Aerial Robotics Kharagpur Software Task Documentation

Nishant Goyal*

***Abstract*— This documentation specifies the approach and results for ARK Software Team tasks 1, 2 and 3.**

For task 1, I have used Minimax Algorithm for cpu to predict the next optimal move. For task 2, I have used Hough Transform to determine the balls and stick. And for task 3, I have used OpenCV's Cascade Classifier to detect the faces and used the Centroid Tracking Algorithm for tracking its motion.

I. INTRODUCTION

A. TASK 1

The first task involves writing code for an agent to play Tic-Tac-Toe game. We do not have to code the game environment instead we need to use gym-tictactoe environment. For the agent I have used the Minimax Algorithm. Minimax Algorithm is used for two player games which gives us the highest value for a move without knowing the other players move. This task was pretty straight forward. We just need to understand the working of this algorithm except this it was simple

B. TASK 2

The second task involves finding the pairs of ball-ball collision and ball-wall collision given the initial image of a billiard pool table with cue balls. There was also an assumption given for simplification that we need to consider all the collisions to be perfectly elastic and also assume that in case of ball to ball collision, incoming ball stops and the incident ball moves the same velocity as the incoming ball. I first used the Hough Transform to detect the circle(balls) and lines(stick) and the slope of line(direction of velocity). I have maintained two lists one containing the information of balls at rest and other containing the information of moving balls which are being updated after each step. This task was also straight forward.

C. TASK 3

The third task involves faces detection, motion tracking and finally making a flappy bird game controlled by motion of our face. We were allowed to use the OpenCV's inbuilt functions for face detection but we were not allowed to use inbuilt functions for Motion Tracking. I have used the OpenCV's Cascade Classifier(Haar Cascade) for face detection which uses a XML file containing the data used for detection, we can also train the classifier for detecting objects whose data is not given. Then I have used the Centroid Tracking Algorithm for tracking the motion of face. Centroid tracking algorithm takes in account the Euclidiean distance between the centroids in one frame and the centroids in other to establish relation between them. The first part (Detecting the face) was pretty straight forward but the second part (Tracking) was relatively tough as we need to determine a fast implementation of algorithm for real-time tracking. And the last part task was making the game for which I have used the randInt function to generate a obstacle of random length. This task was comparatively tougher than the previous two tasks..

*Write anyone who might have helped you accomplish this eg any senior or someone

II. PROBLEM STATEMENT

This should cover one full column of page.

A. TASK 1

The first task involves writing code for an agent to play Tic-Tac-Toe game using minimax algorithm. We do not have to code the game environment instead we need to use gym-tictactoe environment. Using gym-tictactoe environment is compulsory just to test our skills to work on someone else's code. A skeleton for player-vs-minimax agent was also provided here

B. TASK 2

The second task involves finding the pairs of ball-ball collision and ball-wall collision given the initial image of a billiard pool table with cue balls. 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. Moreover it would be better if instead of just printing the coordinates of collision we show this through an animation.

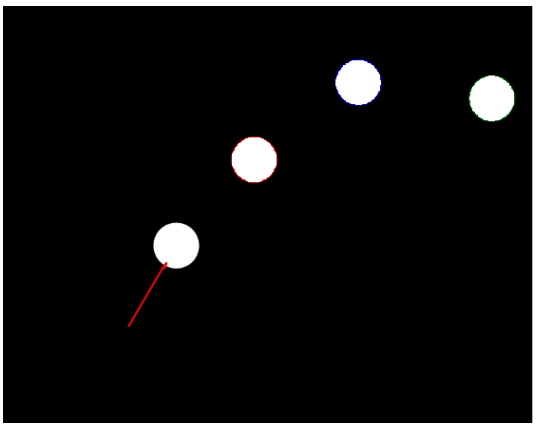

Fig. 1. Billiard pool table

C. TASK 3

The third task involves faces detection, motion tracking and finally making a flappy bird game controlled by motion of our face. For the first part we are allowed to use OpenCV's inbuilt functions for face detection. However for face tracking we were not allowed to use OpenCV's inbuilt functions. Once we have detected and have working face tracking methods, we have to implement a game where the user has to navigate his face through obstacles. This video(only the first ten seconds are relevant) is an example of one such implementation

III. Related Work

Write, in brief, about other approaches used and implemented to tackle the problem and reference them in references.

IV. INITIAL ATTEMPTS

A. TASK 1

For task 1 initial and final attempts were almost similar. I have used the same minimax algorithm for the agent for both initial and final attempts just with few differences. At first I was passing the available actions list directly to the a function (which is getting passed by reference) which was creating lots of errors.

B. TASK 2

For task 2 initial I have used thresholding to get the stick and then used blur for removing noise and it worked well for the one test image but failed for other test images. For finding the angle of stick I have used the houghlines function to find r and theta and then used them to find to points on the line and thus find the angle. For detecting the ball I have used hough cirlces that directly gives me center and radius of cirlcle. And for detecting collision I have taken Euclidiean distance in account.

C. TASK3

For task 3 both my initial and final approaches we same for face detection that I have used Cascade Classifier to detect face. For tracking the face there was difference in my initial and final approach. At first I was directly using the position of first object detected as the new position for face but this gave rise to lot of errors when multiple objects were detected due to noises present.

V. Final Approach

This should cover both columns (full page) excluding images

If report is about implementation, write all the steps of implementing the report with step-wise syntax and details about the errors faced and their solution.

VI. Results and Observation

Compare your results with all the available algorithms which you may have used to tackle the PS. If possible, present your and their results in tabular / graphical format.

Explain the trend of results in details. Mention the drawbacks (if any) of your algo compared to other algo and the reason of picking up the approach over the other if you have implemented any algo over the other.

VII. Future Work

Write about the problems in your algorithm / approach and limitations in testing (if any due to hardware or otherwise) and how to tackle them and any future work which can be done to improve the results further.

CONCLUSION

Write overall about what the problem was, how you solved it, difficulties faced and the net output with it's usefulness to ARK or in general.

References

[1] Khatib, O., "Real-time Obstacle Avoidance for Manipulators and Mobile Robots," International Journal of Robotics Research, Vol. 5, No. 1, pp 90-98, 1986.

[2] Write all the papers and links you have referenced to complete your project.One example for format is given above, Format followed should be ::

[3] Author Names, "Paper Name", Conference / Journal where the paper was published , Year of Publication