Computer vision

CSL7360

Ajit Kumar

Instructor - Prof. Anand Mishra

Question1.

1. Read the image.
2. Divide the image in equal part with respect to width by using slicing in equal part(Shape of image is (386, 700, 3)). So now the shape becomes ((386, 350, 3) and (386, 350, 3)).
3. Take the **X-OR** of both images and then plot it.
4. Take the **Difference** of both images and then plot it.

Google collab link:
https://colab.research.google.com/drive/1NhroULOxb22HOQuUe2DqwO59r6T-KN23

Question2:

1. Read the image.
2. Extract the text from the images and store inside the “result” variable.
3. Format of the output of the easy-ocr is:
   `[([[51, 5], [107, 5], [107, 19], [51, 19]], 'Tniikistal', 0.16009526434695096)]`
4. Calculate the midpoint of state by traversing the list.
5. Finally calculate the euclidean distance between both coordinate.

Google collab link:

https://colab.research.google.com/drive/1LYAOemJxsnPofVInzVpLZ9dutifeUs9p

Question3:

References:
https://scikit-image.org/docs/stable/auto_examples/edges/plot_circular_elliptical_hough_transform.html

1. Read the image in BGR format.
2. Read the image in grayscale.
3. Detect the edge from the gray image.
4. Guess the radius from range.
5. Calculate all the circles for each point in the edge image(edge image is calculated by canny edge detection method).
6. Then peaked the most prominent 1 circle among all the detected circles.
7. Then calculate the perimeter and area.

References:
https://www.geeksforgeeks.org/circle-detection-using-opencv-python/

1. Read the image.
2. Read the image in grayscale.

3. Blur the image to remove the noise if present.
4. Use opencv to detect all the circles present in the image(In opencv it first detects the edge and then detects the circle and for this it takes the parameter in function itself like min_radius,max_radius,min_threshold and max_threshold.
5. Draw the first circle which is detected.
6. At last calculate the area and perimeter.

Google collab link:

https://colab.research.google.com/drive/1cG8k-D_HaiS5U-zTw0ejAF24ultyaYxU

Question4:

References:https://github.com/codespaces

Algorithm:

1. Generate various line on the edge of line that is done by the canny edge.
2. Then find the “r” and “θ” for each the line.
3. The line for which “r” and “θ” is same at each point that line is main line.
4. So in that way, we will find the line and during the hough transformation, we have “θ” value also.
5. So we will subtract the maximum and minimum angle to find the resultant angle.

Google collab link:

https://colab.research.google.com/drive/1nqjwA6tMEyIwWKRhvCll0qGtwEwLlfdO

Question5:

1. Three images downloaded from the given link in the question.
2. Read all the three images one by one and then append them into the list.
3. Resize all the three images by using opencv.
4. Calculate the average of all the three images along x-axis.
5. Subtract the image2 from image1 and then show it.

6. Salt(White) noise adds into the dark region.
7. To add the 5% probability, we first calculate the 5% of total image size. And make the 5% position of total images to 255.
8. To remove the Salt noise, we use the median filter. Generally the median filter is used to remove the salt noise.
9. Construct a numpy array=[[-1, -1, -1], [0, 0, 0], [1, 1, 1]].
10. Then use opencv, 2D filter to convolve over them.

Google collab link:

https://colab.research.google.com/drive/1a3CgB_NCnySVyrW7TKq2hOdnlbTm5zr0#

Question6:

1. Randomly selected 100 images of “**0**” and “**1**” and put them into the two separate folders.
2. Traverse each image of both folders and the folder which contain images of “**0**” only, give them label “**0**” and similarly “**1**” label to images containing “**1**”. Read all the images and compute the horizontal projection profile. Then flatten them and give the label according to the above rule.
3. Then I store all the images and label offline into a desktop using the pickle module in python.
4. Then read the module by loading them and read separately as feature and label.
5. Next divide into **X_train**,**y_train**,**X_test**,**y_test**.
6. Then apply the SVM classifier and KNN from the sk-learn and found the accuracy 100 percent on test data.
7. Then plot the result also to show the result by both classifiers.

Google collab link:

https://colab.research.google.com/drive/1YLGsTSXruJ2JzTL3hHYxGsx5kqQzgjZm

Question7:

References:
https://docs.opencv.org/3.4/d4/d73/tutorial_py_contours_begin.html

For Black text white background

1. Read the image.
2. Convert it into the gray scale.
3. Find the edge using a canny edge detection method.
4. Then find the contours.
5. And draw the contour on the coordinate that is found by the find contour method.

Google collab link:

https://colab.research.google.com/drive/1Scl81aLbPmyjCFE6sQVMoCxT9ZIbU6uO

Question8:

References:

https://docs.opencv.org/4.x/df/dfb/group__imgproc__object.html#gga3a7850640f1fe1f58fe91a2d7583695dac6677e2af5e0fae82cc5339bfaef5038

1. Read the image and the template(Image containing my name “Ajit” and template containing character “A”.
2. Resize the template to 256*256.
3. In template matching, we calculate the distance between template and image and take the region where the difference is minimum.

$$E[i,j] = \sum_{m}\sum_{n}(f[m,n] - t[m-i,n-j])^2$$

So minimize the expression:

Where f is the image and t is the template.

After expanding:

$$E[i,j] = \sum_{m}\sum_{n}(f^2[m,n] + t^2[m-i,n-j] - 2f[m,n]t[m-i,n-j])$$

So generally, maximize the below expression and this is cross-correlation.

$$2f[m,n]t[m-i,n-j])$$

4. So for that we use opencv template matching function.
5. To suppress all the unwanted matching, we set the threshold value to 0.7.
6. Then calculate the location, where matching is above than threshold value.
7. Then draw a rectangle over the matched location.

Google collab link:

https://colab.research.google.com/drive/1lAISHrNTcHbRAEbmh0rnPrNPdarvtxEZ

Question9:

References:

https://docs.opencv.org/3.4/d4/d1b/tutorial_histogram_equalization.html

1. Read the image.
2. Convert into the gray.
3. Using the opencv, write the code to histogram of pixel value with bin size.

   Way histogram plot shows the information:

   a. X-axis contains the gray value of the pixel and the y-axis denotes the number of pixels that have the given range.
4. Then by using opencv perform histogram equalization and observe that

quality of image is enhanced.

Algorithm for the Histogram Equalization:

a. First find the number of pixels for each gray level in the image.
b. Then find the probability of each pixel of gray level.
c. Then find the PDF.
d. Then find the CDF by adding all the previous PDF.
e. Then multiply each CDF to the maximum gray value of the pixel.
f. Then convert each value to the integer.

Google collab link:

https://colab.research.google.com/drive/1hQT8qtiZX7FmTDWviJFbW0ALdpXvs-uf

Question10:

1. Read both images.
2. Extract the text from both images and store into “Result1” and “Result2”.
3. Easyocr extract the text in format of

   [([[51, 5], [107, 5], [107, 19], [51, 19]], 'Tniikistal', 0.16009526434695096)]

4. So by using the indexing and slicing, find the last three digits.

Google collab link:

https://colab.research.google.com/drive/1Otq5lSE7RjaDI4_3Wjm8cbMYnDUe5H5x