Practical Assignment Computational Intelligence

VU University Amsterdam / University of Amsterdam

3 November 2016

Introduction

The main goal of this assignment is to apply computational intelligence techniques in a practical setting. The setting is a simulated car race on the TORCS platform,[†] an open racing car simulator. You will build controllers, using computational intelligence techniques, that a) are able to keep a racing car on track, b) can race against opponents, and c) can cooperate with team members to gain a competitive advantage. You will work on this assignment in groups of three students.

Ultimately, the goal is to develop a competitive controller so you can to beat fellow students' groups. Below is more detail on the various subparts of the assignment, the submissions system, the requirements for the report, and the grading scheme for the assignment.

Getting started

Download the TORCS platform from
http://mac360.few.vu.nl:9090/cicourse2016/resources.jsp.

This will enable you to test your own implementation locally. To familiarise yourself with the platform, you can start by implementing a very simple controller, e.g. one that just drives the car in a straight line.

Part 1: A basic controller

When you have familiarised yourself with the platform, construct a controller that is able to keep the car on the track for an arbitrary circuit. For now, you can assume that there are no other cars on the track. To develop this controller, use a technique from the Neural Networks category. An important issue you must address is training the network. This could for example be established by providing it with a training set (e.g. composed by you driving the car on the track yourself), but possibly you can think of some alternative training scheme (e.g. online learning). A training set will be made available.

Part 2: Controller that races other cars

Extend the Neural Network you have developed in Part 1 so that it can cope with opponents driving on the circuit (collision avoidance, overtaking, etc). For this purpose, you will use an evolutionary approach, such as NEAT[‡] (Stanley & Miikkulainen, 2002). You can elaborate the network from Part 1 or you can develop a new controller in addition to the controller developed in Part 1 and combine the two controllers via a subsumption architecture (Brooks, 1992). Again, you should consider what training procedure is most appropriate to develop good controllers.

Part 3: Team based racing

[†] http://torcs.sourceforge.net/
[‡] http://www.cs.ucf.edu/~kstanley/neat.html
[§] https://www.springer.com/computer/lncs?SGWID=0-164-6-793341-0

The final extension addresses the issue of cooperation between racing cars. Assume teams consisting of two identical racing cars that are allowed to help each other. The performance of a team will be based on the result of the member to finish first. Examples of strategies could be to use one car to slow other competing cars down, allowing the other car to build up a more comfortable leading position (but note that two cars must have identical controllers). In order to develop such controllers, you will use swarm techniques such as appropriate attraction – repulsion rules.

Submission system

An online submission system has been made available where you can submit your controller and get feedback on its performance compared to your fellow students. The submission system will perform regular runs, showing the score of your racing car controller compared to other (possibly earlier) submissions. If your code does not follow the requirements for the TORCS controller you will also receive a message. A more detailed explanation of the submission system is available on Blackboard.

Report

You are required to write a brief report (at most five pages –excluding references, including graphs and images– formatted according to the Lecture Notes in Computer Science template, by Springer[§]). In the report you should at least specify the following:

- The approaches you have used for each of the three controllers (i.e. what techniques, how did you apply them to this domain, etc.);
- Rationale for the approaches you have selected;
- Estimation of the performance of the controllers;
- Interpretation of the results you have obtained.

Grading

The assignment grade will be based on (50%) the grade for the report and (50%) the performance of your final controller (i.e., the one handed in for Part 3; the other controllers are not graded). Controller performance is evaluated as follows (identical to the scheme used in the online submission system):

- Controllers are tested on *n* circuits, with 5 races for each circuit;
- In each race, 10 randomly selected teams of two cars each participate;
- The starting positions during the various races are pre-defined, giving each team an equal distribution of starting positions;
- The sum of the ranks of the best performing car of each team on a particular circuit is the controller score of that circuit;
- The controller's race score is the average rank over all *n* circuits.

The minimum requirement is that the controller is able to drive at least one full lap on half of the circuits. Submissions that fail to meet this requirement will be

---

§ https://www.springer.com/computer/lncs?SGWID=0-164-6-793341-0

graded 5 (out of 10) under the assumption that a working controller has been submitted. Submissions that do meet the requirement of completing at least one lap of half the circuits are ranked according to their race score. The highest-ranking submission gets a 10, the worst gets a 7, and controllers in between are graded on a linear scale between these extremes.

Finally, the 10 highest-ranking submissions are entered into the final race, where they can earn a bonus point. The winner of the final race earns a full point on the overall assignment grade, the runner up wins 0.5 point and the bronze medallist earns 0.25 point.

Presentation

You should prepare a 5-minute presentation on your approach for the final session on 11 December.

Deadline

The deadline for the report and the submission of your controller is 10 December 2015, 10:00AM.

References

Rodney A. Brooks (1986), A Robust Layer Control System for a Mobile Robot, *IEEE Journal of Robotics and Automation* RA-2, 14-23.

Kenneth O. Stanley and Risto Miikkulainen (2002) **Evolving Neural Networks Through Augmenting Topologies**, *Evolutionary Computation* 10(2):99-127.