SYSTEM DESIGN PROJECT 2012

MILESTONE 2 INDIVIDUAL REPORT

MARTIN MARINOV s0932707(ALPHA TEAM – TEAM 8)

PAIR PROGRAMMING

We decided to split the team so we could do pair programming. We thought this would be highly beneficial since it would improve code quality and documentation. Also doing so meant we would lower the chance of undetected bugs slipping through. My teammate is *Laura Ionescu*.

SIMULATOR

Since the vision system was not in functioning order we couldn't proceed with the development of the AI logic. That's why we needed a working simulator as quick as possible. I preferred writing our own Physics engine. The problem with third party libraries was the difficulty that would be required for adjusting forces to make the robot move accurately. On the other hand, third party library would provide better collision detection and response. That is why Laura and I started to experiment with different approaches – with or without libraries. In the end we agreed not to use third party libraries.

NEURAL NETWORKS

We took the idea from last year's *Team 5* to utilize machine learning in the AI. Laura and I did research on different learning algorithms that could be used to control the robot. I started implementing one based on a multi-layer perceptron with the help of the Java library *Neuroph*. The idea is to use manual control for training and thus build the behavior of the robot. During the last week I tried several different approaches to avoid over/under training, experimented with inputs/outputs and varied the number of hidden neurons in the hidden layer. All of the attempts were quantitatively measured to determine the most suitable approach.

So far the best-working solution takes inputs from 10 "virtual distance sensors" (*Figure 1*). Each of them also supplies relative speed of the nearest collision point. Additionally, the distance to the ball and relative robot-ball angle is added to the input. Those 22 features represent the input layer of the perceptron. The hidden layer consists of 45 neurons. There are 9 output neurons (different movement directions ex. *front_right*, *back_left*, *stop*, etc.). The training data is split into training and testing set. The highest accuracy on the test set is kept and if no better becomes available after the next several epochs, the network is assumed to have converged. (*Figure 2*)

Currently the network is capable of successfully moving the robot towards the ball while trying to avoid collision with the opponent. This area is new to SDP and requires more research. The end goal is to use neural networks alongside the normal AI for taking difficult decisions.

Appendix

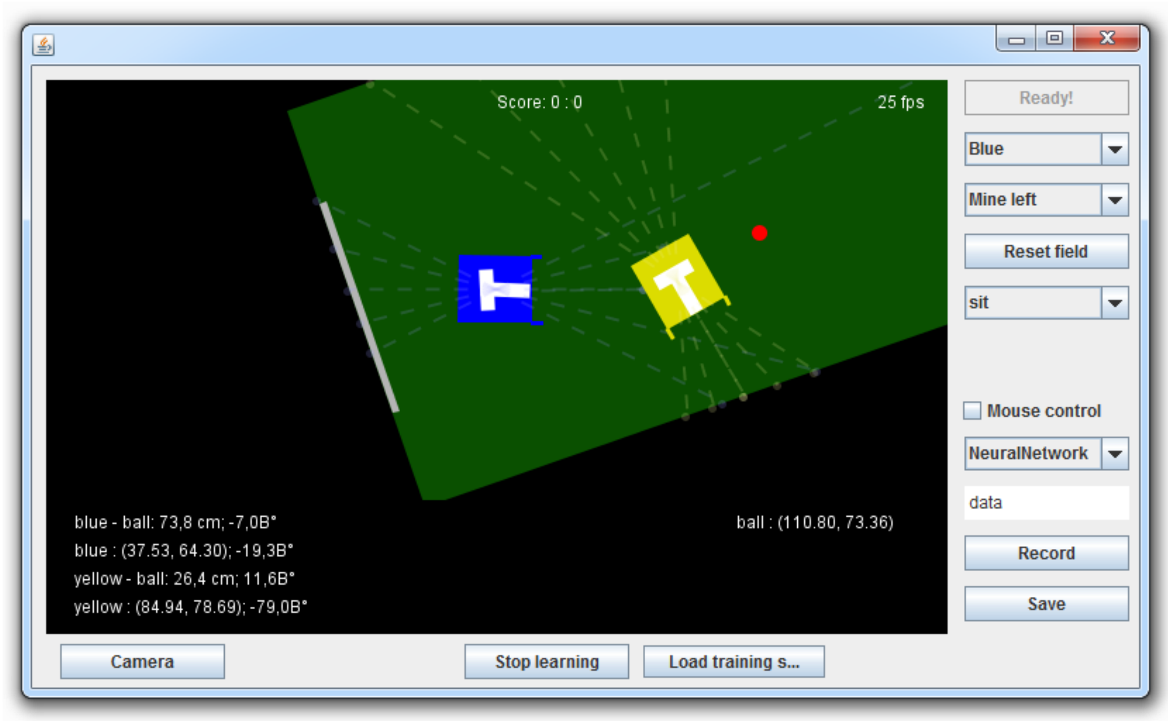

Figure 1 Simulator GUI showing laser-like distance sensors

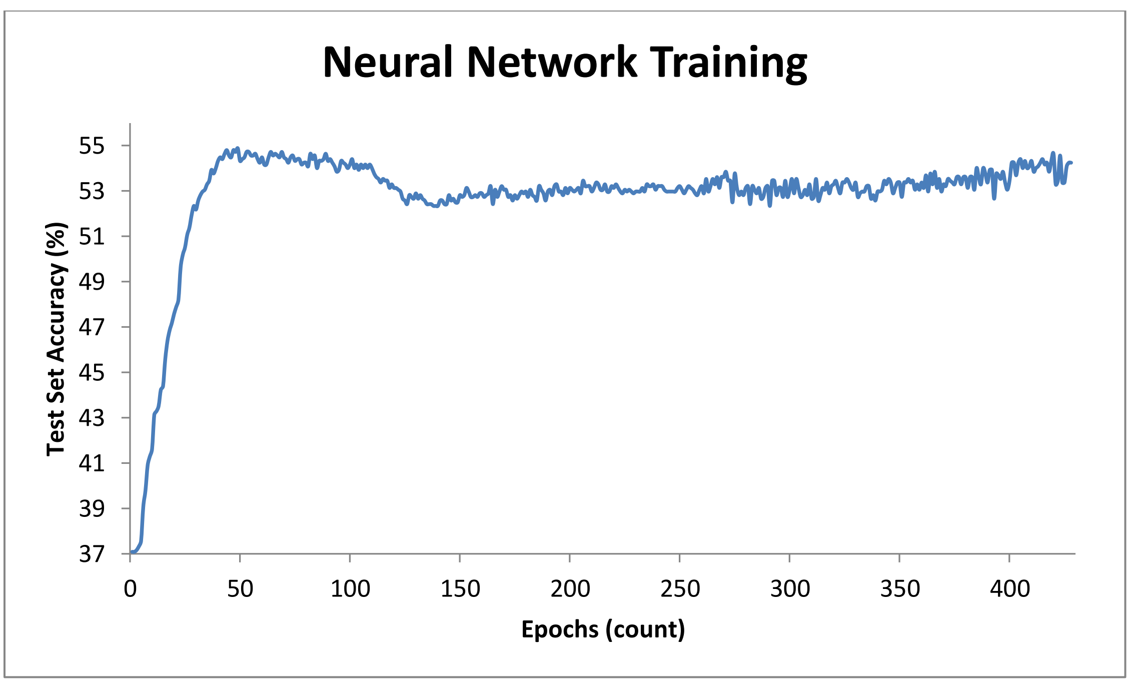

Figure 2 Training history of the (22, 45, 9) neural network