CS 491 Senior Design
Project Specification Report

DriveSim

Group Members

Bora Ecer

Oğuz Kaan Ağaç

Alp Ege Baştürk

Alptuğ Albayrak

Alperen Erkek

Supervisor:

Prof. Uğur Güdükbay

Jury:

Prof. İbrahim Körpeoğlu

Prof. Özgür Ulusoy

Innovation Expert:

Cem Çimenbiçer (TaleWorlds Entertainment)

October 15, 2018

1. Introduction

Machine Learning (ML) has been around for a while. Lately, it’s been a spotlight of the computer science world and most well developed companies are started to solve their market needs by using ML [1]. Popularity of ML algorithms brings new technical requirements needs as well. The data that is required to train the system is one of the most needed requirements and sometimes it's not that easy to gather high amount of data that the training requires. The end results of the training is not perfect but gives a fair enough approximation. This method of approximation is actually used for simulating real world situations in a basic controlled environment such as games to gather pseudo-world data to see if this type of systems are applicable to the real world [2].

Our project will be a machine learning based simulation game where player will design a vehicle and put sensors that will feed the environmental data to our training system. Depending on vehicle design, placement and type of sensors our training system will teach itself how to drive without crashing into walls/pedestrians/road blocking objects. Depending on how well the designed car works (which depends on how well the user creates the vehicle), user will get a score.

1.1 Description

DriveSim will be a single player, offline, third person, sandbox game where the player is going to design a vehicle from scratch using a variety of vehicle parts such as body materials, engines, connectors and sensors. After the design phase, the vehicle will be placed on a map where the vehicle will be trained to reach the finishing line without crashing to any static or dynamic obstacles. The placement of the sensors is crucial for detecting the obstacles because the training phase of DriveSim will be done by using neural networks which will use the sensors as the input. This training process will be the unique aspect of the game since it will involve neural networks and configurable sensors (reach, placement etc.).

The player will be able to preview the training process if he/she wishes to do so or he/she can simply skip it to a specific trained generation. This will allow us to train the network for multiple genomes in parallel without rendering all of them. The actual win/lose conditions of the game will depend on the game mode that the player will choose such as the cargo mode, collection mode or the traffic mode or simple genom simulation mode.

In the cargo mode, the player will design a suitable vehicle to carry some cargo packages to the finishing line. The cargo packages will be able to fall of the vehicle due to the overall map design or obstacles. The main objective of this mode is to carry as many packages as possible to the finishing line.

In the collection mode, the player will design a suitable vehicle to collect as many collectible as possible while avoiding any obstacles. Crashing into the obstacles will cause player to lose points.

In the traffic mode, the main objective is to finish the round with a certain given time limit. In addition to this, there will be more dynamic obstacles, such as other cars, pedestrians, etc. than other game modes and crashing into them will reduce the remaining time that the vehicle has to finish the round.

The simple genom simulation mode is to simulate the very core features of the game. As mentioned above, in this mode the player will be able to preview the training phase of the designed vehicle and observe the improvement genome by genome. In this mode, the player will be able to create various vehicle designs and test them.

Project is planned to be developed using a neural network for the training part. The neural network will be trained using reinforcement learning method. The designed vehicle will be simulated by the neural network for driving. We have two major implementation choices when it comes to implementing the neural network. First choice is to do the calculations on the GPU however, this option requires the physics calculations to be also done on the GPU since otherwise we would create a communication bottleneck between the GPU and the CPU. The second option is to do both calculations on the CPU, for this to be applicable we have to use evolutionary algorithms to optimize the neural network faster, since we can run multiple genomes of a single generation in parallel.

Finally, the game will be empowered by Unreal Engine or Unity and will be created for PC environment. Use of the engine is subject to change because of the aforementioned connection issue of neural network and the engine. Another alternative is using or developing a simpler engine with lower costs as a backup plan.

1.2 Constraints

1.2.1 Technical Constraints

- DriveSim will be a desktop application which will run on Windows Operating System.
- The game will require a decent GPU to be able to run with decent performance.
- The game will be empowered with a commercial game engine like Unreal Engine or Unity. We are considering to select either Unreal Engine or Unity [2], [3]. However, data from the engine should be passed to the neural network for training. This operation is expected to be costly, thus choice of engine will depend on solving this issue. Also a simpler engine might be chosen or developed as a backup plan, if costs with the aforementioned engines are not acceptable.
- The neural network will work on either CPU or GPU. As mentioned before, we have two implementation choices regarding the neural network. Among these, we will find the most efficient one regarding where to do the physics calculations and where to train the neural network.

1.2.2 Social Constraints

- DriveSim will be a single player, offline, third person, sandbox game [5]. After the base game is completed, we may implement a network layer for players to play the game together.

- DriveSim will be in English language.
- After the final demonstration, we might consider inserting other languages to the game.

1.3 Professional and Ethical Issues

The project is not expected to cause professional or ethical issues. All of the assets that will be used in the game will checked for copyright issues. Therefore, it will not contain anything that will violate copyright issues. This will be free to play game and the source code of the game will be provided publicly for educational purposes. Additionally, if we have time to implement online - multiplayer features, to make the game fair for all players the player's vehicle data will not be publicly visible, instead we will provide a "ghost" vehicle that follows the path of the original one. So that the other players cannot copy the original vehicle design to get the highest score.

2. Requirements

2.1 Functional Requirements

- The player will be able to design vehicles using some basic building blocks.
- The player will be able to select sensors with different specifications such as distance, angle of vision, etc. or different sensors such as gyroscopic sensors and etc.
- The player will be able to add sensors to the vehicles. These sensors will be used to guide the vehicle.

- The player will be able deploy the designed vehicle to the training phase.
- The player will be able to preview the training phase of the vehicle with simulation mode.
- The player will be able to skip to a specific generation of the training phase.
- The player will be able to save and load a trained vehicle design.
- The player will be able to select other game modes and play them with the trained vehicle.
- The player will be able to pause/cancel the game modes.
- The player will be able to restart the current game mode.
- The player will be able to configure sound options from options panel.
- The player will not be able to control the vehicles.

2.2 Non-Functional Requirements

- The game interface should be user-friendly.
- Vehicle design mode should be user-friendly and easy to understand.
- Training phase should be completed in a limited time.
- Vehicle movement physics should be realistic.
- Sensor placement should affect the success of the vehicle.
- The game should be bug free.

3. References

[1] "Artificial Intelligence Market Overview 2018 – Neuromation – Medium", *Medium*, 2018. [Online]. Available: https://medium.com/neuromation-io-blog/artificial-intelligence-market-researches-2018-bfc9202a5102. [Accessed: 14 Oct. 2018].

[2] A. Fard, V. Salmani, M. Naghibzadeh and H. Ahmadi, “Game Theory-based Data Mining Technique for Strategy Making of a Soccer Simulation Coach Agent.” *ResearchGate*, 2007. [Online]. Available: https://www.researchgate.net/publication/221141267_Game_Theory-based_Data_Mining_Technique_for_Strategy_Making_of_a_Soccer_Simulation_Coach_Agent. [Accessed: 14 Oct. 2018]

[3] “What is Unreal Engine”, *UnrealEngine*, 2018. [Online]. Available: https://www.unrealengine.com/en-US/what-is-unreal-engine-4. [Accessed: 14 Oct. 2018].

[4] “The world’s leading real-time engine”, *Unity3d*, 2018. [Online]. Available: https://unity3d.com/unity [Accessed: 14 Oct. 2018].

[5] Breslin, Steve. “The History and Theory of Sandbox Gameplay”, 2018. [Online]. Available: https://www.gamasutra.com/view/feature/132470/the_history_and_theory_of_sandbox_.php?print=1. [Accessed: 14 Oct. 2018].