Aerial Robotics Kharagpur Documentation Template
Shashwat Shukla

Ashwary Anand, Other Author/ Contributor Names in Order of their contribution to the project *

***Abstract*— Should be half column long. Should be clear enough to explain your whole documentation. Similar to a TL;DR**

For task 2.1 minimax algorithm was used , for task 2.2 object detection on the basis canny and hough detection was used (but still some errors left) , task 2.3 face detection was done on the basis of haar classifier which uses trained classifier , task 3.2 i tried to plot the 3d graph using matplotlib and predict the trajectory of plane from there.

I. INTRODUCTION

TASK 2.1: In this task i had to write an agent to play the Tic-Tac-Toe game using minimax algorithm.Minimax is the simple way of predicting the result based on all possible moves left. I wrote the code in python. I use the gym tictactoe evironment given earlier in the problem statement. The major problem which i faced was to figure out how the changes should be made so that the computer can play undefeatedly. Once undestood it was easier to do afterwards.

TASK 2.2: In this task i had to write a program which can calculate the number collision of balls in a given situation, it was also required to show the animation how the ball should move. I wrote the code in c++ and used function such as hough gradient for circle detection and also used canny function for better edge detection. I tried looping the sequence of first detecting a circle and then moving it but as i failed in moving a circle so i came to the approach of using circle a point particle at its center and then using it. I also faced the problem of x and y axis as it was inverted.

TASK 2.3: In this task i have to used a haar classifier file for frontal face and then made a rectangle where the face is probably detected and then we find a region of interest.

TASK 3.2: In this task we have to find the potential enemies by using the csv file in which we had to filter out the data by removing noises and then make a trajectory for the most correct path. Firstly as this was diificult for me to understand so i was only able to plot some data from the file and got to know how to plot the 3d graph from excel sheet using python.

II. PROBLEM STATEMENT

This should cover one full column of page.

TASK 2.1: The problem statement was to write an agent to play the Tic-Tac-Toe game using minimax algorithm. Minimax algorithm basically is just an exhaustive search of game space. You'll be using the gym-tictactoe environment.

TASK 2.2: Using the images of a billiard pool table with the cue and ball to be hit indicated find pairs of collisions assuming perfectly elastic ball-ball collisions and ball-table collisions. Obviously, the balls' centers might not be in-line and this would lead to two divergent ball movements. To simplify this, we assume that in case of ball-ball collision, the incoming ball stops and the incident-ball moves with the same velocity as the incoming ball.

TASK 2.3: For the task we're supposed to implement algorithms to detect a face and to track its motion. WE can use OpenCV's inbuilt function for detecting faces.

TASK 3.2: Imagine you're writing the code for a military aircraft radar. Your electrical engineers have given you a csv file con- taining the detected 3D location of enemy air-crafts at all given time frames. However, this detection is not very reliable. There are time-steps on which an aircraft might not appear on the radar, on other instances - a bird or civilian aircraft might pop up as an enemy aircraft. In general the problem is that your detection might report false positives for some small spans of time, and it might miss true positives for some small time stamps. Your job is to filter out all such false positives and report the count of what you think are true positives over all time stamps. Also, report the entire trajectory for some subset of these true positives .

III. RELATED WORK

TASK 2.1: As python was new to me then it was a very difficult task for me. So first i learned python and watched many youtube videos and read many sites on syntax of python and on minimax algorthim which helped me a lot. I first introduced the computer agent playing with minimax and then introduced a value function which gives the value of the next possible move and returns the best possible move.

TASK 2.2: As this task required the use of opencv knowlegde , i first worked on what does the hough gradient for the circle stores what are its parameters in addition i also read about the parameters of canny and then used a loop first keeping taking a circle and then move accordingly , and then after collision change the circle.

TASK 2.3: First i had to learn about the haar classifier file and then i learned about it to apply them but then after some research i found a trained xml file for frontal face and then i made the program and then the face detection was not tough at all. But then many days after i realised that actually i have to make a game which is till incomplete.

TASK 3.2: I have to first search for the way how to convert the excel file into the python readable array so that then i can further use it and then afterwards i tried to plot the plots

*Write anyone who might have helped you accomplish this eg any senior or someone in three dimensional space or graph. I used matplotlib and numpy for converting the excel shhet and then making a 3d graph out of it.

IV. INITIAL ATTEMPTS

TASK 2.1: Initially the attempt was to change the code and also to introduce the minimax algorithm in between the code such that the main code is not changed and it also leads to better playing. First I had many errors as i didn't used the correct recursive function for minimax and was just solving on the basis of given move and next move.

TASK 2.2: Initially there were many problems related to detection of the circle , it was not detecting the correct nummber of the circles, and still i was facing the problems regarding the slope of the line i tried to find the slope of the line with hough line detection but its still not working. So i just gave a random value to the slope and then i ran my code, and it worked . I also used the distance formula which is $(((\text{x2-x1})^2 + (y1 - y2)^2)^0.5)$

between two points and also calculate the distance betwwen the two circle.

TASK 2.3: Initially first i tried to train my computer about the frontal face detection of which i faced a lot of learning problem and then after sometime i found a already found git repo which had trained frontal face classifier which helped me in making my program.

TASK 3.2: Initially i first tried to convert the excel file to python list which was not tough and afterwards i tried to graph the points given in the file with the help of matplotlib. There i faced a very big issue of having empty spaces in between which is still not solved in my program.

V. Final Approach

This should cover both columns (full page) excluding images

TASK 2.1: In this task the final approach was to used a correct recursive function so that it first calculate the result of all the possible move and afterwards till the game ends, as it requires a lot of computation on the basis of recursive function and give the value on the basis of game won, draw or lost by the computer. First a computer agent using this function was introduced which is specifically designed so that never loses the game. The computer agent with the help of the value function calculates all the future possible moves of itself and the human agent and then decides which is the best move to play. File to run is : task1.cpp

TASK 2.2: My final approach to this problem was to was hard-coding the slope and then using it in the further program for the circle to move. first i detected the number of circles and their centers and then i used them accordingly , i first moved a circle and then according to the collision I proceeded when they collided i changed the center of the moving circle and hence the collision could be seen in the animation produced by the program. File to run is : centerd.cpp

TASK 2.3: My final approach towards this task was using a webcam and then detecting the face on the webcam only by first telling about the classifier and then using it to detect the faces. File to run is finalface.py

TASK 3.2: The final approach for this task as told earlier was trying to manipulate the array obtained from the excel sheet in such a way so that i can find the most plausible trajectory of the plane. This was to be done by taking the difference of each consecutive elements of the array and if its in the range of 1.5 or less then it can be possible trajectory of the enemy plane as in my program i had only uptill now have plotted the graph of some elements from the excel sheet. File to run : plot.py

VI. Results and Observation

TASK 2.1: The final result of this task was an A.I. tic tac toe player which never loses , in my program it plays the first move as ”O” and the human plays the second as ”X”. More algorithm was also available such as Q-value learning or genetic algorithm. But as due to many difficulties i faced in minimax it was difficult for me to work on them.

TASK 2.2: The final result of the above task was a kind of animation or changing the frames with different images so that it could look like animation of moving balls, there could be other possible algo of contour detection which could also detect the center of the circle and the slope of the line. Its major drawback is it sometimes going so much close does not collide with the circle and sometimes from far away only it collides with the circle

TASK 2.3: The only algorithm which i could search for was using haar classifier in the program and then finding the region of interest in the video which is the face. It could come handy in many places such as detecting special objects and known obstacles.

TASK 3.2: In my program the result is it just shows the 3d graph of one of the continous part of the excel sheet, and from that part i could observe that there could be two possible enemies.Using matplotlib i plotted the graph.

VII. Future Work

TASK 2.1: The major problem of this algorithm is as this this algorithm is recursive it takes a lot of time for bigger game such as chinese checkers and especially chess as there are so many possible ways or moves by which player can move which will lead to a very long thinking of computer.

TASK 2.2: The problem of this algorithm till now its not perfect and still it does not uses slope and still it misses some collision and has some extra collision. This can be easily improved with detailed study of more opencv function.

TASK 2.3: the major problem which i faced was how to train my computer about the faces and then using it to finding the faces.

TASK 3.2: In future i could improve on program by first by introducing Nan function from numpy so that i can eliminate the empty cells from the sheet. Then i was thinking about introducing possible line chart in 3d so that when the points will be connected it will show a possible trajectory of the file. I was thinking of using pandas also but i didnt used that in my program.

CONCLUSION

TASK 2.1: The problem was quite a tough problem which was based on changing the already made program in accordance to requirement , I solved this problem with the help and thorough study of minimax algorithm , major difficulties was to how to use the predefined function given already in the environment and to understand what it meant. Its net output was a tictactoe playing agent and with this it was helpful in predicting the possible outcomes of a given move.

TASK 2.2: The problem was easy but still my output is not totally correct as required. Difficulties as i discussed was about kind of inversion of x and y axis and other was detection in detecting the number of circle and still the problem left is not taking the slope.

TASK 2.3: the task was quite a bit tricky but still as i forgot that i had to made game out of the face my task is still incomplete.

TASK 3.2: This task was kind of bit analytical type, which included large data handling which would be obviously useful for any area in any field. In this we had to technically deal with the large amount of data and process it according to ourselves.