# Robot Chase Project Report

Zhe Wang (zw420)

***Abstract*— This is a report for the final project of L310 Introduction to Robotics. The GitHub repository for this project can be found at https://github.com/Ayden666/robot_chase.**

## I. INTRODUCTION

This project is aimed at solving a robot chase problem. Two types of robots are involved. One is the police, whose goal is to approach and catch the baddie. The other is the baddie, whose goal is to avoid being caught by the police. The robots obey the following rules:

- A police robot knows the position and velocity of all robots
- A baddie robot doesn't know the position or velocity of any other robots
- A baddie robot is always faster than a police robot
- Once a police robot comes within a certain distance from a baddie robot, the baddie robot is caught and will stop moving
- Once a baddie robot is caught, police robots will ignore it from there onwards

In my implementation of the solution, three baddie robots and three police robots are spawned in an empty arena. The baddie robots move randomly as they don't have information about the police robots' whereabouts and thus have no way of actively avoiding being caught. A simple controller for the police robots was first implemented, with the police robots only chasing a baddie robot if it comes within a certain distance. A more complex controller for the police robots was then implemented, which uses the Hungarian algorithm to assign each police robot to a baddie robot at the start of the game. The more complex controller was then tested with varying speed differences between police robots and baddie robots.

## II. BADDIE ROBOT IMPLEMENTATION

The baddie robot controller subscribes to each baddie robot's ground-truth positions and publishes velocities for the baddie robots. The velocities are determined by randomly choosing a point on the unit circle, and the vector starting from the origin and pointing towards the random point becomes the robot's direction. The baddie robot controller will re-sample the velocities every 3 seconds. During the 3-second interval, the baddie robot will keep moving at the same velocity.

The baddie robot controller also subscribes to the positions of the police robots, but this is only to check if a baddie robot has been captured by the police. If a baddie robot is captured, i.e., it came within a certain distance from a police robot, it will publish its name in a '/captured' topic and stop moving for the rest of the game.

## III. SIMPLE POLICE ROBOT IMPLEMENTATION

The simple police robot controller subscribes to every robot's ground-truth positions and the baddie robots' velocities, it is responsible for publishing velocities for the police robots. The reason why the police robot controller needs the baddie robots' velocities in addition to their positions is that the baddie robots are always faster than the police robots. As a result, the police robots will never catch up with the baddie robots if they set their goal as the current position of the baddie robots. The police robots' actual goal is the baddie robots' position after a short amount of time, which was calculated using the baddie robot's current position and velocity.

$$goal[x] = baddiePos[x] + u * t * \cos(baddiePos[yaw]) (1)$$

$$goal[y] = baddiePos[y] + u * t * \sin(baddiePos[yaw]) (2)$$

Where $goal$ is the goal position for the police robot, $baddiePos$ is the current position and orientation of the baddie robot, $u$ is the current linear velocity of the baddie robot, $t$ is how much time in the future we are predicting.

The police robot subscribes to the '/captured' topic and will ignore any baddie robot that has already been captured. If all baddie robot has been captured, the police robots will stop moving and the game will end.

## IV. COMPLEX POLICE ROBOT IMPLEMENTATION

The simple implementation of the police robot controller leads to large uncertainty about the amount of time needed for the police robots to capture all baddie robots. The baddie robots might wander away from the police robots, making it seem uncertain whether the police robots can catch all baddie robots at all. As a result, a more complex controller is implemented for the police robots.

At the start of the game, once the police robot controller receives the position of all robots, a cost matrix is calculated using the Euclidean distance between each pair of police robots and baddie robots as the cost. The Hungarian algorithm was then used to assign each police robot to a baddie robot, with the goal of minimizing the total amount of distance between each pair of robots. After the assignment, the police robot will chase the baddie robot assigned to it until it is caught.

## V. MISCELLANEOUS

A simple visual simulator for the robots was provided by the TA and can be accessed at https://github.com/ajshank/SimplePointRobots.

A simple statistics node was also implemented to calculate the time it takes for the police robots to capture all the baddie robots. At the start of each game, just when it's publishing the first one of many velocity commands, the police robot controller will also send a timestamped message to the '/stat' topic, marking the start of the game. After all baddie robots have been captured, the police robot controller will send out another timestamped message to the '/stat' topic. The statistics node, which is subscribed to the topic, can then calculate the amount of time it takes for the police robots to capture all baddie robots.

## VI. EXPERIMENTS

For the experiment on the difference in speed between police and baddie robots, the speed of the police robot is fixed at 0.1 m/s, and the speed of the baddie robots is changed. The result of my experiments is shown in the following graph. As the baddie's speed increases, the police take longer to capture them.

TABLE I

AVERAGE TIME OF GAME COMPLETION WITH DIFFERENT BADDIE SPEED

| Police Speed(m/s) | Baddie Speed(m/s) | Average Time(s) |
|---|---|---|
| 0.1 | 0.12 | 32.48 |
| 0.1 | 0.13 | 38.57 |
| 0.1 | 0.14 | 41.80 |

## VII. CONCLUSIONS

For this project, I implemented one type of baddie robot controller and two types of police controller, one is simpler and the other one more complex. The more complex police robot controller also results in a livelier environment. Experiments were then carried out to investigate the effect of the speed difference between police and baddie robots. The result of the experiments suggests that as baddie robots become faster, it is more difficult for the police robots to catch them. The GitHub repository for this project can be found at https://github.com/Ayden666/robot_chase.