ark
aerial robotics kharagpur

Software and Perception team tasks

General Instructions

The following tasks are intended to check your skills in the field of Computer Vision and Path Planning. You would have been added to the MS Teams for the task round. You're encouraged to figure out the problems on your own, however, feel free to contact Venkatesh Naresh, Upayan Chatterjee, or Amrit Lal Singh, message anyone on MS Teams or just ask it in the MS Teams channel. The final code must be submitted on a github repo in C++ or Python along with appropriate documentation. The documentation format will also be provided to you. We had fun creating these tasks and we hope that you'll enjoy them too. Do not copy code from friends as we will use automated plagiarism checkers. Also, if you use any library functions, you might be asked to explain in detail how it works. Ping us for more problems if you don't find this challenging enough. Good luck!

1 Gebe dich nie auf

Yeltsa Kcir was a big fan of π. To show his devotion towards π he hung this picture in his drawing room for his guests to see(look closely at the pixel values of each pixel beginning from top-left corner). However, one day Grant Sanderson, another self proclaimed π lover and jealous of the fame of Mr. Kcir as a π devotee, added some noise to the image(and also corrupted the link to the original image). The resulting pi_image.png is given below. The digits of π that were distorted can be each multiplied by 10*π and converted respectively to the greatest integer less than or equal to them. The resulting numbers can be arranged in descending row major order and used as a 2x2 filter.

Meanwhile in another part of the world Pablo Picasso came up with this artwork. However allegations were made that the art was not original and was a distorted copy of a famous portrait. You were called up to investigate into the matter and decide whether Mr. Picasso was guilty or innocent. From your highly reliable network of spies you have come to know that given Picasso's aptitude in Mathematics(or lack of it thereof!) he only knows to perform 3 operations: bitwise OR, bitwise AND, bitwise XOR. He has a habit of distorting pictures by applying a filter on pre-existing pictures. He does this by putting the filter on top left corner of the picture and performing element by element operation between the filter and picture values and updating the same value in the picture. then he moves the filter to the right by "s" units where s is equal to one of the dimensions of the filter(here 2). You already have the filter(find it!). All you have to do is apply the filter on the distorted image to recover the famous portrait. This portrait will serve as a template to the next part.

Given below is an image(collage.png) which contains the template found above. The template you have found needs to be scaled to appropriate dimensions(100x100) to make it useful for template matching. apply template matching from scratch(without using inbuilt functions of opencv) and find the coordinates of the top left corner of the matched template in the image. Add the abscissa and ordinate, multiply it with pi and round it to the greatest integer less than or equal to it to get the password to the zip file.

In the images provided, perform the RRT-connect algorithm from start to end point and attach the image of the final path formed.

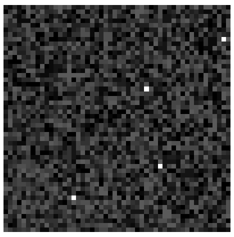

Figure 1: pi_image.png

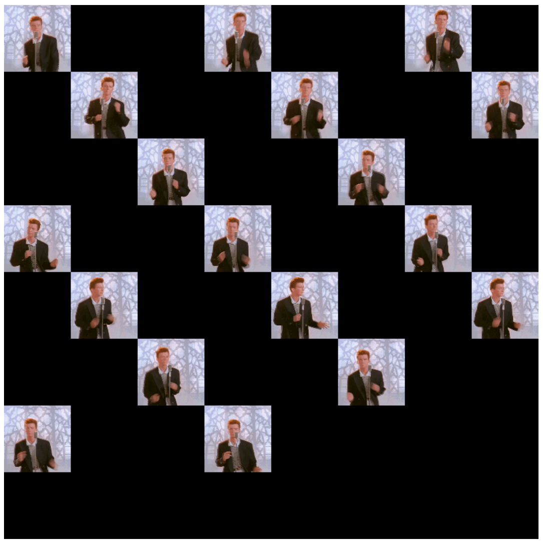

Figure 2: collage.png

2 Finite state machine

In this task you are required to implement finite state machine in python or C++. Read up what finite state machines are and implement a vending machine using FSM logic with the following description:

- The vending machine will show a list of juices available along with their price
- User will enter his choice of drink (the four letter code).
- The user will enter the amount of money he will feed the vending machine, if the amount entered is exact as the cost of the juice, no change is returned, otherwise return the change to the user after vending the juice.
- Initialise each variety of juice to 50. Once the 50 cans are exhausted, display suitable warning message to the user asking him to choose some other juice.
- If all the cans are exhausted, the user needs to type in REFILL to replenish stocks of all the juices, and only then can he use the machine.

Basically you need to define states of the vending machine and device appropriate transition conditions between the different states so as to complete the task. You are not supposed to use if-else statements since the essence of using finite state machine is to avoid clumsy if-else branchings and ease of addition of any new state just by defining a couple of new transition conditions. The different types of drinks and their respective costs are provided below:

Sl. no.	Drink	Code	Cost
1	Pepsi	PEPS	30
2	Mountain Dew	MDEW	30
3	Dr. Pepper	DPEP	50
4	Coke	COKE	20
5	Gatorade	GATO	20
6	Diet Coke	DCOK	30
7	Minute Maid	MINM	25
8	Tropicana	TROP	30

*Note:*You can refer to the following sources to get a starting point for understanding FSMs:

- Rise of finite state machines
- Implementing a finite state machine in C++
- Understanding finite state machines

3 Localization in known environment

When a UAV (Unmanned Aerial Vehicle) needs to navigate from one place to another, it relies on two sources of information: the environment it needs to navigate and its initial position relative to the environment. With these two sources available, the problem can be reduced to solving a maze with given starting and ending positions. In real-life situations, satellite imagery is used to visualize the environment, and Global Navigation Satellite Systems (GNSS), such as GPS, have been used for UAV localization. However, GNSS systems have been prone to inaccuracies caused by the use of radio signals to determine the distance between the UAV and satellites, as well as numerical integration approximations. This has led to the need for an alternative localization method. Localization refers to the process of estimating the position of an object relative to a given reference based on sensor inputs. In the case of drone localization, visual localization involves using camera images to estimate the drone's position in a given environment.

In the context of this problem, you are provided with three files: player.py, utils.py, and MapGeneration.py. (link)

In player.py, an instance of the Player object represents a drone that is lost within a maze, where the position of the drone relative to the maze is hidden. The Player object offers the following methods:

- getMap(): Returns an image of the entire maze.
- getSnapShot(): Simulates the action of the drone's camera and provides a 51x51 image of the environment surrounding the drone.
- move_horizontal(): Controls the movement of the drone along the row.
- move_vertical(): Controls the movement of the drone along the column.

The details regarding the implementation of the Player class can be found in utils.py and MapGeneration.py. Your task is to localize the drone (represented by the Player object) with respect to the maze using the maze image, local snapshots, and control actions. To accomplish this, you need to complete the strategy() function in player.py. At the end of execution of strategy() function, the program should be appreciably certain about the drone's position(i.e. there should be relatively small error in the drone's position), and should print this position.

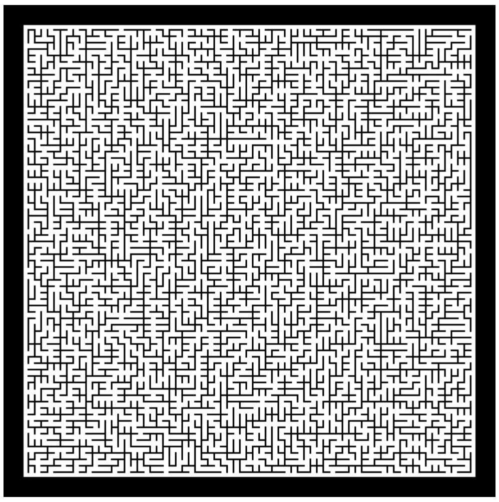
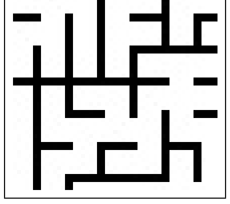

Figure 3: Left image is the maze and right image is the 51x51 environment around the hidden position

4 Unleash Your Genius: Decode the Hidden Image and Outsmart the Kryptonians!

In an unprecedented turn of events, extraterrestrial visitors from the advanced planet of Krypton have graced Earth. Initially planning to conquer our world, the Kryptonians have been captivated by humanity's remarkable progress and now offer us a chance to prove our intellectual mettle. They have shared a perplexing system that conceals an image within it. This system requires an 8-bit, 3-channel image as input and produces a feedback image based on an intricate relationship between the hidden and input images. To rise to this challenge, your task is to develop an efficient algorithm capable of unraveling the hidden image by utilizing appropriate input images and analyzing the generated feedback. Will you prove humanity's intellect by cracking this extraordinary challenge? The fate of the world rests in your hands(unless Superman comes in and saves the day).

You need to implement the solution using a template given in ROS(Robot Operating System).Clone this git repository to find the template.The details about the input and output have been given below:

All the images have 3 channels i.e. each pixels holds 3 values : Red(R),Blue(B),Green(G) .Let X_y denote the value of colour X in the image y

Then the input and output images are related in the following way:

$$R_{input} < R_{hidden} \implies G_{output} = 0$$

$$R_{input} = R_{hidden} \implies G_{output} = 127$$

$$R_{input} > R_{hidden} \implies G_{output} = 255$$

$$G_{input} < G_{hidden} \implies B_{output} = 0$$

$$G_{input} = G_{hidden} \implies B_{output} = 127$$

$$G_{input} > G_{hidden} \implies B_{output} = 255$$

$$B_{input} < B_{hidden} \implies R_{output} = 0$$

$$B_{input} = B_{hidden} \implies R_{output} = 127$$

$$B_{input} > B_{hidden} \implies R_{output} = 255$$

*Note:*Helpful sources:

- ROS Familiarization
- Installing Ubuntu on Virtual Machine
- Basic Terminal Tutorial
- Getting started with Git

5 Get trained to train

In a distant epoch, long after the rise of AI, you, my fellow seeker, emerge as an embodiment of a future human lineage. Amidst a world where machine learning prowess courses through the veins of your kin, you bear the mantle of a transcendent individual. With the echoes of ancient wisdom and the gleam of technological marvels, you embark on a journey to unlock the secrets of existence, bridging realms where destiny and knowledge entwine. In the depths of your advanced lineage, the secrets of training a machine learning model are second nature, ingrained in every member of your kin. Yet, a dormant hunger propels you into an ancient abyss, drawn to a cryptic miniature box labelled “Universal-Serial-Bus”. You had never seen a physical data storage before, you read it and find a folder with unlabelled images and a text containing commandments by an ancient philosopher. You have to train a model on these images, you will be graded according to how well you follow the commandments.

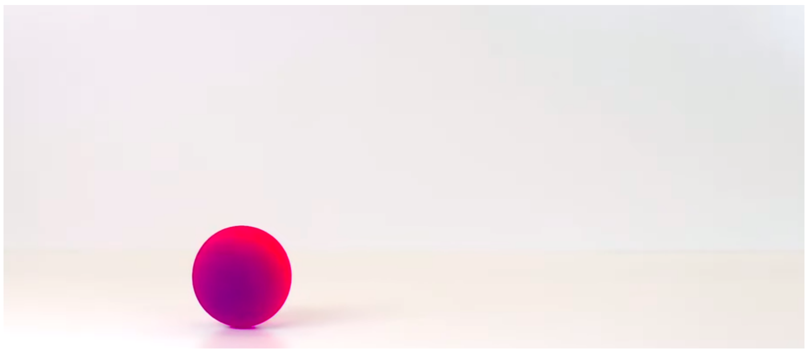

Figure 4: The Input

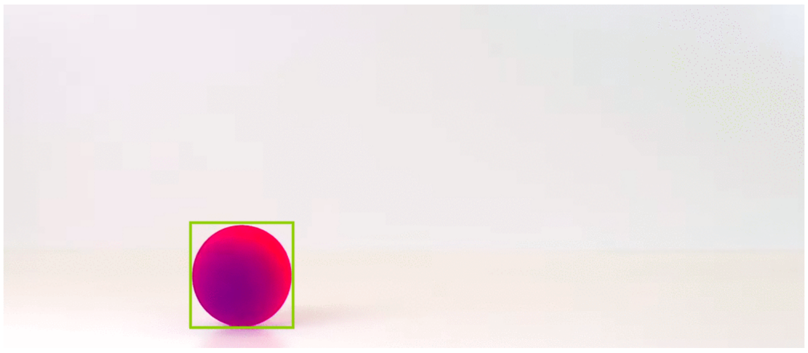

Figure 5: The Expected Output