**Virtual Autonomous Vehicle Simulator**

**Cycle 3 Written Report**

Authors:

Sam Vatany

Kegan van Ginkel

Henry Paek

**Table of Contents**

1. System Metaphor……………………………………………………..………………2
2. Project Introduction……………………………………………………………..…..2
3. User Stories…………………………………………..………………………………….2

3.1 Car Controller…………………………………………………………………....2

3.2 Camera Follow for User-controlled Car……………………………………....3

3.3 Waypoint for AI Pedestrians…………………………………………………...3

3.4 Waypoint Navigator for AI pedestrians………………………………….........3

3.5 Character Navigation Controller for AI Pedestrians………………………....3

3.6 AI Pedestrian Spawner………………………………………………………....4

3.7 Car Node for AI Cars……………………………………………………………4

3.8 Car Wheel for AI Cars…………………………………………………………..4

3.9 Vehicle AI Controller…………………………………………………………….5

3.10 AI Vehicle Spawner……………………………………………………………5

3.11 Waypoint Editor for AI Pedestrians…………………………………………..5

3.12 Waypoint Manager Window for AI Pedestrians…………………………….6

3.13 Traffic Light System……………………………………………………………6

3.14 Pedestrian Traffic Light System………………………………………………6

3.15 Intersection/City………………………………………………………………..6

3.16 First Person Camera…………………………………………………………..7

3.17 Camera Sensor……………………………………………………………......7

3.18 AI Car Agent………..…………………………………………………………..7

1. Design Documentation…………………………..…………………………………8
2. Management Plan…………………………………………………………………….9
3. Lessons Learned……….………………………..…………..………………………..10
4. Test Documentation…………………………………………………..……………..10

Appendix A………………………………………………………………………………..11

1. System Metaphor

This software allows the user to drive through a city with simulated traffic. The traffic, vehicles and pedestrians, will be guided by traffic lights and by random paths. The simulator will collect data using machine learning and display it to the user.

2. Project Introduction

2.1 Previous Development

1. Implemented first person camera view for the user-controlled car.
2. Set up the tools necessary to begin development of AI vehicles through Deep Q-learning.

2.2 Intent This Cycle

1. Modify AI vehicle implementation using Deep Q-learning.
2. Set up an AI car agent and a camera sensor for the development of AI vehicles through deep Q-learning.
3. Developing traffic light animation newly in Unity.

2.3 Future Plans

1. Be able to send inputs to the car agent in Unity from Jupyter.
2. Alternatively develop deep Q-learning model in Unity
3. Create a basic reward/punishment system for training in simple scenarios.
4. Export simulator from Unity as a downloadable program.

3. User Stories

3.1 Car Controller

Summary: As a user, I would like to control a car to experience the traffic within the city.

Description: The user will be able to use the arrow keys to maneuver the car in different directions and the space key to apply the brakes for the car.

* Planned Hours: 4
* Actual Hours: 3.5
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Status: Complete

3.2 Camera Follow for User-controlled Car

Summary: As a user, I would like to view where the car I am controlling is going.

Description: This feature places a camera behind the user-controlled car. It allows the user to modify the distance, height, translational speed, and rotational speed of the camera prior to starting the simulation.

* Planned Hours: 1
* Actual Hours: 1
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Status: Complete

3.3 Waypoint for AI Pedestrians

Summary: As a user, I would like AI pedestrians to have a path they could follow.

Description: Create waypoints through which AI pedestrians can traverse. The waypoints allow the user to determine the area the pedestrian can travel through and the chance of a pedestrian taking a branch waypoint.

* Planned Hours: 1
* Actual Hours: 1
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Status: Complete

3.4 Waypoint Navigator for AI pedestrians

Summary: As a user, I would like the AI pedestrians to know when they have reached their next waypoint, as well as making decisions on which waypoint to take next.

Description: Create a system so that pedestrians know they have reached a waypoint, whether they should take the next waypoint or a branch waypoint, and where the location of the next waypoint is.

* Planned Hours: 3
* Actual Hours: 2
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Status: Complete

3.5 Character Navigation Controller for AI Pedestrians

Summary: As a user, I would like to have the AI pedestrians know how to navigate the waypoint system. I would also like the pedestrians to know whether they can travel through a crosswalk based on pedestrian traffic lights.

Description: Create a navigation controller that will guide the pedestrian to the next waypoint and determine whether the pedestrian should stop due to a traffic light. It also allows the user to modify the movement and rotation speed, stop distance, and animation for the pedestrian.

* Planned Hours: 10
* Actual Hours: 5
* Coder(s): Sam Vatany and Henry Paek
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Status: In Progress

3.6 AI Pedestrian Spawner

Summary: As a user, I would like to determine how many pedestrians are spawned so that I can simulate different types of traffic.

Description: Randomly spawn pedestrians at the location of waypoints. Give the user the ability to determine how many users are spawned in the next simulation.

* Planned Hours: 2
* Actual Hours: 1
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Status: Complete

3.7 Car Node for AI Cars

Summary: As a user, I would like the AI vehicles to have a path they can follow.

Description: Create car nodes through which AI vehicles can traverse. Each car node points to a next car node or an optional link car node, which allows the AI vehicle’s path to be more random.

* Planned Hours: 4
* Actual Hours: 3.5
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Status: Complete

3.8 Car Wheel for AI Cars

Summary: As a user, I would like the AI vehicle’s wheels to turn and rotate in the same direction as the AI vehicle’s wheel colliders.

Description: Modifies the AI vehicle’s wheel transforms based on the status of the AI vehicle’s wheel colliders.

* Planned Hours: 1
* Actual Hours: 1
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Status: Complete

3.9 Vehicle AI Controller

Summary: As a user, I would like the AI vehicles to know how to drive throughout a city. I would like the AI vehicles to be able to avoid collisions with nearby objects. I would also like the AI vehicles to know how to navigate an intersection based on the traffic lights.

Description: The AI vehicle controller allows the user to determine the AI vehicle’s power and brake force, forward and angled sensor lengths, sensor angles, and target distance. The AI controller modifies the AI vehicle’s path based on the location of the next car node. It also consists of five forward sensors and two angled sensors to determine whether the vehicle needs to brake or countersteer depending on which sensors are activated.

* Planned Hours: 20
* Actual Hours: 14
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Summary: In Progress

3.10 AI Vehicle Spawner

Summary: As a user, I would like to determine how many AI vehicles are spawned so that I can simulate different types of traffic.

Description: Spawns an AI vehicle at every car node up to the number of vehicles the user chooses to spawn.

* Planned Hours: 5
* Actual Hours: 6.5
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Summary: In Progress

3.11 Waypoint Editor for AI Pedestrians

Summary: As a user, I would like to see the path that the AI pedestrians can move through.

Description: In the scene view, this script draws the area the AI pedestrians can move through.

* Planned Hours: 2
* Actual Hours:1.5
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Summary: Complete

3.12 Waypoint Manager Window for AI Pedestrians

Summary: As a user, I would like to have a simple window to create new waypoints, add branch waypoints, and remove waypoints for the AI pedestrians.

Description: A window that allows the user to create waypoint, add brach waypoint, create waypoint before, create waypoint after, and remove waypoint.

* Planned Hours: 2
* Actual Hours: 3
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany, Keegan van Ginkel, Henry Paek
* Status: Complete

3.13 Traffic Light System

Summary: Models the functionality of traffics lights in real world scenarios allowing for similar behavior to be exhibited in the simulation.

* Planned Hours: 15
* Actual Hours:14
* Coder(s):Kegan van Ginkel, Henry Paek
* Tester(s):Kegan van Ginkel, Henry Paek, Sam Vatany
* Status: In Progress

3.14 Pedestrian Traffic Light System

Summary: Implement a system that controls the behavior of AI pedestrians at traffic lights.

* Planned Hours: 6
* Status: Not started.

3.15 Intersection/City

Summary: As a user, I would like to have a road system/city to traverse through.

Description: The user will have a city he/she could drive through with traffic lights to guide AI cars/pedestrians.

* Task 1: Design a road network with sidewalks and crosswalks for pedestrians.
* Planned Hours: 20
* Actual Hours: 6
* Coder(s): Kegan van Ginkel
* Status: In Progress

3.16 First Person Camera

Summary: As a user, I would like to have a realistic point of view when using the simulator.

Description: The first person camera script gives the user a driver’s point of view when running the simulation. It allows the user to look left and right by the pressing the z and c keys, respectively. It also gives the user a backwards view when the user is attempting to reverse.

* Planned Hours: 3
* Actual Hours: 2.5
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany
* Status: Complete

3.17 Camera Sensor

Summary: As a user, I would like to be able to train the AI vehicles using a camera placed on top of the car to gather input data for the deep Q-learning model.

Description: The camera sensor captures every frame and encodes the frame as a JPEG. The encoded frame is then available as a byte array for the deep Q-learning model.

* Planned Hours: 3
* Actual Hours: 3
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany
* Status: Complete

3.18 AI Car Agent

Summary: As a user, I would like an AI vehicle that is able to receive and send information to the deep Q-learning model.

Description: On request, the agent logs its current steering, throttle, brake, velocity, acceleration, and location values.It also will modify its throttle, brake, and steering values based on the output from the deep Q-learning model.

* Planned Hours: 6
* Actual Hours: 9
* Coder(s): Sam Vatany
* Tester(s): Sam Vatany
* Status: Complete

4. Design Documentation

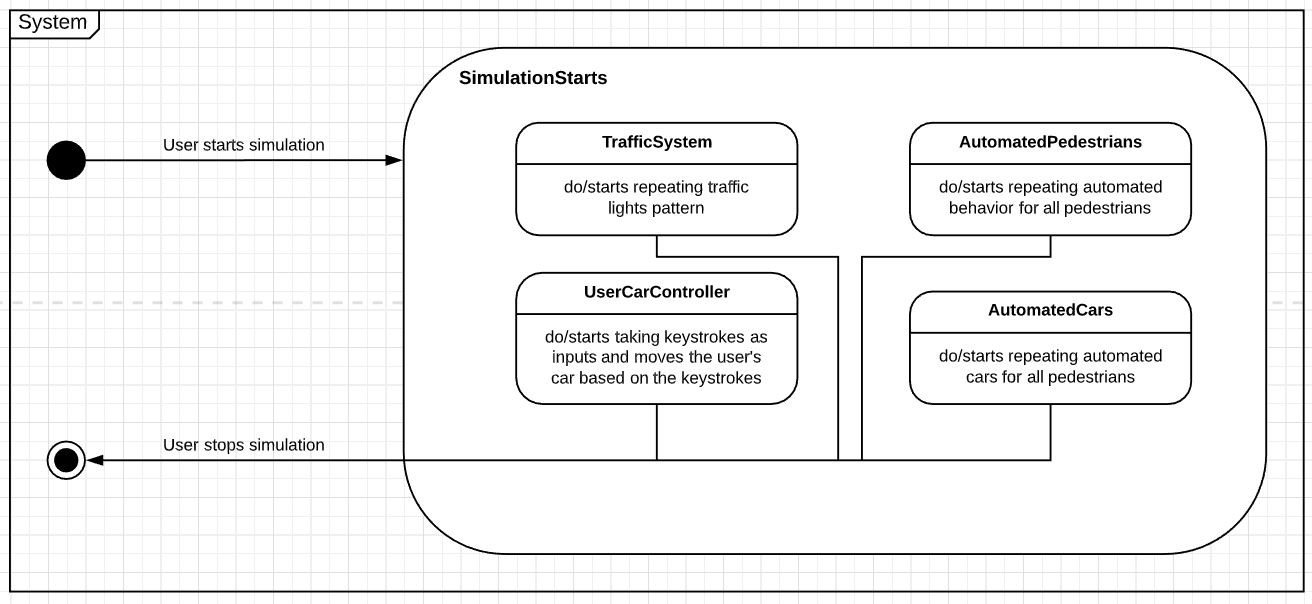
A. Tools/Language

For the project, we are using Blender to design objects and an intersection in the simulation, C# for the behavior and logic of the objects, and Unity for visualizing all the objects and 3D environments and for running the simulation. We chose Blender for modeling because it’s one of the most common 3D modeling tools and is very interoperable with Unity. We chose C# as the language for coding the behavior and logic of the objects based on our group members’ proficiency in the language. Lastly, we chose Unity as a 3D simulation engine for its advantage over other engines; it’s a very specialized engine for the fast development of projects or games involving 3D environments and it is very user-friendly and accessible.

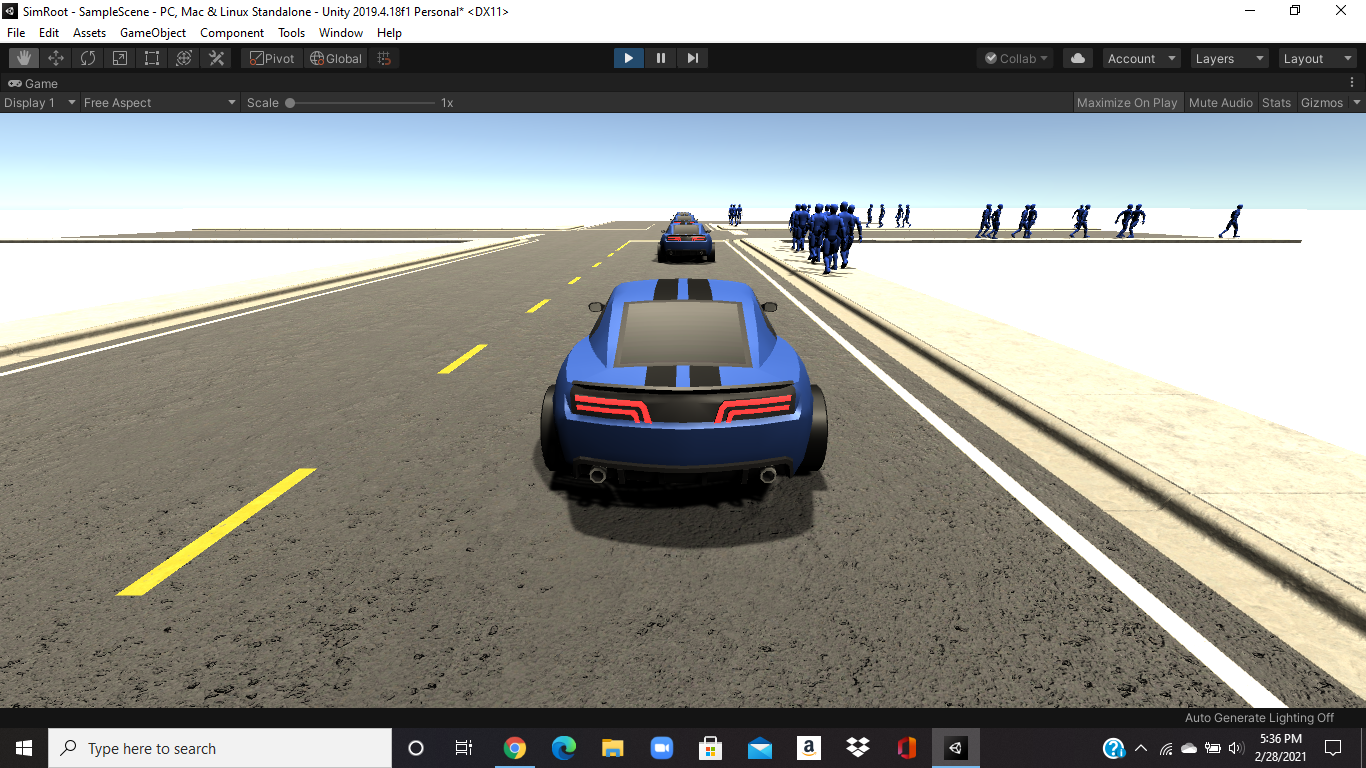
B. UI Layout

Currently, the simulation is only able to be run by using Unity. However, we plan on making our software downloadable which will include a simple start menu on startup. The start menu will include play, options, and exit buttons.The play button will simply run the simulation. The options button will allow the user to modify some variables for the next simulation (e.g., the number of pedestrians or vehicles that will spawn). Lastly, the exit button will close the program.

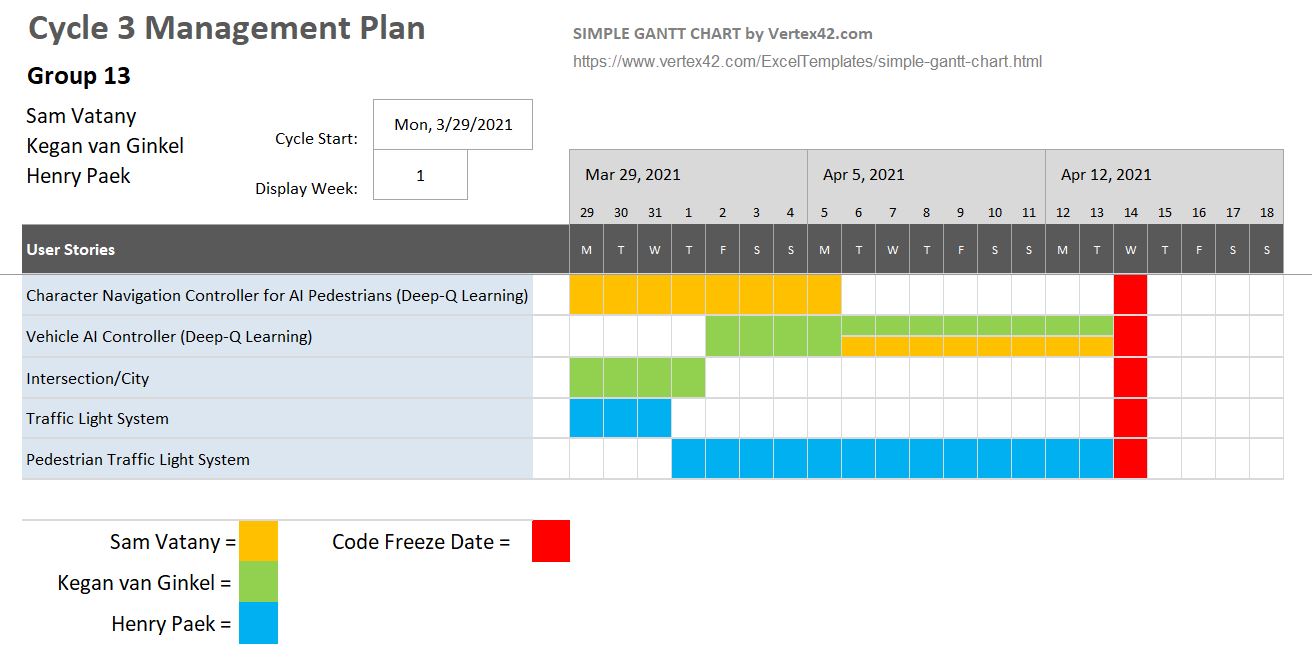
C. State Diagram for Overall Simulation



D. Visualization of Simulation in Unity



5. Management Plan



6. Lessons Learned

Before starting this project, we all had very little experience with game development and the use of tools like Unity and Blender. During these last few weeks, we have done a lot of research on how objects can be designed in Unity and how we can use scripts to allow objects to interact with the environment and the user’s inputs. We have developed a better understanding of how objects can be 3D-modeled using blender. Lastly, we have improved our skills in C# to develop better tools and to have a better understanding of how objects can be manipulated in Unity.

We have also learned that communication is a very useful tool. At first, we decided to just choose a component to work on and implement it how we see fit. We have now realized that it is important to discuss our implementation ideas, so that we can develop components that are better suited to work with each other.

7. Test Documentation

Since we are creating a simulator, we have decided that the best method of testing our software is to simply run the simulator and then check every aspect of the new component that has been added. We also implemented a strategy many game development testers use which is to try to break the game when we run it to find any holes in our software.

Appendix A

Supporting Documents

A.1 Status Reports

1st Status Report:

<https://docs.google.com/spreadsheets/d/1MPOE2_umkrF7eSnCivIDLwlZCMDBoaCLOlQXOx7wGaw/edit?usp=sharing>

2nd Status Report:

<https://docs.google.com/spreadsheets/d/1Sg3rop8thc-0OYRdHXjfHwvf43UEeSU4NtYLN80rhOs/edit?usp=sharing>

A.2 Meeting Minutes

Meeting minutes is reflected in the cycle status reports.

A.3 Size Estimation Documentation

Does not exist.

A.4 Problem Reports / Change Requests

No problem reports or change requests.

A.5 Memoranda

<https://docs.google.com/document/d/12cLe2X1ocV6AkMTfNAKoXLXMe20MdOITdo1lDxWIxYw/edit?usp=sharing>

A.6 Source Code

Source code is on github in this repository under the master branch. <https://github.com/Sam-Vatany/VirtualAutonomousVehicleSimulator.git>