Project plan

Martin Blom   Felix Jönsson   Hannes Kaulio   Marcus Schagerberg
Jakob Windt

2023-01-23

Bachelors Thesis
Department of Computer Science and Engineering
Chalmers University of Technology

Contents

Glossary

Agent: Autonomous systems that inhabits an environment and act based on predefined rules.

Agent Based Model (ABM): A computer simulation model in which agents interact with each other and their environment to produce emergent behavior.

Data Structure: A way of organizing and storing data in a computer so that i can be accessed, manipulated, and modified efficiently. Some common examples are arrays, stacks, and linked lists.

Information Visualization: Field that focuses on creating meaningful and easy to to interpret graphical representations of data.

MonoBehaviour: Base class for Unity scripts. Provides access to event functions such as Start(), Update(), and so on.

Pooling: A technique used in programming to improve performance by reusing objects instead of creating new ones.

Prefab: A reusable object in Unity that stores a configuration and can be used as a template for creating assets.

Scrum: The scrum agile project management framework provides structure and management of work and is popular among software development teams.

Unity: Cross-platform game engine.

Unity Asset: A file containing reusable content that can be imported into Unity projects. Can be accessed through Unity's official platform or imported via third-party repositories.

User Testing: A method of testing and evaluating a product by observing and gathering data from real users.

UX: User Experience (UX) refers to the overall experience of the actual user of a product. The goal of good UX design is to create intuitive and enjoyable products.

1 Introduction

Traffic congestion is the result of the demand for road/ -and railway travel exceeding the supply. This problem can be seen all around the world[1] and it impacts our quality of life. As vehicular traffic builds up other vehicles like bikes, cars, buses and trams can move significantly slower resulting in a higher increase of time and fuel wastage. Moreover, delaying transportation of goods can lead to an increase in economical costs, food waste and general inconvenience for the effected parties.

Not only does this affect our society on the larger scale, but also on an individual scale. If the average commuting time does not go down, the average citizen will spend around a year of their life commuting. According to Trafikanalys data of traffic habits[2], the average Swedish citizen's daily commuting time during 2019 was just under 1 hour and dropped to around 45 minutes post-COVID. Some amount of commuting time is inevitable in our current society. But if you look at some of the bigger cities in the world such as London, an estimate of 156 hours per person was lost in just traffic delay alone during 2022[1].

Other than leading to loss of time and resources, congestion and traffic in general leads to air pollution which poses health hazards and also lowers the quality of life[3]. A study of air pollution in connection to cars made in the USA during 2022, shows that transportation stood for 27% of the total greenhouse gas emissions[4]. By reducing the amount of combustion sources that contribute to air pollution we can work on solving multiple problems at once.

In section 2 we will present how we aim to create a tool to help with both understanding why, and solving societal, economical and resource problems related to traffic and congestion.

2 Purpose

The purpose of this project is to create a traffic simulation tool that will allow the user to design a road network, and see real-time statistics about its traffic flow and environmental impact. In addition, the user should be able to manipulate different aspects of the simulation such as the amount of cars.

By changing some aspects of the simulation, the user will be able to see whether their design improves the road network. These improvements can result in less traffic congestion and a smaller environmental impact.

3 Problem

In order to produce a traffic simulation that clearly and visually shows environmental impact as well as any resulting traffic congestion, several problems will need to be solved.

There are multiple ways to simulate a traffic flow, however based on the purpose and the established limitations of our project we have chosen to implement an agent-based model[5] (ABM). This would offer a fitting high level of granularity of interplay between the environment and the individual agents.

To start of, the environment to which all agents respond and interact within will have to be specified and created. Static environmental objects such as roads and corresponding traffic signs will have to be generated in a modular manner to allow the end user to set up and run their customised simulation. Furthermore, all individual traffic elements such as cars, buses, traffic lights, and roads will need to be simulated. In order to accomplish this, an ABM will be used as mentioned previously. All these traffic elements will be simulated as either individual agents abiding to a set of rules, or environmental objects that relay information to the agents. The challenge will be to decide the different rules and logic for the various agents. Vehicle agents will, for example, need to interact with the different traffic signs and follow the specified road rules.

There also exists implicit and more subtle rules that govern movement in public spaces, such as the observed phenomenon of personal space radius being dependent upon the velocity of a person in western cultures, achieving its maximum size when the person is stationary[6]. Similar behavior can be observed in traffic where interactions are not only governed by the laid out traffic rules, but psychological and social factors as well[7]. A challenge therefore lays in sufficiently integrating these variables into the simulation to allow for a realistic depiction of the real life scenario it is trying to emulate.

As a result of implementing an ABM, performance will run a high risk of becoming an issue. All agents in the simulation will have to be continuously updated according to their rules. This can become computationally expensive when the number of agents increases. Performance-based design choices will have to be made for city-scale simulations to be possible such as exploring the options for pooling or constructing customized data structures.

Other issues that will have to be resolved are how to process the order of execution both from a low level perspective such as order of execution of functions within an agent or environment object. This includes a higher perspective such as order priority amongst different entities in the simulation. Failing to do so might otherwise result in subtle faulty behavior that can be hard to track down and fix. Unity has some native support for this in the form of different update events with varying priorities. This enables Unity objects, i.e. classes inheriting from Unity's base class MonoBehaviour, to customize their order of execution. However, it is not clear at the moment whether this will be enough to ensure an easy configuration of execution priority or offer the necessary transparency. Decisions will have to be made if we should implement our own ABM framework or make use of existing unity asset such as the open-source Agent-Based Modelling Framework for Unity3D[8].

The goal of our project is to offer a user-friendly tool which would require a user interface for the end-user to interact with. This interface needs to clearly communicate which parameters the user can tweak in the simulation and offer an intuitive design to convey this. Appropriate User Experience design patterns will have to be reviewed and implemented.

Finally, the interface will also need to display relevant information with regards to statistics of the simulation in a manner that is meaningful and easy to comprehend. To achieve this, we will have to research and make use of concepts from Information Visualization to display relevant information in a comprehensive manner. This will involve using techniques such as data visualization, charting, and graphing to display the statistics of the simulation in a way that is easy to understand and visually appealing.

4 Limitations

To limit the scope of the simulation tool, it is decided that we should set boundaries on what the tool should and should not include.

To begin with, the tool will only simulate vehicles such as cars and buses since the inclusion of pedestrians would add unnecessary complexity to the simulation. This is because pedestrians have too small an impact on traffic congestion. In addition, having too many movable objects would impact the overall performance of the tool with both vehicles and pedestrians.

Furthermore, a discussion about whether the tool would display the map in 2D or 3D was necessary. Creating the tool in 2D would take less time to complete, since the overall complexity of the project would decrease. However, showing the map in only two dimensions would lead to bridges, tunnels, and highway exits and entrances being harder to interact with when compared with 3D. Also, since the tool is meant to be used by a third party there is a need to make it appealing. On this basis, the group made the decision to create the tool in 3D. It allows for more precision when designing road networks to the user, while also appearing more intuitive.

Lastly, the decision was made to only include a few models of cars with different pollution amounts. These amounts will be similar to Sweden's average emissions by car. It will not be exact because the data will be sourced from the user. The reason behind this is that the average emission depends on the city's exact location, as well as when the data was collected due to its continuous changes.

5 Method

The simulation tool will be developed in Unity, a cross-platform game engine. The workflow we aim to implement is comprised of several distinct areas, each of which plays a crucial role in the overall development. Below follows a short description of each area and the method we have chosen to implement with an accompanying motivation. These are 1) Game Engine and Libraries, 2) Modeling, 3) Testing, and finally 4) Organisation. The chapter will then conclude with stating the criteria by which we will judge the success of the project.

5.1 Game Engine and Libraries

The simulation tool will be developed in Unity, a cross-platform game engine. Unity is a cross-platform game engine and development environment which comes with a plethora of useful tools for developing software that communicate with a 2D or 3D environment. It is currently the most used game engine, holding a market share of 38% within the game engine market[9]. From initially being marketed as a highly accessible game engine, it has grown into a versatile tool that can be used for a wide range of applications. These include animation, simulation, VR, architectural visualisation and more. With the above in mind, together with the massive amount of high quality documentation available and high number of asset plugins available, we motivate Unity as the choice of engine to actualise the simulation.

5.2 Modeling

We motivate our choice of an agent based approach by its ability to capture and convey complex behaviour that would otherwise have required highly complex mathematics. After we have configured the simulation environment, we only require the ability to decide and manipulate how the agents react to the it. This will be enough to allow us to capture an interesting emergent system that would otherwise require the use of complex differential equations if it were modeled from a macroscopic perspective.

Our goal is to make a product that can easily convey both the statistical information of the simulation as well as the first-hand experience of how traffic actually behaves. This will be in the form of exploring the environment in real-time 3D. This further motivates our decision to go with an ABM approach. While ABM is classified as a microscopic model, it still allows for gathering data from each autonomous agent and aggregating it in the form of statistics. On the other hand, by rendering the simulation and each agent in real-time, it will allow the user to see the effects of the parameters they supplied for the environment. For example, this could be in the form of observing a crossing, or following one specific agent throughout its journey towards some destination.

ABM also offers a high degree of flexibility where the model makes it easy to construct and inject new agents with different behaviors, heuristics, and so on. This will be important for both the development team and the final end user experience. The property will allow us to iteratively adapt the complexity of our simulation as the development process proceeds, and be easily offered to our users as a way of tweaking different behaviors. The ability to easily adjust the complexity of the model is highly valued by the development team as well, because it allows us to adapt and make changes as needed during the course of the project. It is particularly important to note that the team has not previously worked on a project of this nature, and thus, flexibility serves as a safety net for any potential shortcomings in the product planning.

5.3 Version Control

In order to collaborate, the project will be stored in a Git repository. This also allows for version control and the opportunity to revert to previous versions in the case of identification of errors. Since a Unity project does not consist of pure text or code files, additional steps have to be taken in order to avoid merge conflicts or other issues that can arise when collaborating in a version control system. Therefore each developer will have their own scene in Unity and updates to the project will mainly take place as changes to prefabs, which are reusable composite objects that can be used as templates.

5.4 Testing

Due to the challenging nature of validation and testing of agent-based simulations, our main method of making sure our software is working correctly will be in the form of in-house user testing. A set amount of hours will be allocated each week once testing becomes relevant, where developers will explore and observe the simulation under different parameters. The goal of this will be to identify any faulty system behaviour or usability issues, and also gather valuable feedback on the overall design and functionality of the simulation.

5.5 Organisation

In order to have a clear organizational structure we decided to employ an interpretation of the scrum agile framework [10]. In scrum, the work is split into weekly partitions called sprints. Before each sprint there is a planning meeting in which you plan the upcoming sprint - what to do and who should do it. After the sprint, a sprint review is held to reflect about whether all goals were met. The sprints should be relatively short as the scrum framework is based around the idea of creating software through iterations. This means new iterations should be produced at a regular pace. We have decided on one week long sprints, with a weekly demo meeting acting as our sprint review in which all members demonstrate their completed work during the past sprint. This also serves an additional purpose - sharing the progress between all members so everyone is on board with how far the software as a whole has evolved and opening up possibilities for input and discussion.

After the demo meeting, sprint planning will occur every week in order to assign tasks to members and plan the goals for the upcoming week. Each weekly sprint planning originates from a goal set for a longer time period which describes the next iteration of the software. The next iteration is collectively decided upon as a specification of requirements during a planning meeting. The planning meeting is centered around the monthly goals set in 6 Time plan. We have also assigned a project leader responsible for taking this next iteration and splitting it up into tasks to be completed during the sprints, making sure there is a logical order of completion for the tasks.

The aim of combining the longer time frame iteration plan with the short term sprint goals is to develop a fast-paced iterative software development environment. This enables new functioning versions of the software to be released at a regular interval. This makes sure the software is continuously developed in the right direction and splits the work into small manageable parts with a clear progression between them.

5.6 Criteria of success

The success of the project will be analyzed in three dimensions: functionality, user-friendliness, and computation capacity.

Functionality will be judged on how many meaningful and correctly functioning features our final simulation offers. The features need to be meaningful in the way of allowing the user to observe and gain insights of how different traffic flow set-ups produce interesting results depending on the supplied simulation parameters.

User-Friendliness will be analyzed on how easy it is for the user to understand the interface and interpret the produced statistics. The user should be able to set up simulations with ease and be able to easily extract insights from the results. Out of house user testing might be necessary for this, though this is currently not in the project plan.

Computation capacity will measure how well the simulation can handle complex traffic scenarios and its ability to function under high computational workload. Different kind of stress tests will have to be constructed and carried out to make sure that the simulation is capable of handling a sufficient amount of agents to represent realistic scenarios.

6 Time plan

A time plan was created in order to structure the project and plan the different parts in time, see appendix A Time plan. It is an aid during the sprint plannings as well as when defining the requirements for the next version of the software. The time plan is split into two different sections, administration and product, indicated by the colours. Each task in the time plan has a duration under which we plan on finishing each task, as well as the stated deadline.

The administrative tasks in blue have definitive deadlines, whereas the green product tasks have deadlines based on when we estimate them to be completed. The product tasks are split into the different phases, where the goal at the end of each phase are the software iterations with increasing capability. In order to explain the product tasks, a brief description of each one is presented below.

6.1 Autonomous vehicles

The goal of this task is to create a working vehicle model that can be controlled by the user as well as through code. This will allow the vehicles to be controlled autonomously, enabling the simulation of vehicles traversing roads in our model.

6.2 Navigation

A navigation system is required to direct the vehicles around the road network, allowing them to travel between different places on the map. As it is possible to create a graph representation of the road network, the navigation system can be implemented as a graph search algorithm.

6.3 UI

To control the simulation a user interface needs to be designed. It should allow the user to control the parameters of the simulation, such as simulation speed and vehicle counts. It should also give the user additional controls, including camera switching and providing context menus with information for example when selecting a car.

6.4 Statistics

As part of the UI the statistics related to the simulation needs to be displayed. The statistics should allow for different aggregates, for example displaying the average speed of the entire road network, as well as for a single road or even a single vehicle.

6.5 Map editor

In order to create the map to be used in the simulation a map editor needs to be created. This will also allow the user to update the map, examining how different road networks affect the traffic flows.

6.6 Demo city

A city used for demonstrating the product needs to be created. This should incorporate the different features such as intersections, public transport routes and identification of bottlenecks in the road network.

6.7 Public transport

Travelling by public transport instead of with your own vehicle reduces both the emissions as well as the traffic strain. Finding a good balance between personal vehicles and public transport is difficult but this task aims to aid with that, by also simulating public transport. The product should allow the user to choose the how many people travel by public transport, and seeing how it affects the simulation.

7 Relevant societal and ethical aspects

Ethical aspects can be broken down into two parts, 1) aspects related to the method of the project, and 2) possible consequences for users of the final product and society as a whole. There are no obvious ethical issues related to the method of our project. For example, there is no use of personal data, no significant security risks, and no use of questionable technology or software. If we choose to introduce any user-testing outside of the development team this will have to be reconsidered and proper implementation of data privacy will be necessary.

The ethical aspects of the finished product however, is accompanied by more careful considerations. One of the goals of this project is to create a tool that can be used by different end-users of various occupation connected to traffic planning, and offer these users insight about the efficiency and emission associated with different set up of road networks. Since these insights might lead to real-life decisions regarding actual infrastructure, careful consideration will have to be taken regarding the design we choose to implement and what sort of consequences these might have in the finished product. To instill model credibility and prevent model realism bias, we will have to communicate any assumptions and limitation of the model in an easy and accessible manner. ABMs are generally considered challenging with regards to validation and traceability[11], and failing to mitigate these might lead to decisions being implemented on obscure premises.

References

[1] Inrix, "Inrix 2022 global traffic scorecard," https://inrix.com/scorecard/, accessed: 2023-01-30.

[2] "Resvanor," https://www.trafa.se/kommunikationsvanor/RVU-Sverige/, May 2022, accessed: 2023-01-30.

[3] A. J. Cohen, H. R. Anderson, B. Ostro, K. D. Pandey, M. Krzyzanowski, N. Künzli, K. Gutschmidt, C. A. Pope III, I. Romieu, J. M. Samet *et al.*, "Urban air pollution," *Comparative quantification of health risks: global and regional burden of disease attributable to selected major risk factors*, vol. 2, pp. 1353–1433, 2004.

[4] K. Gallagher, "How much air pollution comes from cars?" https://www.treehugger.com/cars-are-causing-air-pollution-we-breathe-new-study-finds-4856825, Aug 2022, accessed: 2023-01-30.

[5] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," https://www.pnas.org/doi/10.1073/pnas.082080899, accessed: 2023-01-24.

[6] K. Cheliotis, "Agent-based modelling of public space activity in real-time," Ph.D. dissertation, UCL (University College London), 2019.

[7] W. Wang, L. Wang, C. Zhang, C. Liu, L. Sun *et al.*, "Social interactions for autonomous driving: A review and perspectives," *Foundations and Trends® in Robotics*, vol. 10, no. 3-4, pp. 198–376, 2022.

[8] K. Cheliotis, "Abmu: An agent-based modelling framework for unity3d," *SoftwareX*, vol. 15, p. 100771, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352711021000881

[9] "Did you know that 60% of game developers use game engines?" Dec 2022. [Online]. Available: https://www.slashdata.co/blog/did-you-know-that-60-of-game-developers-use-game-engines

[10] C. Drumond, "A guide to scrum: what it is and how it works," https://www.atlassian.com/agile/scrum, accessed: 2023-01-30.

[11] M. Mora-Cantallops, S. Sánchez-Alonso, E. García-Barriocanal, and M.-A. Sicilia, "Traceability for trustworthy ai: A review of models and tools," *Big Data and Cognitive Computing*, vol. 5, no. 2, 2021. [Online]. Available: https://www.mdpi.com/2504-2289/5/2/20

Appendix

A Time plan