Artificial Intelligence (DT8012) Laboration 1 Report

Andreas Häggström

November 20, 2019

Contents

1 Introduction

This lab-report is a part of my studies in the course *Artificial Intelligence* , **DT8012** at Halmstad University. In section 2 the first part of the lab is covered, here the assignment is to implement four different AIs for operating a robot within **V-REP**[1]. Section 3 covers the second part where the different AIs are to be implemented for 3 Card poker. Both tasks are written using **Python-3.7.2 64 bit**.

2 Task-1

In this task, four different agent implementations are to be added to a mobile robot, **Pioneer P3-DX** within a virtual robot simulator.These are as stated below.

1a. Random Agent

1b. Fixed Agent

1c. A simple Reflex Agent

1d. Memory Agent

The robot is to pick up red energy blocks spread out over the environment. To be able to achieve this the robot is equipped with sensors and actuators.

2.1 Task-1 Sensors

The sensors available for this tasks are the following.

- Ultrasonic Sensor: These are 16 spread out evenly around the robot.
- Energy Sensor: Can see distance of and direction to nearest energy block.

2.2 Task-1 Actuators

The actuators available are the following.

- Left Motor: Controls the speed of the left motor
- Right Motor: Controls the speed of the right motor
- Energy Block Collector: Picks up the nearest energy block if within range

2.3 Task 1.a - Random Agent

The implementation for the random agent ignores the sensors and only operates using the actuators. Every 1000 ticks it sets a random value between -2 and 2 to both the left and right motor. It tries to collect a block every 10 000 ticks.

2.4 Task 1.b - Fixed Agent

The Fixed agent follows a predefined path to follow, reminding of a Beebot [2]. In the best case it collects three blocks, but the robot will fail if any block is spawned in the path of any of these. This since this agent also ignores all sensor data and only operates using the actuators.

2.5 Task 1.c - Reflex Agent

The reflex agent is implemented to move towards the nearest energy block and try to collect it. This agent utilizes the actuators as well as the energy sensor. The motors are set to make the robot align in direction D_{eb} where D_{eb} is the direction to the nearest energy block. Both motors are then set to the same value to make the robot move towards the block and finally the Energy Block Collector is activated when the block is within range.

2.6 Task 1.d - Memory Agent

The memory agent is to change strategy when the current one is no longer optimal. This is implemented by using multiple ultrasonic sensors stated below.

- Ultrasonic Sensors
  - Sensors 3 to 6
    Front sensors used to determine distance to wall
  - Sensors 8 and 9
    Parallel sensors of the right side of the robot, used to align with walls

By utilizing these together with the actuators and the Energy Sensor, the memory agent can align towards the nearest energy block just as the Reflex Agent in 2.5. It can also detect if the it's path is prevented by a wall by using ultrasonic sensors 3 to 6 . If this is the case, it will save the distance d_{eb1} to the block, turn to the left and align with the wall using sensors 8 and 9. It will then follow this wall until the distance to the nearest block is less than d_{eb1}.

2.7 Bonus task

The bonus tasks for this assignment were the following.

1 Your robot agent can collect more than half of the energy blocks.

2 Your robot can quite reliably avoid obstacles, i.e., it does not crash into walls.

Both these criteria are met in the implementation of the Memory Agent in 2.6. During testing, the robot succeeded to collect all blocks several times and almost always collected more than half. The obstacle avoidance is implemented by having the robots motor speeds depend on the readings from the ultrasonic sensors. If there is a wall close on the right side of the robot, the speed of the left motor is lowered and vice versa.

3 Task-2

In this task, three different 3-card poker playing agents are to be implemented as well as an environment for the poker agents to interact with. The agents are to receive three cards, place three bets and finally reveal their cards and deciding who the winner is. The various tasks for this part is as stated below.

2.a Random Agent

2.b Fixed Agent

2.c Build the environment of the game

2.d Analyse the result of the game with a random agent vs. fixed agent

2.e Implement a simple reflex agent and play it against a random agent.

3.1 Task 2.a - Random Agent

The random agent ignores the values of the cards within its hand. Instead it simply bets a value between 0 and 50.

3.2 Task 2.b - Fixed Agent

The fixed agent also ignores the values of its cards and will instead bet 10$ as its first bet and then increment with another 10$ each of the three betting rounds.

3.3 Task 2.c - Environment

The environment of the poker game contains the classes stated below.

- PokerEnvironment
- PokerPlayer
- Hand
- Card

The *PokerEnvironment* is to be seen as a dealer, with it evaluating which of the players won the game as well as dealing the cards. The *PokerPlayer* receives a *Hand* object containing three *Card*s. It then stores this and uses this to calculate its bets unless its a random or fixed agent.

3.4 Task 2.d - Fixed Agent vs Random Agent

For comparison between the fixed and the random agent, the following data is chosen to be of interest to measure performance.

$$Performance = \frac{MoneyWon}{GamesWon} \tag{1}$$

This is since whether an agent wins or looses is simply depending on the random factor of card dealing. Therefore an agents ability to win more money per win is more of interest to measure performance. Agent A could win more money than B simply because it's had the luck to receive better cards more often.

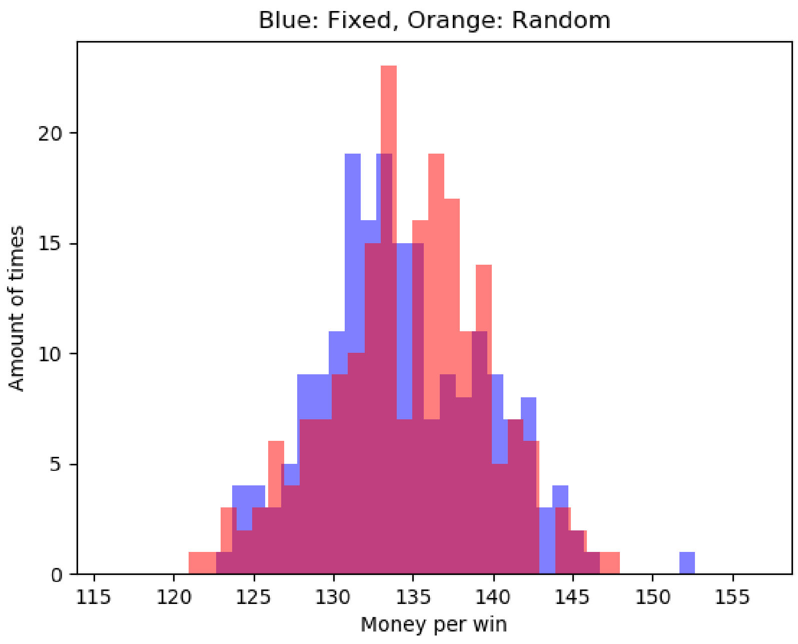

Figure 1: Comparison between Fixed Agent and Random Agent

For this test, the agents faced off for 50 rounds then the performance for the agents was measured and stored in a JSON file. This process was repeated 200 times, after this the JSON file was read in order to generate the histogram seen in Figure 1. From this histogram, it's hard to draw any conclusions whether the fixed agent (blue) or the random agent (orange) is superior. It would though be viable to reason that the random agent tends to acquire higher performance values This could depend on the implementation of the fixed agent since it varies between rounds.

3.5 Task 2.e - Reflex Agent vs Random Agent

This test was performed in the same way as the Fixed vs Random test in section 3.4. The main difference being that the random agent now is playing against a the reflex agent, which will actually vary its bet depending on its cards. The reflex agent calculates its bid using the following equation.

$$betAmount = (rankCount - 1) * 19 + (highestCardValue - 2) \tag{2}$$

This will give an even bet amount between 0 to 50$, with gaps between the rank counts. An agent won't able to bet e.g 18$ since the highest one can bet without a pair is 12$ while the lowest an agent with one pair can bet is 19$.

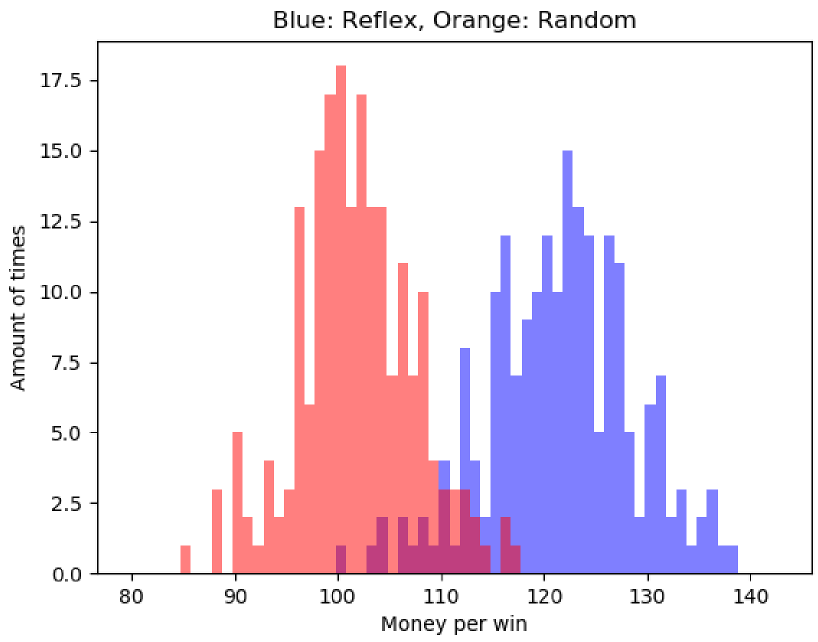

Figure 2: Comparison between Reflex Agent and Random Agent

Here, one can more easily see that the reflex agent is superior to the random agent. Since its performance values clearly is tending to be higher than the ones for the random agent.

3.6 Bonus task

The bonus tasks for this assignment were the following.

1 You implement a poker agent (with memory) which makes betting decisions based on itscurrent hand strength and the amount of money opponent bet last round.

2 Have your poker agent with memory play against the three types of the agents 2a, 2b and2e and make it:

   a deduce which type of the agent it is playing

   b outplay other agents, by adapting its strategy to exploit the weaknesses of the fixed and reflex agents

I chose to only finish bonus task 1 for this task. This was implemented through having a *biddingFactor* initially set to the value of 1. This factor is multiplied with Equation2 in the function to calculate a bid. When the agent receives the bet of a opponent, it then compares this to its own and lowers the factor if the opponent has laid a higher bet.

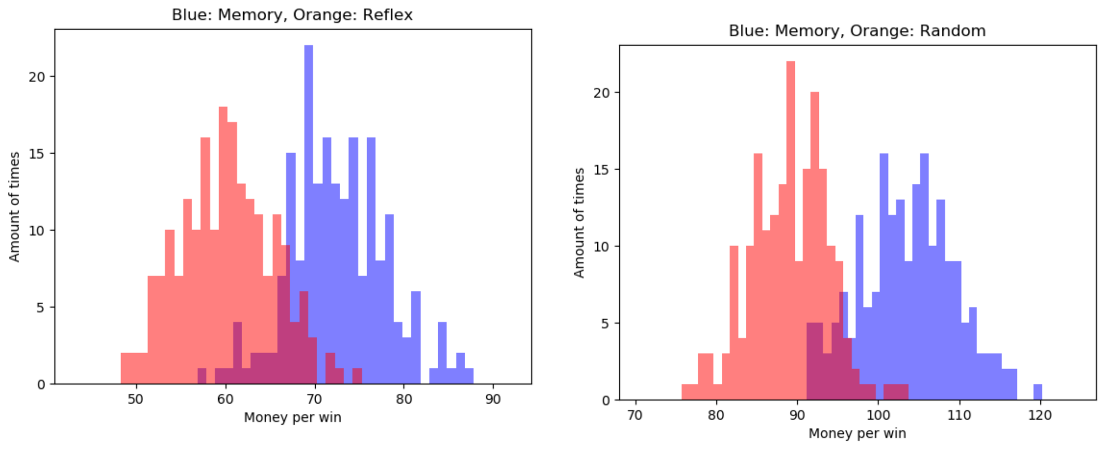

Figure 3: Comparison between Memory Agent and Reflex/Random agents

This agent outplayed the random agent just as the reflex agent did. It also performed better than the reflex agent itself. This is probably since it lowered the amounts that the reflex agent won by lowering its bets. The memory agent clearly had a bigger effect on the reflex agent than the random agent, this can be seen if one compares the performance in the figures above to figure 1.

References

[1] Coppelia Robotics. V-rep robot simulator. URL: http://coppeliarobotics.com/.

[2] Terrapin. Beebot. URL: https://www.terrapinlogo.com/beebot.html.