_right = img[0:half_height, half_width:width]
bottom_left = img[half_height:height,0:half_width]
bottom_right = img[half_height:height,half_width:width]

cv2.imshow('Top Left', top_left)
cv2.imshow('Top Right', top_right)
cv2.imshow('Bottom Left', bottom_left)
cv2.imshow('Bottom Right', bottom_right)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

2 2)Write a program to showrotation, scaling, and translation of an image.

[2]:
```
import cv2
import numpy as np

img = cv2.imread('fruit4.jpg')

angle = 60
scale = 1.5
tx, ty = 50, -30


h, w = img.shape[:2]
```

```
M_rotate = cv2.getRotationMatrix2D((w//2, h//2), angle, scale)

M_translate = np.float32([[1, 0, tx], [0, 1, ty]])


img_rotated = cv2.warpAffine(img, M_rotate, (w, h))
img_transformed = cv2.warpAffine(img, M_translate, (w, h))

cv2.imshow('Original', img)
cv2.imshow('Rotated', img_rotated)
cv2.imshow('Transformed', img_transformed)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

3 3)Read an image, first apply erosion to the image and then subtract the result from the original. Demonstrate the differencein the edge image if you use dilation instead of erosion.

[1]:
```
import cv2
import numpy as np

img = cv2.imread('fruit4.jpg')

if img is None:
    print('Image not found or cannot be read.')
else:
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    kernel = np.ones((5, 5), np.uint8)
    erosion = cv2.erode(gray, kernel, iterations=1)
    # Continue processing the image as desired
dilation=cv2.dilate(gray,kernel,iterations=1)

edges_erosion=gray-erosion

edges_dilation =dilation-gray

cv2.imshow('Original',img)
cv2.imshow('Eroded', erosion)
cv2.imshow('Dilated',dilation)

cv2.imshow('Edges (Erosion)', edges_erosion)
cv2.imshow('Edges(Dilation)',edges_dilation)
```

```
cv2.waitKey(0)
cv2.destroyAllWindows()
```

4 4)Read an image and extract and display low-level features such as edges, textures using filtering techniques

[1]:
```
import cv2

img=cv2.imread('fruit4.jpg',cv2.IMREAD_GRAYSCALE)

blurred = cv2.GaussianBlur(img,(5,5), 0)

edges = cv2.Canny(blurred,100,200)

laplacian = cv2.Laplacian(blurred,cv2.CV_64F)

kernel = cv2.getGaborKernel((10,10),3,0,10,0.5,0,ktype=cv2.CV_32F)
gabor = cv2.filter2D(img,cv2.CV_8UC3,kernel)

cv2.imshow('Original', img)
cv2.imshow('Canny Edges', edges)
cv2.imshow('Laplacian Edges', laplacian)
cv2.imshow('Gabor Filtered', gabor)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

5 5)Demonstrate enhancing and segmenting low contrast 2D images.

import cv2

img = cv2.imread('fruit4.jpg', cv2.IMREAD_GRAYSCALE)

clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8)) clahe_img = clahe.apply(img)

thresh_val, thresh_img = cv2.threshold(clahe_img, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)

contours, hierarchy = cv2.findContours(thresh_img, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE) cv2.drawContours(img, contours, -1, (0, 0, 255), 2)

cv2.imshow('OriginalImage', img) cv2.imshow('Enhanced Image', clahe_img) cv2.imshow('Segmented Image', thresh_img) cv2.waitKey(0) cv2.destroyAllWindows()

[ ]:
```
import cv2
```

```
img = cv2.imread('fruit4.jpg', cv2.IMREAD_GRAYSCALE)

clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
clahe_img = clahe.apply(img)

thresh_val, thresh_img = cv2.threshold(clahe_img, 0, 255, cv2.THRESH_BINARY +␣
 ↪cv2.THRESH_OTSU)

contours, hierarchy = cv2.findContours(thresh_img, cv2.RETR_EXTERNAL, cv2.
 ↪CHAIN_APPROX_SIMPLE)
cv2.drawContours(img, contours, -1, (0, 0, 255), 2)

cv2.imshow('OriginalImage', img)
cv2.imshow('Enhanced Image', clahe_img)
cv2.imshow('Segmented Image', thresh_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

[ ]:

[ ]: