Boids On Wheels

A Proof of Concept Study of the Boid as a Vehicle

Aastha Kakaria

Master's by Research

University of York

Computer Science

September 2016

Abstract

The following project is a proof of concept study exploring the feasibility of simulating traffic as a multi-agent system, with the individual vehicle being implemented as a boid as defined by C. Reynolds in 1987. Furthermore, the simulator is to serve as a tool for urban planning.

This thesis first explores the growth in using computers to simulate traffic since the 1940s, along with a brief review of the work done on Reynolds' boids. This is followed by a discussion of tool selection and the reasons behind it.

Secondly the thesis discusses the different steering behaviours developed for a boid vehicle, as well as their implementation. This is followed by a description of the preliminary evaluations carried out and the problems encountered.

Finally the thesis concludes with some ideas for further research and the conclusion that within the scope of this project, implementing the boid as a vehicle was found to be feasible, and ripe for further work.

Contents

List of Figures

Acknowledgements

I would like to express my deepest appreciation to my supervisor Dr Polack for her never ending patience and knowledge, without which this project would never have left the garage.

Author's Declaration

Chapter 1: Introduction

1.1 About the Road

The need to accurately forecast future traffic flow has been an integral part of urban planning ever since the first proto agrarian communities first settled down. The one unifying factor between the first civilizations of India, Rome and Egypt are the long straight roads punctured by sharp right angles for the easy passage of not just the general population but armies, as well as providing little room for invaders to barricade themselves.

Two examples of the importance that future movements of traffic are the founding of Singapore and reconstruction of London after the great fire of 1666. The great fire of 1666 happened in the shadow of the execution of Charles the first and the English civil war. As such the authorities were naturally reluctant to rebuild the city in the likeness of other European capitals like the Italian city states with their sweeping plazas that invited disgruntled citizens to gather, protest and possibly rise up and overthrow unsatisfactory ruler. As such, to inhibit too much easy movement of citizens the city was rebuilt along the old lines preserving the narrow streets and cramped squares. Whether this bottlenecking of the populace is responsible for the lack of further civil unrest and the resulting rise in nationalism was the cause is not clear but the anecdote does illustrate the weight that road planning holds, as London held the dubious honour of being Europe's most congested city in August 2015[1].

Another illustrative example would be the founding of Singapore in 1819. Despite great attention being paid to the initial selection of the site of the city no further thought was given to urban planning despite it being meant as an important naval base. The city's population exploded from 150 to 185,000 in less than a hundred years. Roads originally meant for foot traffic was choked by automobiles. The ensuing congestion and overcrowding could not be adequately addressed until the 1970s [2] Singapore may be taken as a microcosm of a problem familiar across the world, as populations grow so do the number of registered vehicles. Fig 1 shows the increase in registered vehicles between 1994 and 2014[3]. It should be noted that this increase is occurring against a backdrop of relatively unchanging road systems that require expensive maintenance to keep up with the demand, and said renovations often fail to allow for future increases.

Before the 19th century, road systems were designed to accommodate crowds or mobs, with vehicles being restricted to carts and carriages. As such any predictions city planners of old may have made about future traffic flow would have been restricted to crowd movements. Roads today accommodate much more traffic and of a somewhat different variety. While newer towns and cities may be built with better foresight, older cities that have endured through the centuries require their old, narrow roads to be constantly rebuilt and opened.

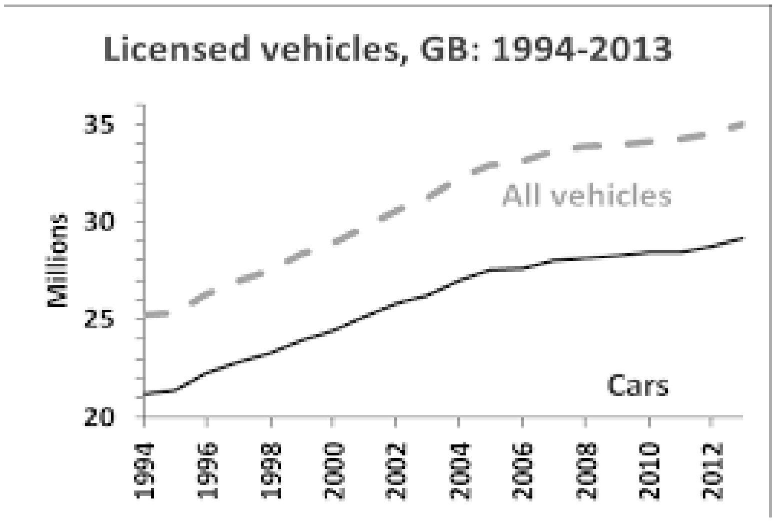


Figure 1 Increase in Registered Vehicles in Great Britain 1994-2004[3]

London evolved from a Roman city (43AD), and covered no more than one square mile for its first 17 centuries. The subsequent growth to over 600 square miles has also seen the growth of public transport, motorised transport, railways, and commuting. [4] Even with newer cities, it is more often the case that developers build with the required capacity of today in mind rather than the needs of the populace in a few years' time. This combined with the fact that cities by their very nature tend to be attractive to commuting workers and it is almost inevitable that the roads will end up with much more load than they were originally designed for.

There is then a need for a simple and efficient way to simulate traffic that is easily malleable so that one can observe the effects of planned changes and new development before actually carrying it out. That is, a simulation in which one may make the desired changes and observe the reactions of drivers with minimal cost and where the changes can be reverted with ease.

1.2 About the Question

At its simplest, the flow of traffic may be defined as a crowd with a common goal. While it is true that each driver is steering a vehicle, at the end of the day, they are still a mass of individuals, in a restricted space, with a similar aim, not unlike a flock of migrating geese or a school of fish.

A crowd is defined as a collection of people with a shared experience. [5] Traffic can be a specific form of crowd, since a road defines a gathering of drivers with the shared experience of navigating the road on the way to an individual destination.

One of the simplest representations of crowd dynamics is the boid. [6] Each individual member of the flock is represented as a boid with some basic steering mechanism: each member seeks to follow the general direction of the flock, while avoiding collisions or straying too far from its flock mates. In this research, it is assumed that the behaviours of boids can model the behaviour of a vehicle driver: the vehicles travelling along a carriageway conform to the general direction of the traffic without veering off course; drivers try to keep up with other vehicles; and drivers try to avoid getting too close to avoid collisions.

The purpose of this project is two-fold: firstly, to conduct a feasibility study on the usefulness of a simple rule-based agent system for modelling complex traffic systems. Due to the similarities between the average driver and a boid as defined by Reynolds in 1987 [6] as discussed above, the decision was made to conduct a feasibility study on whether the boid could be implemented as a vehicle in a manner that would serve to create a tool for a realistic simulation of traffic.

Reynolds himself developed a basic model of the boid as a vehicle in 1999, along with some steering behaviours, which are discussed in chapter 4. Correctly implementing them to achieve the right balance such that the emergent behaviour of the vehicular flock resembles traffic flow in the real world required a lot of finessing work.

Reynolds' Boid displays three behaviours: collision aversion, velocity matching, and flock cohesion. Of the three, collision aversion is the behaviour that holds the most importance for a vehicle and so the decision was made to focus on developing the best implementation of it. This was done due to the small scope of this project.

To avoid distractions from other behaviours, the decision was made to develop the boid with the following restrictions were placed on the model:

- The simulation is restricted to one way street
- Each vehicle is assumed to be generally lawful

The second purpose is to begin constructing a traffic simulator that serves as an easy to use tool for urban planning that allows developers to easily observe the effects of proposed changes on the flow of traffic.

Such a tool should ideally have the following features:

- Easy to add and remove different city modules such as buildings, crossings, lanes, traffic lights etc.
- Easy to add and remove non-vehicular agents such as pedestrians, stray animals, and other animated obstacles
- Easy to add different road modules such as lanes, roundabouts, link roads etc.
- Separation between the boid agents and the city/road modules so that the changes in one do not affect the other.
- The city/road modules should not involve complex coding as they are meant to be handled by users who are probably not experienced in coding.

The main aim of this thesis is to show that Reynold's Steering Mechanism [6], originally intended for a three-dimensional model of flocking behaviour, can be sufficiently modified to represent a two-dimensional flow of traffic. That is, to show that, with the steering behaviour coded into the boids can be appropriately adapted to represent, vehicles as agents, avoiding colliding while managing to maintain a steady path. Its secondary goal is to create a simulator that allows both the road structure and the vehicles to be modified independently of each other, thus creating a tool for urban planning that allows builders to correctly estimate the effect of a given road design on the flow of traffic.

The project aims to observe and evaluate the behaviour of vehicle agents coded to act like boids in a restricted environment and to evaluate the boids inspired crowd centric approach to simulating traffic. This research represents a novel use of boid-like agents. Furthermore, there is no target dataset against which to test simulation results. Instead, evaluation focuses on the robustness of the basic boid model and documents issues that arise.

To evaluate the feasibility of boid-like vehicle simulation, the simulator will be configured to represent a single-carriageway road. Each behaviour will be challenged by introducing appropriate numbers of vehicles. Thus, the vehicle's behaviour as a boid will be tested first singly, and then in small numbers. Due to the limited scope of the feasibility project, proper evaluation of the first two points in the above bullet points will be reserved for future development. However, the next two points will be covered by choosing the appropriate software as outlined in chapter 3.

1.3 Research Questions

How do boids "see" each other?
Deciding on the visibility range of the boid to resemble a real vehicle most closely is the first challenge, with said range changing as vehicles overtake or fall behind each other.

How does one accurately balance the different risk factors in the steering mechanism? In the wild, flocks have more room to evade risks, operating in a three-dimensional space, as well as the freedom to veer wildly off-course and away from the flock to avoid a collision, whereas a vehicle on the road has a much more restricted space for navigation (it cannot veer off the road in most cases) as well as being restricted to a two-dimensional frame.

Moreover, the obstacles facing each individual vehicle are not identical: it is much more fatal to crash into another car than it is to veer onto the grass for instance, but even this becomes complicated if there are other objects on the grass. Balancing different priorities is essential for the steering mechanism and while this project confines the dilemma to the boid needing to choose between crashing into another vehicle or veering onto the grass, possible mechanisms for navigating more complex decisions are discussed in the chapter on prospects.

1.4 Motivation

As shall be seen in chapter two, traffic as a crowd is a new and exciting approach to traffic simulation where the scope for research is vast, as simple agent modelling is not a method widely used in simulating traffic. By creating a small-scale representation of a boid based traffic system, this project lays the groundwork for a much larger project to model and simulate a city's road system using boid-like agents.

Chapter 2: Background

In this chapter, a brief introduction is given to some common concepts in traffic simulation to make the following literature review easier to understand, followed by an overview of the growth of computer based traffic simulations, and a discussion of the different implementations of Reynolds' boids.

2.1 Basic Concepts

Traffic simulation is the mathematical modelling of transportation systems (e.g., freeway junctions, arterial routes, roundabouts, downtown grid systems, etc.) through the application of computer software to better help plan, design and operate transportation systems. Simulation is important as it allows for the studying of models too complex for analytical or numerical methods, as well as study relationships that would be lost in mathematical development, along with providing an easy to understand visual representation.

Traffic simulations can be divided into three broad categories:

- Macroscopic: deals with aggregated characteristics of transportation elements, such as aggregated traffic flow dynamics and zonal-level travel demand analysis
- Mesoscopic: analysis is of transport elements in small groups, within which elements are homogeneous
- Microscopic: studies individual elements of the traffic system, such as driver or vehicular behaviour.

This project is a microscopic simulation. An important component of microscopic traffic simulations is the car following model, a class of scientific models whose dynamic variables represent microscopic properties like position and velocity of single vehicle units. [7]. While there have been a variety of car following models since 1958 [8], the one that inspired the mechanisms in this project is Newell's Car Following method. [9]. The main idea behind this model is that a driver will seek to maintain a minimum distance and time gap between itself and the vehicle that precedes it. Therefore if the velocity of the preceding vehicle changes, the following vehicle will change accordingly. This trait is like the behaviour of a boid, which changes its velocity based on its neighbours as well. It is well suited for modelling more congested (i.e. urban) settings and is thus ideal for developing a tool for simulating city road structures.

2.2. A Brief History of Traffic Simulation

The development of traffic simulation models closely parallels the growth of multiframe and highly powerful computers. In the early 1950s digital computers was slow, in low supply, with poor memory, and no high-level languages or operating systems. The environment was hardly ideal for the development of simulation software. In this, traffic simulation parallels the rise of agent based simulation, as computers grew smaller and more powerful, as well as more easily available, computer scientists were able to come up with more and better ways to tackle problems.

Edward B. Lieberman [10] recounts the following anecdote in his Brief History of Traffic Simulation as an illustration of the difficulties a plucky programmer would have faced:

“To illustrate this environment, I relate the experience (as told to me) of Jim Kell during his final year as a graduate student at UC Berkeley. He chose as his Master’s thesis the development of a simulation model to analyse traffic flow at two intersections controlled with STOP signs. He learned how to program the IBM 701 computer which was available, and collected and analysed field data to design and calibrate his model. Finally, near the end of the school year, he reserved time on the computer to generate the results. As that day dawned, he entered the Computer Lab and was greeted by an empty space where the computer had been. Upon inquiry, he was told that the computer was just shipped out that morning and would be replaced by an IBM 704 computer. Knowing that the two computers were not compatible, Jim rushed to the loading dock to find the 701 on the truck about to be driven away. Fortunately, Jim was a large man who could persuade the dock workers to move the computer off the truck and back into the Lab, whereupon he was able to complete his thesis.” [11] as quoted in [10]

Perhaps one of the earliest traffic simulations was produced by Harry H. Goode in conjunction with C.H. Polimer and J.B. Wright in 1956. The simulation was used to predict the traffic delays at a road intersection based on a turn factor, how long a traffic light remains green, and the number of cars [12]. Modelled on an IBM 704, while it was in theory meant to aid traffic engineers to correctly predict the flow of traffic at a series of intersections (the simulator worked for up to four), Goode was unable to compare the simulation with real traffic flow and thus the model was not tested in the real world [13].

Due to the limitations associated with computer based simulation, advancements were mostly confined to the theoretical. One of the most significant in the field of traffic simulation was Wardrop's "equilibrium" laws [14], related to, but developed independently of, the Nash equilibrium in game theory. [15] These laws state firstly that all users choose to reduce their travel time, and that user equilibrium is reached when none may lower their travel time any further. This led to a simplification of the complex mathematical models then in use and is still used in simulating networks today [16].

Further developments in the 1950s included advancements in fluid flow analogies which form the basis for many macroscopic models, [17] [18]. car following techniques, which influences several microscopic models [8].

The end of the 1960s bought the next significant development, the TRANSYT signal optimisation [19] [20], developed by the Transport Research Laboratory in the United Kingdom to model junctions and traffic signals and still in use today. The traffic flow model was in the form of a cyclic macroscopic simulation model and embedded as a component of a signal timing iterative procedure, rather than as a stand-alone evaluation tool for evaluating the inclusion and utility of traffic signals. It is an early example of an integrated simulation model.

The 1980s were when the greater availability of high-performance computers, and the standardisation of programming practices, as well as the development of high level programming languages like C, led to an environment better suited to more complex simulations.

Another boost to the popularity of traffic simulation came in the form of portable simulation software like NETSIM which was ported to PCs by the Federal Highway Administration (FHWA) [21]. The FHWA also sponsored the development of animated software for simulating vehicles on PCs [22].

Some researchers modelling the flow of traffic through busy intersections use Gawron's queue model as exemplified by Cetin et al [23]. They propose a model for assigning vehicles to link roads based on Gawron's queue model [24]. Their simulation is a mesoscopic simulation and utilises parallel computing to speed up dynamically allocating routes to vehicles based on how congested conditions are. Apart from some issues in spillback, that is, there was some difficulty with correctly simulating vehicles being pushed into busier intersections because congestion prevents access to less busy ones. this approach has been successful in accurate simulation. [25]

Developments in the last two decades in both affordable hardware and penetration of personal computers have led to an explosion in agent based approaches to simulating traffic, with several different approaches now being utilised as discussed below. Several of the problems in managing traffic can be addressed with practices common in multi agent system programming, such as optimal resource allocation and most of the actors in a traffic system can be easily "agentified" [26]. That is, they can be represented as an independent entity, with a goal and certain guiding principles and available actions that help it achieve that goal. Previous decades focused more on traffic as a fluid, macroscopic simulations, and mathematical models. The difficulty with representing traffic as a flow is that it removes the role of the driver. Perhaps that is why there has been a tendency to move away from "flow" based macroscopic simulations and towards driver based microsimulations focusing on individual agents acting in a group.

Dia [27] introduced a representation of driver response to real time information and the effect it has on trip time, number of trips and other aspects of a traffic system. An Intelligent Agent was created based on the Belief-Desire Interface(BDI) framework. Derived from philosophical roots, agents using the BDI framework have a set of beliefs/desires, goals and set plans to achieve them. Each agent constantly re-evaluates the current situation to choose the best way to attain its goal

[27]. Dia uses this agent to model dynamic driver behaviour on a congested commuting corridor in Brisbane and the results have been encouraging on a small scale.

In another example in which driver behaviour is varied, Paruchuri, et al. [28] present a multi-agent simulation of unorganised traffic where the emergent behaviour is a result of drivers with differing temperaments interacting with each other and their environment. Although the environment is rudimentary and the agent behaviour basic, it is an example of using the emergent behaviour of a group of agents to simulate traffic.

Similarly, [29] use the emergent behaviour of multi-agent systems to realistically simulate traffic at road junctions. The simulation models the opportunistic individual behaviours of drivers which can violate the norm, the simulation also allows variation in the he anticipatory individual abilities of simulated drivers, allowing critical situations to be detected. The authors validate their model against a real intersection.

2.3 Boids and Traffic Simulation

The previous section discussed the growth of traffic simulation in over the last seven decades, as well as some agent based traffic simulations inspired by nature. However, flocking has not yet been used as a model for traffic although it has been adapted to model pedestrian behaviours such as the multi-layer boid developed by Fasheng Qiu that models pedestrians as a flock of flocks, with boids in different flocks tending to avoid each other and stay with their own. [30]. A representation of traffic using flocking needs to model the behaviour of individual vehicles (drivers) such that realistic traffic behaviour emerges, just as simulations of crowds in malls or emergency evacuation scenarios seek to model individual behaviours that lead to emergent collective behaviours.

Reynolds first proposed the boid in 1987 [6] to represent a bird-like agent. The collective actions of a group of these agents replicates the observed emergent behaviour of a flock. The word boid is used for such agents even if the individual being represented is not a bird - boid simulations include fish in a school, animals in a herd, people in a mob, etc. Reynolds [6] proposed that the movement of discrete birds causes the behaviour of the flock each of which follows a set of simple rules on the basis of its own observations. The behaviour of the flock is the result of interactions between individual behaviours of birds and thus an example of emergent behaviour.

Reynolds [6] based his model on a particle system, but there are some crucial differences between the two. First, a boid has a geometric shape but this only comes into play on a visual level. Secondly a boid has volume, a vital consideration when it comes to collision. Thirdly, a boid's geometric direction is part of its state, unlike a particle. Fourth and most crucially, a boid must interact with its fellows if it is to avoid colliding or being left behind, which in turn means that a boid's behaviour is dependent on two states: the internal (geometric direction) and the external (observation of the behaviour of surrounding boids).

An interesting aspect of flocks in nature is that they lack an upper size limit. The only ceiling on the size of a flock is the availability of individuals - for instance during their annual migration to their mating ground, schools of herring can number in the millions [31]. In Reynolds' model each member is aware of itself and its two or three closest neighbours, to whose behaviours it reacts according to three simple rules. This collective behaviour causes the overall actions of the flock or school. By only concentrating on the actions of close neighbours, the computational complexity of participating in a flock is reduced.

Since Reynolds' pioneering work, other researchers have taken his work further. Olfati-Saber [32] created a set of algorithms focusing on recombining a fragmented flock by implementing obstacle aversion, with α-lattice geometric models being used for mapping flocks. The algorithm developed was a theoretical attempt for the design and analysis of distributed flocking system.

To steer a boid-like agent, approaches such as AFP (artificial potential fields) can be used [33,34]. This approach is widely used, for instance in motion planning for robots [35]. To "steer" a robot, the robot evaluates its behavioural rules according to its monitoring of its immediate environment. The steering response is determined by treating the robot as a point in potential field. AFP gives good results unless the robot is subject to balanced attraction and repulsion effects, in which case it becomes stuck. Furthermore, the robot may fail to find a path if, for example, the repulsion from two objects is too high for the robot to find a way between them.

A flock composed solely of Reynolds' boids can appear somewhat random due to their leaderless nature. Flocks in nature do not necessarily behave this way - geese for instance will migrate in a v-formation with a distinct leader. To represent this behaviour Hartman more accurately and Beneš propose inserting an external leading force [36]. They introduce a complementary steering mechanism, in which, if a boid is on the edge of the flock, it will attempt to break away, thus causing the rest of the boids to change direction and follow.

While most research with boids focuses on the individual boid and how it creates the actions of the flock, Croitoru [37] proposes beginning with a top-level view of the flock, and then move backwards to work out the actions of individual boids. Croitoru have presented a methodology for estimating the boid steering behaviour that can explain the observed emergent behaviour.

Interesting aspects of flocking also include how robust a flock is when exposed to turbulent flow fields, i.e. when groups of marine micro-organisms are exposed to turbulent waters. Flierl et al. [38] investigated how turbulent flow fields affect flocks and found that grouping behaviour is enhanced up to a certain level.

Delgado-Mata et al [39] propose a different approach, expanding the boid's basic behaviour to include a model of the emotion of fear. They model a herd of deer who can communicate their fear to other herd mates and act accordingly. They conclude that a combination of reaction and naturalistic behaviours is important for a realistic simulation.

The application of flocking behaviour has been useful in several fields other than modelling flocks in nature. In robotics, layered behaviours were first used in 1986 [40] when Brooks showed that a robot control system could be accurately decomposed into behaviours rather than functional modules.

2.4 Conclusion

One reason why there are so many different ways to model traffic is because there is no one unifying theory of traffic flow. Attempts to provide a mathematical model of traffic flow date back to the 1920s, when Frank Knight first proposed the bare bones of what would later become Wardrop's principle of equilibrium [14]. This is because the flow of traffic is chaotic and difficult to predict mathematically due to the presence of the human element in the form of the driver. It is not feasible to mathematically represent the irritable anxiety of a tired office worker returning home after a long day or to balance his desire to get home as soon as possible with his desire to not break the law. Thus, the idea of representing the flow of traffic, not as an equation, but as a crowd of individuals, or a swarm of agents balancing their individual wants against a set of constraints like the speed limit begins to take shape as a possible solution. As covered in the Introduction, it then becomes clearer that boids provide yet another avenue for simulating traffic. Moreover, the representation of traffic as a flock is yet to be explored.

Therefore, moving forward, the simulation being developed shall be microscopic, as the focus is on the emergent behaviour of a group of individual drivers, with each driver being a modified boid agent.

Chapter 3: The Tools

Before one can begin building, the appropriate tools must be acquired. There are a wide variety of options available for building simulation, from pixel art to representing the system as a graph. However, correctly simulating road traffic required a number of calculations that, while necessary for an accurate representation, were not only time consuming but also required extensive study. These included, but were not limited to: torque, road resistance, air resistance, effects of vehicle mass on turning and reaction time, coding the graphical interface and representation etc. As such the idea of coding a simulator from scratch was discarded almost immediately.

The idea of modifying an existing open source simulator such as MATSim [41] held several advantages over simply coding a simulator from square one, such as not needing to worry about the physical aspects, only needing to code the actual agents representing the vehicles, and having pre-existing simulations freely available for comparative purposes. However, this approach did not permit the second aim of this project, namely the creation of an easily modifiable simulator that would allow the user to observe the changes in road layout and number of vehicles on traffic flow, as such a task would necessitate taking apart the entire simulator and rebuilding it to allow for modifications in environment. Instead it only allowed for the first aim: to test whether using flocking behaviour to model traffic was a feasible area of research.

At this point it was decided that what was needed was a tool that would handle both the physics and graphical aspects while allowing modifications to both to suit the coder. As such the idea of using a game engine was explored.

"A game engine is a software framework designed for the creation and development of video games. Developers use them to create games for consoles, mobile devices, and personal computers. The core functionality typically provided by a game engine includes a rendering engine ("renderer") for 2D or 3D graphics, a physics engine or collision detection (and collision response), sound, scripting, animation, artificial intelligence, networking, streaming, memory management, threading, localization support, scene graph, and may include video support for cinematics. The process of game development is often economized, in large part, by reusing/adapting the same game engine to create different games, or to make it easier to "port" games to multiple platforms." [43]as quoted in [42]

The use of games for scientific research or education is a well-documented phenomenon called "serious gaming" [44]. The serious gaming approach to developing a boid based traffic system is adopted because it not only allows a more visually appealing result but the in-built physics engine that most game engines use allows a model of the environment that is an appropriate analogue of real world situations.

The list of available game engines is long but it is not too difficult to filter out the more specialised engines, such as those dedicated to first person shooters or role playing games. This leaves, a shorter list of general purpose game engines.

The engine being sought needs to fulfil a few basic requirements: it has to be free to use, it needs to provide a physics engine to simulate environmental effects such as friction, collision, etc., and ideally it needs to be able to create software that is portable.

On the basis of these criteria the options were narrowed down to the following:

1. CryEngine (CryEngine 2015)
2. Unity (Unity Technologies 2015)
3. Unreal Engine (Epic Games 2015)
4. Id Tech 4 (id Software 2011)

Of the four, CryEngine was rejected, despite its powerful engine, as it requires a monthly subscription or the purchase of a corporate license, neither of which was feasible.

Id Tech 4 is open source and freely available under the GNU GPL license [45], but its latest version is proprietary and has not yet been made freely available [46]. There is also a lack of documentation and it is not easy or intuitive to use.

Unity 5 was created by Unity Technologies. The version released in March 2015 was considered. It uses C# as its scripting language, works on multiple platforms, has extensive documentation and is free for personal use. With a 45% share of the market it is one of the most popular engines [47]. The professional version may be purchased outright for 1500$ or subscribed to for 75$ a month.

Unreal Engine is available free to use for academic purposes, has extensive documentation on GitHub [48] along with examples [49]. It has a proven superiority in processing performance with autonomous agents [50]. It comes with its own visual scripting language for quick implementation as well as using C++, which the author has personal experience with. It is the game engine chosen for this project.

Created in 1998 by Epic Games [51], Unreal Engine (UE) was first used to develop the first-person shooter game, Unreal [52]. Epic Games is a US video game company known for such games as Gears of War [53]. Now in its 4th version, UE is used to create games for a variety of consoles including the PlayStation, Xbox, PCs, and mobile games.

The physics engine can be used to simulate several real-life situations. Lewis et al [54] used the Unreal Tournament 3 engine to create a game modification (a mod) to reproduce the appearance of a series of visual impairments. The mod was then presented to three different groups for testing: a set of professionally trained opticians, a visually impaired consultant, and a group of students. It was concluded that "the Unreal Engine 3 was a suitable platform for developing an effective and accurate simulation of visual impairments. It also demonstrates that, following exposure to the virtual environment, users of the simulation reported and demonstrated a greater awareness of the nature of visual impairments. Furthermore, they reported improved understanding of the difficulties visually impaired people face.

UE has also been used to develop the widely used USARsim, a popular simulation system for studying a wide variety of tasks related to mobile robots [55].

UE comes with a highly-integrated physics library using NVIDIA PhysiX [54], which is a physics library used in popular games like Watch Dogs and Assassin's Creed 4: Black Flag [56]. It also comes with an integrated vehicle object, which is used in this project.

3.1 Using UE

As stated above, games can be developed in UE using two different approaches: the built-in Blue-print visual script language, or a more powerful medium using C++ in conjunction with Visual Studio. Scripting allows one to develop the city modules (buildings roads, pedestrians) separately from the boids (agents) themselves. The ease of using the Blueprint tool can be seen in Figure 2, which shows the code required to generate an instance of a generic player class, complete with positions to be targeted by said player, as well a range of positions within which the player is to be randomly generated. The two approaches can be combined. Here, so that the easier. Blueprint approach is used to develop the city modules, thus providing a platform to develop a simulator that can be easily learnt by laymen and thus fulfilling one of the aims of this project.

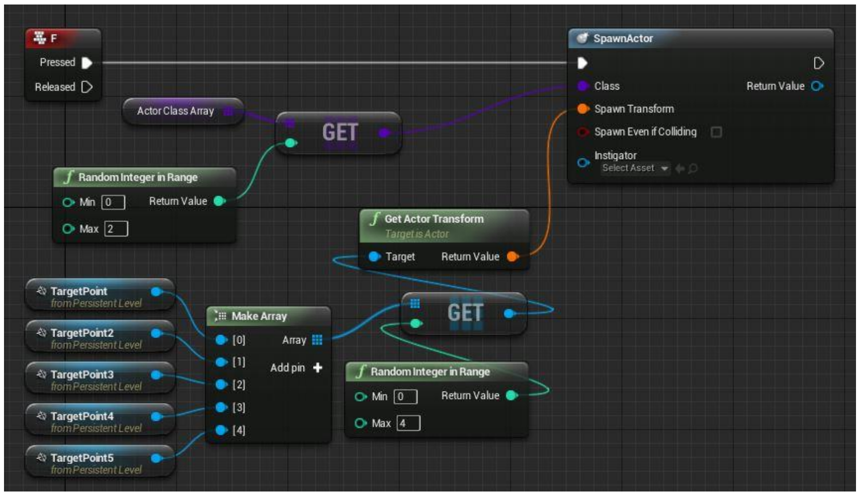


Figure 2 Sample Blueprint Code for Generating a Player Agent

UE allows ease in adding various city modules to the simulator without needing to interfere with the vehicle. These may be added from the vast library available.

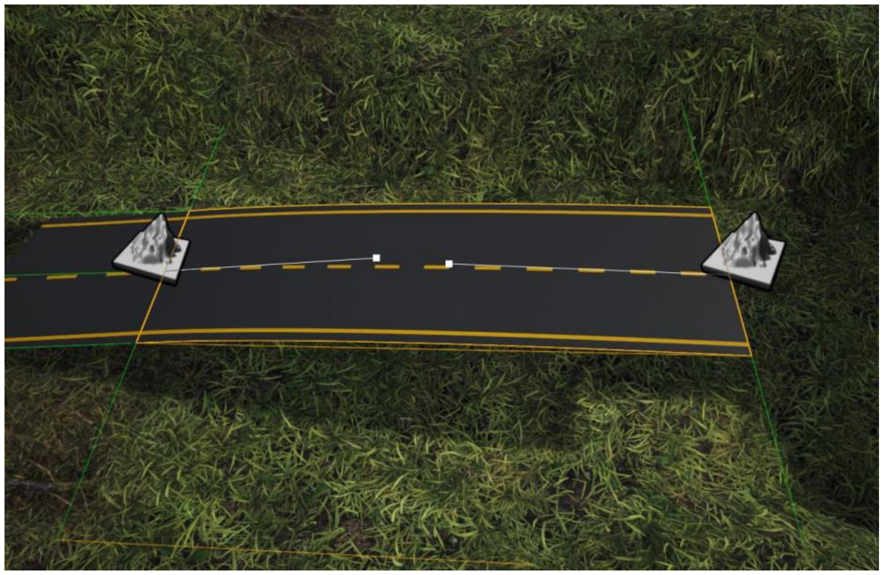
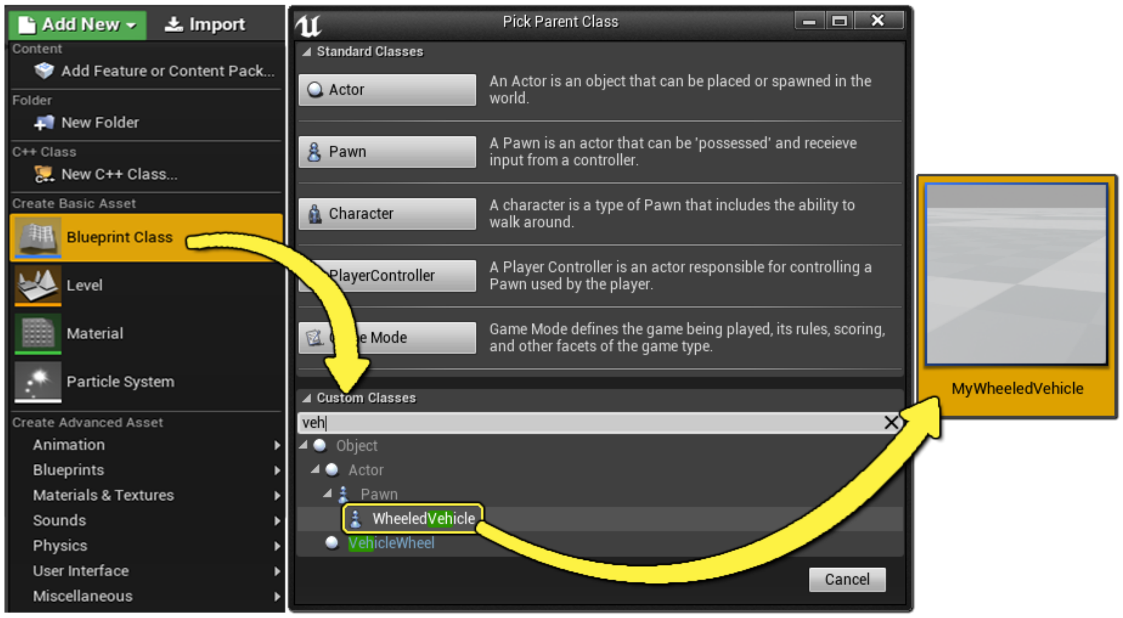


Figure 3 The creation of a road, created using UE's spline tool

In this proof of concept, a single road is introduced. The road can be created using UE's spline tool as shown in Figure 3. The spline tool is a simple drag and drop tool that allows the user to construct a road of the desired dimensions. The road asset comes with a pathway that defines the heading of the road, and the width of the road, the two of them together provide the roadmap our boids will be using.

UE comes with a pre-modelled vehicle type. Vehicles are spawned by creating a spawner at the "beginning" of the road. In this simulation, minimal changes have been made to the actual vehicle model: the speed has been adjusted to spawn with a random speed of between 50 and 60 km/hr. However, UE allows the users to make several, more complex changes to the basic vehicle model from wheel width and height to more complex engine set up. It is to be noted this vehicle model only refers to the basic representation of the physical aspects of the car and not the more complex implementation of the steering behaviour that will be discussed in chapter 5.



Actor: An Actor is an object that can be placed or spawned in the world.

Pawn: A Pawn is an actor that can be 'possessed' and receieve input from a controller.

Character: A character is a type of Pawn that includes the ability to walk around.

PlayerController: A Player Controller is an actor responsible for controlling a Pawn used by the player.

Game Mode: Game Mode defines the game being played, its rules, scoring, and other facets of the game type.

Figure 4 Creating a new vehicle in UE

The spawne5r generates a random number of vehicles between a lower limit *m* and an upper limit *n* per hour. For the purposes of this project *m* and n have been set as 1000 and 5000 respectively.

To demonstrate that heterogeneous vehicles and behaviours can be supported in the boid-like traffic simulation, we introduce two different vehicle models though they both function the same for the purposes of this project: a blue sedan pictured in figure 5 and a red vehicle otherwise identical to the sedan to break up the visual monotony and to allow for greater ease in distinguishing between vehicles while observing the results.

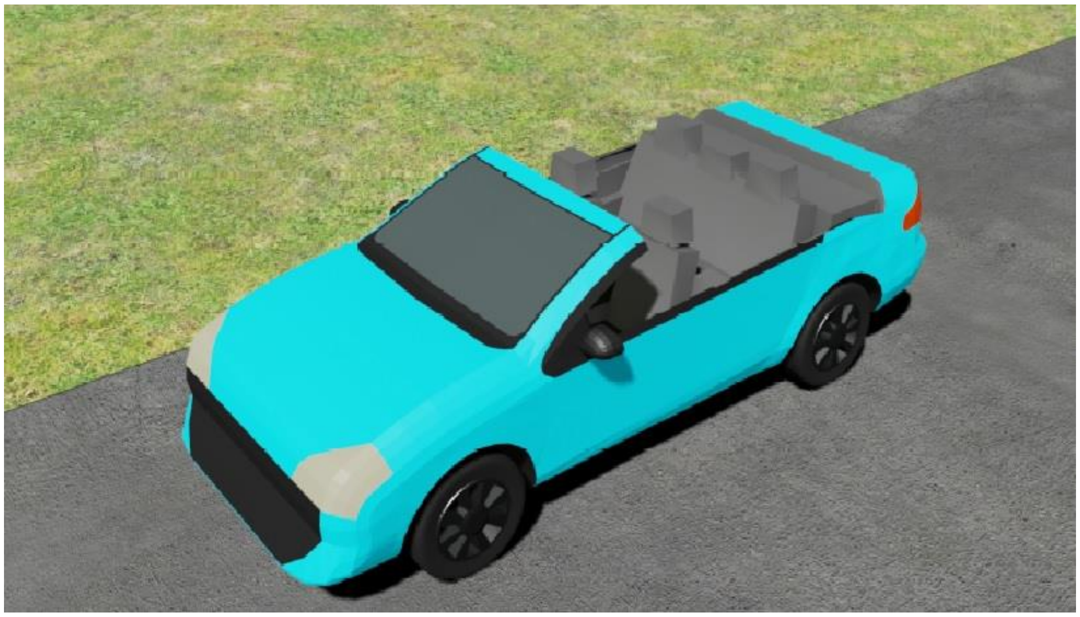


Figure 5: Example of UE's Generic Blue Sedan

A standard built in vehicle in UE consists of:

- A Skeletal Mesh: the basic structure of the vehicle, it consists of a root body
- Physics Asset
- An Animation Blueprint: decides how the vehicle will look in motion, this setting is left as default
- A Vehicle Blueprint
- One or more Wheel Blueprints: there are at minimum two types of Wheel blueprints: one that is affected by steering (the front wheels usually) and one that isn't (in this case the rear wheels)
- A TireType Data Asset: for setting friction

3.2 Tool Summary

In this chapter, various platforms and approaches for agent simulation have been reviewed. The key features of the simulation have been identified. The review of possible simulation approaches led to the decision to use a game engine, because of its ability to support the physics of the environment, and to provide a visually appealing simulation. The free UE game engine is selected, and the key environmental and agent-vehicle features identified.

Chapter 4: Boid Mechanisms

Having selected UE as the platform of choice, this chapter goes into the behaviours defined by Reynolds for a vehicular boid in some depth and covers five basic behaviours: cohesion, seek, flee, pursue, and evade.

Although Reynolds provided a rudimentary implementation of the boid as a vehicle [57], it needs to be expanded to more closely resemble the behaviour of a driver on the road. Here, the various steering behaviours derived from Reynolds' initial model are outlined, before the actual implementation is described in the following chapters.

It is important to note that the text occasionally refer to boids having "aims" or otherwise suggest that a boid has a degree of sentience. This is done because the project is seeking to use the boid to represent a sentient driver. For instance, we refer to a vehicular boid as wanting to get home quickly while not breaking the speed limit as part of the justification for the inclusion of the implementation of velocity matching. Remembering that the boids are meant to represent sentient beings is necessary as future developments will seek to introduce varying degrees of morality in drivers as well as obstacles that represent a negligible physical threat but a significant moral one, such as a small child running at a crossing. It is however important to remember that these "sentient" behaviours are emergent, derived from rules that do not in themselves have a sentient aim.

4.1 Steering Vectors

The emergent behaviour of a flock is the result of two opposing behaviours: the desire to avoid collisions and the desire to flock [55]. The desire to avoid colliding with other individuals has obvious evolutionary advantages, in-air collisions being catastrophic. The desire to flock has several different underlying benefits: protection from predators, greater chance of survival for individuals with similar genetic material (this is particularly true of animals that travel in packs of one male and several females like lions or family units like wolves), greater chances of finding food, extra protection for the young and vulnerable, and mating and social behaviours [28].

These behaviours may be simulated by the following rules outlined by Reynolds [57]:

1. Collision aversion: avoid hitting other boids and objects
2. Velocity matching: do not get left behind or surge ahead and outstrip the others
3. Centring: stay close to flock mates

This project seeks to implement these basic rules in a vehicular boid, and discusses further in chapter 5.

Collision aversion is carried out by assessing the distance between the boid and other boids that it can "see" or obstacles in the environment. If the distance is too small, a steering vector is returned contained the direction in which the boid should turn to avoid a collision and is thus one of the important behaviours to consider while implementing traffic as a boid.

For the purposes of this project, a steering vector is defined as the average of weighted sums of desired velocities individually calculated by behavioural functions. These behavioural functions are representations of the steering behaviours described below.

The rules that control a boid allow it to avoid collisions by not going either faster or slower than its flock mates. This is naturally not a huge factor in simulating traffic, especially as the same effect might be achieved by coding a range of speed limits. This is not to imply that any of Reynolds' rules has an end aim: they are simply rules that avoid follows, which, in the individual appears to priorities collision aversion, and in a collective models the emergent behaviour of a flock. A realistic assumption for the simulation is that this mechanism is contradictory to the nature of a vehicle, as each individual driver naturally seeks to arrive at their destination as soon as possible, and thus has no interest in matching speeds. Where it does come in however is in matching the direction of vehicles travelling in the same lane. Most drivers however will seek to match speed with their neighbours for other purposes: to not go above or below the speed limit, to avoid falling behind etc. In fact, if a driver notices that all other vehicles noticeably slow down at certain points on the road, they may infer that there is a speed trap nearby. These and similar actions require speed matching.

Flock centring is what keeps a boid close to its flock mates. The rule encourages the boid to stay close to the centre of those boids that it can "see", and thus determines the compactness of the flock. While the urge to centre is minimal in the middle of a flock, boids on the edge have a much larger desire to centre. This may be inverted while simulating multi-lane road systems, where drivers in slower lanes have a greater urge to "centre" on faster lanes, but that is beyond the scope of this thesis.

Once again it is important to remember that a boid's cohesion behaviour appears to be a higher priority near the edge of a flock only because the boid has fewer flock mates nearby, and the cohesive behaviour has a lower priority near the centre of the flock because the boid is surrounded by flock mates. A boid's actions are thus based on perceptions of its local environment only, it has no global knowledge of the flock. A driver's view is similarly limited.

4.2 Summation of Steering Vectors

The behaviour of a boid is the sum of the individual behaviours prompted by the three simple rules, which in turn are determined by the boid's monitoring of its surrounding environment. Each behaviour uses the available assessment of the local environment and triggers certain behaviours if required conditions are met. If a behaviour determines that a change of movement is necessary, then it returns a vector, called a steering vector, which contains a direction and a speed. The direction determines the new heading or orientation of the boid, while the speed determines how fast it is to go. The overall action to be taken is determined by taking the weighted sum of individual steering vectors.

A weighted sum is used because some behaviours must have a higher priority than others. When using boid-like agents to represent vehicles, collisions with other vehicles and vulnerable objects like pedestrians must be avoided at all costs, so need to be assigned a larger weight than the vector that encourages a vehicle to stay close to other vehicles or to stay on the road. Therefore, although collision aversion is a behaviour that will only return a vector if any external objects get too close, it will still be assigned a larger weight. Thus, weighted sums ensure that critical behaviours are given priority while making sure other behaviours are not left out.

Weighted sums are not the only way to determine the overall behaviour of a boid. Reynolds in 1987[3] proposed using prioritised acceleration allocation (PAA). PAA accumulates steering vectors up to a certain threshold and simply ignores any that exceed the threshold. PAA rewires prioritising critical behaviour even at the risk of ignoring others and is thus not suited for a smooth or accurate simulation.

4.3 Implementing Steering Vectors In Vehicles

In 1999, Reynolds described a basic model for implementing the boid as a vehicle [57]:

Simple Vehicle Model:

mass scalar
position vector
velocity vector
max_force scalar
max_speed scalar
orientation N basis vectors

The mass scalar is the vehicle mass (required to implement friction, and thus deceleration, accurately), position is the vehicle position, and velocity is the vehicle velocity. Max_force, is the maximum torque produced by the engine of the vehicle. max_speed is the maximum speed of the vehicle, and orientation is the heading of the vehicle.

Reynolds' vehicle model is used as the basis to create the four basic behaviours that will be used later on: seek, evade, flee, and pursue.

Pursue is when the boid is looking for its nearby flock mates, hoping to keep them in sight, so as to not fall behind. As a vehicle, the boid shall attempt to keep a certain maximum distance between itself and the vehicle in front. If the maximum distance is exceeded, it triggers an increase in speed as well as a correction of the directional vector if required.

Evade is when the boid is getting too close to its flock mates. As a vehicle, the boid shall attempt to keep a certain minimum distance between itself and other cars. If this limit is breached, the boid shall try and either reduce speed, or if this causes it to collide with the vehicles behind, it will attempt to veer onto the grass.

Flee is the behaviour that shall keep the vehicle on the road and off the surrounding terrain, unless there is a danger of collision with other cars. In a more complex simulation, this behaviour will need to either be bifurcated into "avoid going off road" and "avoid crashing into pedestrians/pil-lars/buildings etc.", or combined with the evade mechanism.

Seek is when the boid shall actively seek locations that are not other vehicles: a prime parking spot, an open space in a faster lane, a chance to overtake, an opening in a traffic jam, a green light etc.

As can be seen "seek" and "pursue" are behaviours inspired by the flock cohesion mechanism, while "evade" and "flee" are inspired by the collision aversion mechanism, with velocity matching being folded into both as required.

$$\begin{aligned}
\mathbf{v}_{dir} &= p_{current} - p_{target} \\
\mathbf{v}_{des} &= \frac{\mathbf{v}_{dir}}{||\mathbf{v}_{dir}||} * max\_speed \\
\mathbf{v}_{steer} &= \mathbf{v}_{des} - \mathbf{v}_{cur}
\end{aligned}$$

Figure 6 Reynolds' Equations for determining desired velocity for seek behaviour [6]

Reynolds determined the seek behaviour by normalising the vector between the current and seek position. This is done by first determining the steering direction by subtracting the b target position from the seek position. This is then normalised and multiplied by the speed to achieve the desired velocity. The current velocity is then subtracted from the desired velocity to obtain the new steering vector, as illustrated in figure 5.

Mathematically, seek and flee are very similar with the main difference lying in the direction of the vector. In seeking, the boid aligns itself towards the target while in fleeing the boid aligns itself away from the target. Therefore, the equation for flee behaviour is like the seek equation shown in figure 5 except that the steering vector is obtained by subtracting the desired velocity from the current velocity.

Figure 6 shows illustrates the effect of the seek and flee behaviours and how they combine to steer a boid. In Figure 6, the green arrow represents the current velocity, while the grey arrow represents the desired velocity. The blue and red arrows represent the steering vectors for seek and flee respectively, while the blue seek path represents the path the vehicle would take to reach its goal. The red flee path shows the same if the vehicle were actively seeking to escape the target rather than approach it.

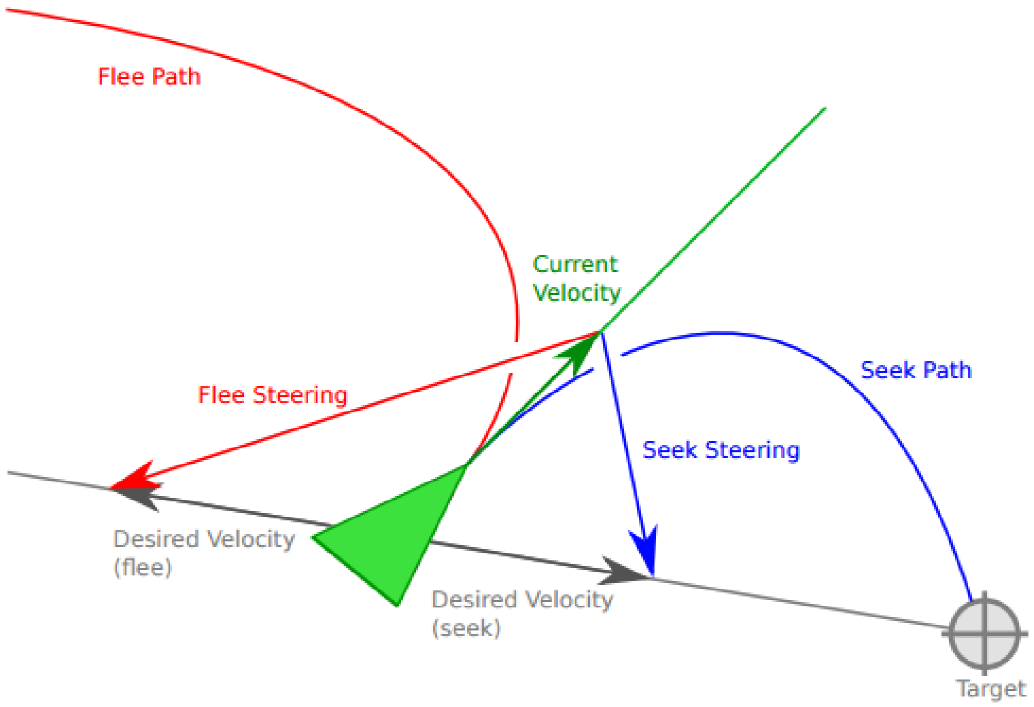


Figure 7 Seek and Flee behaviour. Adopted from Reynolds 1999 [57]

Figure 8 illustrates the pursue/evade behaviour. The main difference between “seek/flee” and “pursue/evade” is that in the former pair the target is stationary while in the latter it is dynamic. Therefore, to correctly calculate the steering vector, the target’s position needs to be continuously re-evaluated. To ensure an accurate vector, the target position needs to be re-evaluated as often as possible, i.e. the time between predictions of target positions T should be close to 0. However, to save computational cost and to ensure speedy run time, T may be reasonably large when the target is distant and gradually decrease as it becomes closer. Mathematically however, seek and pursue are alike, as are evade and flee. The main difference is that the target position is constantly being re-evaluated.

Figure 8 illustrates the boid changing direction based on a steering vector calculated by taking in the quarry’s present location and speed and using it to predict its future location. While a simulated boid may simply turn on the spot to change its path (as signified by the blue and red arrows) in real life, a vehicle’s steering would adopt a more curved path, as illustrated in figures 7 and 8. It is an important point that even though we are attempting to represent traffic as a crowd, cars, unlike people cannot be reduced to a particle system illustrated by points. They have a much larger mass (which affects acceleration and deceleration behaviour) and occupy greater space. This point is especially worth remembering if one were to represent a larger variety of vehicles, from tiny sedans to eighteen wheel trucks, each of which would react at different speeds to threats and with different levels of manoeuvrability.

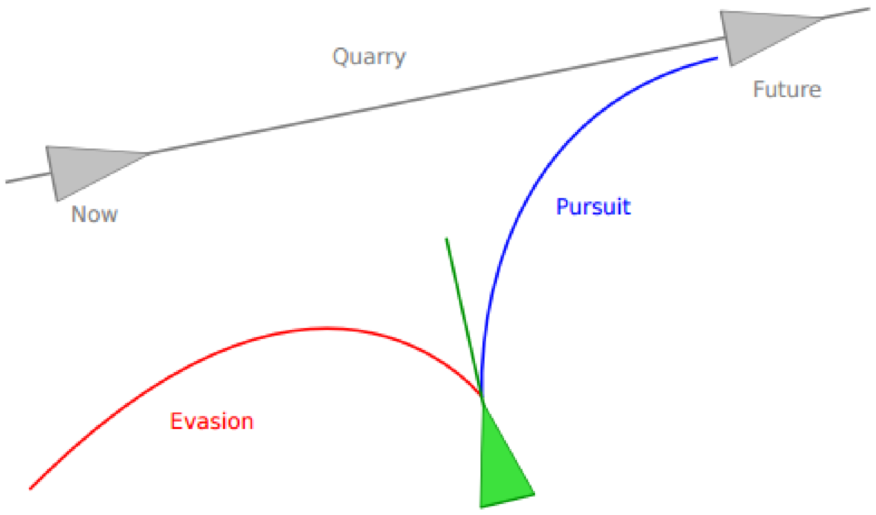


Figure 8 An illustration of pursue and evade behaviours [57]

A major difference between the real world and a computer simulated model is that in a computer model the environment is not continuous but discrete and refreshes at fixed intervals. Therefore, the steering vector needs to be updated multiple times a second to avoid a jagged path.
For example, if a vehicle's speed is 50km/hr, then it is moving at approximately 14m/s. If the steering vector is only updated once per second, then the vehicle will appear to “jump” fifteen metres every second and if the direction changes as well as the speed, then it becomes difficult to map the path accurately.

While these vectors may be implemented in a number of ways, for the purposes of this discussion, they are represented as 2D vectors as the vehicles are moving on a 2D plane. Once again, in a more complex city system that makes use of flyovers, multilevel parking lots and the like, this would need to be modified, though not necessarily heavily enough to resemble the regular 3D implementation required when representing flocks of birds or schools of fish. This might be necessary, as the boid would be forced to visualise nearby boids in a three-dimensional space.

The final behaviour to consider is alignment. It allows a boid to align itself with fellow boids in both direction and speed. One way of aligning a boid is to take an average of the velocities of nearby boids and then assign the average to the boid. The steering vector will then be the difference between the boid's old velocity and the current velocity.

It would be interesting in future work to observe the effect on alignment when nearby boids are themselves poorly aligned, or what happens if a boid is deliberately coded with a poor alignment mechanism to simulate the effects of an unreliable driver or a driver under the influence.
Figure 9 visualises the alignment behaviour. The boid observes the velocities of nearby boids in its field of vision and averages them. It then calculates its required steering vector (represented by the red arrow and adjusts accordingly. The left-hand line on the green (central) boid represents the desired velocity.

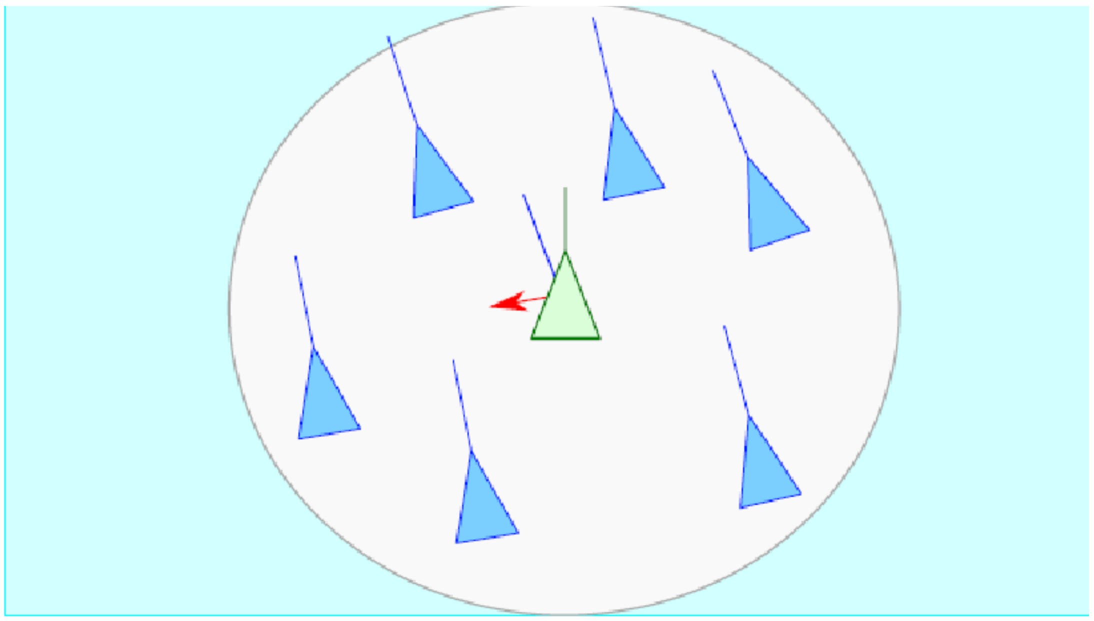


Figure 9 Alignment Steering behaviour [57]

These basic behaviours combine to create more complex behaviours when they occur simultaneously across a group of boids. The next chapter considers how, in this project, the resultant steering vector is calculated, using a weighted sum of the individual vectors. Depending on the steering vector a boid may prioritise avoiding a collision over correctly aligning with fellow boids.

Chapter 5: Architecture and Development

The basic goal of this project is to test whether simple flocking using the behaviours outlined in chapter 4, may be used to effectively simulate traffic. As described in chapter 3, this proof of concept models traffic boids on a single lane road, using the UnReal Engine. For simplicity, the road is completely flat and all calculations take place in the plane of the road on an xy axis.

The system architecture is heavily influenced by how UE is structured. UE is class based and uses both inheritance and composition. All actors (that is all components that participate in the simulation such as the road and vehicles) in the simulation are instances of descendants of the Actor class.

The algorithm that determines the movement of each vehicle uses a layered model as illustrated in figure 10.

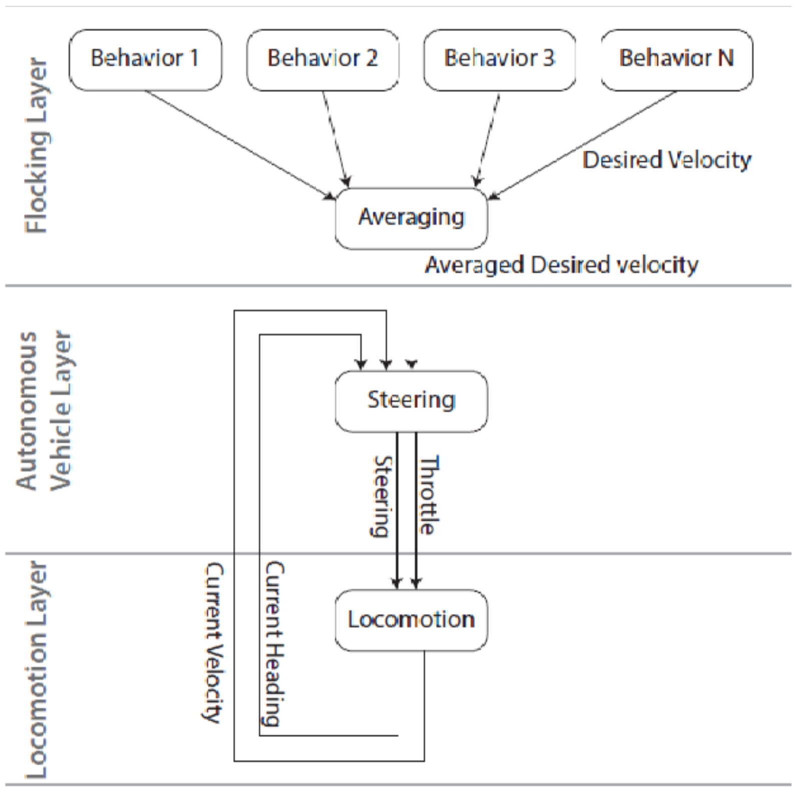


Figure 10 An Overview of the Layered Model Structure of Boid Model

The highest or flocking layer is what controls the flocking mechanism. This layer takes in environmental cues which trigger the four behaviours discussed below. The triggered behaviours return a certain velocity. The flocking layer then averages a weighted sum of these behavioural velocities and returns the desired velocity to the steering layer, which uses this along with the current velocity to determine the new steering vector which is then passed to the locomotion layer. The locomotion layer is basic vehicular model discussed in chapter 3. The only arguments passed to the locomotion layer are the changes a driver would make whilst behind the wheel: speed and throttle.

Therefore, for ease of understanding: the flocking layer is the driver assessing their surroundings and deciding on their desired velocity (i.e. the desired speed and heading), the steering layer is the driver calculating the steering vector (Desired velocity-Current Velocity) and then passing said vector to the locomotion layer which then makes the necessary changes to the vehicle’s speed and direction as well as factors in the effects of friction, wind resistance, the action of gravity and other physical effects.

The flocking layer takes the average of the weighted sum of four behaviours derived from the three defined in chapter 4. These six behaviours are: pursue and flee, and seek and evade.

5.1 Flocking Layer

As discussed above, this is the layer that represents the driver taking in his environment and making decisions on steering. This section will discuss how the driver vies the road and orients himself, and the implementation of the seek, flee, pursue, and evade behaviours.

5.1.1 Orientation and Vehicular Vision

In chapter 3 we discussed some steering mechanisms derived from Reynold’s boids [57] and what they would mean for a vehicular boid.

To reiterate, the boid as defined by Reynolds is defined by three basic rules:

1. Collision aversion: avoid hitting other boids and objects
2. Velocity matching: do not get left behind or surge ahead and outstrip the others
3. Centring: stay close to flock mates

There are some differences between a boid in the wild and a vehicle. Firstly, there is a radical difference between herd animals and vehicles. Animals in a flock have a natural advantage in staying close, as there is safety in numbers as well as a greater proximity to mates, food sharing and rearing of the young (particularly noticeable in mammals like hyenas and lions). By contrast for vehicles on the road greater proximity means greater chances of collision [58]. Furthermore, we assume that reaching his own end goal is of greater importance to the average driver than remaining close to other cars. However, this does not mean there is no place for the cohesion mechanism when implementing the boid as a vehicle. For vehicles, cohesion effects can be applied between vehicles travelling in the same direction, and can help to achieve velocity matching. If the simulation imposes speed limits, cohesion can also be of use. However, the cohesive behaviour would be discarded entirely if the nearby boid were oncoming as the only conceivable result would be a collision.

Returning to the three rules, to follow them, the vehicular boid needs to be aware of both the speed and orientation of nearby vehicles. It was decided that the velocities of nearby vehicles would be stored as private variables in the Vehicle Actor class of which each boid is an instance. One variable would store the velocity of vehicles travelling in the same direction while the other stores the velocity of vehicles travelling in the opposite direction. These variables are used to calculate the boid's velocity anytime one of the four main behaviours are triggered.

A vehicle needs to take into account the direction of travel of cars in the same direction as itself and the direction of travel of vehicles coming the from the opposite direction, as well as of vehicles entering from side roads, intersections, and roundabouts.

The cohese vector contains the direction and speed of nearby boids travelling in the same direction, while the align vector contains the direction and speed of nearby boids travelling in the opposite direction. The cohese vector is assigned when the vehicle is spawned. As the current project consists of a single lane, one way road, the distinction is immaterial but in future iterations, when multiple lanes and link roads will be added, the align vector will take on more importance and will probably need to be refined further to account for nearby boids with different directions of travel. Now, the align vector serves as a placeholder for future development.

The cohese vector is non-empty at the point of spawning, and the class constructor generates an error message if it is empty upon being created. Roads in UE are created using a spline tool and contain a road plan defining the centre of the road and the radius or width of the road. These two combined provide a 2D roadmap that vehicles can query to determine the heading of the road as well as the occupancy of nearby spaces.

Vehicles in UE come with a built-in vector called ForwardVector that orients the vehicle. It contains the direction pointing to the vehicle's front. The ForwardVector, together with heading of the road, is used to determine the cohese vector of the vehicle. If the cohese vector and ForwardVector are not identical, this will trigger a reorientation of the vehicle until it is facing the right direction.

The presence of a queryable roadmap allows the boid to determine the location of nearby obstacles. To more accurately represent the average driver, the queryable distance is restricted to twice the length of the vehicle in all directions. Accurately representing the "field of vision" of the driver proved to be challenging, as did determining representation of the velocities of nearby boids.

Firstly, it was necessary to determine how far a boid could "see". A driver on the road has, barring any obstructions, a reasonable level of visibility and is able to make note of and account for cars well beyond the limit decided on for this project. However, allowing such a large range of vision resulted in the vehicles behaving less like a flock and more like a swarm of connected agents, as each vehicle was able to make calculations and adjustments to its steering behaviour based on the actions of a boid well ahead or behind. It is important to remember that the vehicle see its surroundings by querying a roadmap, a facility unavailable to the average driver. This would allow the boid to see vehicles from around corners, behind larger vehicles and other blind spots, which of course is not a realistic representation of road traffic.

After some trial and error, a viewable distance of twice the length of the vehicle was decided upon, although this does leave some questions unsatisfied, such as the fact that drivers on the road do not have such a clear-cut restriction on how far they can see or that drivers do make changes in their behaviour based on events visible much further down than the restriction placed here. For instance, a driver, upon noticing that the flow of traffic further ahead is much slower, may decide to switch to a different lane well before he infringes the two-vehicle limit implemented here. A possible solution, discussed as further work later, is to add a secondary layer of "imperfect vision" which would only notice significant events outside the two-vehicle limit such as accidents, bottlenecks, and the like. These events would only be seen when the velocity of vehicles outside the "perfect sight" limit falls below a certain level.

While in flocking, a boid takes note of its flock mates' positions to stay *close* to them, as a vehicle a boid would use the same information but in order to stay *away* from them. However, there is an interest on the driver's part to not stray too far from other vehicles as this usually means that he is either going too fast and outstripping them or too slow and lagging behind, thus going against his goal of reaching his destination as quickly as possible.

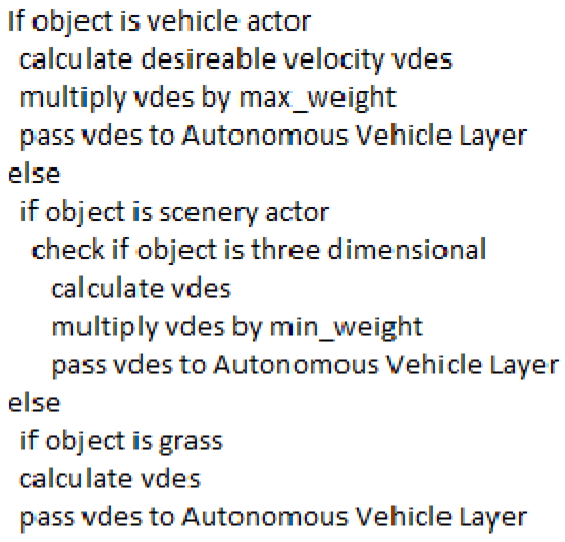
Furthermore, it was noticed that, as the road system grew more complex, and the number of vehicles increases, there would be multiple directional vectors to take into account, as vehicles began to move in opposite directions, or when, with the addition of link roads and roundabouts, the vectors were no longer simply parallel to each other. In a North-South two way road, the orientation of vehicles travelling north could be represented as 1 and the ones travelling south as -1 but this can no longer apply when vehicles start travelling at angle to each other.

Therefore, cohese and align variables would be used for comparative purposes by the pursue and evade behaviours, and if a change in the steering behaviour were required, the appropriate weights would be assigned.

As stated above, the importance of cohesion while simulating traffic is debateable. In a project focused on automated driverless cars undertaken by two students at the Norwegian University of Science and technology in 2015, it is argued that the only benefit cohesion brings is that it increases the chances of vehicles platooning and thus reduces drag [58]. As such they modify the cohesion vector to only account for vehicles travelling the same way, and to only allow for lateral cohesion.

While reducing cohesion for automated cars has merit, the Norwegian approach does not suit this project as our vehicles are acting more as a crowd than a robotic swarm; a vehicle with a sentient driver is not likely to strive to stay close to other cars on the road. Furthermore, there is also a need to define a cohesion behaviour in response to oncoming vehicles as they can prove to be a much greater threat than a vehicle travelling in the same direction as the boid.

The collision aversion behaviour requires adjusting because of the difference in threat levels of objects that need to be avoided. While the aversion of active threats, like predators, maps clearly onto a vehicle's desire to avoid oncoming traffic, leaving the flock’s path is not a “free” action on the road. As a bird, the boid is free to leave the flock to avoid a predator or other obstacles and return once the threat has been avoided but a vehicle does not have this luxury. For vehicle boids, it is necessary to expand the definition of “objects to avoid” to include the off-road area but, since this is not on the same threat level as an oncoming vehicle, the implementation either requires a complicated aversion function with a nested if-else statement listing possible threats and assigning the appropriate weights, or requires the splitting of the behaviour into two aversion mechanisms: flee and evade.



```
If object is vehicle actor
 calculate desireable velocity vdes
 multiply vdes by max_weight
 pass vdes to Autonomous Vehicle Layer
else
 if object is scenery actor
  check if object is three dimensional
    calculate vdes
    multiply vdes by min_weight
    pass vdes to Autonomous Vehicle Layer
else
 if object is grass
 calculate vdes
 pass vdes to Autonomous Vehicle Layer
```

Figure 11 Discarded approach to implementing collision aversion

Initially, the nested conditional approach (Figure 11) was tried. This approach is inelegant. More importantly, it goes against one of the main aims of this project, to develop a simulation tool that allows independent changes to both the boid and the surrounding environment without affecting the other. The above approach would require that the checklist be updated anytime a new environmental feature were added. Instead, the behaviour was dividing into flee and evade, as discussed below. The flee behaviour is triggered if and only if the boid gets too close to the edge of the road, while evade is only triggered if the boid gets too close to other vehicles.

5.1.2 Seek and Flee, Pursue, and Evade

The main four behaviours are seek and flee, pursue, and evade. While the pairs may sound similar, for the purposes of this project, they have some distinct differences. Seek and flee react to stationary objects while pursue and evade react to objects in motion.

Pursue and evade react to other boids in motion. Pursue is a function within the Actor class that represents vehicles that contains a maximum distance the boid may maintain from nearby vehicles. At each time step the location and velocities of nearby boids is passed to this function. If the location of the other boids is more than the maximum distance allowable by the variable discussed above by 50%, the pursue behaviour is triggered, and the necessary adjustment to the velocity is calculated. For example, the current pursue function holds a maximum distance value of 64 metres. If the distance between the boid and the vehicle in front is more than one and a half times the maximum distance (i.e. 96 metres), then the pursue action is triggered. The 64-metre limit was chosen as being twice the minimum safe stopping distance of a vehicle travelling at 60 km/hr.

At this stage of the project, this simply means an increase in speed. Evade defines a minimum distance that the boid must maintain from nearby vehicles. If the distance between two boids is less than the minimum distance times three quarters of the minimum distance, then the evade mechanism is triggered. This limit was settled on as it was considered more in line with safety procedures for a drier to begin taking evasive measures before he has actually breached the safe limit.

However, the evade mechanism has a few extra steps that pursue lacks. Depending on the direction of travel of the other boid, an extra weight is added before the desired velocity is returned. So, if the other boid vehicle is travelling in the opposite direction added emphasis is placed so that the evade behaviour is weighted twice: once before being returned and again before the average of the weighted sum is calculated. As this project focuses on a one-way street, this particular function has not yet been tested. Evade is a partial realisation of the collision aversion behaviour described for Reynolds' boid, with the other part being realised in flee.

Another extra step for the vehicular boid as opposed to the bird in a flock, is that the evade function checks the location of the other boid. If the other boid is behind and following the vehicle, then two separate desired velocities are calculated: one in regards to the boid in front and another to the one behind. A small weight is then multiplied with the desired velocity calculated in relation to the boid behind. Then an average of the two is returned. This adjustment is needed because if the boid were equidistant between two other vehicles, and if both were within the minimum distance that triggered the flee mechanism, the boid would either freeze in place or, rather distressingly, start spinning on the spot until the vehicle behind crashed into it.

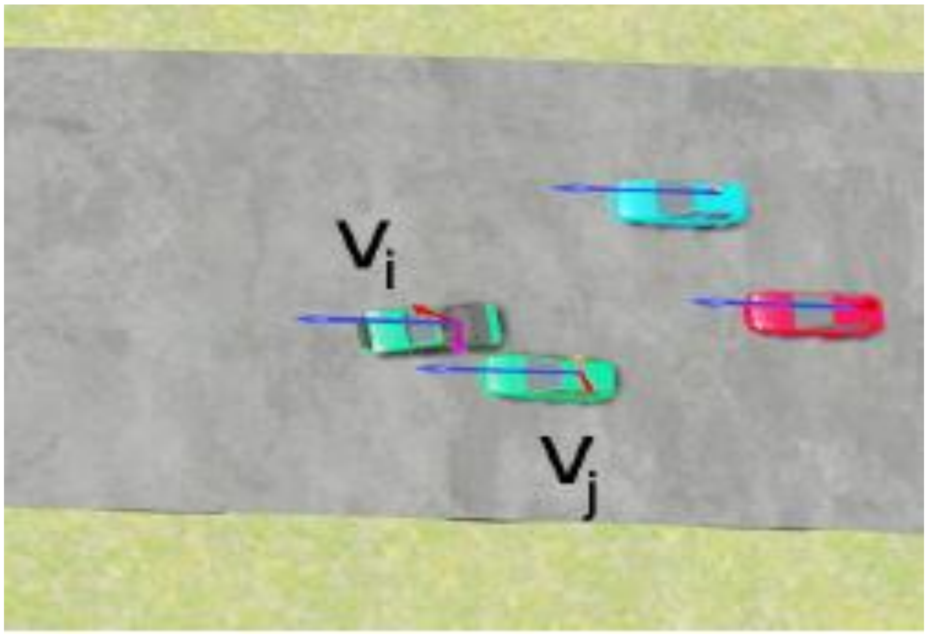


Figure 12 An Instance of Evade in action

Figure 12 illustrates this change in the evade mechanism. Trapped between the boid marked *vi* and the red car at the extreme right, boid *vj* has prioritised distancing itself from the red car even if it temporarily has to get too close to *vi*, falling well within the minimum distance it needs to maintain from other boids. This behaviour is achieved by multiplying a small weight to the red car's velocity before the average was taken, which skewed the desired velocity in such a manner that *vj*'s steering mechanism opted to overtake *vi* at the cost of collision.

The decision to priorities aversion of collision with a vehicle behind the boid was taken because it is more likely to cause a collision as it is speeding towards the vehicle, as opposed to a vehicle in front as it is speeding away. This is not however entirely satisfactory as a driver is more likely to notice a vehicle in front getting too close than a following vehicle.

This extra step was added as otherwise the vehicle would freeze if faced with the threat of collision with both the vehicle in front and behind. To solve this, it was decided that the vehicle would prioritise avoiding the vehicle behind it as slowing down to avoid the vehicle in front would only increase the chances of collision with the boid behind.

Seek and flee behaviours react to stationary objects. Flee operates on a similar principle to evade. It contains a minimal distance to be maintained from all stationary objects, which at this stage merely means the off-road area, but as the layout that the boid navigates grows more complex, it will gain further significance. There will be a need to differentiate different threat levels of stationery objects: it is safer to go off road in a field than on a bridge or a flyover. In the context of this project however, flee serves as a way to keep the vehicle on the road when it tries to avoid colliding with other vehicles.

In early iterations, both seek and flee were left as placeholders to be developed in the context of a more complex layout where the behaviour could be more accurately tuned.

It was observed however that, due to there being no weight attached to keeping the vehicles on the road, they would simply go off-road to avoid collisions, instead of trying to manoeuvre into a safer position, even when the threat was not drastic as evading a vehicle was given a significant weight. This behaviour was very different to that of a typical driver, so the flee function was implemented.

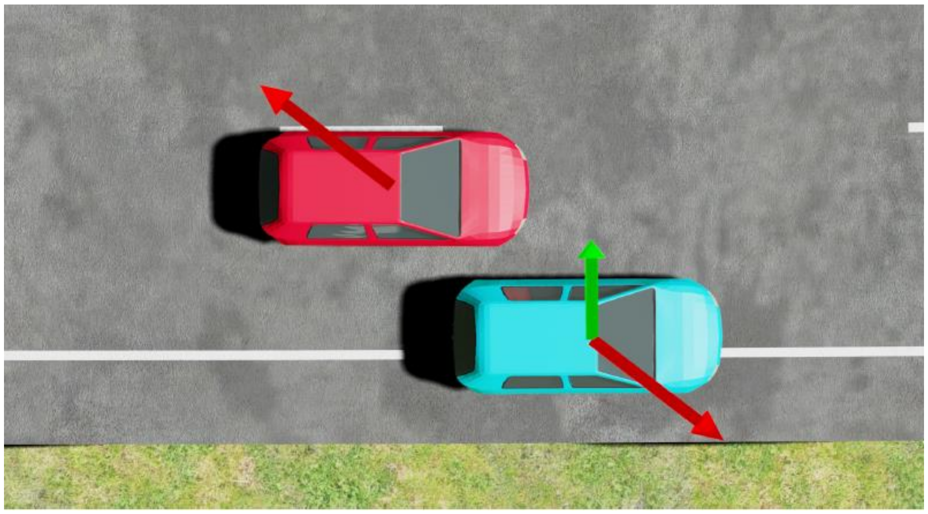


Figure 13 An Instance of Flee in Action

Seek is the behaviour that will come into play when the vehicle needs to move towards a stationary object such as a free parking spot or the shoulder of the road etc. Right now, the seek behaviour is merely triggered by the end of the road. In the context of the current project it is not of much significance because stationary objects have not yet been included in the environment. Stationary objects are only apparent temporarily, such as an opening to overtake. However, the full seek behaviour may be achieved by an implementation of the overtaking leadership behaviour observed in flocks of geese in the wild such as that of Hartman and Benes [34].

5.1.3 Calculating the Steering Vector

Once the locations and velocities of the boid vehicles have been noted, and the individual behaviours have calculated the desired velocities, there is a need to calculate the final vector that decides what action is to be taken. This decision is taken by the averaging function, which is called at each time step and which in turn calls the behavioural functions. If the behavioural functions' conditions are triggered, then the individual functions' desired velocities individually calculated and returned to the averaging function. The averaging function then multiplies the individual desired velocities with the associated weights and then averages them to find the velocity-that is the speed and orientation-the vehicle will take.

Deciding to use a weighted sum for the final desired velocity was in some measure intuitive. The movement of a boid in a flock, while predicated by the rules devised by Reynolds, does not take into account every rule equally. There is then a need for a way to combine the velocities in a manner that models the behaviour of a driver with some degree of accuracy. This cannot be achieved by a simple averaging as that attaches equal importance to all six behaviours and does not take into account the fact that collision aversion functions must take precedence over velocity matching functions.

When considering the above, taking the average of the weighted sum of the behavioural velocities seems like a natural solution.

Deciding the appropriate weights proved to be a difficult process that still requires finessing. Correctly balancing the various behaviours to achieve a realistic steering mechanism that represented the average driver with reasonable accuracy was not immediately intuitive.

The evade function is particularly illustrative. At first glance it seemed obvious that it should hold the highest weight as avoiding collisions with other vehicles is of a priority for the driver. Therefore, it seemed simple enough to assign evade the highest value. However, as illustrated above, it was then found necessary to further weight the velocity of vehicles following the boid if they got too close as they were more likely to cause collisions, as well as to break the “freezing” effect boids caught between two equidistant vehicles exhibit.

Similarly, assigning a minimal weight to pursue and leaving flee unweighted was initially assumed to be sufficient to keep the vehicles on the road. This however proved not to be the case, as the simplest option for a vehicle is to go off the road to avoid a collision. A small but not insignificant weight is added to the desired velocity generated by the flee function.

$$\mathbf{v}_{des} = \sum_{b \in B} \mathbf{v}_b * w_b$$

Figure 14 The averaging function for boid vehicle vectors

The above figure shows the averaging function used to achieve the final desired velocity, which is then passed to the Autonomous Vehicle Layer. The individually generated behavioural velocities v_b are multiplied by the assigned weights w_b *and* then the average is taken of the sum of these products.

5.2 Autonomous Vehicle Layer

Returning to figure 10, we see that the autonomous vehicle layer serves as a bridge between the flocking layer and the locomotion layer. At this point in the project, it serves to provide a throttle and steering variable to the locomotion layer. It references the current velocity of the boid, accepts the desired velocity and calculates the necessary change that needs to be made, that is the difference between the current and desired velocity.

The autonomous vehicle layer then translates the necessary change into a change in the speed and heading of the boid, and passes these to the locomotion layer.

It is possible in future iterations to use the autonomous vehicle layer to implement some constraints in cases where simply following the output of the velocity layer may not be desirable. One such scenario may be that the flocking layer's output would cause the boid to go off road in a hazardous area such as a bridge or a flyover.

5.3 Locomotion Layer

The vehicle implementation in Unreal Engine is a wrapper around the vehicle implementation in NVIDIA PhysX.As discussed in chapter 3, the physical vehicle model provided by UE is left largely unchanged. When creating a vehicle class in Unreal Engine, multiple parameters can be tuned. These parameters include, but are not limited to:

- Wheel radius and width
- Wheel mass
- Wheel damping rate
- Braking behaviour
- Steer angle (including which wheels can steer)
- Tire friction
- Tire stiffness

- Max brake torque
- Vehicle Mass Drag Coefficient
- Chassis width and height (used to calculate drag force)
- Torque curve (torque for different motor RPM)
- Engine Setup (things like Max RPM, moment of inertia, damping of different parts)
- Differential setup (front or rear drive, or both, how the force is split between the wheels)
- Transmission setup (Gear ratios and number of gear)
- Steering curve

In the proof-of-concept simulation, most of these parameters were not manipulated. The most significant ant change made was to the braking behaviour. It was observed that the boids were breaking too smoothly when approaching too close to a vehicle in front, and it was felt that this behaviour was not representative of a driver in the real world. On the road, if a car gets too close to another, the driver would not attempt to decelerate gently but would rather brake sharply. This desired behaviour was achieved by increasing the friction upon braking variable.

5.4 Preliminary Evaluations and Achievements

This section describes how the prototype simulator developed in this feasibility study was tested. The goal is to demonstrate that the vehicle boids behave in the ways expected on their own and in small groups.

5.4.1 Test Driving the Boid

The purpose of the first experiment is to observe whether a single vehicle behaves in the way that would be predicted from simple application of the rules.

A solitary vehicle was generated at a preliminary velocity of 55 km/hr in order to test ordinary behaviour of vehicle. It was observed that the vehicle kept gaining speed as it progressed, although it did not change orientation or exceed the programmed safe speed. This was expected. The seek function caused the vehicle to accelerate until another vehicle was detected, but no other vehicle existed. As there were no obstacles (stationary or otherwise) in its path, none of the other three behaviours were triggered. As there were no other cars on the road to trigger either the flee or pursue behaviour, the boid kept accelerating until it reached the speed limit, then maintained maximum speed until reaching the end of the road.

5.4.2 Test Driving Two Boids

In the second experiment, two vehicles were generated, one five seconds after the other in order to test the effect on behaviour due to the pursue and flee behaviour. The second boid was observed to accelerate until there was just the minimum safe distance between the two vehicles, which was the how the boid was expected to behave when the pursue behaviour is triggered. At this point it decelerated slightly, since the evade behaviour was triggered. This too was expected behaviour. The vehicle generated first showed the same behaviour as the vehicle in the first test, as was expected.

5.4.3 Test Driving In Large Groups

At this point it was decided that a larger number of vehicles need to be generated in order to properly observe the simulator behaviours. Therefore, several runs of the simulation were carried out, each run generating between 1000 and 5000 vehicles. The vehicles were generated in batches of 50 vehicles, with an interval of two minutes between batches. There was a gap of half a second between each vehicle’s generation. This experiment is designed to mimic the periodic arrival of groups of vehicles, often observed in heavy but uncongested traffic flow.

Therefore, each experiment began by randomly generating an integer between 1000 and 5000, which determined the number of vehicles to be generated. The vehicles were then generated at half second intervals, with a two-minute gap after every fifty vehicles.

Apart from some outstanding outlying issues discussed in chapter 6, the observed behaviours were appear to indicate that the vehicle agents interact as expected. satisfactory. Due to the velocity of each boid falling within a specified range, rather than being assigned a hard value, instead of showing a neat line of vehicles equidistant from each other, there was observable acceleration and deceleration as each vehicle's pursue or evade behaviour was triggered. Furthermore, vehicles accelerated to the maximum speed limit as they neared the end of the road, which was their destination, and thus triggered them seek behaviour.

This chapter has outlined the design decisions and implementation of the steering behaviours discussed in chapter 4, and has discussed some of the resolution of various issues that arose. Testing has shown that the prototype simulator, in which cars travel "safely" along a single carriageway road, produces behaviour that matches expectation. The next chapter discusses some further problems encountered and their possible solutions.

Chapter 6: Outstanding Issues During the Simulation

Due to the lack of previous practical work done in the implementation of Reynolds' boid as a vehicle, as well as the relatively small scope of this project, developing a methodology and experiments proved to be challenging as well as failed to provide any meaningful information. As one of the aims was to implement Reynolds' boid as a vehicle, the most accurate test would naturally be to replicate a scenario for which an accurate simulation and data already exists and check if the two match up. This is necessary as it is not simply enough to produce a simulator where the boids flock without crashing or going off road but all reach their destination safely. This is because the entity being modelled as a boid is a flesh and blood driver, and thus achieving a consistent result where each boid arrives safely and without incident would actually be considered unrealistic and a failure.

As stated above, Johansen and Løvland [58] carried out a similar project to explore the behaviour of driverless vehicles using boids. They had a narrower question which necessitated the designing of repeatable experiments in order to test their hypothesis that driverless vehicles made for safer roads. The fact that the vehicles in their project were driverless removes the human element that our own project is seeking to model.

In our work, the restrictions imposed by only implementing a single lane, one way road makes it difficult to compare the observed vehicle and emergent traffic flow behaviour to any real-life scenario. It also prevented the full testing of some features like the seek function and the align variable. In future iterations, a second lane will be added to test how the velocity matching behaviours might be used to determine which lane is faster, and to support switching lanes, etc. In this chapter, we focus on the main issues remaining and the possible solutions that may be implemented.

6.1 Cohesion

Implementing the cohesion behaviour proved to be difficult, particularly as visualising a use for it when representing a vehicle was counter-intuitive. The issues with implementation and the result are discussed in chapter 5. This final result however has not proven satisfactory.

The main difficulty lies in keeping track of the current locations of nearby boids. As stated in chapter 5, this is achieved by querying the roadmap within a strict two-vehicle limit. However, this approach only works when all vehicles move at a uniform speed, a scenario that does not hold true for the real world. Unfortunately, when implementing varying speeds for vehicles, the location became much more difficult to calculate accurately, with vehicles taking action based on erroneous information.

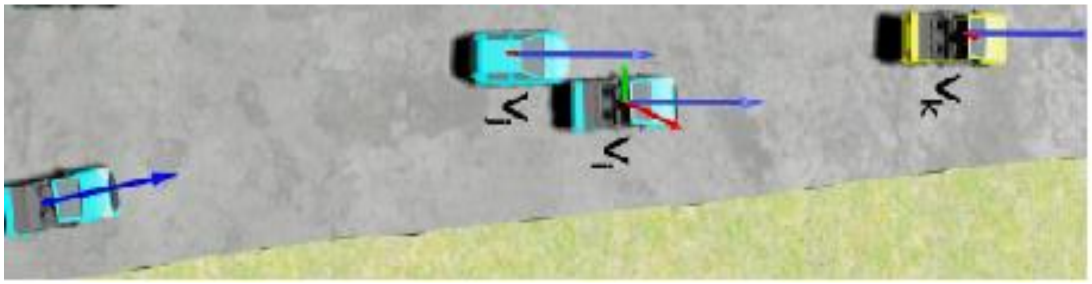


Figure 15 Erroneous Location Position

Figure 15 illustrates the problem of the current cohesion mechanism. v_i, $v_{j,}$ and v_k are travelling in a line. $v_{i\ has}$ queried the roadmap for the positions of nearby boids. The location of v_j has been determined to be at a safe distance, but the location of v_k is too close. Hence v_i reduces speed to avoid colliding with $v_{k.}$ However, within this time step, v_j has come too close, and v_i ended up going off-road to avoid a collision. This is a recurring issue, and it is not obvious why the simple calculation of distance (distance = speed * time) is not producing the anticipated results. One possible solution is to change the simulation time step, to avoid conflicting vehicle adjustments. However, this would require a decoupling of the velocity calculation time step and the UE's GUI refresh rate, as the simulator visualisation consumes a lot of computational resource.

6.2 Evade and Flee

Assigning appropriate weights to the vector containing the location of a following vehicle has proven to be somewhat of a challenge, owing in part to the difficulties in correctly calculating location outlined above. In some instances, it was observed that the weight applied was too insignificant and the resulting steering vector froze the boid in place, causing it to spin on the spot until the following vehicle crashed into it, triggering a pile up.

This is an issue that arises also in part because of the weight attached to the flee behaviour, which keeps the boid from going off road. Without the flee weighting, the boids rarely freeze, but often go off-road to avoid a collision. Subjective validation of course suggests that the earlier behaviour is the one more accurate to real life scenarios, but this is only marginally so. In real life, a driver will see going off road as a last resort, to be taken only when a collision is imminent.

It is difficult to accurately determine the ideal weight for the different behaviours in a project with a scope this small, as discussed below.

In future iterations, a larger road network and more boid vehicles will allow systematic parameter tuning, and better understanding of how to set parameters to achieve realistic emergent traffic flows.

6.3 Environment too small

As mentioned in chapter one, the proof-of-concept project required that the scope be kept small. The road environment is a single lane one way road, and design and implementation have focused on adapting the boid rules for use in boid vehicles which could in future iterations be further expanded to more fully realise the goal of modelling sentient driver behaviour and the resultant emergent traffic flows. The advantage of this approach has been the focus on the boid mechanism, without needing to worry about the effects of the environment on the flock.

However, this approach comes with a significant disadvantage that it some required features cannot be tested. In particular, the align vector is not relevant because there are no vehicles travelling in a different direction to the boid. Also, seek served only as a placeholder as stationary objects such as a free parking spot or an opening to overtake were not present.

The limitations of the environment used in this proof-of-concept may also be a factor in the boid’s early propensity to go off road as there was very little scope for overtaking.

However, this early restriction was considered necessary to allow for more room to emphasis the implementation of the anti-collision behaviour as it was felt that a driver would prioritise not crashing over other concerns.

The end results indicate that while a reasonably good anti-collision function was implemented in the evade function, it requires further development of the other behaviours to truly represent a realistic driver and these need a larger environment with more varied obstacles to be developed further.

Chapter 7: Future Proposals

As discussed in chapters 5 and 6, while the small scope of this project has been met in part, there is still much more that can be achieved in this topic. This chapter discusses three areas of interest that shall be tackled in future iterations. These are: environmental, behavioural, and evaluatory.

7.1 Environmental

The use of UE allows the addition and removal of different city modules without affecting the implementation of the boid. This is a feature which was one of the aims in building the simulation. However, outside of selecting UnReal Engine to build the simulator, we were unable to exploit in the proof-of-concept. By selecting UnReal Engine, we have at least ensured that the development tool supports this feature. An expansion of the existing environment is essential to provide an avenue for assessment of behavioural features like the align vector and the seek function.

A step by step expansion is proposed, with the next step being the inclusion of a two-lane road that the existing road shall merge into. This will allow for assessment of the behaviour of boids when merging with other boids. This will test the converging nature of the vehicular flock and allow further tuning of the cohesive behaviours. Subsequent iterations will add two-way roads, intersections, etc.

The ultimate goal is to build a five-road roundabout, modelled on a roundabout in a major city, for which there is data available on traffic flow and which serve as a standard for comparison as illustrated in the section on future means of evaluation.

Along with the above proposed extensions in road systems, future iterations will add non-vehicular obstacles. These will mainly consist of actors like pedestrians and small animals. They will carry a higher priority of aversion than vehicles as they are much more vulnerable.

A third category of city modules to be added is a different variety of vehicles, such as buses and emergency vehicles, which will have different behavioural rules than regular vehicles.

7.2 Behavioural

As the road system grows more complex, there is room for the development of more behavioural functions, one of which is the Offset Pursuit.

Offset Pursuit is a variation of the pursue behaviour. Instead of plotting a path straight for the target, it plots a path to a certain spot at a certain predetermined distance from the target, but which is still close to it. This has potential for implementing the overtaking behaviour necessary for an accurate representation of real world traffic but has proven difficult to implement using the functions developed thus far.

Another behavioural change that will need to be developed is the cohesion behaviour. In the real world, the appearance of an emergency vehicle triggers the nearby vehicles to make way for it. Vehicles too far to see said vehicle may still move to the side to make room as their drivers will recognise the actions of surrounding vehicles and deduce the reason behind it. This is a clear-cut example of velocity matching in traffic and highlights the importance of further development of the cohesion behaviours.

The introduction of emergency vehicles will also bring a variance of assignation of weights by vehicle type. An emergency vehicle needs to give less weight to collision aversion behaviours as its driver knows that it has the right of way, and other vehicles will make way for it.

A vehicle's behaviour does not change just based on its type but also based on its driver. It would be interesting to observe the implementation of a morality system for drivers, with different drivers having different levels of respect for road safety rules.

This would be implementing a morality system modelled after the Dungeons and Dragons Alignment system.

Each boid would be assigned a moral index from 0-8, with 0 representing lawful good, and 8 representing chaotic evil. Different moral indices would weigh the steering behaviours differently. A chaotic good driver for example may be lax about traffic rules, except for the ones that would be dangerous to vulnerable pedestrians.

LAWFUL GOOD
The needs of the many outweigh the needs of the few.
NEUTRAL GOOD
We can admit that we're killers... but we're not going to kill... today.
CHAOTIC GOOD
I only use it for medicinal purposes.
LAWFUL NEUTRAL
It's incredible that people can exist like this.
TRUE NEUTRAL
Gentlemen, I'm simply an honest businessman!
CHAOTIC NEUTRAL
It's not nice to laugh at people.
LAWFUL EVIL
You humans make a brave noise. There are ways to convince you of your errors.
NEUTRAL EVIL
We will do well in your century, Captain. Do you have any other questions?
CHAOTIC EVIL
Didn't I say you'd better be good to me?

Figure 16 A somewhat subjective representation of the D&D alignment chart using characters from Star Trek the Original Series

That is, he may run a red light but would go slow in a school zone or take particular care not to hit a small animal. Similarly, a chaotic evil boid would be the equivalent of a driver under influence. The addition of a morality index would add the human element deemed necessary for a realistic simulation.

7.3 Evaluation

The ultimate goal of the next iteration of this project is to build a representation of the Old Street roundabout in London. This was chosen for two reasons: first as a reasonably well known feature with its own Wikipedia page [citation to be added], it has available_data that can be fed into an existing and reliable simulator like MATSIM [58], which will provide a point of comparison.

Secondly the Old Street roundabout is slated for renovations to increase road safety because it is one of the more dangerous junctions for cyclists. As such it makes a good test subject to test the second part of our aim, namely to build a simulator that will allow the user to see the effects of proposed changes easily. This will allow us to test and evaluate the research goals outlined in chapter 1, that were not in the scope of this initial project.

Chapter 8: Conclusion

In this project, we have developed a proof-of-concept implementation of Reynolds boid as a vehicle. Beginning with Reynolds' three basic behaviours for modelling the emergent behaviour of a flock, first a theoretical set of vehicular behaviours was developed. These behaviours were then further refined by implementing them as a set of two location vectors and four basic functions.

The boids were first tested individually, then in pairs to observe the behaviours in action and were found to be satisfactory. Testing in larger groups uncovered some interesting results which were outlined in chapter 6. While there is a need for a larger environment in order to further develop the four behaviours, initial observations indicate that the evade behaviour performs to expectations, with occasional issues arising from a difficulty in determining accurate weights for the different behaviours.

Furthermore, it has been demonstrated that the issues still remaining are not because of an incompatibility in method or principle but rather a result of the small scope of the project. Expanding the size will give the boid more room to flex its behaviours and will give more opportunities to further refine the behaviours.

References

M. Balmer, N. Cetin, K. Nagel and B. Raney, "Towards Truly Agent-Based Traffic and Mobility Simulations", in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS, 2004, pp. 60-67.

M. Bando, K. Hasebe, A. Nakayama, A. Shibata and Y. Sugiyama, "Dynamical model of traffic congestion and numerical simulation", *Physical Review E*, vol. 51, no. 2, pp. 1035-1042, 1995.

J. Barceló, *Fundamentals of traffic simulation*. New York: Springer, 2010.

A. Bazzan, "Opportunities for multiagent systems and multiagent reinforcement learning in traffic control", *Autonomous Agents and Multi-Agent Systems*, vol. 18, no. 3, pp. 342-375, 2008.

R. Brooks, "A robust layered control system for a mobile robot", *IEEE J. Robot. Automat.*, vol. 2, no. 1, pp. 14-23, 1986.

"Build software better, together", *GitHub*, 2016. [Online]. Available: https://github.com/. [Accessed: 29- Sep- 2016].

P. Busettea, C. Ghidini, M. Pedrotti and Z. Menestrina, "Briefing Virtual Actors: A First Report on the PRESTO Project", in *50th Annual Convention of the AISB*, 2014.

S. Carpin, M. Lewis, J. Wang, S. Balakirsky and C. Scrappe3r, "USARSim: a robot simulator for research and education", in *IEEE International Conference on Robotics and Automation*, 2004, pp. 1400-1405.

N. Cetin, A. Burri and K. Nagel, "A Large-Scale Agent-Based Traffic Microsimulation Based On Queue Model", in *Swiss Transport Research Conference*, 2003.

N. Cetin, K. Nagel, B. Raney and A. Voellmy, "Large-scale multi-agent transportation simulations", *Computer Physics Communications*, vol. 147, no. 1-2, pp. 559-564, 2002.

R. Chandler, R. Herman and E. Montroll, "Traffic Dynamics: Studies in Car Following", *Operations Research*, vol. 6, no. 2, pp. 165-184, 1958.

A. Croitoru, "Deriving Low-Level Steering Behaviours from Trajectory Data", in *IEEE 13th International Conference on Data Mining Workshops (2009)*, Miami Florida USA, 2009, pp. 583-590.

C. Delgado-Mata, J. Martinez, S. Bee, R. Ruiz-Rodarte and R. Aylett, "On the Use of Virtual Animals with Artificial Fear in Virtual Environments", *New Generation Computing*, vol. 25, no. 2, pp. 145-169, 2007.

H. Dia, "An agent-based approach to modelling driver route choice behaviour under the influence of real-time information", *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 5-6, pp. 331-349, 2002.

H. Espita and J. Soltony, "Path planning of mobile robots using potential fields and swarms of Brownian particles", in *IEEE Congress on Evolutionary Computation*, 2011.

"Fast Facts", *Unity*, 2016. [Online]. Available: https://unity3d.com/public-relations. [Accessed: 29- Sep- 2016].

G. Flierl, D. Grünbaum, S. Levins and D. Olson, "From Individuals to Aggregations: The Interplay between Behavior and Physics", *Journal of Theoretical Biology*, vol. 196, no. 4, pp. 397-454, 1999.

"Game engine", *Wikipedia*, 2016. [Online]. Available: https://en.wikipedia.org/wiki/Game_engine. [Accessed: 29- Sep- 2016].

"GameWorks PhysX Overview", *NVIDIA Developer*, 2014. [Online]. Available: https://developer.nvidia.com/gameworks-physx-overview. [Accessed: 29- Sep- 2016].

C. Gawron, "An Iterative Algorithm to Determine the Dynamic User Equilibrium in a Traffic Simulation Model", *International Journal of Modern Physics C*, vol. 09, no. 03, pp. 393-407, 1998.

"Gears of War - Official Site", *Gearsofwar.com*, 2016. [Online]. Available: https://gearsofwar.com/en-gb. [Accessed: 29- Sep- 2016].

"The GNU General Public License v3.0- GNU Project - Free Software Foundation", *Gnu.org*, 2016. [Online]. Available: http://www.gnu.org/copyleft/gpl.html. [Accessed: 29- Sep- 2016].

H. Goode, C. Palmer and J. Wright, "The Use of a Digital Computer to Model a Signalized Intersection,", *Highway Research Board*, vol. 35, pp. 548-557, 1956.

H. Goode and C. Wendell, "Simulation and Display of Four Interrelated Vehicular Traffic Intersections", 1958.

K. Graft, "id Tech 5 Rage Engine No Longer Up for External Licensing", *Gamasutra.com*, 2016. [Online]. Available: http://www.gamasutra.com/view/news/120702/id_Tech_5_Rage_ Engine_No_Longer_Up_For_External_Licensing.php. [Accessed: 29- Sep- 2016].

C. Hartman and B. Bene, "Autonomous boids", *Comp. Anim. Virtual Worlds*, vol. 17, no. 3-4, pp. 199-206, 2006.

A. Hay, J. de D. Ortuzar and L. Willumsen, "Modelling Transport", *Transactions of the Institute of British Geographers*, vol. 18, no. 1, p. 153, 1993.

G. Hernandes and C. Welborn, "Constrained 3D flocking behavior", in *Consortium for Computing Sciences in Colleges*, 2011, pp. 29-36.

Highway Capacity Manual. Washington, DC, 1985.

"History of Unreal - Part 1 - BeyondUnreal", *Beyondunreal.com*, 2005. [Online]. Available: https://www.beyondunreal.com/articles/history-of-unreal-part-1/?page=2. [Accessed: 29- Sep- 2016].

S. Johansen and A. Løvland, "Flocking for Road Traffic Efficiency Improvement a Concept Study", Masters, Norwegian University of Science and Technology, 2015.

J. Kell, "Analyzing vehicular delay at intersections through simulation", *HRB Bulletin*, vol. 356, 1962.

J. Lewis, D. Brown, W. Cranton and R. Mason, "Simulating visual impairments using the Unreal Engine 3 game engine", in *IEEE 1st International Conference on Serious Games and Applications for Health (SeGAH)*, 2011, pp. 1-8.

M. Lighthill and G. Whitham, "On Kinematic Waves. II. A Theory of Traffic Flow on Long Crowded Roads", *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 229, no. 1178, pp. 317-345, 1955.

"London 'most congested city in Europe' - BBC News", *BBC News*, 2016. [Online]. Available: http://www.bbc.co.uk/news/uk-england-london-34044423. [Accessed: 29- Sep- 2016]

E. Lieberman, "A Brief History of Traffic Simulation", *Annual Meeting of the Trans portation Research Board,* no. 93, 2014.

"MATSim | Multi-Agent Transport Simulation", *Matsim.org*, 2016. [Online]. Available: http://www.matsim.org/. [Accessed: 29- Sep- 2016].

J. Nash, "Non-Cooperative Games", *The Annals of Mathematics*, vol. 54, no. 2, p. 286, 1951.

R. Olfati-Saber, "Flocking for Multi-Agent Dynamic Systems: Algorithms and Theory", *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401-420, 2006

A. Paolo Masucci, Kiril Stanilov and Michael Batty (2013) The growth of London's street network in its dual representation

"Remembrances of Games Past, Part # 25 - Unreal (1998) (PC) (Epic Games)", *YouTube*, 2016. [Online]. Available: https://www.youtube.com/watch?v=BcxPszRZBBA. [Accessed: 29- Sep- 2016].

C. Reynolds, "Flocks, herds and schools: A distributed behavioral model", *ACM SIG GRAPH Computer Graphics*, vol. 21, no. 4, pp. 25-34, 1987.

C. Reynolds, "Steering Behaviours For Autonomous Characters", in *GDC*, 1999

P. Richards, "Shock Waves on the Highway", *Operations Research*, vol. 4, no. 1, pp. 42-51, 1956.

D. Robertson, "TRANSYT: A Traffic Network Study Tool", *Road Research Laboratory Report*, 1969.

D. Robertson, "TRANSYT: Traffic Network Study Tool", *Fourth International Symposium on the Theory of Traffic Flow*, 1968.

"Serious Games Society: About", *Academy.seriousgamessociety.org*, 2016. [Online]. Available: http://academy.seriousgamessociety.org/about. [Accessed: 29- Sep- 2016].

E. Shaw, "The Schooling of Fishes", *Scientific American*, vol. 206, no. 6, pp. 128-141, 1962.

S. Sibley, "NETSIM FOR MICROCOMPUTERS", *Federal Highway Administration*, vol. 49, no. 2, pp. 54-59, 1985.

S. Tan, *"Home, work, play." Urban Redevelopment Authority*. Singapore, 1999.

"Unreal Engine 4 vs. Unity: Which Game Engine Is Best for You?", *Digital-Tutors Blog*, 2014. [Online]. Available: http://blog.digitaltutors.com/unreal-engine-4-vs-unity-game-engine-best/. [Accessed: 29- Sep- 2016].

"Vehicle licensing statistics: 2013 - Publications - GOV.UK", *Gov.uk*, 2014. [Online]. Available: https://www.gov.uk/government/statistics/vehicle-licensing-statistics-2013. [Accessed: 29- Sep- 2016].

J. WARDROP, "ROAD PAPER. SOME THEORETICAL ASPECTS OF ROAD TRAFFIC RESEARCH.", *Proceedings of the Institution of Civil Engineers*, vol. 1, no. 3, pp. 325-362, 1952.

C. Warren, "Global Path Planning Using Artificial Potential Fields", in *IEEE International Conference on Robotics and Automation*, 1989, pp. 316-321.

"What is a Game Engine?- GameCareerGuide.com", *Gamecareerguide.com*, 2016. [Online]. Available: http://www.gamecareerguide.com/features/529/what_is_a_game_.php. [Accessed: 29- Sep- 2016].

"What is Unreal Engine 4", *Unrealengine.com*, 2016. [Online]. Available: https://www.unrealengine.com/what-is-unreal-engine-4. [Accessed: 29- Sep- 2016].

M. Yedlin, A. Lieberman, A. Phlegar, A. Kanaan and A. Santiago, "The New TRAF- NETSIM: Version 4.0", in *73rd Annual Meeting of the Transportation Research Board*, Washington D.C., 1994.

Bibliography

[1]"London 'most congested city in Europe' - BBC News", *BBC News*, 2016. [Online]. Available: http://www.bbc.co.uk/news/uk-england-london-34044423. [Accessed: 29- Sep-2016].

[2] S. Tan, "Home, Work, Play." Urban Redevelopment Authority. Singapore, 1999.

[3]"Vehicle licensing statistics: 2013 - Publications - GOV.UK", *Gov.uk*, 2014. [Online]. Available: https://www.gov.uk/government/statistics/vehicle-licensing-statistics-2013. [Accessed: 29- Sep- 2016].

[4] A. Paolo Masucci, Kiril Stanilov and Michael Batty (2013) The growth of London's street network in its dual representation

[5] Carroll J. Glynn, Susan Herbst, Garrett J. O'Keefe, and Robert Y. Shapiro (2004) Public Opinion

[6] C. Reynolds, "Flocks, herds and schools: A distributed behavioral model", *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, pp. 25-34, 1987.

[7] M. Bando, K. Hasebe, A. Nakayama, A. Shibata and Y. Sugiyama, "Dynamical model of traffic congestion and numerical simulation", *Physical Review E*, vol. 51, no. 2, pp. 1035-1042, 1995.

[8] Chandler, F. E., R. Herman, and E. W. Montroll, (1958). Traffic Dynamics: Studies in Car Following, Operations Science Proceedings of the 3rd International Symposium on Research, 6, pp. 165-184.

[9] Newell G.F. (2002) A simplified car-following theory: a lower order model. Institute of Transportation Studies, University of California, Berkeley.

[10] E. Lieberman, "A Brief History of Traffic Simulation", *Annual Meeting of the Transportation Research Board*, no. 93, 2014.

[11] J. Kell, "Analyzing vehicular delay at intersections through simulation", *HRB Bulletin*, vol. 356, 1962.

[12] H. Goode, C. Palmer and J. Wright, "The Use of a Digital Computer to Model a Signalized Intersection,", *Highway Research Board*, vol. 35, pp. 548-557, 1956.

[13] H. Goode and C. Wendell, "simulation and Display of Four Interrelated Vehicular Traffic Intersections", 1958.

[14] J. WARDROP, "ROAD PAPER. SOME THEORETICAL ASPECTS OF ROAD TRAFFIC RESEARCH.", *Proceedings of the Institution of Civil Engineers*, vol. 1, no. 3, pp. 325-362, 1952.

[15] J. Nash, "Non-Cooperative Games", *The Annals of Mathematics*, vol. 54, no. 2, p. 286, 1951.

[16] J. Barceló, *Fundamentals of traffic simulation*. New York: Springer, 2010.

[17] M. Lighthill and G. Whitham, "On Kinematic Waves. II. A Theory of Traffic Flow on Long Crowded Roads", *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 229, no. 1178, pp. 317-345, 1955.

[18] P. Richards, "Shock Waves on the Highway", *Operations Research*, vol. 4, no. 1, pp. 42-51, 1956.

[19] D. Robertson, "TRANSYT: A Traffic Network Study Tool", *Road Research Laboratory Report*, 1969.

[20] D. Robertson, "TRANSYT: Traffic Network Study Tool", *Fourth International Symposium on the Theory of Traffic Flow*, 1968.

[21] M. Yedlin, A. Lieberman, A. Phlegar, A. Kanaan and A. Santiago, "The New TRAF-NETSIM: Version 4.0", in *73rd Annual Meeting of the Transportation Research Board*, Washington D.C., 1994.

[22] *Highway Capacity Manual*. Washington, DC, 1985.

[23] N. Cetin, A. Burri and K. Nagel, "A Large-Scale Agent-Based Traffic Microsimulation Based On Queue Model", in *Swiss Transport Research Conference*, 2003.

[24] M. Balmer, N. Cetin, K. Nagel and B. Raney, "Towards Truly Agent-Based Traffic and Mobility Simulations", in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS, 2004, pp. 60-67.

[25] C. Gawron, "An Iterative Algorithm to Determine the Dynamic User Equilibrium in a Traffic Simulation Model", *International Journal of Modern Physics C*, vol. 09, no. 03, pp. 393-407, 1998.

[26] N. Cetin, K. Nagel, B. Raney and A. Voellmy, "Large-scale multi-agent transportation simulations”, *Computer Physics Communications*, vol. 147, no. 1-2, pp. 559-564, 2002.

[27] H. Dia, "An agent-based approach to modelling driver route choice behaviour under the influence of real-time information", *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 5-6, pp. 331-349, 2002.

[28] Praveen Paruchuri, Alok Reddy Pullalarevu, and Kamalakar Karlapalem. 2002. “Multi agent simulation of unorganized traffic”. In Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1 (AAMAS '02).

[29] Arnaud Doniec, René Mandiau, Sylvain Piechowiak, Stéphane Espié,” A behavioral multi-agent model for road traffic simulation”, Engineering Applications of Artificial Intelligence, Volume 21, Issue 8, December 2008, Pages 1443-1454, ISSN 0952-1976

[30] F. Qiu and X. Hu, "Modeling Dynamic Groups for Agent-Based Pedestrian Crowd Simulations," 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Toronto, ON, 2010, pp. 461-464

[31] E. Shaw, "The Schooling of Fishes", *Scientific American*, vol. 206, no. 6, pp. 128-141, 1962.

[32] R. Olfati-Saber, "Flocking for Multi-Agent Dynamic Systems: Algorithms and Theory", *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401-420, 2006.

[33] C. Warren, "Global Path Planning Using Artificial Potential Fields", in *IEEE International Conference on Robotics and Automation*, 1989, pp. 316-321.

[34] G. Hernandes and C. Welborn, "Constrained 3D flocking behavior", in *Consortium for Computing Sciences in Colleges*, 2011, pp. 29-36.

[35] H. Espita and J. Soltony, "Path planning of mobile robots using potential fields and swarms of Brownian particles", in *IEEE Congress on Evolutionary Computation*, 2011.

[36] C. Hartman and B. Bene, "Autonomous boids", *Comp. Anim. Virtual Worlds*, vol. 17, no. 3-4, pp. 199-206, 2006.

[37] A. Croitoru, "Deriving Low-Level Steering Behaviours from Trajectory Data", in *IEEE 13th International Conference on Data Mining Workshops (2009)*, Miami Florida USA, 2009, pp. 583-590.

[38] G. Flierl, D. Grünbaum, S. Levins and D. Olson, "From Individuals to Aggregations: The Interplay between Behavior and Physics", *Journal of Theoretical Biology*, vol. 196, no. 4, pp. 397-454, 1999.

[39] C. Delgado-Mata, J. Martinez, S. Bee, R. Ruiz-Rodarte and R. Aylett, "On the Use of Virtual Animals with Artificial Fear in Virtual Environments", *New Generation Computing*, vol. 25, no. 2, pp. 145-169, 2007.

[40] R. Brooks, "A robust layered control system for a mobile robot", *IEEE J. Robot. Automat.*, vol. 2, no. 1, pp. 14-23, 1986.

[41]"MATSim | Multi-Agent Transport Simulation", *Matsim.org*, 2016. [Online]. Available: http://www.matsim.org/. [Accessed: 29- Sep- 2016].

[42]"What is a Game Engine? - GameCareerGuide.com", *Gamecareerguide.com*, 2016. [Online]. Available: http://www.gamecareerguide.com/features/529/what_is_a_game_.php. [Accessed: 29- Sep- 2016].

[43]"Game engine", *Wikipedia*, 2016. [Online]. Available: https://en.wikipedia.org/wiki/Game_engine. [Accessed: 29- Sep- 2016].

[44]"Serious Games Society: About", *Academy.seriousgamessociety.org*, 2016. [Online]. Available: http://academy.seriousgamessociety.org/about. [Ac cessed: 29- Sep- 2016].

[45]"The GNU General Public License v3.0- GNU Project - Free Software Foundation", *Gnu.org*, 2016. [Online]. Available: http://www.gnu.org/copyleft/gpl.html. [Accessed: 29- Sep- 2016].

[46] K. Graft, "id Tech 5 Rage Engine No Longer Up for External Licensing", *Gamasutra.com*, 2016. [Online]. Available: http://www.gamasutra.com/view/news/120702/id_Tech_5_Rage_Engine_No_Longer_Up_For_External_Licensing.php. [Accessed: 29- Sep- 2016].

[47]"Fast Facts", *Unity*, 2016. [Online]. Available: https://unity3d.com/public-relations. [Accessed: 29- Sep- 2016].

[48]"Build software better, together", *GitHub*, 2016. [Online]. Available: https://github.com/. [Accessed: 29- Sep- 2016].

[49]"What is Unreal Engine 4", *Unrealengine.com*, 2016. [Online]. Available: https://www.unrealengine.com/what-is-unreal-engine-4. [Accessed: 29- Sep- 2016].

[50]"Unreal Engine 4 vs. Unity: Which Game Engine Is Best for You?", *Digital-Tutors Blog*, 2014. [Online]. Available: http://blog.digitaltutors.com/unreal-engine-4-vs-unity-game-engine-best/. [Accessed: 29- Sep- 2016].

[51]"History of Unreal - Part 1 - BeyondUnreal", *Beyondunreal.com*, 2005. [Online]. Available: https://www.beyondunreal.com/articles/history-of-unreal-part-1/?page=2. [Accessed: 29- Sep- 2016].

[52]"Remembrances of Games Past, Part # 25 - Unreal (1998) (PC) (Epic Games)", *YouTube*, 2016. [Online]. Available: https://www.youtube.com/watch?v=BcxPszRZBBA. [Accessed: 29- Sep- 2016].

[53]"Gears of War - Official Site", *Gearsofwar.com*, 2016. [Online]. Available: https://gearsofwar.com/en-gb. [Accessed: 29- Sep- 2016].

[54] J. Lewis, D. Brown, W. Cranton and R. Mason, "Simulating visual impairments using the Unreal Engine 3 game engine", in *IEEE 1st International Conference on Serious Games and Applications for Health (SeGAH)*, 2011, pp. 1-8.

[55]"GameWorks PhysX Overview", *NVIDIA Developer*, 2014. [Online]. Available: https://developer.nvidia.com/gameworks-physx-overview. [Accessed: 29- Sep- 2016].

[56] S. Carpin, M. Lewis, J. Wang, S. Balakirsky and C. Scrappe3r, "USARSim: a robot simulator for research and education", in *IEEE International Conference on Robotics and Automation*, 2004, pp. 1400-1405.

[57] C. Reynolds, "Steering Behaviours for Autonomous Characters", in *GDC*, 1999.

[58] S. Johansen and A. Løvland, "Flocking for Road Traffic Efficiency Improvement a Concept Study", Masters, Norwegian University of Science and Technology, 2015.