PRCO304: Project Initiation Document (PID)

Toy car driven with Artificial Neural Networks and Neuro-evolution

Charlie Creech // 10493048

INTRODUCTION

Self-driving cars are moving towards the focus of the public eye. Such technology being integrated into everyday life seems much more attainable with the advancements made in recent years. Toys are also becoming smarter than ever – some involving technology as advanced as facial recognition that can determine if a child is smiling or not and if they are looking at the toy.

This project is to create software for an autonomous car that could be easily integrated into a simple toy car controlled by a Raspberry Pi computer. The car will have two different "modes" of learning, both utilizing an Artificial Neural Network (ANN) to control its left and right wheels independently. The final product of this project could have potential clients such as toy companies that specialize in remote controlled cars.

One learning mode will be completely autonomous, where the car teaches itself to drive from scratch. The other will initially require a user to give the toy car 'driving lessons' by manually controlling it whilst it gathers data from sensors and actuators.

Comparisons of the results of each learning mode will be made in the hopes of determining the most effective method.

BUSINESS CASE

BUSINESS NEED

Due to the trail-and-error methods of learning, the product of this project is not to be seen as a solution to the control of *real* self-driving cars as that would be extremely dangerous. The intended market for this product is exclusively the toy market.

Currently, the market for remote controlled cars is missing a self-driving toy car that can truly be marketed as *smart*. Other self-driving and "smart" toy cars on the market are simply given a list of commands by the user and follows them consecutively; or the car might be controllable from the user's mobile device.

This project will result in a controller that could be easily integrated into any simple toy car with independently spinning front wheels. The car can be marketed as a *smart* car because it is controlled by an ANN (essentially a small virtual brain).

The concept of teaching a miniature car to drive with this controller by giving it "driving lessons" could be very appealing to consumers of the toy market.

PROJECT OBJECTIVES

1. To design and implement a software environment capable of running and displaying a real-time simulation of an autonomous toy car navigating through a track.
2. To implement a system that allows a user to design tracks for the toy car to navigate.
3. To implement a learning mode for the toy car that utilises evolutionary algorithms to train an ANN by altering the connection weights. The ANN will have a fixed topology. The car will start with absolutely no driving skill but will learn, over many generations, how to effectively navigate a track on its own.
4. To implement another learning mode for the toy car. This mode will require the user to manually control the car through a track. Data collected will be used as training, validation and test data to train an ANN to control the toy car.
5. To provide the user a choice on which of the above learning techniques (objectives 3 and 4) to use before training starts.
6. To conduct experiments to determine which learning mode is most efficient at training the car to effectively navigate a test track.

INITIAL SCOPE

The system is wholly dependent on these features being complete (core deliverables):

1) The proposed system will have a GUI that allows the user to…
   a) Choose one of the two proposed learning modes for the car to use
   b) Run and display a real-time virtual simulation of a car that is being trained to drive around a track
2) Implement the learning mode outlined in project objective 3
   a) Create a ANN where its connection weights are adjustable by evolutionary algorithms
3) Implement the learning mode outlined in project objective 4
   a) Implement a system to manually control the simulated toy car (arrow keys?). Only 2 controls - speed of the left and right wheels.

Not integral to the completeness of the project:

1) Implement a system that allows users to create custom tracks that can be used in the simulation.
2) Conduct experiments on the effectiveness of each learning mode.
   a) Record data (Possible data: % of track complete, time taken to complete track, no. of times a wall was hit, e.c.t.) for both learning methods
   b) Compare data with graphs to make determinations about each learning mode.

Low-priority achievements that would ideally be completed if there is time:

1) Implement the system to work with a real toy car

a) Construct a toy car using wheels, proximity sensors, a body and a Raspberry Pi as a controller
   i) Proximity sensors will be used to detect how close track walls are in several directions
b) Construct a test track out of cheap materials such as cardboard. This will be used to test the toy car.
c) Install an implementation of the controller that will work with a Raspberry Pi onto the car's built-in Raspberry Pi

Time constraints make the following objectives impossible and will not be achieved:

1) Add support for racing multiple cars on the same track at the same time – and for them to be able to interact with each other (e.g. Account for collisions, overtaking, e.c.t.)

RESOURCES AND DEPENDENCES

This project is critically dependent on these resources:

- A computer with an IDE capable of programming C++ installed.
  - C++ will be needed for programming the GUI portions of the project, this includes a window containing a real-time representation of a toy car and the track.
- MATLAB
  - Easy to use libraries for implementing neural networks exist.
  - MATLAB code will be integrated into the main C++ code

If the system is to be implemented into a real toy car controlled with a Raspberry Pi, then several electronic components will be required

- 2 motors for driving the front two wheels
- 5 Proximity sensors for measuring the distance to walls at the front, sides, and front-side-diagonals.
- A Raspberry Pi computer
- A chassis to attach these too

METHOD OF APPROACH

The method of approach for developing this project will be incremental. Core features will be added and tested in an order that makes sense. The GUI will be tackled first, followed by the ANN controllers for the toy car. Less essential features (see initial scope) will only be added once the core features are complete.

Possible technologies are C++ and MATLAB. C++ would be used as the main language, handling GUIs and general processing. MATLAB would be used for any neural network processes and computations. These are subject to change during the project development.

INITIAL PROJECT PLAN

Subject to change as the project proceeds.

Stage	Expected start/finish date	Objective
1. Project Objective 1 and 5	8 Feb – 1 Mar	Design and create an appropriate GUI. This will take a long time.
2. Project objective 3	1 Mar – 16 Mar	Implement an ANN based controller that starts with random connection weights that are altered through evolutionary algorithms.
3. Project objective 4	16 Mar – 1 Apr	Implement a way to control the toy car manually whilst collecting data. Implement a ANN that uses this data as training, validation, and testing data to appropriately control the wheels of the car to navigate a test track.
4. Project Objective 2	1 Apr – 21 Apr	Allow the user to create custom tracks in some way.
5. Testing	21 Apr – 23 Apr	Test final system.
6. Project Objective 6	24 Apr – 25 Apr	Conduct experiments
7. Finalise report	27 Apr – 5 May	PRCO304 Report

CONTROL PLAN

At the conclusion of each stage a highlight report will be written to give an overview of the successes and failures of that stage. This is to counter contingencies that result in deviance from the project plan.

Any meeting, discussed in the communication plan, will be revised. Ad-hoc meetings will be arranged as necessary and reviewed to. Both of these are to prevent too much deviance from the project plan and schedule overrun.

COMMUNICATION PLAN

Weekly meetings with the project supervisor will be conducted to discuss progress. Meetings with other relevant lecturers will be made if there is a specialized problem.

INITIAL RISK LIST

Subject to change as the project proceeds.

Risk	Risk management
Schedule overrun	Precautions for schedule overrun have been made in the project plan. Weekly meetings will provide contingency and prevent going too far off-track.
Difficulty implementing ambitious technology	As the two learning modes are based off of complex concepts (neural networks and evolutionary algorithms for example) problems will likely be encountered during implementation. Using MATLAB will allow these things to be designed, coded, and tested in an isolated environment and later be implemented into the main solution.
Difficulty integrating the controller into a Raspberry Pi controlled toy car	Meetings with relevant lecturers will be conducted to learn how to use a Raspberry Pi to control two wheels and read input from proximity sensors.
Technology failure	Backups will be made before any attempt to implement any new feature.

INITIAL QUALITY PLAN

Subject to change as the project proceeds.

Requirements	Periodically, the current state of the project will be checked against the objectives to make sure that the requirements, especially those of the core objectives, are still relevant and achievable.
Design Validation	The design of the GUI will also be checked against the project objectives. The GUI, and the project as a whole, will be checked for conformity to HCI guidelines and screen-design acceptance.