|  |  |
| --- | --- |
| **ELECTRONIC ASSIGNMENT COVERSHEET** | Murdoch_land_RGB |

|  |  |
| --- | --- |
| Student Number | 34829454 |
| Surname | Sims |
| Given name | Cameron |
| Email | cameronissacsims@gmail.com |
|  |  |
| Unit Code | ICT206 |
| Unit name | ICT206 Intelligent Systems |
| Enrolment mode | Internal |
| Date | 21/08/2024 |
| Assignment number | 1 |
| Assignment name | Proposal |
| Tutor | Mahmood Golzarian |

|  |
| --- |
| **Student’s Declaration:**   * Except where indicated, the work I am submitting in this assignment is my own work and has not been submitted for assessment in another unit. * This submission complies with Murdoch University's academic integrity commitments. I am aware that information about plagiarism and associated penalties can be found at http://www.murdoch.edu.au/teach/plagiarism/. If I have any doubts or queries about this, I am further aware that I can contact my Unit Coordinator prior to submitting the assignment. * I acknowledge that the assessor of this assignment may, for the purpose of assessing this assignment:  -reproduce this assignment and provide a copy to another academic staff member; and/or   -submit a copy of this assignment to a plagiarism-checking service. This web-based service may -   -retain a copy of this work for the sole purpose of subsequent plagiarism checking, but has a   legal agreement with the University that it will not share or reproduce it in any form.   * I have retained a copy of this assignment. |
| I am aware that I am making this declaration by submitting this document electronically and by using my Murdoch ID and password it is deemed equivalent to executing this declaration with my written signature.  Signed  Cameron Sims | |

**Description of Project**

**Introduction**

Predicting how cars work in the modern day is a large industry, Google and Apple maps both have features which predict and project traffic and reports accidents as well as other types of parameters such as if a road has a speed camera.

However, most of this data is reactive using historic data and not pro-active. It doesn’t prevent traffic nor does it predict very much. This is an opportunity I would like to investigate.

My project would revolve around graph theory, which represents a road network to simulate car crashes. These car crashes would change their probability depending on certain factors, such as amount of drivers on a road, the speed of the road and how much nodes the road connects to.

This would be a “rule-based system” where certain rules dictate how the road works and interacts.

**Description of Project**

**Road Structure**

|  |  |
| --- | --- |
|  | This is an example of what the road-connection system is based around.  The white circles are “nodes” or “vertices” these are connections where the crashes occur.  The blue lines are “edges” these dictate the length and speed of the driver going across. |

For example, if we have 5 drivers on a road and if it was raining, there would be a significantly more likely probability that one of the drivers would crash. If a driver crashes, they are taken out of the simulation and will not effect the future of the program, however they will effect the crash percentage of the node. The information for roads and connections are read from their csv files located in the data directory.

**Driver**

The simulation runs on hours, drivers will try find a node to go to and will make their way over there, since roads have lengths they will go across until they reach a node. If they crash on their journey, they will not make it and they will be taken out of the simulation and recorded.

**Description of Project**

**Inputs**

The inputs of the system are mainly derived from two files “roads.csv” and “connections.csv”. The “roads” file dictates the “edges” in a graph, they connect node to node, they have at this moment in time 2 main variables (speed, length). Speed is how fast drivers drive across the road and length is how long it is. Both are not in any specified unit, so will just be referred to as *‘units’* and *‘units per second.’* The “connections” CSV is mainly to keep consistent with graph theory and provide a way to easily find what roads connect to one, by hopping from one node to the next.

Along with this, there is 2 terminal inputs at the beginning of the program, there is a driver amount, which is how many drivers you want on the road at any given point. As well as how many months you want the simulation to go on for. These both control the quality and quantity of the data obtained.

**Description of Project**

**Future / Prospective Rules**

Since this project is quite large, it might be wise to add a bunch of rules to maximise the functionality of the program and try simulate real behaviour as much as possible. Some future rules to add to the system could be:

* Driver type: Changes how likely the driver is to crash i.e.: Reckless, Cautious, etc
* Driver vehicle: Motorcycles would be more likely to crash than cars.
* Days: Sundays could have more crashes than Mondays, etc.
* If a road has too much crashes, and is not the only connection to a node, then it can be closed.
* If too much crashes occur on a road, drivers might avoid it.
* Path Finding Algorithms measuring different weights (Least crash potential, quickest way [by km/h], shortest way [by km]).

**Goal**

The goal of the project is to create a rule-based simulation that hopefully accurately represents how real drivers drive across a node network, and generates statistics which may or may not represent reality.

A sample of the project may be found on my Github:

<https://github.com/cameronsims/ICT206-Project-Demo>

**Possible AI Methods and Tools**

**Pyvis:** Pyvis is a Python Library which displays graphs based on an input, PyVis is going to be used in the project to show a visual display of how the program performed, and what deaths/incidents occurred on each road and node. This is helpful as it provides a nice visual representation and makes more sense to somebody with little experience in the program.

We need to heavily rely on graph theory for this program, so including more Python libraries might be a good investment and help simplify the program.

Having the Drivers have a bit of logic would be an excellent addition which would prevent the program from constantly randomly picking destinations, but rather having rules to which node they connect to. Having an A.I. make decisions based on some kind of data would help increase realism and get a pattern which reflects real-world driving.

Having Artificial Intelligence be integrated with the program would be a good addition to generate both the “roads.csv” and “connections.csv” file, as the C program I attached to generate these files is a bit finicky and doesn’t guarantee 100% connections between all nodes. ChatGPT can generate the C.S.V. files that are required fine, however it struggles to create anything meaningful over 16 entries.

**Evaluation Method**

We will know when we meet our goal by having a program that is a rule-based system that simulates drivers on a network semi-realistically. As the demo on my GitHub is as of 26th of August, there is too much driver fatality, as we know the crashing for major cities is far less than 80%. Toning this down for the simulation and making it more realistic would definitely be an improvement and reflect realism, which would generate a more useful simulation.

As a report by the Victorian Government showed, the number of crashes of vehicles is dramatically lower than the amount of drivers.

755 vehicle crashes (pedestrian, bicycles, etc aren’t counted) were reported over 5 years, with 236 dying on roads each year (Government of Victoria, 2024).

This is in compared to the ~85,000 drivers that are in Victoria, as we can see the crash percentage from these numbers is ~0.888% of drivers crashing (Government of Victoria, 2023).

Comparing these numbers to the demonstration of the simulation which generated a total of 401 out of 500 drivers (80%) there is an obvious realism problem. Improving this would be a sign of completion and would cause a far more realistic simulation.

**Evaluation Method**

However, this approach we have set is not without its own problems:

* Due to the way that references work in Python, connecting nodes can be quite costly.
* Hard to implement actual roads and road data due to unique design of the system.
* If we have actual data, it will be very costly to transfer the information from one format to another format.
* Path finding algorithms, if done often and constantly can slow down and impact the performance of our simulation.
* Our simulation has no guarantee of being accurate or performing well, this will require understanding and testing.
* Adding too much features may make the program too slow and unfunctional.

All these problems need to be addressed and optimised so we can maximise the output with reducing the errors.

If the simulation doesn’t reflect reality or is not meaningful it doesn’t really have a purpose to exist. This means that we must try reflect reality and give a report which can hopefully reflect real world activity.

**Time Schedule**

|  |  |
| --- | --- |
| Simple Demonstration | 3 weeks – (COMPLETED) |
| Adding Complex Path Finding | 2 days |
| Adding Non-Static Crash Percentages | 2 day |
| Complex Driver AI (Avoiding “bad” roads) | 1 week |
| Generating a .CSV report from each generation | 1 day |
| Refactoring and improving realism of simulation (editing and modifying values to reduce deaths) | 2 days |
| Performance/Optimisation/Realism monitoring | 1-2 days |
| Final changes + additions | 1-5 days |
| Ironing Out and improving performance | 1 week |

**References**

Government of Victoria. (2023, November 15). *Reforming Commercial Passenger Services.* Retrieved from www.vic.gov.au: https://www.vic.gov.au/reforming-commercial-passenger-services

Government of Victoria. (2024, Feburary 28). *Road User Statistics*. Retrieved from Transport Accident Commision: https://www.tac.vic.gov.au/road-safety/statistics/summaries/road-user-statistics

The submission will be marked using the following rubric:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CATEGORY | **4** | **3** | **2** | **1** | **0** |
| **On Time/Late**  **Without Extension** | On time  0 marks | 1-2 days late  -1 marks | 3-4 days late  -2 marks | * 1. days late   -3 marks | 7 or more days late  -4 marks |
| **Presentation**  **(coversheet, diagrams, captions, references in IEEE format)** | Very good  +4 marks | Good  +3 marks | Acceptable  +2 marks | Poor  +1 mark | Very Poor  0 marks |
| **Description**  **(problem domain, inputs/outputs clear, clear measurable goal)** | Very good  +4 marks | Good  +3 marks | Acceptable  +2 marks | Poor  +1 marks | Very Poor  0 marks |
| **Methods & Tools**  **(what knowledge needed, where it will come from, what tools will be used)** | Very Good  +4 marks | Good  +3 marks | Acceptable  +2 marks | Poor  +1 mark | Very Poor  0 marks |
| **Proposed Evaluation**  **(how the solution should be evaluated, what problems could complicate this)** | Very Good  +4 marks | Good  +3 marks | Acceptable  +2 marks | Poor  +1 mark | Very Poor  0 marks |
| **Writing quality**  **(clarity, style and grammar)** | Very Good  +4 marks | Good  +3 marks | Acceptable  +2 marks | Poor  +1 mark | Very poor  0 marks |

**Tutor’s Comments**