Aerial Robotics Kharagpur Documentation
gebe dich nie auf

Kumar Anant 22MA10030

***Abstract*— This documentation belongs to task 1 of the perception task round for the selection of 1st years in ARK.**

I have completed task 1, except for implementing the RRT-connect algorithm (the reason for not being able to do it is unavoidable, will explain in the interview) on *'maze.png'*. However, I have read some papers on the RRT-connect algorithm and how it works. I have used the OpenCV library and some other libraries like math, mpmath, numpy libraries of Python to code. The environment I have used is Jupyter Lab.

I. Aim

This task aims to find the password to a Zip file and implement RRT connect algorithm to the maze Image. The maze image is locked in a Zip file. The password to the Zip file can be found by performing some image processing on provided images as directed in the question.

II. Objectives are to :

- Find the distorted digits in pi_image.png
- Find the filter from distorted digits
- Apply the filter by using bitwise operators [OR, AND, XOR]
- Apply Template matching [from scratch]
- Get the password and then Image
- Perform RRT – Connect Algorithm

Now, Let's Discuss how I got able to clear the above objectives!

III. Finding the distorted digits

if we look closely at 'pi_image.png', we will find that while printing each pixel value of 'pi_image.png', the values are like –

$$30, 10, 40, 10, 50, 90 \ldots, 255 \ldots 20$$

If we consider only the first digit of each pixel value of pi_image.png, they are

$$314159 \ldots 2 \ldots 2$$

, which are nothing but digits of π except some of the digits, since pi_image.png is distorted some of the digits are replaced, my goal is to find those distorted digits. I have made two lists, 'actual_pi_list' and 'distorted_pi_list', to store the digits of real pi (up to 2500 digits as distorted pi contains 2500 digits)and distorted pi, respectively.

Now, I have counted the occurrence of every digit appearing in actual_pi_list and distorted_pi_list.

The result is below :

Actual pi:
3 : 226
1 : 247
4 : 255
5 : 267
9 : 272
2 : 262
6 : 258
8 : 259
7 : 238
0 : 216
Total - 2500 digits

Distorted pi:
3 : 224
1 : 247
4 : 255
5 : 267
9 : 271
2 : 267
6 : 258
8 : 258
7 : 238
0 : 215
Total - 2500 digits

So, distorted digits are clearly visible: $\mathbf{3}, \mathbf{9}, \mathbf{8}, \mathbf{0}$

IV. Finding Filter

So,

$$\lfloor \{3, 9, 8, 0\} \cdot 10 \cdot \pi \rfloor = \{94, 282, 251, 0\}$$

where $\lfloor x \rfloor$ is greatest integer $\leq$ x
Filter = {94, 282, 251, 0}
t_Filter = sort(Filter)
t_Filter = {282, 251, 94, 0}
Filter = np.clip(t_Filter, 0, 255)
np.clip will prevent elements from exceeding 255

$$\mathbf{Filter} = \begin{bmatrix} 255 & 251 \\ 94 & 0 \end{bmatrix}$$

V. Applying the Filter (using bitwise operators)

Bitwise operators in OpenCV are used to apply some operations between two images using bitwise operators like OR, AND, and XOR.

If we have a completely dark image and a completely white image, then applying OR resultant of two will give a white image.

If we have a completely dark image and a completely white image, then applying AND resultant of two will give a dark image.

If we have a completely white image and another completely white image, then applying XOR resultant of the two will give a dark image.

I have applied all three bitwise operators (OR, AND, XOR) as told in the question.
Below are the functions for applying it:-

- cv.bitwise_or(Filter , artwork_picasso)
- cv.bitwise_and(Filter , artwork_picasso)
- cv.bitwise_xor(Filter , artwork_picasso)

The following are the images that I got after applying the $Filter$ on the famous Portrait:

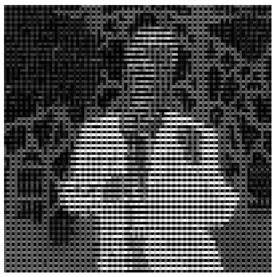

Fig. 1. This is the image we get after applying the AND operator

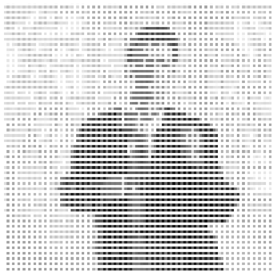

Fig. 2. This is the image we get after applying the OR operator

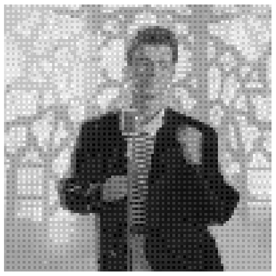

Fig. 3. This is the image we get after applying the XOR operator

VI. Applying Template matching

Template matching uses some mathematical relation to determine how much a template is similar to the provided image. There are various mathematical relations for Template matching.

I have used a relation that is famous with the name TM_SQDIFF.
The formula for Template matching which I used:

$$R(x,y) = \sum_{i\,,\,j} \left(T(i\,,\,j) - I(x+i\,,\,y+j)\right)^2$$

Where $R(x,y)$ is the result of the above formula at (x,y) and i and j are just variables used for iterating over the template and image.
Since, After looking at the collage provided in the question, it is clearly visible that we can get our image in a multiple of 100, so I have incremented my iteration by 100, i.e.....$'for\ \ i\ \ in\ \ range(0,800,100)\ :'$ and also, we can choose any one image for template matching. I have used the XOR image (see Fig. 3) for template matching (since it has more clarity than others)

Since subtraction is used in the relation, it is obvious that location (x, y) having a minimum value of R(x, y) has the highest similarity with the template.

So, in my code, I have accordingly made a function match_template() that takes the template and image and returns the location in the image where the template matches the highest.

The location at which Template matches the most :

$$(\mathbf{100}, \mathbf{100})$$

VII. Get the password and then Image

So, The Password will be:

$$\lfloor (100 + 100) \cdot \pi \rfloor = \lfloor 628.318\ldots \rfloor = \mathbf{628}$$

Hence, we got our maze image which was locked in a Zip folder:

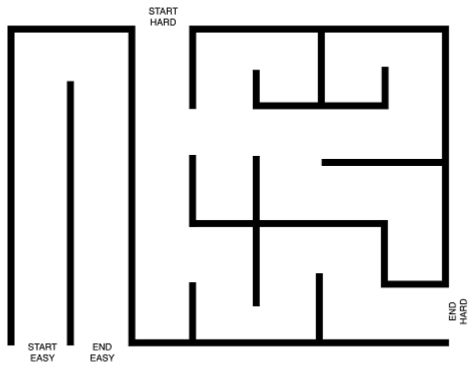

Fig. 4. maze.png

VIII. Perform RRT – Connect Algorithm

RRT-Connect algorithm is an algorithm that uses RRT data structure for path planning. RRT stands for Rapidly exploring random trees. RRT-connect has two RRT expanding trees, one at the start and the other at the end. When they grow, an attempt is made to connect both trees. If possible, then two trees are connected, and backtracking is used to find a path between the start and end point.

References

[1] OpenCV Official Documentations
[2] kuffner_icra2000.pdf in www.cs.cmu.edu/afs/cs/academic/class/15494-s12/readings/